8

Open-Source GIS

8.1 What Is Open-Source GIS?

In a kitchen, a chef may be privileged to have a wide array of tools: a German 8 in, "chef's knife" for chopping vegetable, a santoku 7 in. blade for mincing, an immersion blender for making sauces, and so on. To a casual observer, some blades and appliances might look the same yet each has distinct advantages or disadvantages depending on the task they are used for. A chef may also choose a tool based on subjective reasons, such as experience or comfort level. In a similar way, being aware of different ways to approach GIS problems using different software tools is a valuable skill. But in other ways this analogy quickly falls apart. While software programs are tools, they are complex and learning and keeping up with multiple GIS programs can be difficult. To add to this complexity, it is common for GIS software programs to significantly change between versions, with changes to tools and other functionality.

In this book, we focus on the ESRI ArcMap program, one of the market leaders in GIS software across private industry and governmental sectors in the United States. However, other GIS programs such as QGIS have traction and a wide user base, especially in other parts of the world such as Europe and Asia. QGIS is a desktop GIS program, just like ArcMap, with many of the same features as ArcMap. Although the tools within QGIS and ArcMap might have different names, GIS professionals can usually accomplish the same tasks in both QGIS and ArcMap, though the speed of completing tasks may differ. However, a major distinction between QGIS and ArcMap is that ArcMap is a "closed source" proprietary system while QGIS is an "open source" program. This means that the core code that powers the ArcMap program is copyrighted and cannot be used for free, whereas the code for QGIS is freely available without cost. QGIS has been built entirely by volunteers creating and sharing code to build and improve upon the software over time.

This gives QGIS (and other similar open-source GIS programs) some key situational advantages not shared by paid software. First, some organizations such as nonprofit organizations or government offices in developing countries lack adequate resources to invest in paid GIS software. While GIS companies often give steep discounts on their paid software to nonprofit companies or for educational purposes, this is not guaranteed. By comparison, when a GIS software product is released as an open-source program, it will always be free.

There are several-open source GIS programs besides QGIS with their own histories. GRASS GIS (Geographic Resources Analysis Support System) dates back to 1982 and is an open-source program that was originally created by the US Military. In the 1990s, the GRASS was put into the public domain and its software development was taken over by Baylor University, later released with a permissive "GNU General Public License", allowing people to make changes and contributions to the code as long as the code changes were transparent and viewable by all parties. Today, GRASS GIS is a stable release that can be used through the command line or with a GUI. GRASS GIS's functionalities and core features are especially tied to scientific uses, and there are some uses of GRASS GIS that are especially powerful in these areas due to a long history of the codebase. Furthermore, due to collaboration between the QGIS and GRASS project teams and volunteers, many of these tools and features can be called directly in QGIS using the GRASS plugin, augmenting the existing features of QGIS.

There are many other open-source desktop GIS releases. To name a few, gVSIG, which is especially good for 3D visualization, SAGA GIS (with a focus on scientific GIS applications similar to GRASS), and GeoDa (which is excellent for exploratory spatial data analysis). But remember that you ultimately want the best tool for whatever GIS task you have at hand. For a complex application such as GIS desktop software, it's important to choose software that is actively updated by a large userbase of volunteers. This helps make sure that the tool keeps pace with current trends, bugs and issues in programs are timely addressed, and security defects that could harm your computer are kept to a minimum.

8.2 Getting Started with QGIS

QGIS is installed as an application, just like ArcMap or other desktop GIS. QGIS is available across several different platforms: Windows, Mac, and even Linux. "Plugins," community-developed code packages that add features to QGIS, work independently of the operating system QGIS is installed on. As of this writing, QGIS is installed by visiting their main website, "https://qgis.org" (Figure 8.1).

In this chapter, version 3.0 of QGIS is used, at the time of writing the latest QGIS release, but QGIS (and many open-source projects) also has a version 2.18 "long-term release." QGIS, along with other open-source software projects, releases two versions supported at the same time; one including the most latest features that may not be fully tested for stability, and the other a

FIGURE 8.1
The homepage for QGIS, at " https://qgis.org." The different tabs at the top, Discover QGIS, For Users, Get Involved, and Documentation, are worth exploring to understand the philosophy behind QGIS as well as to get up to speed on how to download and install QGIS. For Users provides the base installer for QGIS.

"long-term release" designed with stability in mind. If you experience unexpected issues such as crashing or tools not working, you may want to try a long-term release instead of the latest release.

We will begin by getting data into QGIS. The program supports a wide array of GIS data types, CSV, shapefiles, GeoJSON, raster formats, and PostGIS databases, but CSV files and shapefiles are fairly ubiquitous in the GIS world so importing CSV files is a good start to get familiar with file importation. We will start with a CSV file downloaded from the Chicago Data Portal, at https://data.cityofchicago.org, where you can find a whole catalog of data related to city issues. Some of the files on the Chicago Data Portal can be downloaded in shapefile format and used without much processing in a GIS project, but many are in Excel or CSV format and need a bit more work to be mapped within GIS software. Figure 8.2 shows a snapshot of the "Map of Urban Farms" dataset with the different icons for managing/exporting the data. Using the search bar in the middle of the site (or navigating directly to the dataset at https://data.cityofchicago.org/Environment-Sustainable-Development/Map-of-Urban-Farms/uti6-fp3f) look for the "Map of Urban Farms" dataset and export it as a CSV.

If you were to open up this file in a spreadsheet program such as Microsoft Excel, you would see that it has both addresses and latitude/longitude coordinates in fields within the sheet. If we import this CSV into QGIS, with just a few clicks we can map the points within the program, and from there, export the file into other formats.

Once QGIS is installed, and the example CSV is downloaded, click the "Data Source Manager" on the far left (the icon that appears as three sticky notes with a plus sign) or alternatively, either click the Layer tab and click Data Source Manager, or click the Layer tab, click "Add Layer, and then click "Add

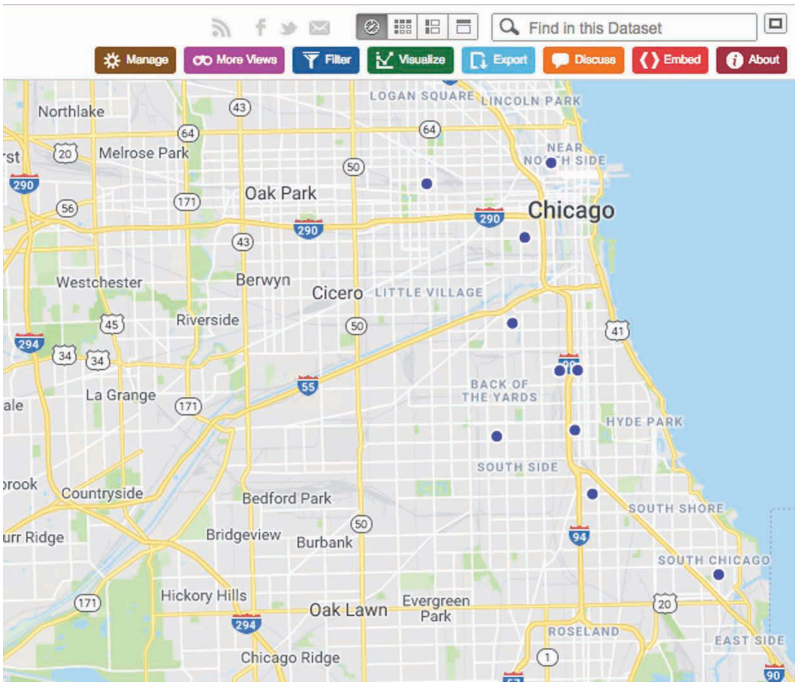


FIGURE 8.2
"Map of Urban Farms" from the " Chicago Data Portal," a source for information, GIS and otherwise, relating to the City of Chicago. By clicking the "Export" tab, this particular dataset can be exported in CSV format.

Delimited Text Layer." (Since CSV files are a type of Delimited Text Layer, this is the correct choice to add this file type.) As you can see, there are a wide variety of layers types that can be brought in to QGIS, with buttons corresponding to Raster layer file types, Vector shapefiles, and so on. You will even see layer types such as "ArcGIS Map Server" and "ArcGIS Feature Server." These two layer types are ArcGIS web maps hosted online that are linked to the QGIS application and mapped inside the desktop. Even though these types of online maps can only be hosted using paid ArcGIS products, QGIS can use them in proprietary layers. However, note that if we were attempting to import a GIS dataset in shapefile format instead of a CSV, all we would have to do is choose "Vector" in the Data Source Manager and select our shapefile, a process not much different from importing shapefiles in ArcMap.

Once you have specified that you want to bring in a Delimited Text layer, a prompt will appear giving you the opportunity to specify the location of the file and set different flags. The three periods to the right of the "File Name" prompt allow you to choose the file, at which point it should automatically set the "File Format" flag to "CSV." However, you will still have to set the "Geometry definition" so QGIS will know how to display the CSV file as mapped points.

You might intuit that we should set the *X* and *Y* fields to the respective Longitude and Latitude field values. However, understanding the dataset that we are working with is prudent to avoid unnecessary headaches. The Chicago Data Portal is maintained using a service called "Socrata," which is a bit hidden on the website. Clicking the "Documentation" button at the bottom right of the page brings us to the "Socrata" home page, but it might not be obvious that the website runs using Socrata, nor why that matters. But if you were to dig in to Socrata's documentation, you would learn three important things:

1. Socrata CSV datasets with geometry information default to the EPSG:4326 coordinate system.
2. Socrata coordinates are encoded in longitude, latitude order.
3. They are DD (Decimal Degrees) rather than DMS (Degrees, Minutes, Seconds).

Figure 8.3 shows the QGIS dialog with these things. If we did not uncheck "DMS coordinates" in the dialog, set the correct coordinate system, or

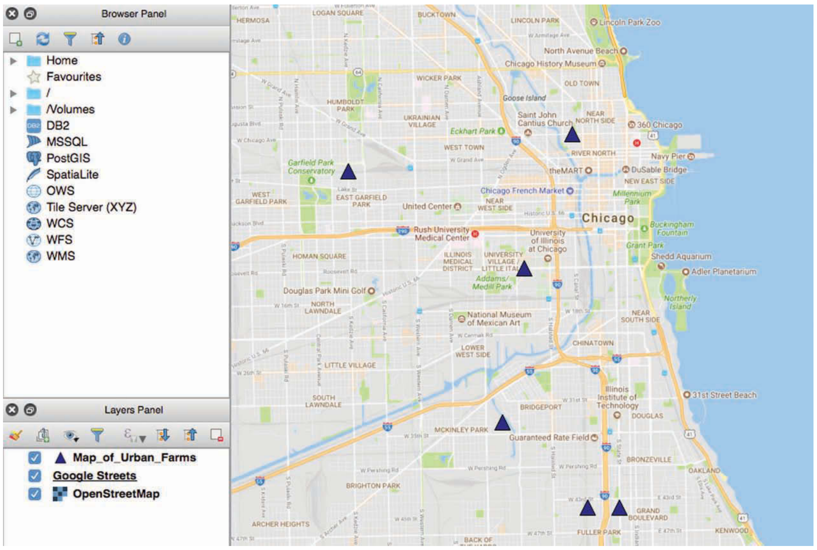
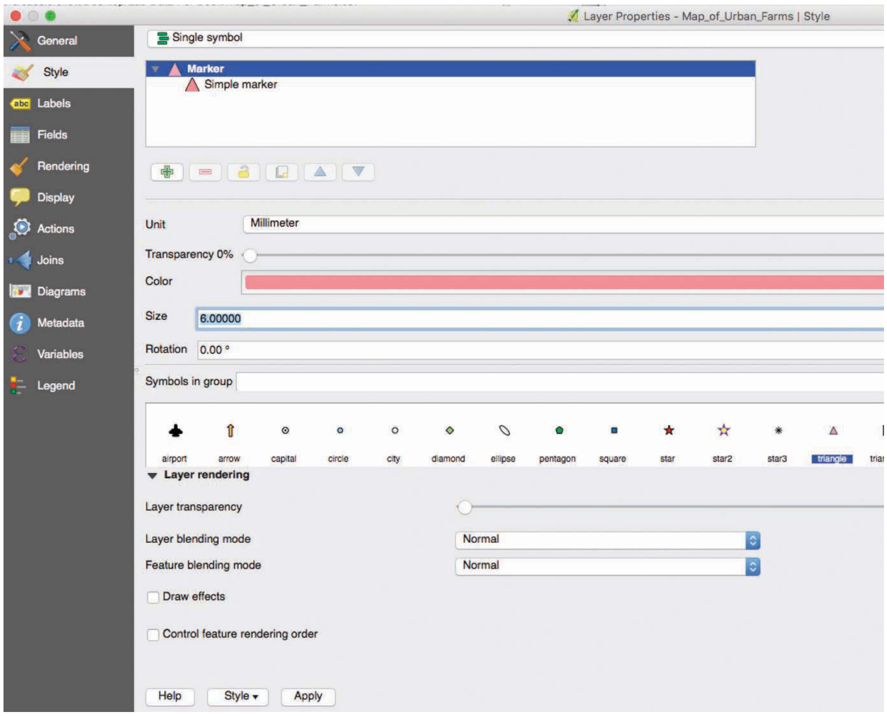


FIGURE 8.3
CSV Import dialog. (a) Farmers markets mapped. Notice the "Layers Panel" in the bottom left. If Map_of_Urban_Farms is below the Google Streets basemap, it will not show since the top layers of the map were drawn first (and thus cover our vector point layer entirely). To access basemaps in QGIS, go to "Plugins," "Manage Plugins," and then search for the "OpenLayers" plugin. You can then bring in several types of basemaps as layers by clicking "Web" and then navigating to the OpenLayers plugin. If you are using QGIS 3.0 or above, look for the QuickMapServices plugin instead.

(*Continued*)



Layer Properties - Map_of_Urban_Farms | Style

Single symbol

Marker
Simple marker

Unit: Millimeter
Transparency 0%
Color
Size: 6.00000
Rotation: 0.00 °
Symbols in group

Layer rendering
Layer transparency
Layer blending mode: Normal
Feature blending mode: Normal
Draw effects
Control feature rendering order

FIGURE 8.3 (CONTINUED)
(b) Layer Properties settings, similar to ArcMap.

indicate the *X* and *Y* fields correctly, the CSV file would be imported into QGIS, but it would not properly display the points. In Figure 8.3a you can see the urban farms mapped inside QGIS, denoted by large triangles. Similar to ArcMap, the symbology can be adjusted by right-clicking the layer in the sidebar (called "Layers"), entering the "Layer Properties" settings, and scrolling to the Symbology tab. Figure 8.3b shows the different settings that can be adjusted within the Layer Properties. As noted, in the Symbology you can adjust the size, color, and shape of symbols, choose choropleth maps with classification methods (Quantile, Natural Breaks, Equal Interval), and change other symbology settings. Other settings can be controlled through the Layer Properties: The Labels setting allows you to use a field to label points, polygons, or lines and the Joins tab allows for table joins and relates, similar to how this is done in ArcMap.

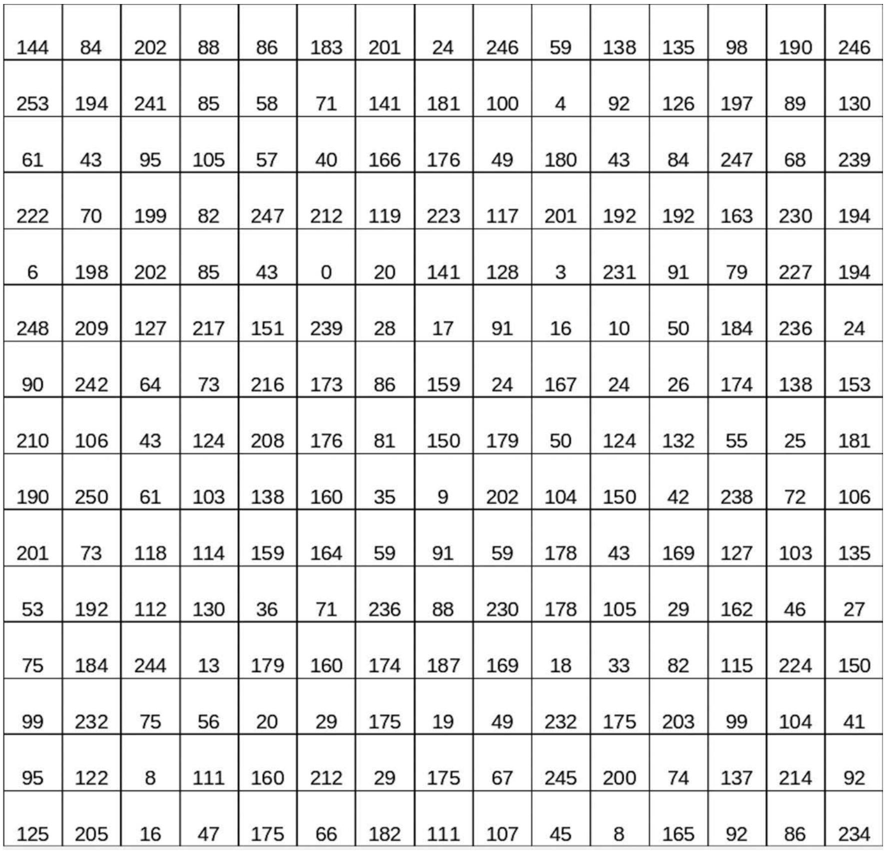
We are working with a CSV file that includes coordinate data, but what if we need to bring in a CSV file that only has addresses? We would have to geocode the file, a process that is slightly different in QGIS than in ArcMap. As a reminder, geocoding is the process of looking up addresses and geographic information and translating them into coordinate information, allowing geometry to be associated with a dataset. When geocoding in

ArcMap, it is typical to conduct the process using an ArcGIS Online account, calling an ESRI service that returns information about the coordinates of a given address. This typically requires a small amount of "ArcGIS Online service credits" such that large geocoding requests over time end up costing money. While geocoding in QGIS is free there is a different catch. Since geocoding uses servers available for the public such as Google Maps and OpenStreetMap, attempting large geocoding projects can cause you to hit "usage limits" quickly.

8.3 GDAL and Raster Data

GIS analysis often uses raster datasets. While vector datasets are represented by points, lines, and polygons and contain an attribute table, raster datasets are structured instead as an array of cells, or pixels. Think of a digital image; even with a very good camera, if you zoom in close enough, you will start to see small squares of a single color. Those squares are called pixels. The resolution of a picture signifies how many pixels in total are in the picture; a resolution of 600×400 means the image is 600 square pixels of width and 400 square pixels of height. In a digital photo, each pixel contains a value that is used to map color to each pixel. In a grayscale image, one value alone for each cell is typically enough to determine an intensity from black to white, as seen in Figures 8.4 and 8.5. In color images, "bands" create a composite of three different values for each pixel; these three bands represent red, green, and blue, or RGB. This is how cameras store their digital images, and it is also how aerial photography, earth monitoring satellites, and drones equipped with cameras record and store data.

The processing and analysis of satellite imagery, aerial photography, and other similar raster datasets is the primary focus of a sister technology called "remote sensing." Operations and analysis for remote sensing raster datasets demand completely different tools and operations than vector datasets. However, because raster datasets represent continuous data, they are especially effective for certain types of analysis. Imagine we have a raster dataset that has been generated from a satellite in space. This satellite has an instrument that can sense the surface temperature in an area while it passes over in space. Since the satellite can record extremes in temperature, we can look for extremes of temperature and map active fires. In simple terms, this is how NASA scientists use thermal data from the Moderate Resolution Imaging Spectroradiometer (MODIS) in orbit to map active wild fires here on earth, as shown in Figure 8.6. Other satellites can record real color images with RGB color. These are often used as "basemaps" in regular GIS programs, with vector datasets overlaid. The power of remote sensing imagery is that, using a particular imaging satellite, you can sweep the same area year to year

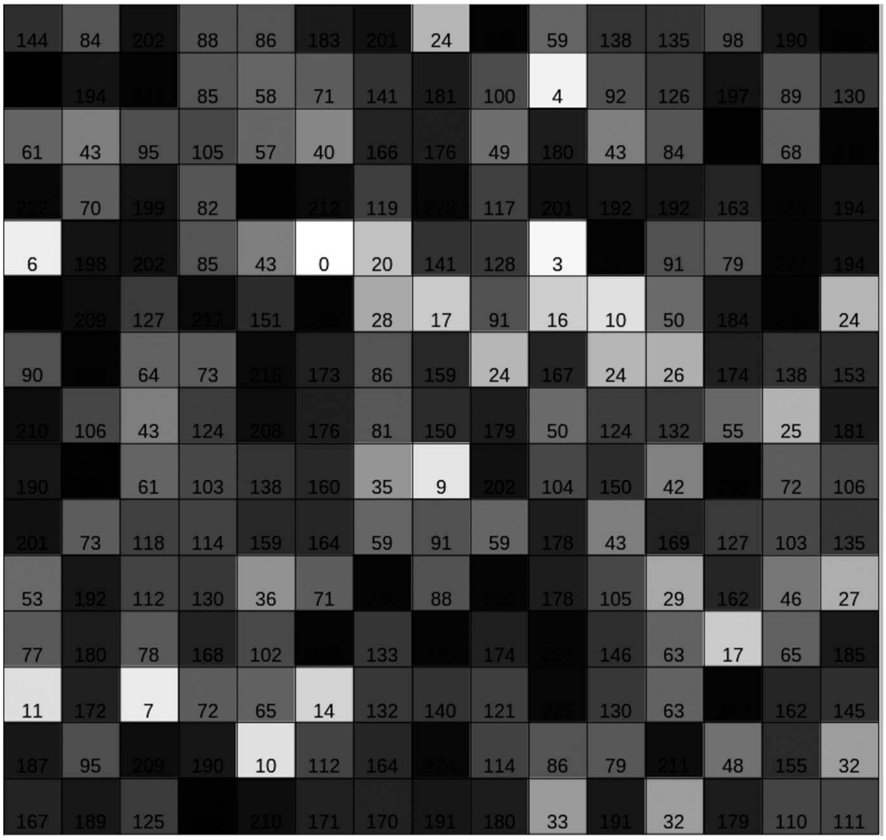


144	84	202	88	86	183	201	24	246	59	138	135	98	190	246
253	194	241	85	58	71	141	181	100	4	92	126	197	89	130
61	43	95	105	57	40	166	176	49	180	43	84	247	68	239
222	70	199	82	247	212	119	223	117	201	192	192	163	230	194
6	198	202	85	43	0	20	141	128	3	231	91	79	227	194
248	209	127	217	151	239	28	17	91	16	10	50	184	236	24
90	242	64	73	216	173	86	159	24	167	24	26	174	138	153
210	106	43	124	208	176	81	150	179	50	124	132	55	25	181
190	250	61	103	138	160	35	9	202	104	150	42	238	72	106
201	73	118	114	159	164	59	91	59	178	43	169	127	103	135
53	192	112	130	36	71	236	88	230	178	105	29	162	46	27
75	184	244	13	179	160	174	187	169	18	33	82	115	224	150
99	232	75	56	20	29	175	19	49	232	175	203	99	104	41
95	122	8	111	160	212	29	175	67	245	200	74	137	214	92
125	205	16	47	175	66	182	111	107	45	8	165	92	86	234

FIGURE 8.4
A grid of values ranging from 0 to 255, without color assigned to each value.

and have the same resolution of the resulting image, and look for changes between time periods. For instance, you could compare the chlorophyll in tree cover in a national park over the course of a decade to gauge overall tree health, or compare land cover change in a city in terms of development, as Figure 8.7 shows.

Both proprietary and open-source tools are available that work with raster datasets. ArcMap has the "Raster Calculator" toolset among other tools, and other products such as ERDAS IMAGINE and ENVI are specifically designed to work with raster datasets and remote sensing imagery. However, because of the long history of scientists and public organizations creating their own tools and algorithms to work with raster datasets, there are robust and well-supported open-source programs that offer the same functionality, and in some cases, unique functionality. QGIS has raster tools comparable to ArcMap, GRASS GIS has a legacy of tools and libraries going back decades,



144	84	202	88	86	183	201	24	[illegible]	59	138	135	98	190	[illegible]
[illegible]	194	[illegible]	85	58	71	141	181	100	4	92	126	197	89	130
61	43	95	105	57	40	166	176	49	180	43	84	[illegible]	68	[illegible]
222	70	199	82	[illegible]	212	119	222	117	201	192	192	163	[illegible]	194
6	198	202	85	43	0	20	141	128	3	[illegible]	91	79	[illegible]	194
[illegible]	209	127	217	151	[illegible]	28	17	91	16	10	50	184	[illegible]	24
90	[illegible]	64	73	216	173	86	159	24	167	24	26	174	138	153
210	106	43	124	208	176	81	150	179	50	124	132	55	25	181
190	[illegible]	61	103	138	160	35	9	202	104	150	42	[illegible]	72	106
201	73	118	114	159	164	59	91	59	178	43	169	127	103	135
53	192	112	130	36	71	[illegible]	88	[illegible]	178	105	29	162	46	27
77	180	78	168	102	[illegible]	133	[illegible]	174	[illegible]	146	63	17	65	185
11	172	7	72	65	14	132	140	121	225	130	63	[illegible]	162	145
187	95	209	190	10	112	164	224	114	86	79	211	48	155	32
167	189	125	[illegible]	210	171	170	191	180	33	191	32	179	110	111

FIGURE 8.5
The same grid from Figure 8.4, with color values assigned, such that a low value is a low intensity (0 is completely white) and a high value is a higher intensity (255 is completely black). Color ramps are assigned by the user, so this could easily be reversed in the GIS program such that 0 is pitch black, and 255 is completely white.

and NASA releases tools like Worldview to interact with NASA remote sensing datasets. The Geospatial Data Abstraction Library (GDAL) is an open-source software library that dates back to the year 2000 and now powers raster analysis in several programs, both proprietary and open source. A "software library" refers to computer code designed to provide specific functions (such as raster analysis) that can be called from other software. For example, in ArcMap you can convert a raster to another raster format using the "gdal_translate" utility, while in QGIS, the same "gdal_translate" function is used to accomplish the same goal. In other words, if you are familiar with and understand the GDAL software library, you can perform raster analysis regardless of which software you use.

Older maps completed before GIS or even computers were common can be scanned and brought into GIS applications as raster datasets as well. If

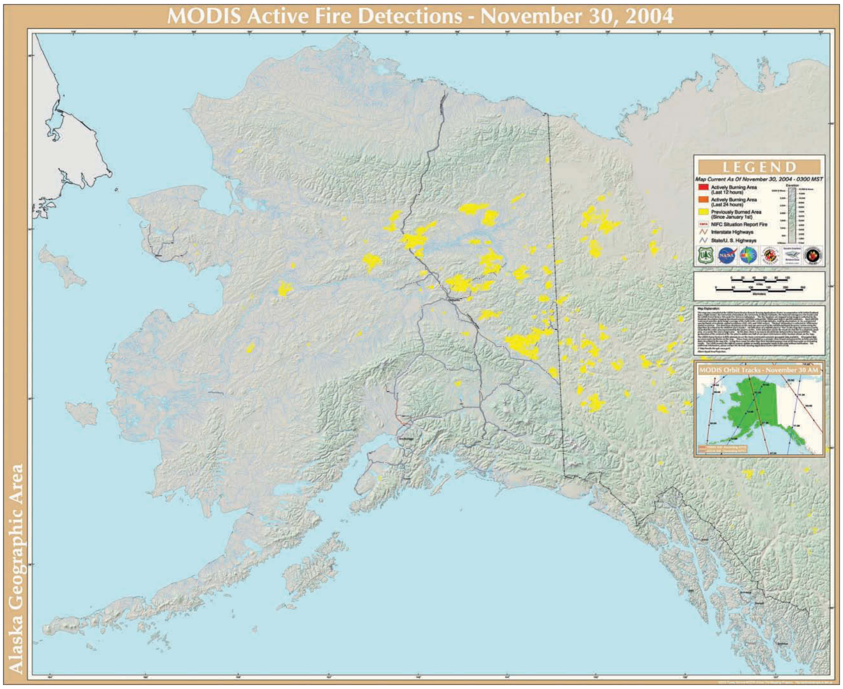
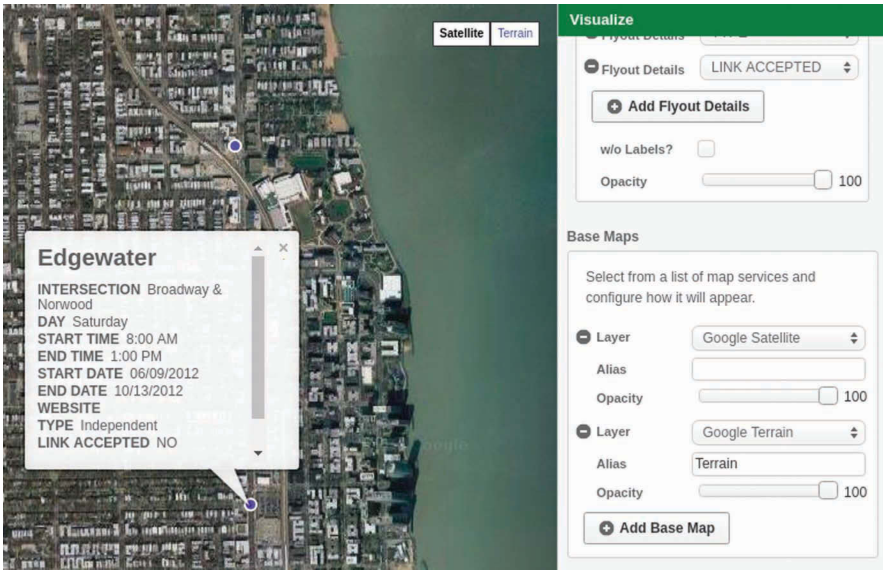

MODIS Active Fire Detections - November 30, 2004
Alaska Geographic Area
LEGEND
Map Current As Of November 30, 2004 - 0300 MST
Actively Burning Area (Last 12 hours)
Actively Burning Area (Last 24 hours)
Previously Burned Area (Since January 1st)
NIFC Situation Report Fire
Interstate Highways
State/U. S. Highways
Map Explanation:
MODIS Orbit Tracks - November 30 AM

FIGURE 8.6
MODIS fire production.



Edgewater
INTERSECTION Broadway & Norwood
DAY Saturday
START TIME 8:00 AM
END TIME 1:00 PM
START DATE 06/09/2012
END DATE 10/13/2012
WEBSITE
TYPE Independent
LINK ACCEPTED NO

Flyout Details LINK ACCEPTED
Add Flyout Details
w/o Labels?
Opacity 100

Base Maps
Select from a list of map services and configure how it will appear.
Layer Google Satellite
Alias
Opacity 100
Layer Google Terrain
Alias Terrain
Opacity 100
Add Base Map

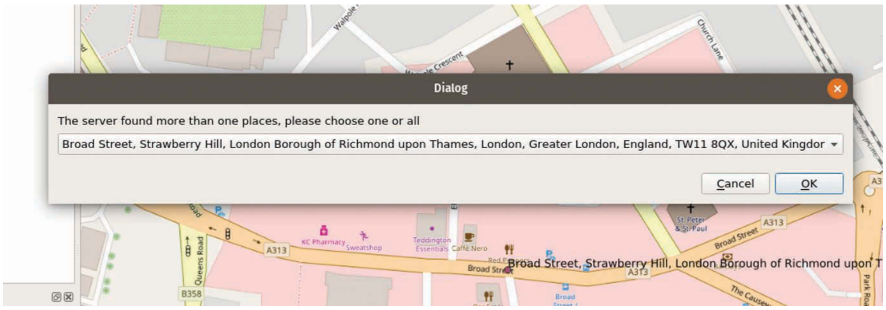
FIGURE 8.7
LANDSAT basemap.

we wanted to take an older map of a city from the eighteenth century and overlay it over a current map of the city, we could not just simply import the older map into the GIS system and be done because the current map would most likely have a map project and coordinate system set, and the older map would not. Even if we applied the correct coordinate system and projection to the older map, we would also have to warp the map so that the corners matched the current extent and position of the subject area being analyzed.

Consider the map created by Dr. John Snow investigating the cholera epidemic surrounding the Broad Street water pump in London 1854. You can find the map in the supplemental files corresponding to this chapter. First, if you have not already, install QGIS. If you are using QGIS 2.18 or earlier, you should also install the plugin "OpenLayers." If you are using the newer QGIS 3.0, you should enable the "QuickMapServices" plugin. Whichever version you are using, enable the "OSM Standard" layer from either OpenLayers or the QuickMapServices plugin. Zoom down to the small neighborhood around Broad Street. An easy way to do this is by clicking the GeoCoding button and, when prompted, entering "Broad Street, London, UK." This will prompt you to select from a few different possible full addresses that have the complete location (as shown in Figure 8.8). You should now have a single point geolocated in the neighborhood that you can right-click in the Layer List sidebar to zoom in to.

Now we will bring in the John Snow cholera map. Go to QGIS "Plugins" in the menu and then to "Manage and Install Plugins." Type in "Georeferencer GDAL" and ensure the plugin is activated. It should now be available as the "Georeferencer" tool under the Raster tab. Click on the 'Georeferencer' tool to start the plugin up and click on "Add Raster." Next, add the cholera map as shown in Figure 8.9, ignoring any questions about applying transformations (as you do not yet have control points yet).

Figure 8.10 shows the Georeferencer tool with the 1856 map imported as a raster. Review the supplemental lab for naming conventions for the raster. When hovered over, there are icons in the navigational bar of the georeferencer


Dialog

The server found more than one places, please choose one or all

Broad Street, Strawberry Hill, London Borough of Richmond upon Thames, London, Greater London, England, TW11 8QX, United Kingdor

Cancel
OK
Broad Street, Strawberry Hill, London Borough of Richmond upon T
A313
Queens Road
B358
Church Lane
Broad Street

FIGURE 8.8
Geocoding tool.

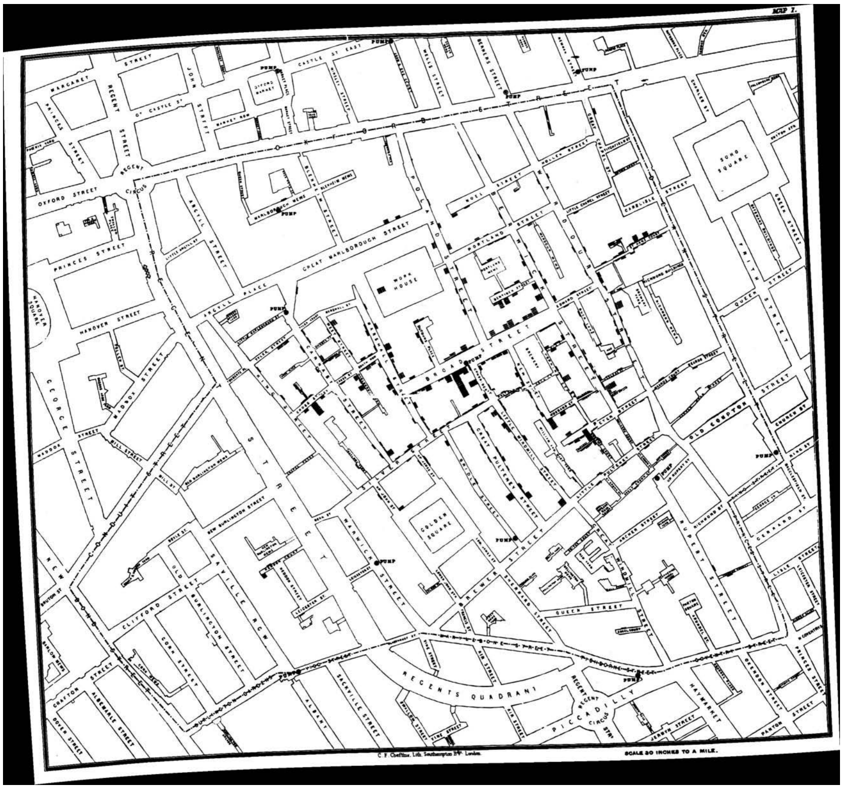
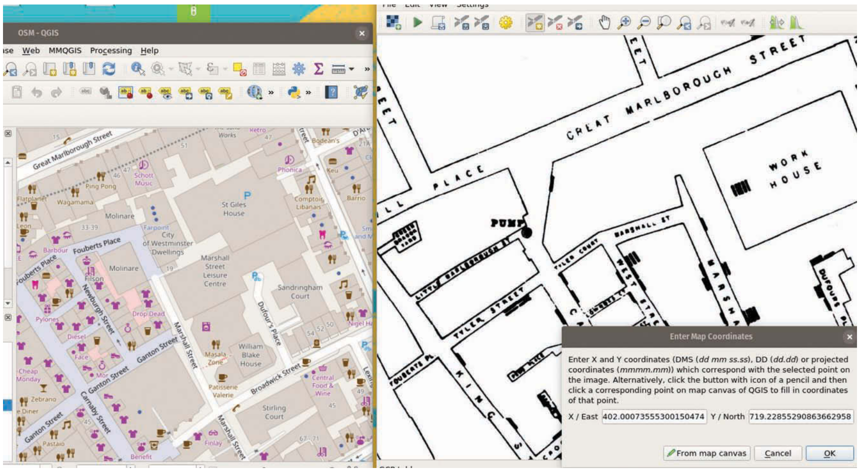


FIGURE 8.9
Adding the cholera map.



Enter Map Coordinates

Enter X and Y coordinates (DMS (*dd mm ss.ss*), DD (*dd.dd*) or projected coordinates (*mmmm.mm*)) which correspond with the selected point on the image. Alternatively, click the button with icon of a pencil and then click a corresponding point on map canvas of QGIS to fill in coordinates of that point.

X / East 402.00073555300150474 Y / North 719.22855290863662958

From map canvas Cancel OK

FIGURE 8.10
The Georeference tool with John Snow's cholera map imported.

tool that relate to importing rasters, applying transformations, and applying control points. As navigation menus often change between releases, refer to lab documentation or, ideally, the Help files related to this plugin to learn more about your particular version. But briefly, from left to right, the checkboard is the "Add Raster" button, the Play symbol applies the transformation and georeferences the image (so do not click this until you are ready!), and the gear is the Settings icon. The icons that look like small satellites (one of them is noted by the "1" arrow as shown in Figure 8.10) relate to the GCP, or Ground Control Points. As noted, the "GDAL" library powers many QGIS operations, including the Georeferencer. The GCP are used to tell the GDAL "gdal_translate" utility how to "translate" the pixel coordinates in the map to be georeferenced to match the system of your QGIS project. "Translate" refers to moving a set of points in a given direction. If you have ever played with image processing software such as Microsoft Paint, you know that when you move an image using your mouse, you are telling the computer to geometrically translate the object in a given distance of pixels according to the path of the mouse. However, since older maps often inconsistent, we need to apply more complex translations that are not necessarily applied in a uniform direction across the entire map. Since the Golden Square park in the cholera map is still present in our current maps, we can place a GCP right in the middle of Golden Park, in both map views, to tell the gdal_translate utility to "translate this image so that Golden Park will be rubbersheet (seamlessly overlap) the same Golden Park in modern maps, as much as possible. Figure 8.11 shows the results of georeferencing in the Golden Square area.

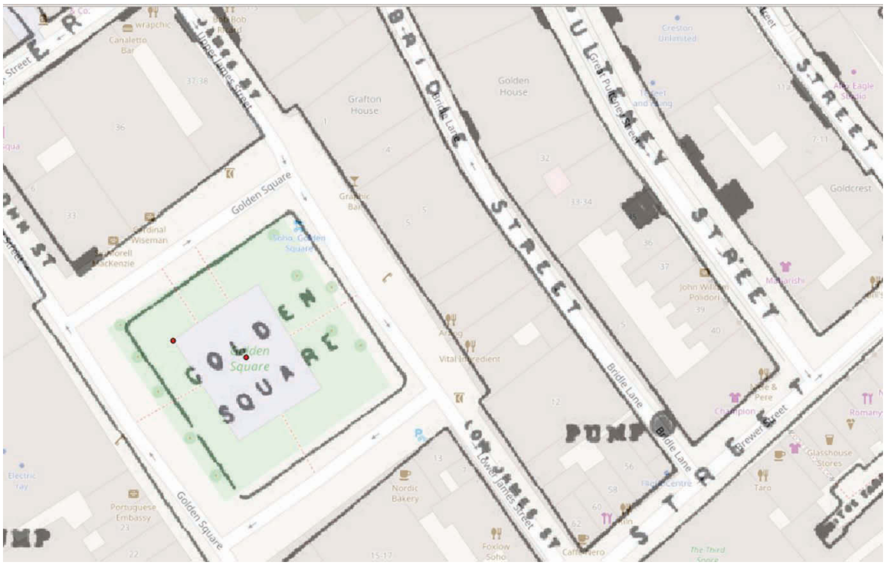


FIGURE 8.11
The overlaid John Snow map with slight transparency.

The small dots in the middle and at the top left represent the GCP used to make sure that Golden Square overlaps after the transformation is applied; the map is set to a slight transparency so that the OpenStreetMap Basemap layer with the contemporary layout of the neighborhood can be seen.

Figure 8.12 shows the Transformation dialog that allows us to apply a transformation algorithm to the translation. You will see several different possible methods, several of which are included in ArcGIS suite of products, but we will not discuss each method in length here. Rather, when you want to georeference a map, ask yourself "What is the coverage of this map? Is it on a regional level, or a global one? What is the coordinate system of my project? Is it spheroid?" Some transformation methods such as Thin Plate Spline are more appropriate for low-quality originals, which might apply to our cholera map, but in this case, the Polynomial Second Order is appropriate. It is a nonlinear algorithm (linear algorithms can only translate, not transform), and it is a fairly popular and standard approach that is a good "first pass" before experimenting with other transformations. The appropriate minimum number of points for Polynomial Second Order is about three or four control points; control points should be strategically placed where you think the greatest distortion between the two maps might occur. In the case of an urban map, you may want to choose parks (such as Golden Park), known street intersections shared between both maps, and landmarks. Once you have placed the control points and set the transformation, you are ready to complete the georeferencing. Figure 8.12 shows the resulting map, with transparency set to view the contemporary street. You may notice that although some features appear aligned, others

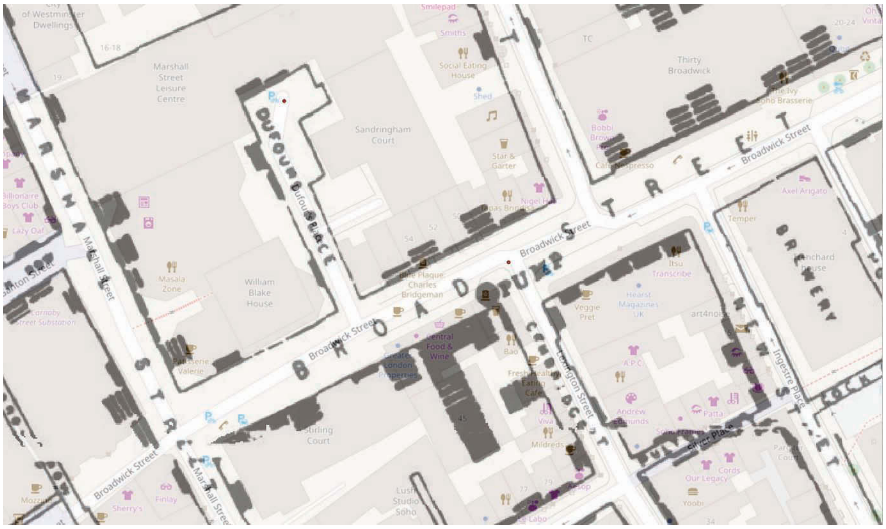


FIGURE 8.12
Transformation dialog within the Georeference tool.

seem "off." This could be due to a variety of reasons. Maybe the control points could have been placed better, or maybe there needs to be some real-life ground truthing and understanding of city records to know how best to translate the city plan across 150 years! In Chicago, for instance, a city built on swampy muck soils, early buildings were said to shrink as much as an entire foot after construction! Notice that we also have the (albeit blurry) marks that signify deaths in certain places. A next step might be to convert these marks into marker features in a shapefile or similar format that record the total deaths over each point, to "modernize" this flat map into a GIS file that can be queried, etc.

8.4 GeoDa and Open-Source GIS Databases

GeoDa is a "spatial data analysis" tool suited for exploratory data analysis. When GIS professionals conduct operations that involve spatial statistics such as regression analysis, they must first ask, "Do I have parametric distribution of my data, or is it skewed toward certain values? Are my observations spatially autocorrelated? A proper explanation of spatial statistics is outside the scope of this chapter, but there it is valuable tool for GIS professionals to approach and understand, especially if in making scientific claims about a dataset. A basic understanding of spatial statistics helps to determine whether points or polygons are "positively spatial autocorrelated" (i.e., whether similar values are clustered together in space). Imagine segregation of neighborhoods where richer neighborhoods cluster together or negative spatial autocorrelation where a wide range of values occur together in space (imagine a more diverse neighborhood where people with different incomes reside near each other). Exploring GIS data in this way is important both for understanding the nature of the data you are working with but is also a prerequisite for attempting certain forms of spatial statistics analysis as certain operations have assumptions about the distribution of level or lack of spatial autocorrelation, among other things.

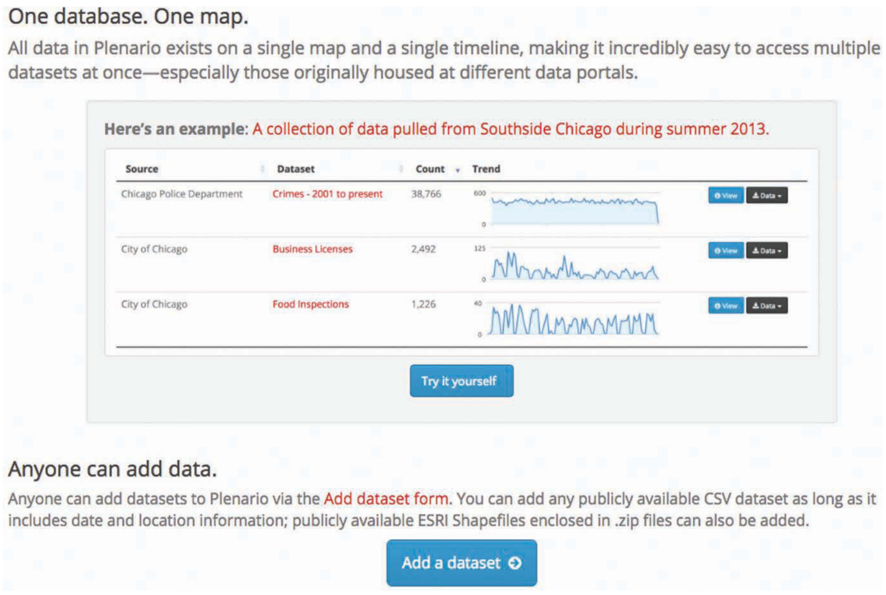
We will cover a few basic tools in GeoDa, but readers are encouraged to also approach the excellent tutorials on the GeoDa homepage (https://geodacenter.github.io/index.html) as well. But first, in the words of Monty Python, "now for something completely different." We will briefly discuss the GeoJSON data type as well as open-source GIS servers. This activity (located in the supplemental lab materials) uses a GeoJSON count of food inspections during a time period in Chicago, from the Plenario platform (based on PostGreSQL and PostGIS), an open-source database technology. GeoJSON is the standard for web GIS, and has many advantages over the shapefile format. Most obvious is the fact that in only uses one file rather than five or six separate files, making it an ideal export format for a web service.

Plenario, a "big data" repository for GIS datasets, is maintained by the Urban Center for Computation and Data at the Computation Institute of the University of Chicago and Argonne. Plenario does not host static links to datasets. Instead, all data in Plenario is hosted in a PostGreSQL database instance using PostGIS. When you import a CSV file, even though it is just rows and columns separated by commas, a GIS program such as ArcMap or QGIS converts it into a relational database table. If you have ever tried a GIS operation using what you think is valid GIS file and are told that you are missing a GUID field (for example), the GIS software you are using assumes that your dataset is organized according to the rules of relational tables and that every row can be uniquely identified via a primary key or unique ID. This and other assumptions of your data are essential for operations such as table joins and relates, union operations, and selections (Select by Location, Select by Attribute), GIS operations that use both SQL and relational algebra, key aspects of database design and management.

GeoJSON is an extension of JSON, or JavaScript Object Notation. Since JavaScript is so fundamental to web applications today, it is used for sending, receiving, and storing data. A JSON file wraps data in "objects" in name/value pairs. Since these name/value pairs are in human readable text, the file is both lightweight and human-readable. However, one disadvantage of the JSON format is that there is no standardized or formal way to determine what a name/value pair actually is outside of the rules of JavaScript and the internal rules of an application working with the JSON data. For example, if you running an online banking service for your customers, you might decide that the name/value pair that represents customers should be named "customer." But another bank might choose to represent that name/value pair using "customerName." In the world of GIS, this lack of formalization in JSON would cause issues because GIS data often needs to be shared, and applications need to know how data should be interpreted and processed for basic GIS functionality to work. This is the rationale behind GeoJSON. At its core it is JSON, but it also is extended with formal rules such that certain keywords, such as polygon, point, and coordinates, are more than simply words with semantic meanings. Instead, they are standardized so that GIS applications that work with GeoJSON data know to process and project the name/value pairs in a GeoJSON file as points or polygon, in an efficient and lightweight manner. But note two key disadvantages of GeoJSON here. As noted in the specifications all GeoJSON should use the WGS84 (World Geodetic System 1984) reference system (so other transformations must be done through GIS applications and in other formats), and GeoJSON is inadequate for some advanced representation of features, such as topology or expanding beyond the elements of point, line, or polygon geometry objects. For many web GIS applications, these constraints do not prevent GeoJSON from being efficient, including Plenario.

As noted, Plenario (accessible via web address "plenario.io") is built using the open-source database PostGreSQL. The site hosts many different datasets from various cities, with the goal of making public data more accessible, as shown in Figure 8.13. Since they are all represented and stored on the platform in a database structure, these datasets can be queried to get the exact time frame, extent, and desired fields before being exported into the file format of choice (GeoJSON, shapefile, or KML). But note that event datasets can only be exported in GeoJSON or CSV format because it is much more efficient for simple datasets that would be represented as vector points to be exported as GeoJSON or CSV rather than an ESRI shapefile. This is because compared to the ESRI shapefile format, GeoJSON files are easier to create, run processing commands on to add or select records, and are smaller file sizes.

In Data Explorer we can filter based on an event dataset of interest; in this case, we will search "food inspection" and choose the Chicago dataset "City of Chicago: Food Inspections." The Data Explorer can be accessed at plenar.io/explore/discover, navigate to that web page to follow along. Figure 8.14 shows the event dataset and the ability to refine or filter it before exporting (we could, alternatively, download the "raw" complete dataset). It is often best to preview how the data is structured before applying such queries. In this case, we chose to select all recorded failed food inspections during the period between December 8, 2017 and March 8, 2018 by setting the start and end dates, and then setting "Select Field" to Results, the relational operator to "=", and the value for the selection to "Fail." This is a simplified user-friendly version of what may otherwise be constructed as "SELECT from "Food_Inspections" WHERE "results" = "Fail." We will only know this selection will work if we preview the data and see that the possible values


One database. One map.

All data in Plenario exists on a single map and a single timeline, making it incredibly easy to access multiple datasets at once—especially those originally housed at different data portals.

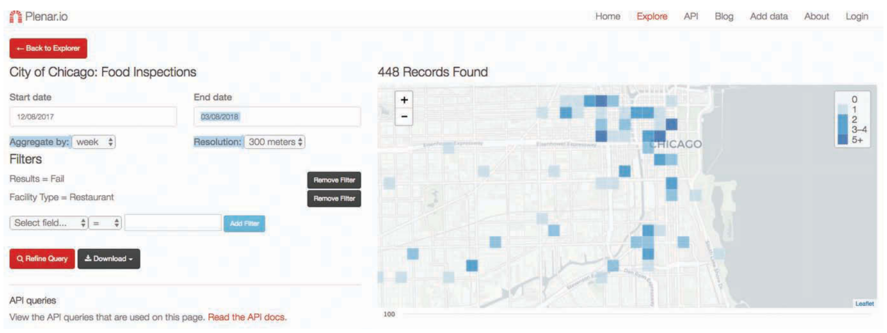
Here's an example: A collection of data pulled from Southside Chicago during summer 2013.

Source	Dataset	Count	Trend
Chicago Police Department	Crimes - 2001 to present	38,766	
City of Chicago	Business Licenses	2,492	
City of Chicago	Food Inspections	1,226	

Anyone can add data.

Anyone can add datasets to Plenario via the Add dataset form. You can add any publicly available CSV dataset as long as it includes date and location information; publicly available ESRI Shapefiles enclosed in .zip files can also be added.

FIGURE 8.13
The Plenario homepage.



City of Chicago: Food Inspections

Start date 12/08/2017

End date 03/08/2018

Aggregate by: week

Resolution: 300 meters

Filters

Results = Fail

Facility Type = Restaurant

Select field... =

448 Records Found

API queries

View the API queries that are used on this page. Read the API docs.

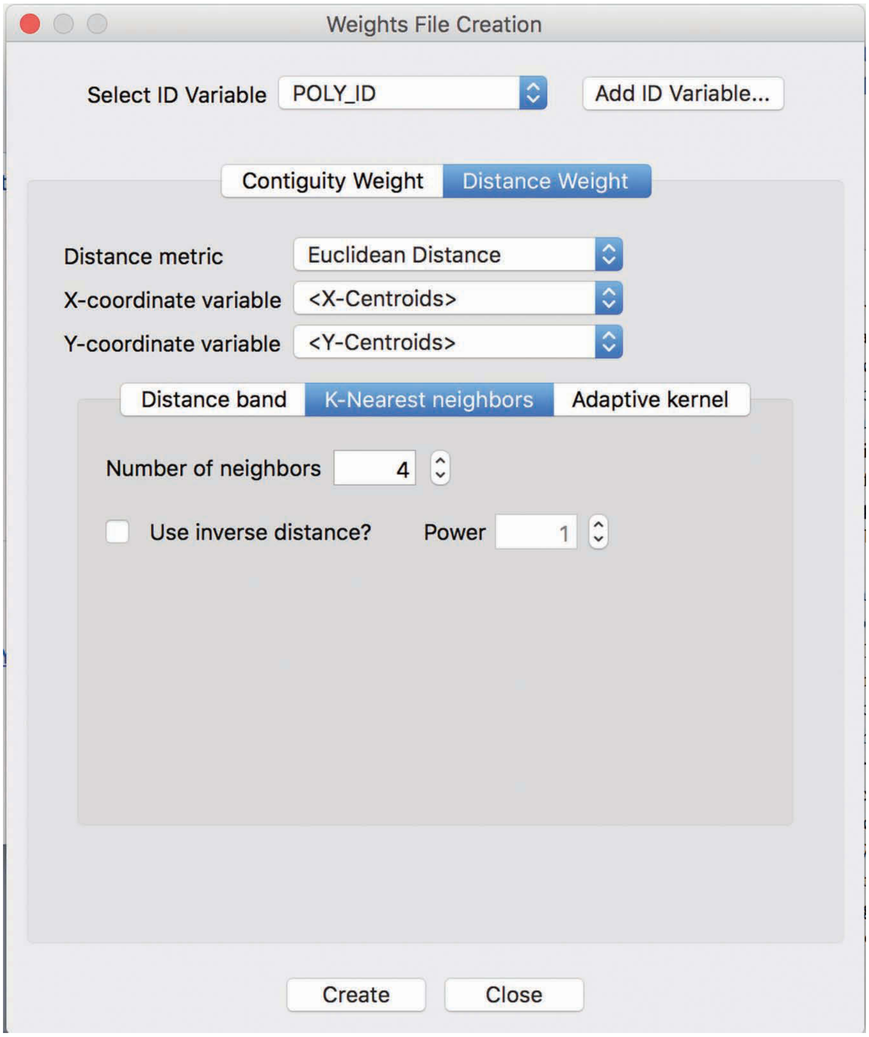
FIGURE 8.14
Plenario's Data Explorer, with the Food Inspections of Chicago dataset selected.

are "Pass," "Fail," and some other possible values such as "Not Ready" and "Out of Business." But note that we are running a query based on a given parameter, in this case, "Return a downloadable view of this dataset where all values of this column are typed as "Fail." What happens if an entry in the Results column is mistyped, such as a value being "Fail"? It will not be returned in the results, because the SQL query is not "smart" enough to catch these types of human mistakes. Once we click "Add Filter," we can add another filter, either applied to the same field or another; in this case, we add the Facility Type=Restaurant filter to limit the results to restaurant types. When we click "Refine Query," a map is generated on the right with aggregated counts of events meeting our filters in small gridded polygons. When we click the "Download" button and select "Complete Raw Data," we will get a GeoJSON (or CSV) file that includes only those food inspections that failed. If we export as a map grid, we will get a map grid that represents polygons aggregated with the counts of the failed food inspections.

But what if we wanted to determine if there is a pattern to food inspections in the city of Chicago? Do certain parts of the city receive fewer food inspections than other parts of the city? This could be a precursor to an investigatory analysis of whether certain neighborhoods face more food inspections after controlling for population density and number of restaurants. While investigating this issue, we may want to investigate whether food inspections are spatially autocorrelated. GeoDa is especially suited for this type of analysis, ESDA, or Exploratory Spatial Data Analysis. You can download GeoDa from https://geodacenter.github.io/index.html. In addition to this simplified demonstration of GeoDa, a robust and continually updated tutorial is available on the website.

After installing and opening a session of GeoDa, click on the folder button on the far left and select your dataset. Plenario will export a dataset in GeoJSON format, but the resulting file will still be labeled as a .json file and so on. Don't worry—they will still adhere to the GeoJSON standards and be interpreted as GeoJSON file once imported. You will see a mapped grid of the food inspections, with each box having the value of counts (how many food inspections took place across the specified period of time within that gridded area). Before we investigate spatial autocorrelation with this dataset, we have to establish "spatial weights." Spatial weights establish the logic that determines whether observations are neighbors. To access the spatial weight settings, click Tools in the GeoDa navigation bar and then Weights Manager (or the shortcut represented by a large W), and then click "Create." We need to establish a spatial weight before investigating spatial autocorrelation to determine the basis for whether observations are "close" or "distant" from each other. You can choose between contiguity weight or distance weight, but in this dataset, several grids are very distant from each other and would not be contiguous (touching), so we will investigate using the distance weight (click the Distance Weight) tab. Since we do not have IDs for each record, we must first click the "Add ID Variable" and select the default POLY_ID. Our options are Distance Band, K-Nearest Neighbors, and Adaptive Kernel. Distance Band uses a set distance (which corresponds to the units used by the projection of the dataset) to determine which observations might be clustered based on being inside or outside of that distance, with the option to weight that distance by an inverse power distance. Adaptive Kernel is appropriate for nonparametric datasets when investigating spatial covariance (i.e., how much two variables change alongside each other spatially. For this activity, as shown in Figure 8.15, we only demonstrate and discuss the K-Nearest neighbor option, which is a machine learning algorithm that calculates the distance between different observations. We will look at the distances in the aggregate to determine the proper threshold for which neighbors would be considered close or far and then assign more significance to closer neighbors than further neighbors as a weight. Click "create" and save the weight file to your desktop.

We now have weights we can use to run a Univariate Local Moran's I analysis, which investigates spatial autocorrelation. Click the Space tab and then click "Univariate Local Moran's I", being sure to set "count" as the "First Variable" and the weights to those you created in the last step. You will get a window asking what results you want; check all of them as shown in Figure 8.16. You will see a chart of the Moran's I results across different counts, a LISA (Local Indicators of Spatial Autocorrelation) significant map that shows where the Moran's I values are significant, and a LISA Cluster Map that shows the clustering of different values, as shown in Figures 8.17 and 8.18. Figure 8.18 also shows the basemap underlaid, illustrating that the clusters of high counts of food inspections surrounded by other high counts of food inspections appear to be concentrated in the heart of downtown of Chicago as well as the northside. Downtown Chicago is a hub for dining and tourist restaurants, so a cluster there makes sense. But it's possible this exploratory data analysis may lead us in other directions, such as generating a map of known restaurants in Chicago covering this period to see if the food inspections match up with a map of restaurants. By overlaying


Weights File Creation

Select ID Variable: POLY_ID

Add ID Variable...

Contiguity Weight | Distance Weight

Distance metric: Euclidean Distance

X-coordinate variable: <X-Centroids>

Y-coordinate variable: <Y-Centroids>

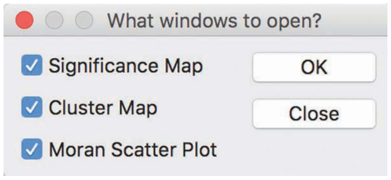
Distance band | K-Nearest neighbors | Adaptive kernel

Number of neighbors: 4

☐ Use inverse distance? Power: 1

Create | Close

FIGURE 8.15
Weights File Creation menu (MacOS).


What windows to open?

✓ Significance Map

✓ Cluster Map

✓ Moran Scatter Plot

OK | Close

FIGURE 8.16
GeoDa dialog box asking which results to save.

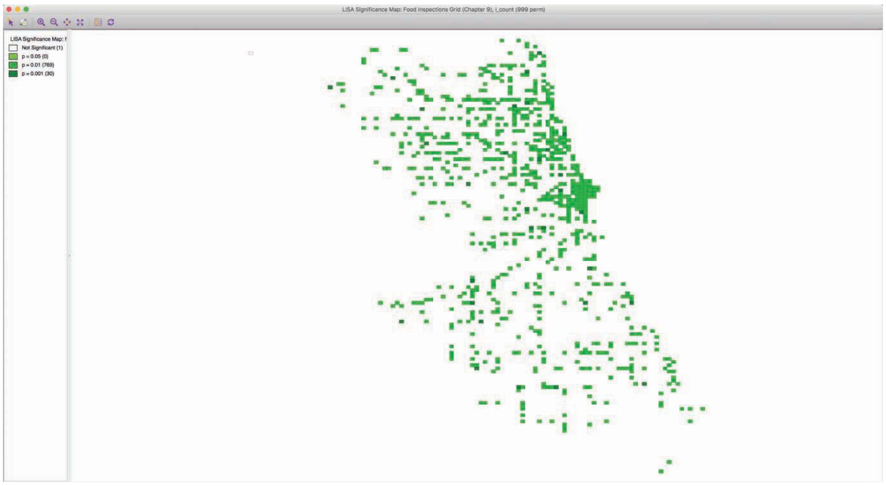


FIGURE 8.17
Moran’s I significance map.

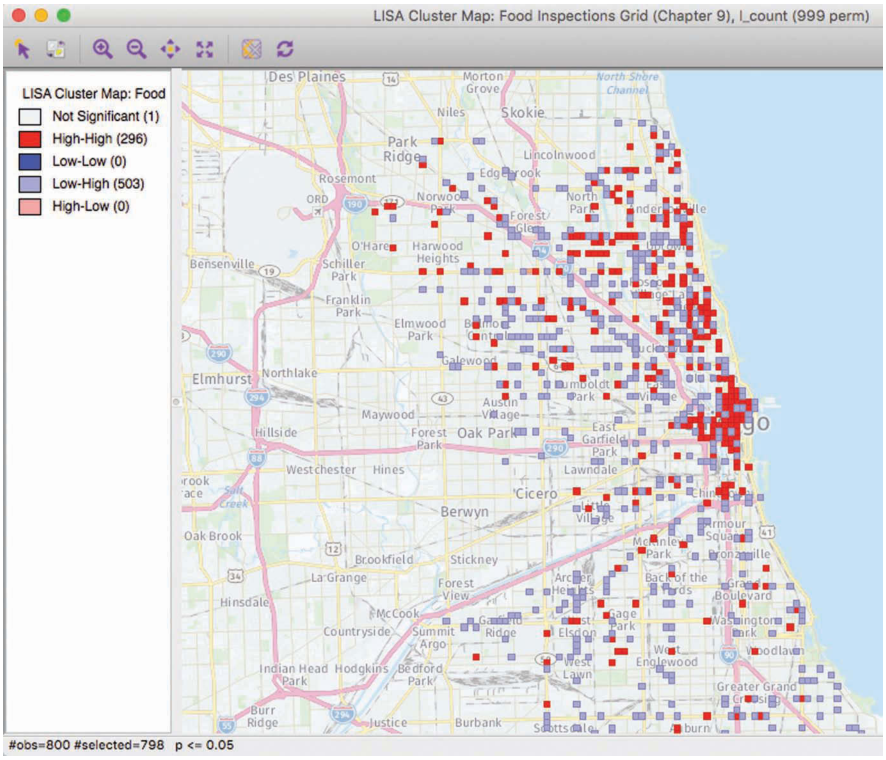

LISA Cluster Map: Food Inspections Grid (Chapter 9), I_count (999 perm)
LISA Cluster Map: Food
Not Significant (1)
High-High (296)
Low-Low (0)
Low-High (503)
High-Low (0)
#obs=800 #selected=798 p <= 0.05

FIGURE 8.18
LISA clustering map.

these two datasets, we could do a Bivariate Moran's I and determine if food inspections are clustered around areas with more restaurants, or randomized. One of the advantages of GeoDa is that a wide array of spatial statistics tools needed to approach such questions are available and easy to use, such as spatially weighted regression and time series analyses. The "GeoDa Workbook" compiled by Luc Anselin (the lead developer of GeoDa and a noted figure in spatial econometrics) is an excellent resource for learning spatial statistics and GeoDa at the same time, and is highly recommended.

8.5 The Future of Open-Source GIS

In the early days of GIS, tools and applications were built for highly specialized tasks and audiences, and for a time, the dominant tools were noncommercial. Now GIS is in demand and usage by nonspecialists, and while commercial tools are available for specialists and nonspecialists alike, open-source GIS are considered as viable alternatives. The time and energy spent by volunteers around the world on open-source GIS projects to keep these programs competitive with commercial GIS offerings is valuable.

References

Anselin, L. and S. J. Rey. *Modern Spatial Econometrics in Practice: A Guide to GeoDa, GeoDaSpace and PySAL*. GeoDa Press LLC: Chicago, IL, 2014.

Anselin, Luc. GeoDa Workbook. Documentation | GeoDa on Github. Accessed April 30, 2018. https://geodacenter.github.io/documentation.html.

Graser, A. *Learning QGIS: The Latest Guide to Using QGIS 2.14 to Create Maps and Perform Geoprocessing Tasks with Ease*. Packt Publishing: Birmingham, 2016.

McKee, L. OGC History. OGC History (detailed) OGC. Accessed April 30, 2018. http://www.opengeospatial.org/ogc/historylong.

Menke, K., R. Smith, L. Pirelli, and J. Van Hoesen. *Mastering QGIS: Go beyond the Basics and Unleash the Full Power of QGIS with Practical Step-by-step Examples*. Packt Publishing: Birmingham, UK, 2016.

Momjian, B. *PostgreSQL: Introduction and Concepts*. Addison-Wesley, Boston, MA, 2001.

Neteler, M., and H. Mitasova. *Open Source GIS: A GRASS GIS Approach*. Springer: New York, 2010.

Obe, R. O. and L. S. Hsu. *PostGIS in Action*. Manning, Shelter Island, NY, 2015.

QGIS Documentation. Documentation. April 30, 2018. Accessed April 30, 2018. https://www.qgis.org/en/docs/index.html.

Wegmann, M., B. Leutner, and S. Dech. *Remote Sensing and GIS for Ecologists: Using Open Source Software*. Pelagic Publishing, Exeter, 2017.