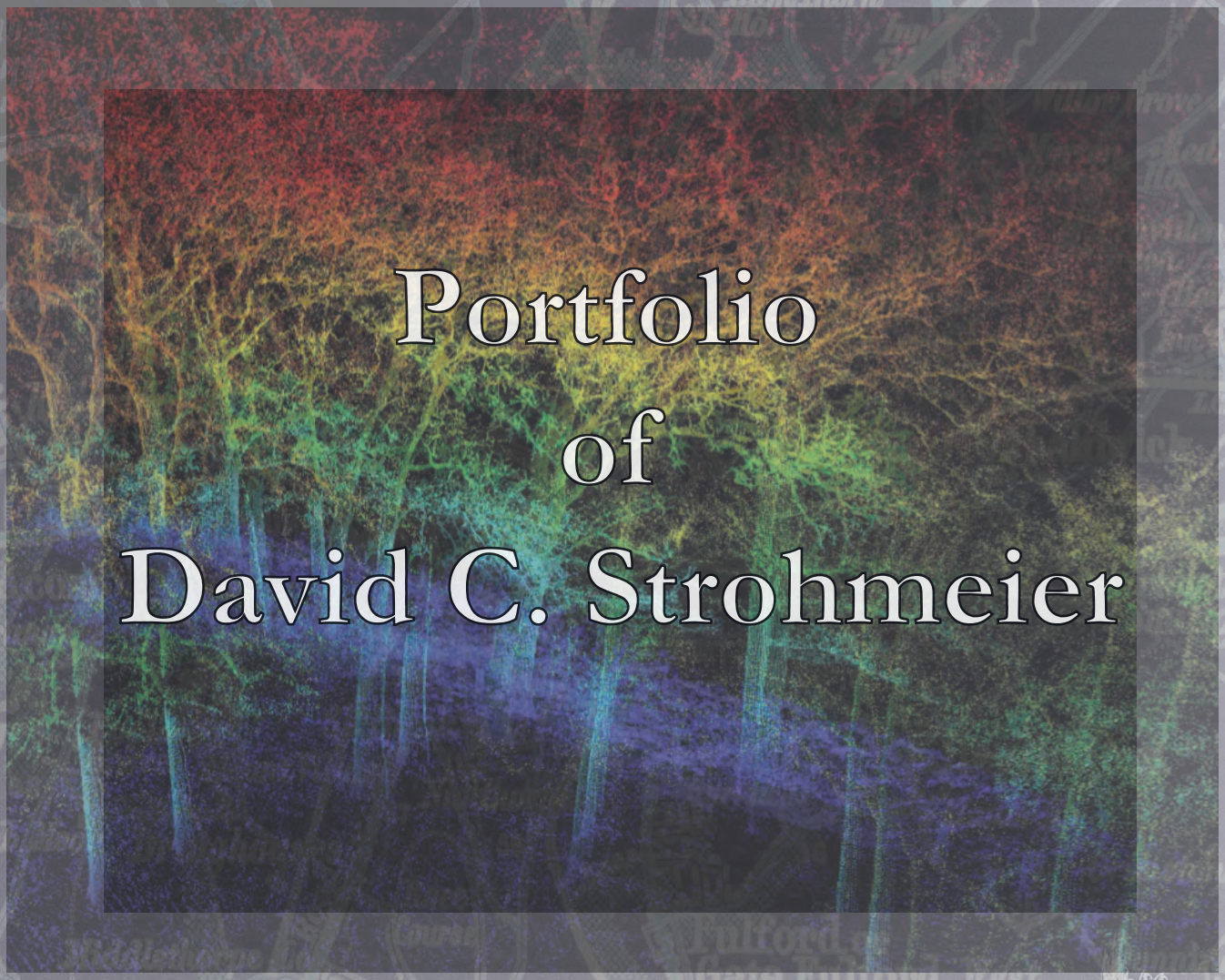


Portfolio of David C. Strohmeier

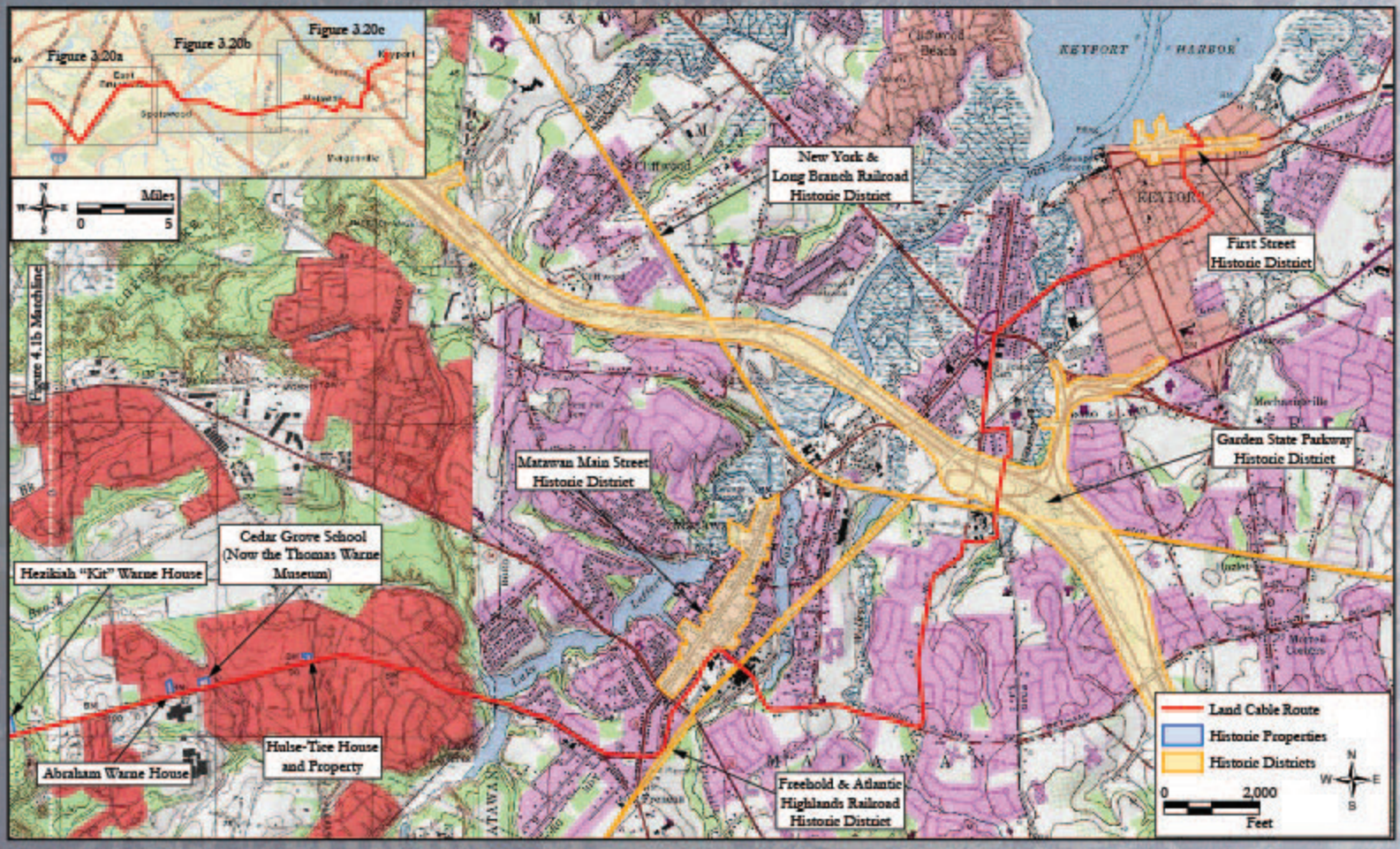
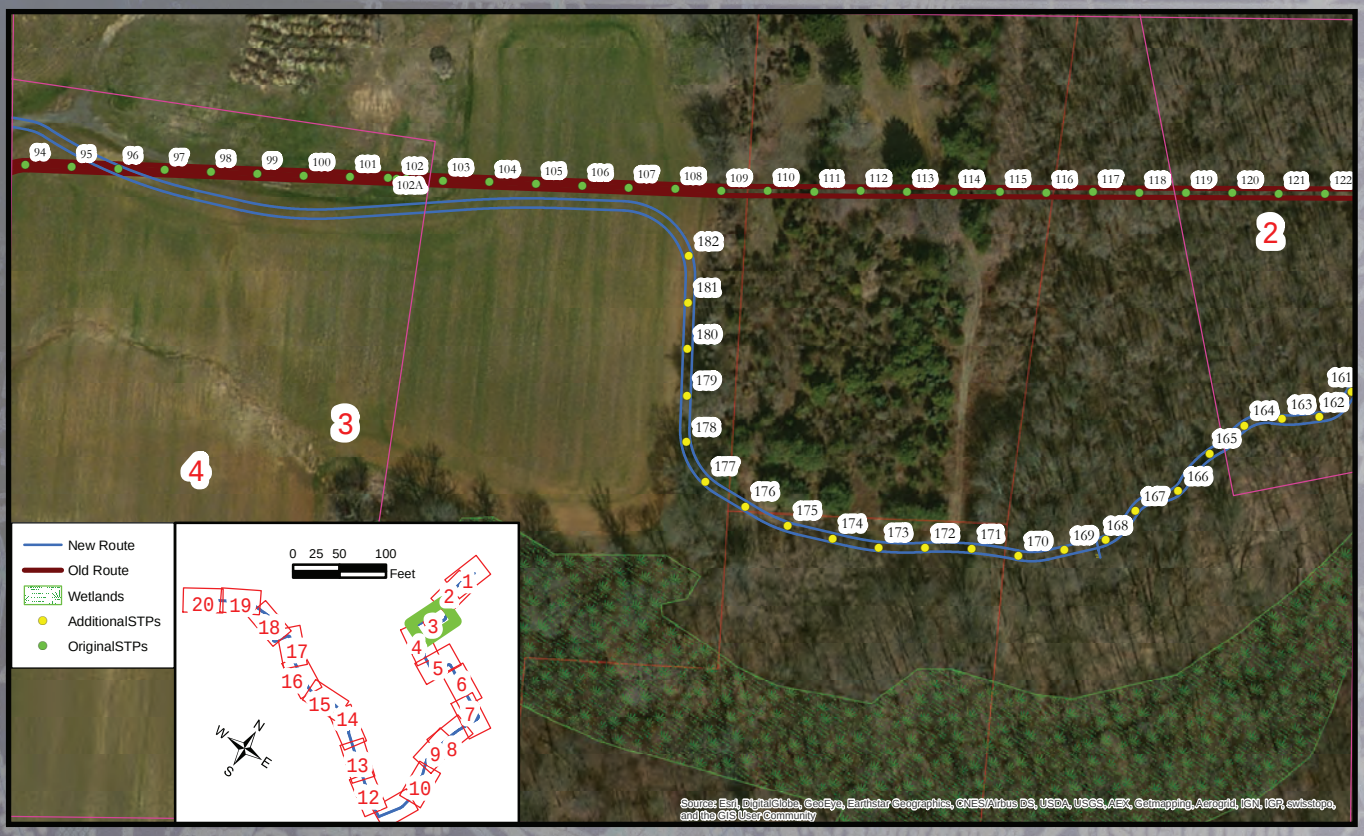


Figure 3.20a
Figure 3.20b
Figure 3.20c
Miles
0
5
Figure 4.1b Matchline
New York & Long Branch Railroad Historic District
First Street Historic District
Garden State Parkway Historic District
Matawan Main Street Historic District
Cedar Grove School (Now the Thomas Warne Museum)
Hezikiah "Kit" Warne House
Abraham Warne House
Hulse-Tice House and Property
Freehold & Atlantic Highlands Railroad Historic District
Land Cable Route
Historic Properties
Historic Districts
0
2,000
Feet


Legend
Listed Historic Districts
Eligible Historic Districts
NHL Property
Listed Property
Eligible Property
500 Foot Buffer
Tower Not Visible
Tower Visible
0
1,000
Feet

Map Books

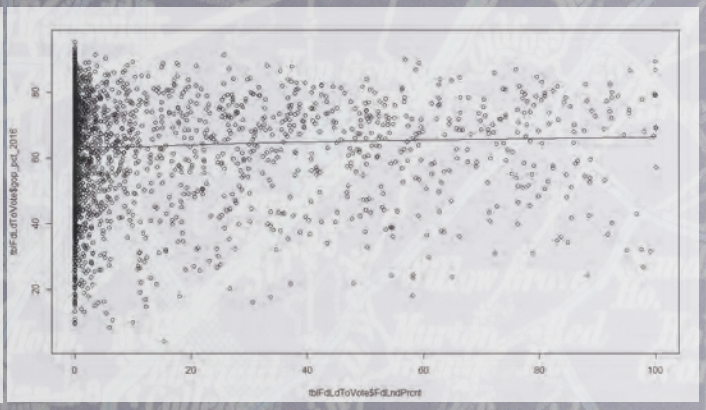


Many project areas are not contained to a single contiguous parcel of land but rather are linear in nature spanning several thousand feet or perhaps even miles. In cases like these map books are required using ESRI's Data Driven Pages to produce overlapping indexed maps for better scalability and overall easier use.

Charts & Graphs

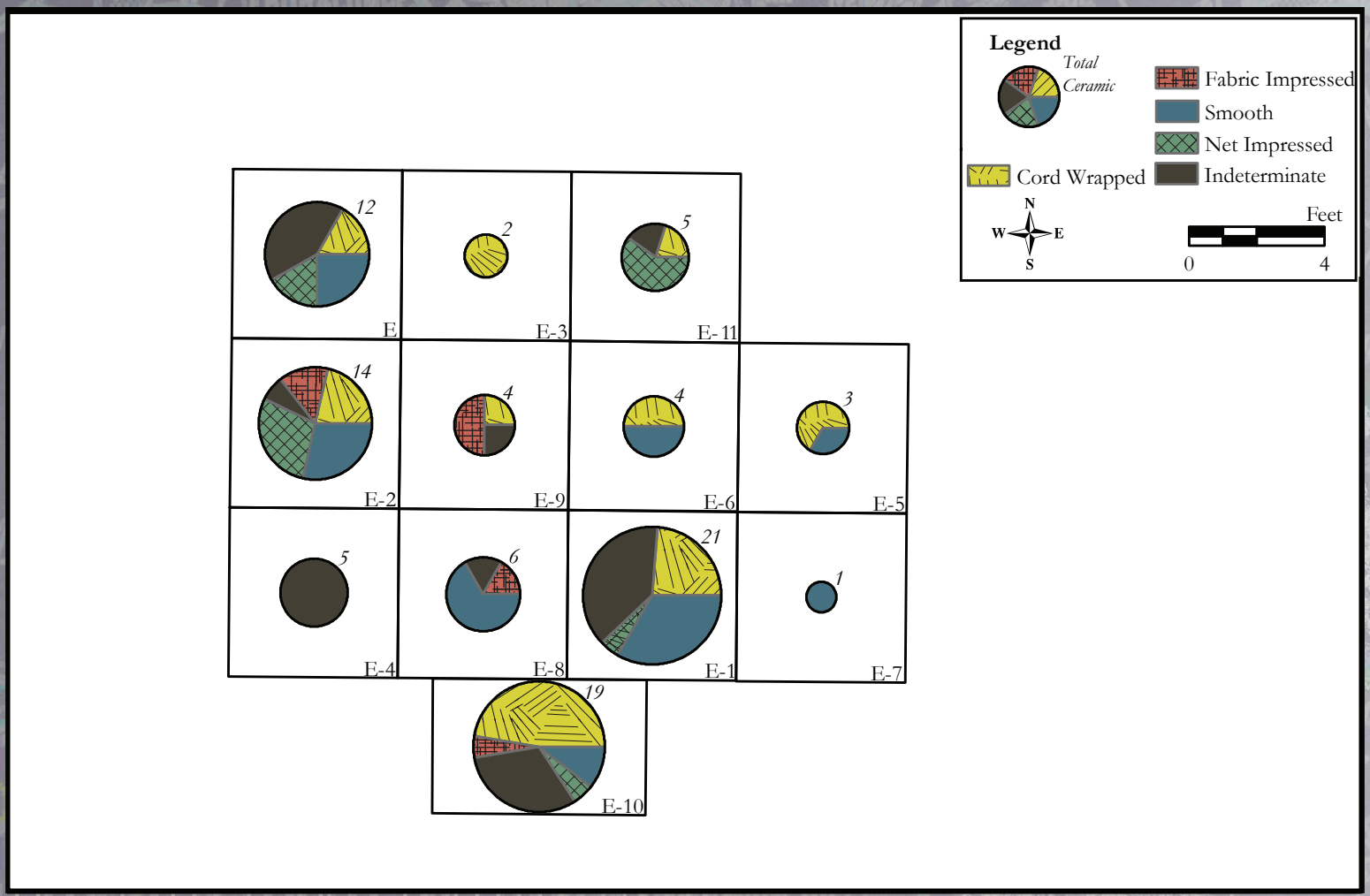
The use of graphical programming languages such as R can be a quick and easy way to produce graphic figures of your data.

```
LocCntyFedPer = "C:/Users/davestrohmeier/Desktop/Geoviz/CountyFedLandPer.csv"
tblCntyFedPer = read_csv(LocCntyFedPer)
tblCntyFedPer= rename(tblCntyFedPer, fips_code=GEOID)
tblFdLdToVote= left_join(tblCntyFedPer, tblElection, by= "fips_code")
plot(tblFdLdToVote$FdLndPrcnt, tblFdLdToVote$gop_pct_2016)
tempfed= filter(tblFdLdToVote, FdLndPrcnt > 80)
plot(tempfed$FdLndPrcnt, tempfed$dgop_pct_2016)
caliBound= left_join(tblFdLdToVote, tblHealthOriginal, by= "fips_code")
nocal= select(caliBound, State, fips_code)
no_WestCoast= filter(caliBound, State != "CA", State != "OR", State !="WA")
plot(no_WestCoast$FdLndPrcnt, no_WestCoast$gop_pct_2016)
california= filter(caliBound,State=="CA")
plot(california$FdLndPrcnt, california$gop_pct_2016)
washington= filter(caliBound, State=="WA")
plot(washington$FdLndPrcnt, washington$gop_pct_2016)
tenpct= filter(tblFdLdToVote, FdLndPrcnt < 10)
plot(tenpct$FdLndPrcnt, tenpct$gop_pct_2016)
pointfive=filter(tblFdLdToVote, FdLndPrcnt < .5)
plot(pointfive$FdLndPrcnt, pointfive$gop_pct_2016)
zeros=filter(tblFdLdToVote, FdLndPrcnt < .001)
```



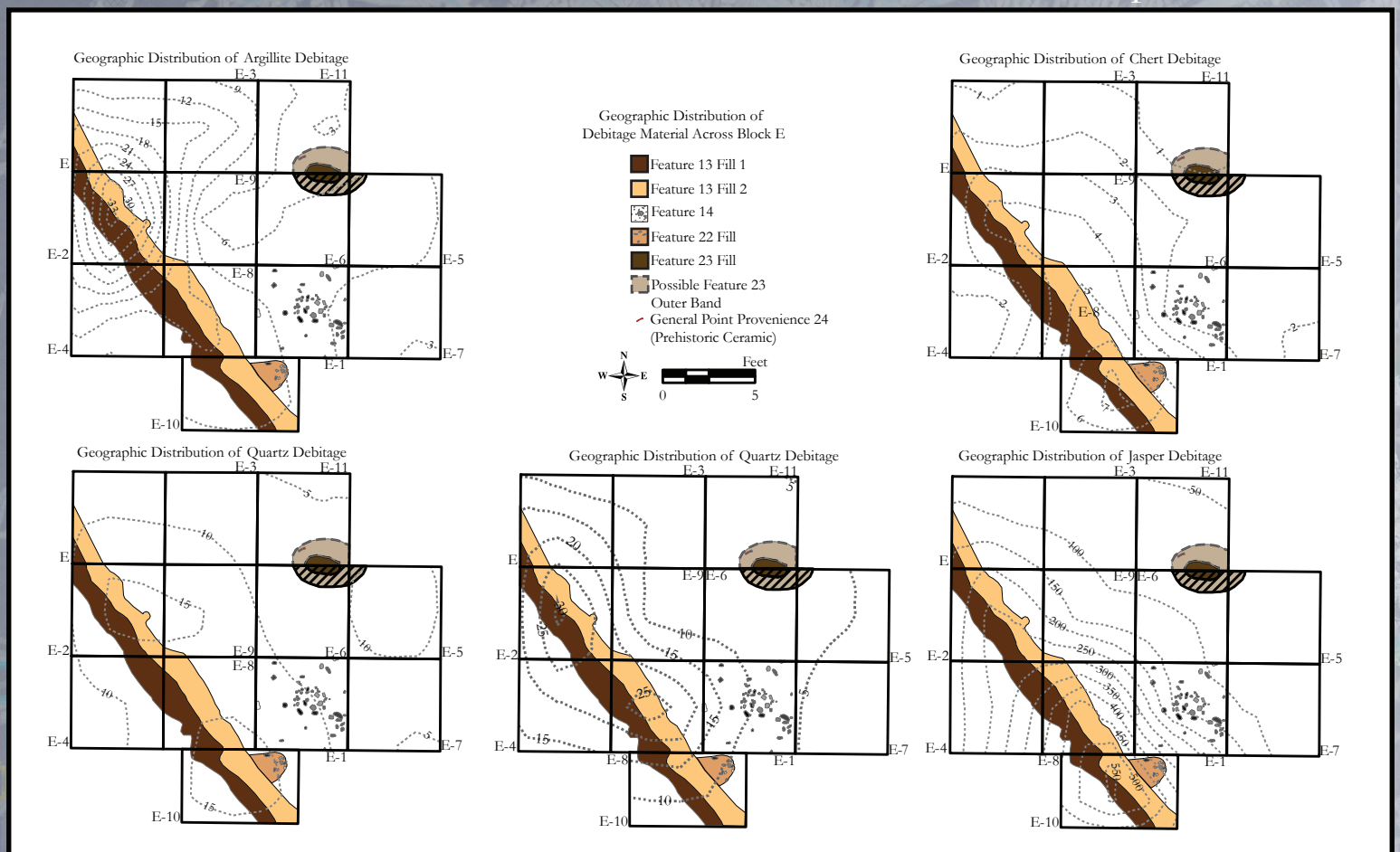
Information regarding counts of specific attribute types obtained from artifact catalogs can be joined with correlated spatial data to create proportional symbol pie charts such as the one seen below as a way to represent the distribution of multiple multiple attributes in a single figure.

Distribution of Exterior Treatment of Prehehistoric Ceramic
Across Excavation Unit Block E

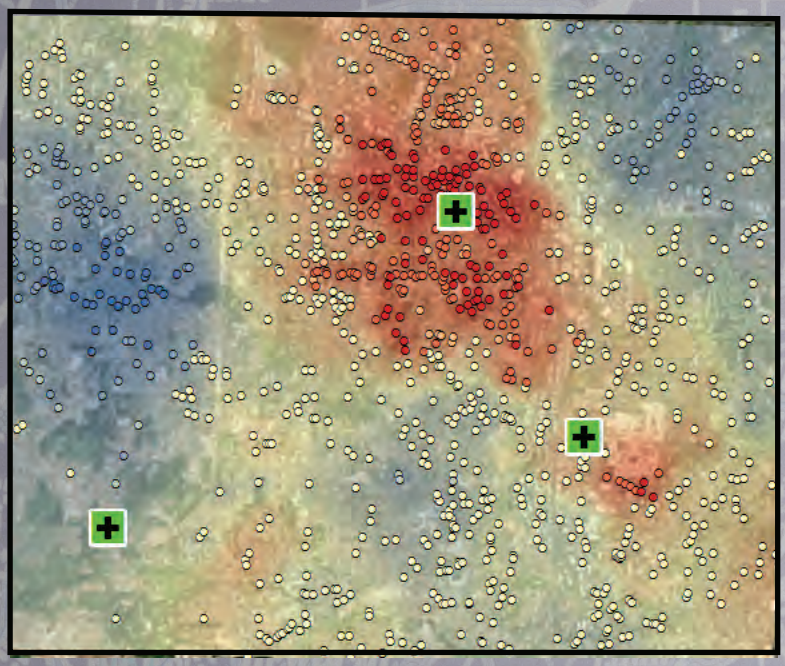


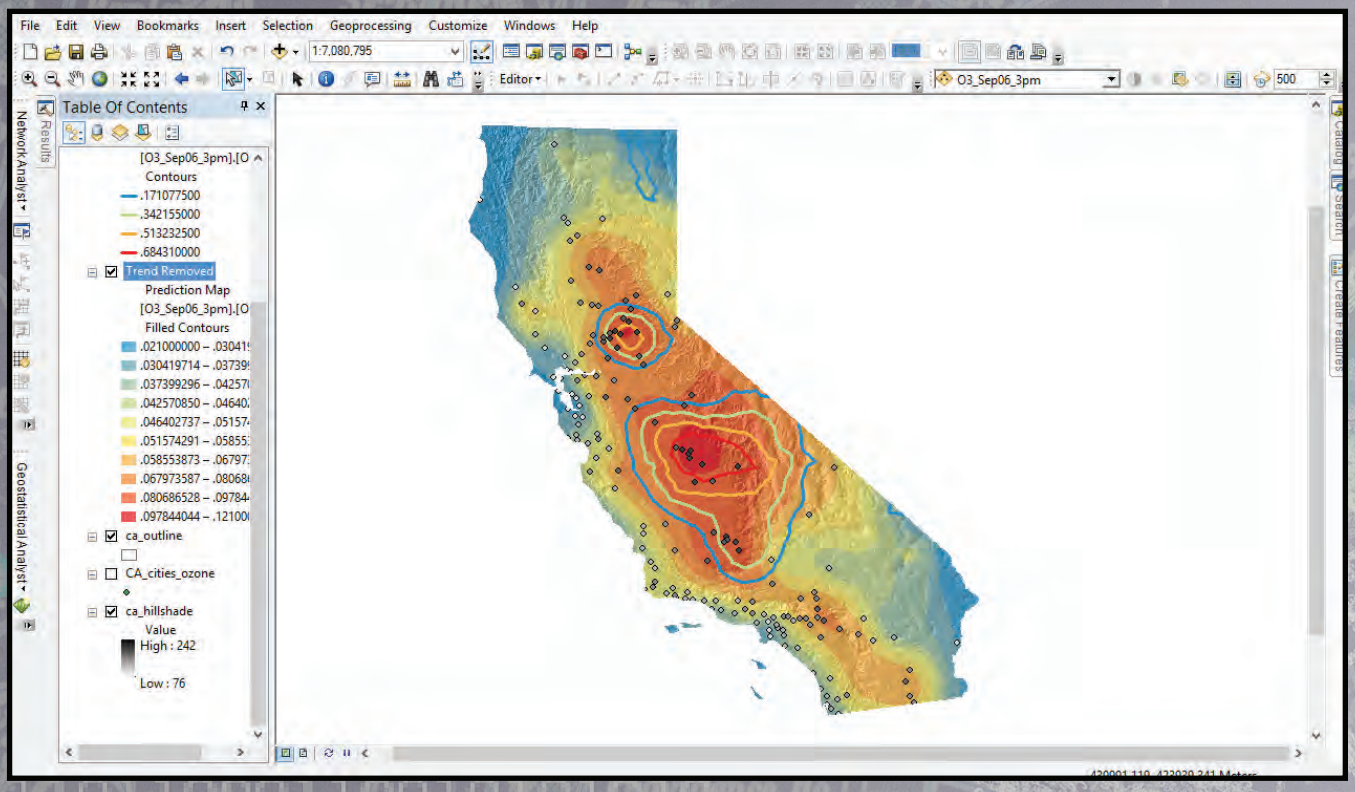
Interpolation..

Interpolations can be used as an easy way to convey meaning to spatial data. By designing large attribute tables form artifact catalogs, archaeologists can reveal hot spots and create interpolated isopleths based on any attribute and and attacing it to its relative collection point.



Interpolation

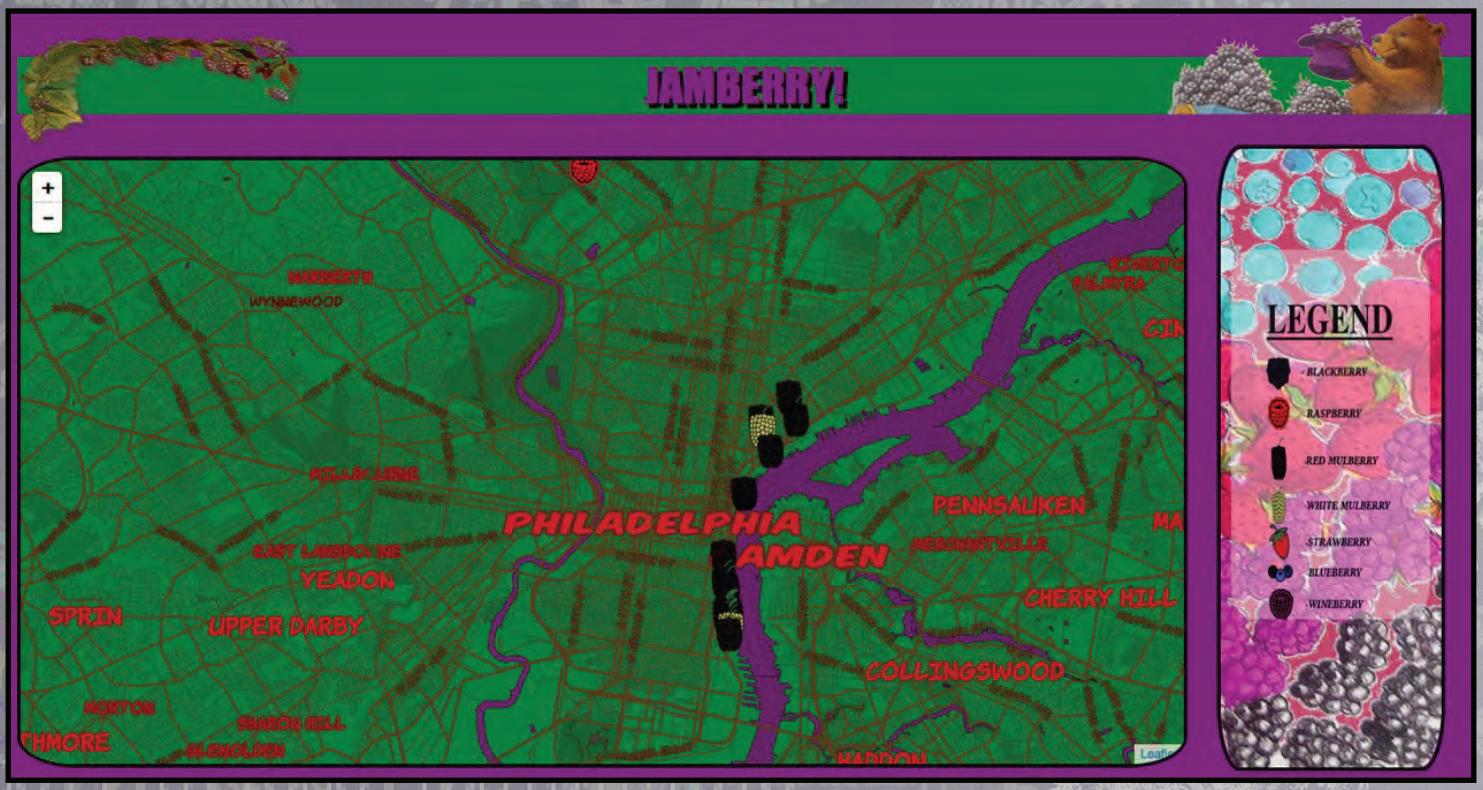


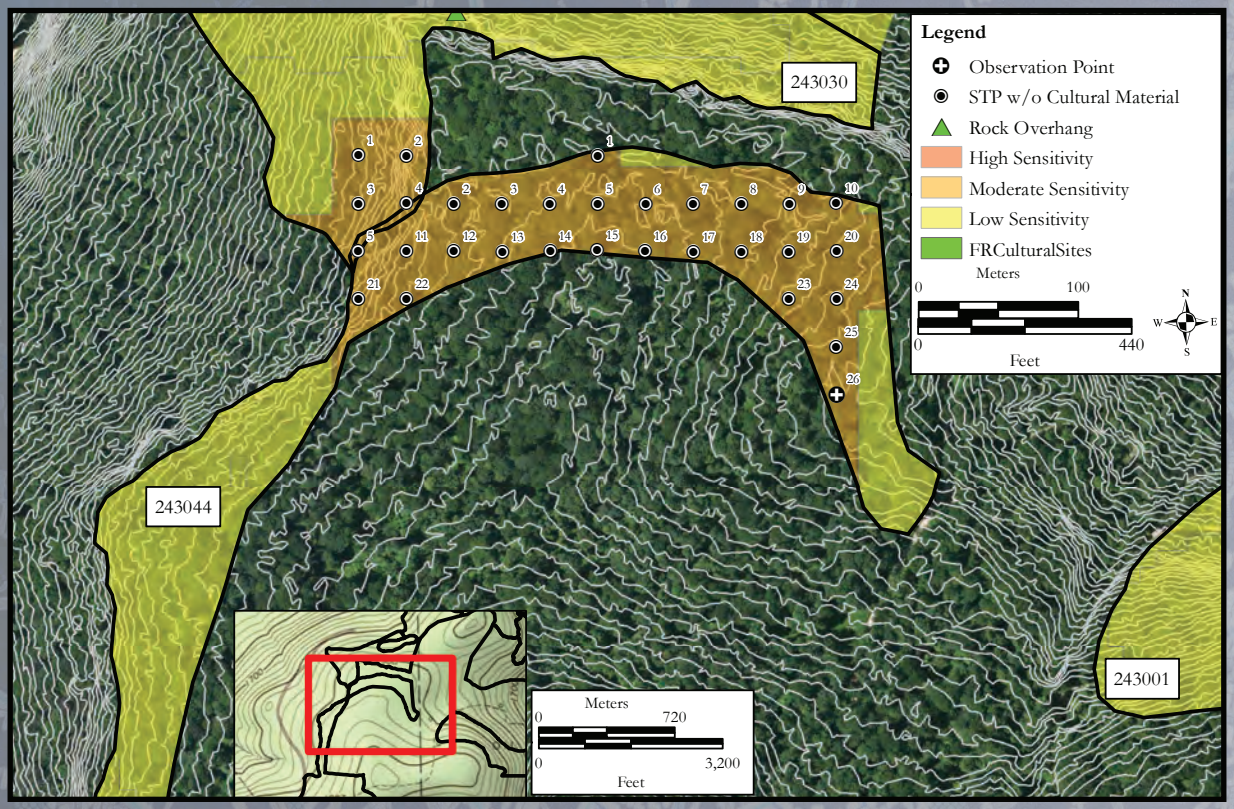


HTML, CSS & JavaScript

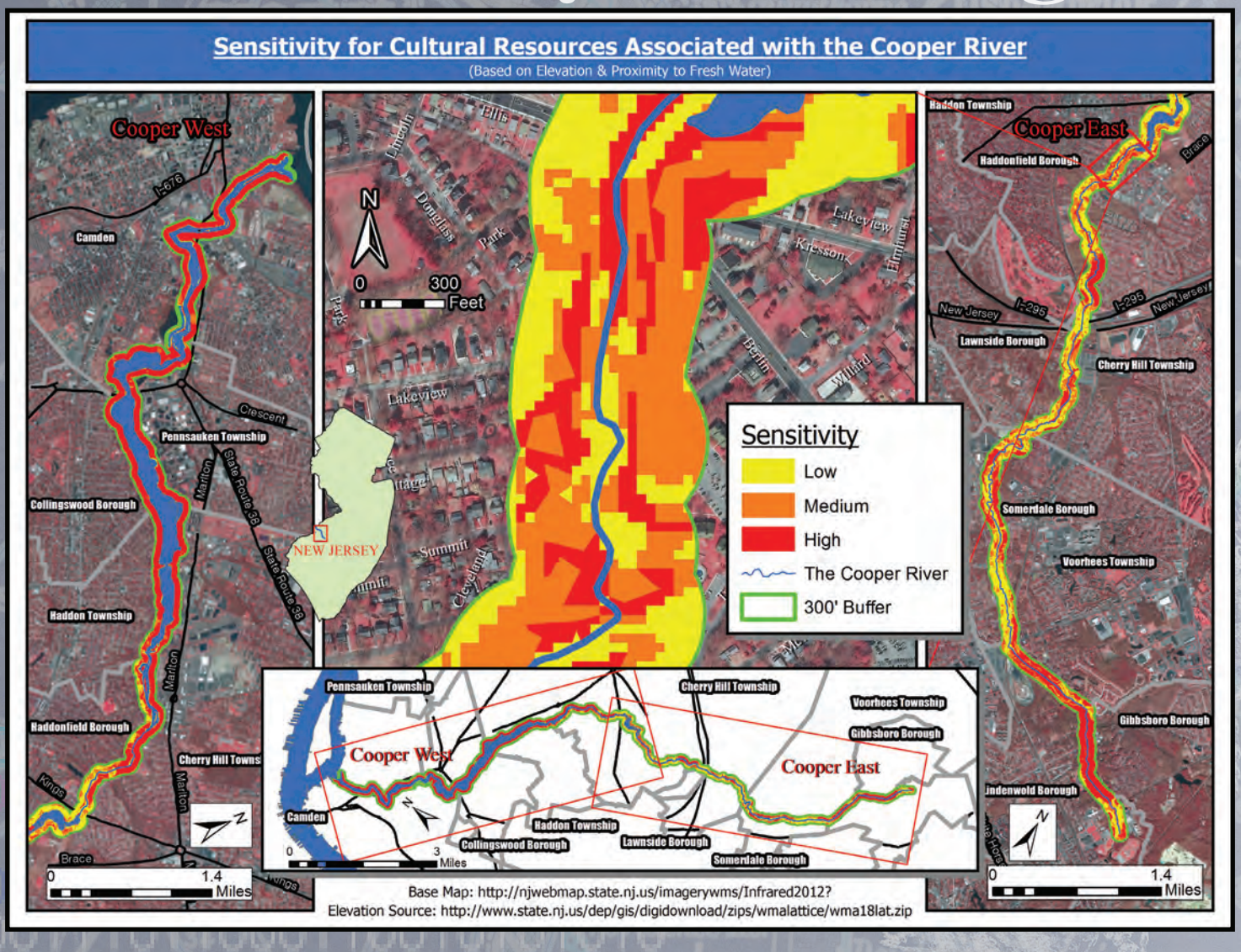
Using web GIS tools like Mapbox and Cartodb and APIs such as leaflet I have learned to develop HTML code using CSS and JavaScript to create interactive webmaps.

```
<div id="map"></div>

<script>

// initialize the map
var map = L.map('map').setView([39.952584, -75.165222], 13);

// load a tile layer
L.tileLayer('https://{s}.tiles.mapbox.com/v4/davestrohmeier.a4fe2e7a/{z}/{x}/{y}.png?access_token=pk.
eyJ1IjoiZGF2ZXN0cm9obWVpZXIiLCJhIjoiZGYxOWI2NjlmNWU5YTM2NTFlM2UxZjMxNjQ2YTNhZGYifQ.h6K9YfvzwcFg2jD92xjY1Q').addTo(map);


    // load GeoJSON from an external file

$.getJSON("Redmulb.js",function(data){
  var berryIcon = L.icon({
  iconUrl: 'redmulberry.png',
  iconSize: [40,40]
});

// add GeoJSON layer to the map once the file is loaded
  L.geoJson(data,{
  pointToLayer: function(feature,latlng){
  var marker = L.marker(latlng,{icon: berryIcon});
    marker.bindPopup("<strong>" + feature.properties.Berry_type + "</strong><br/>" + feature.properties.Ripe_ + '<br/>' + feature.
      properties.Comments + '<br/>' + feature.properties.Refenerenc + '<br/>' + feature.properties.Locale + '<br/>' + feature.
      properties.Description + '<br/>' + feature.properties.Accessible
);
      return marker;

  }
  }).addTo(map);
  });

  $.getJSON("Whitemulb.js",function(data){
  var berryIcon = L.icon({
  iconUrl: 'whitemulberry.png',
  iconSize: [40,40]
});
  // add GeoJSON layer to the map once the file is loaded
  L.geoJson(data,{
  pointToLayer: function(feature,latlng){
  var marker = L.marker(latlng,{icon: berryIcon});
    marker.bindPopup("<strong>" + feature.properties.Berry_type + "</strong><br/>" + feature.properties.Ripe_ + '<br/>' + feature.
      properties.Comments + '<br/>' + feature.properties.Refenerenc + '<br/>' + feature.properties.Locale + '<br/>' + feature.
      properties.Description + '<br/>' + feature.properties.Accessible
```

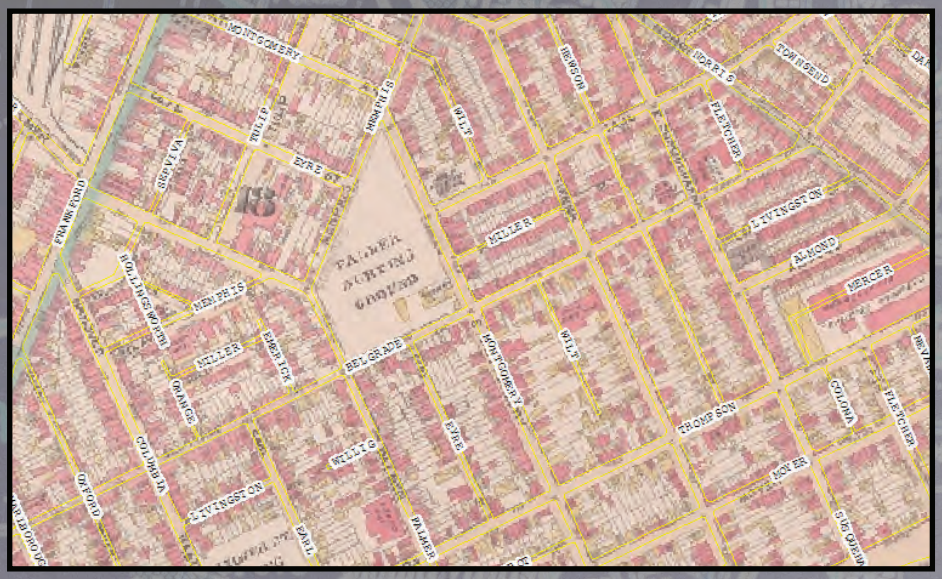


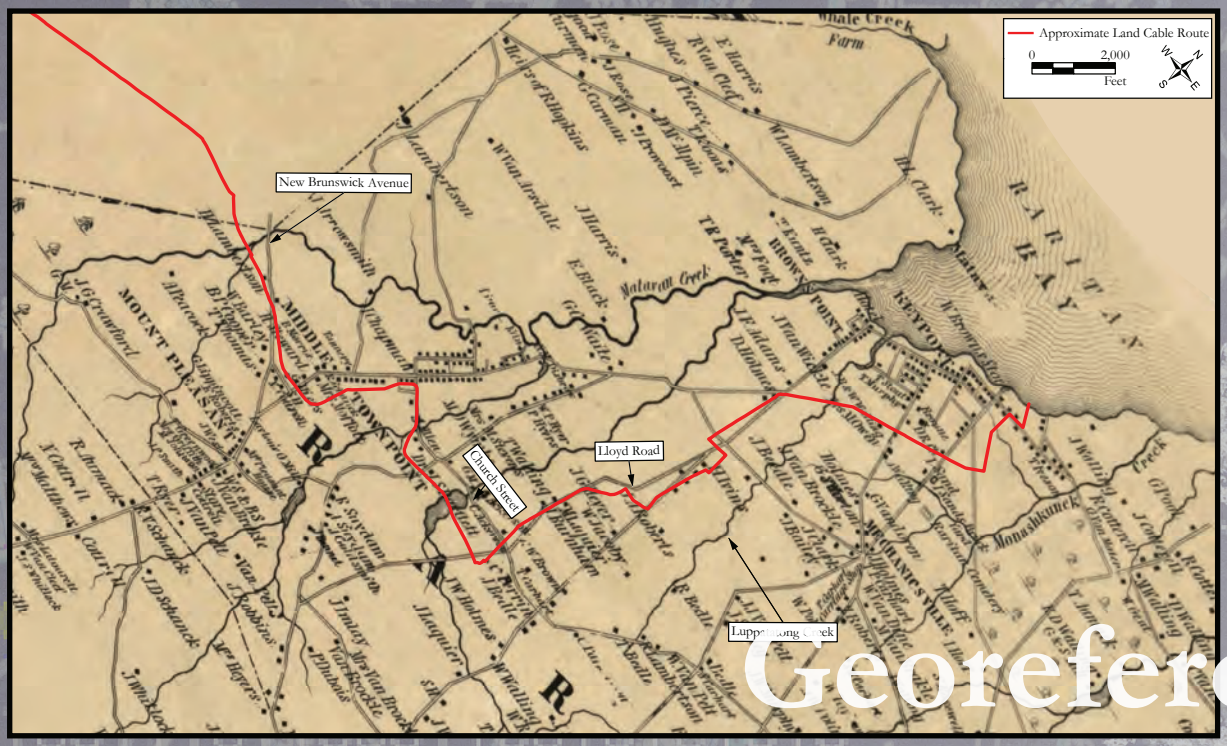


Sensitivity Modeling



From PDF project plans to .dwg files, assigning spatial information to data and images are tasks I regularly perform. In Cultural Resource Management, there is also the tricky task of georeferencing historic maps so that project areas can be represented in historical context.

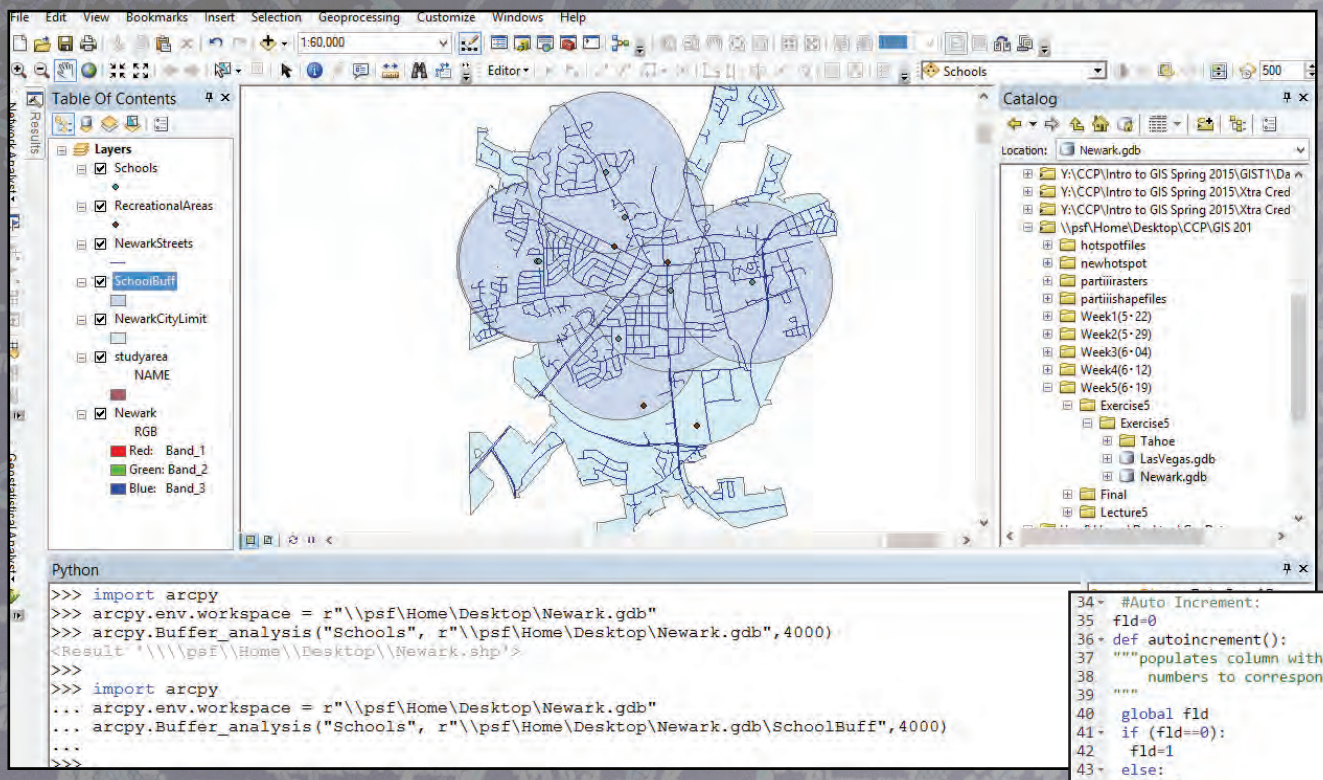




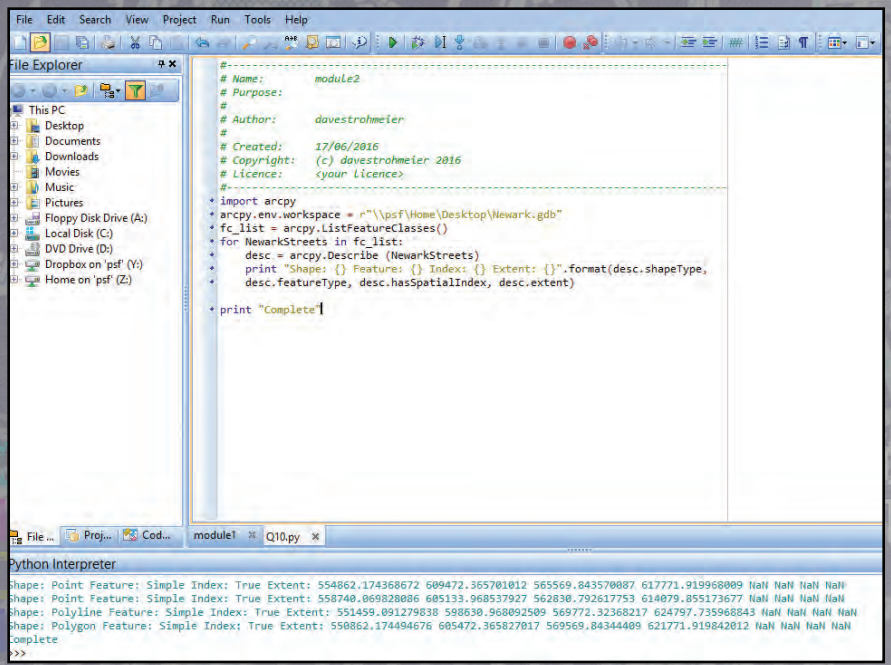
Georeferencing

Python

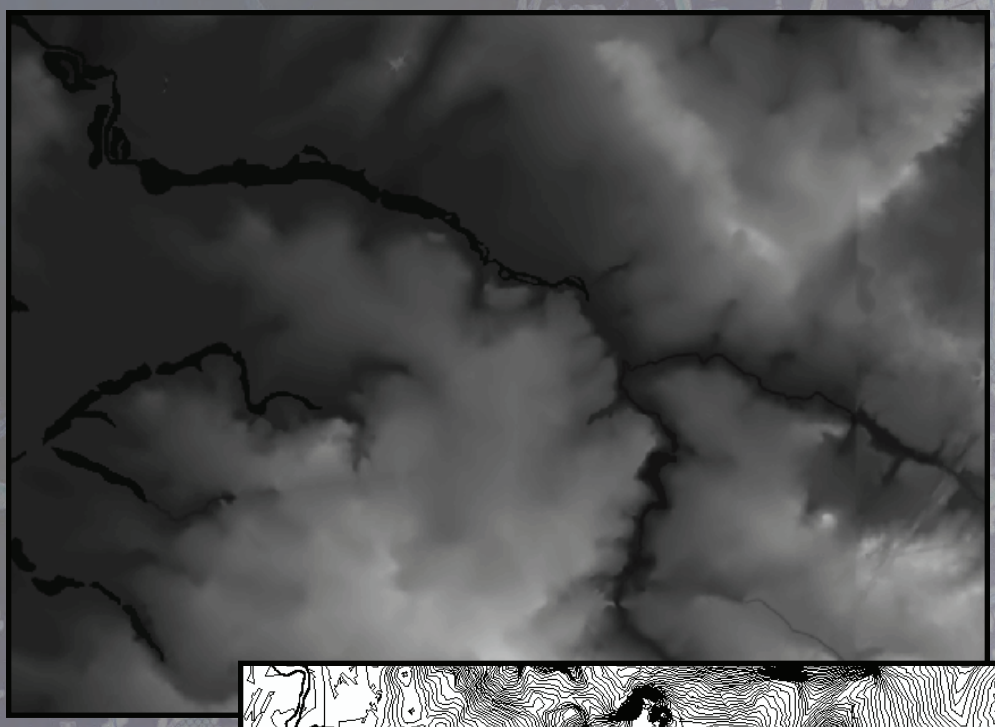
```
21 def rotate_word():
22     a= input(str("Enter a word:\n"))
23     b= input("Enter a number:\n")
24     x= [chr(ord(i)+int(b)) for i in list(a)]
25     result= ''.join(x)
26     print("Your new word is",(result))
27
28 def main():
29     rotate_word()
30
31 if __name__=='__main__':
32     main()
33
```



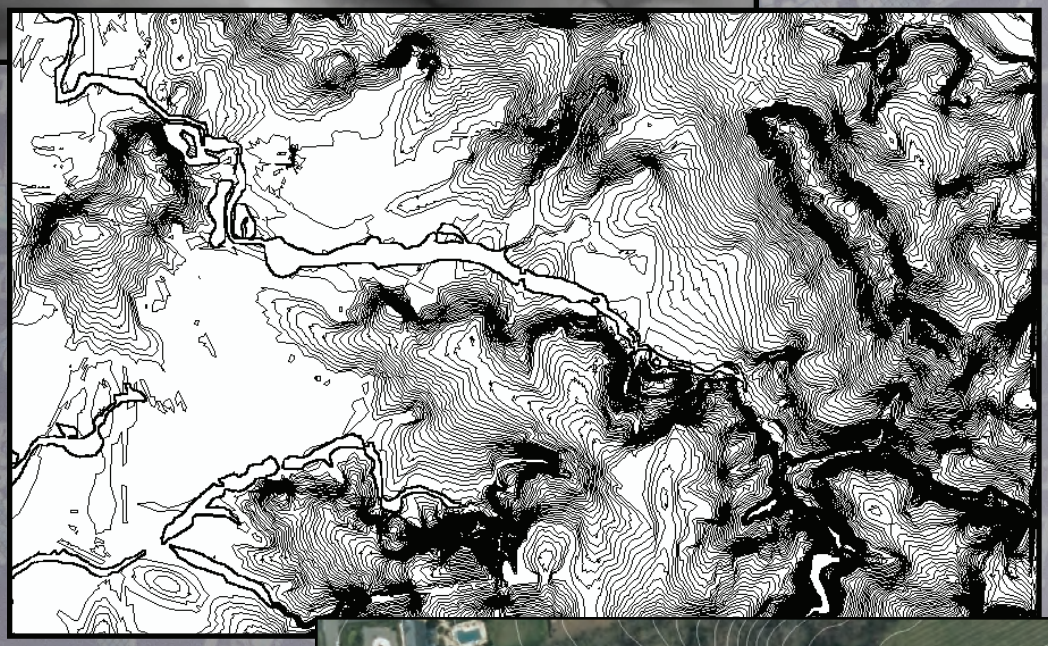

```
34  #Auto Increment:
35  fld=0
36 def autoincrement():
37  """populates column with incremented
38      numbers to corresponding rows
39  """
40   global fld
41  if (fld==0):
42   fld=1
43  else:
44   fld+=1
45  return fld
46
47 def main():
48      autoincrement()
49
50 if __name__=='__main__':
51      main()
52
```



Elevation

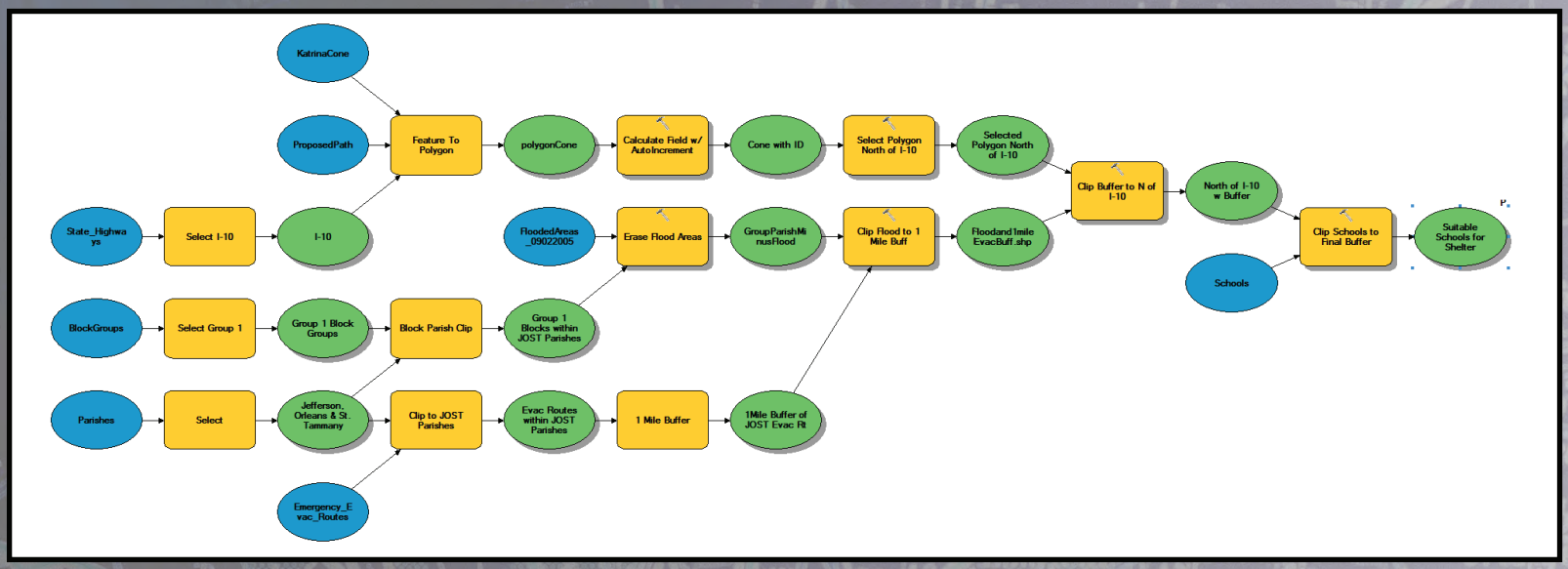


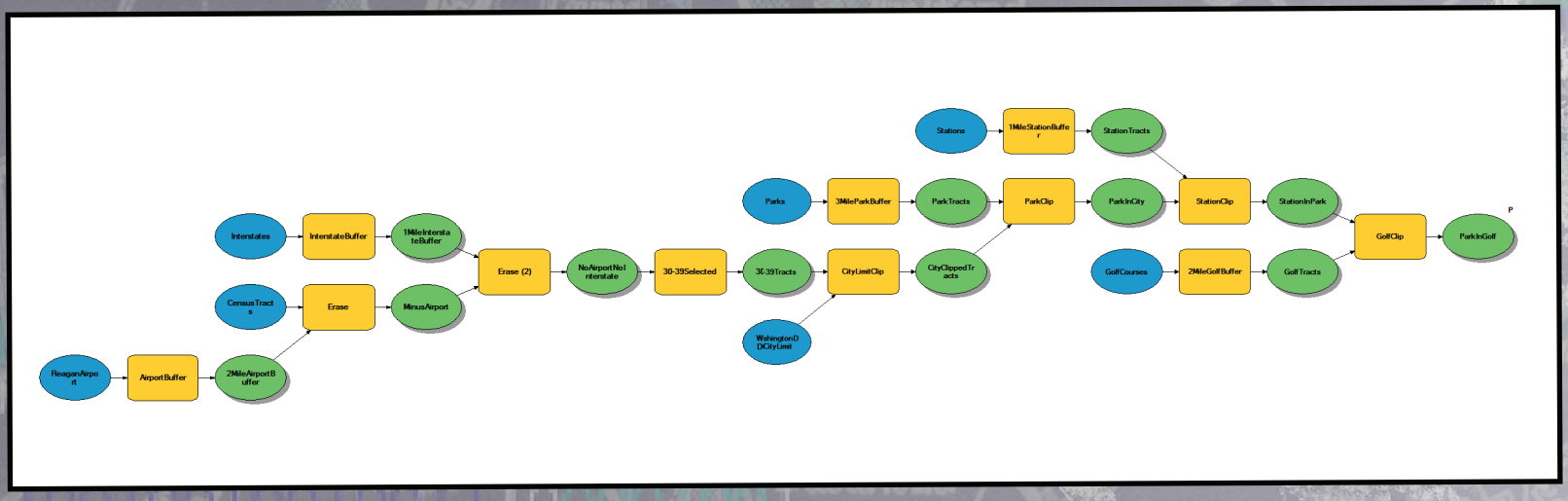
Contours



Isopleths

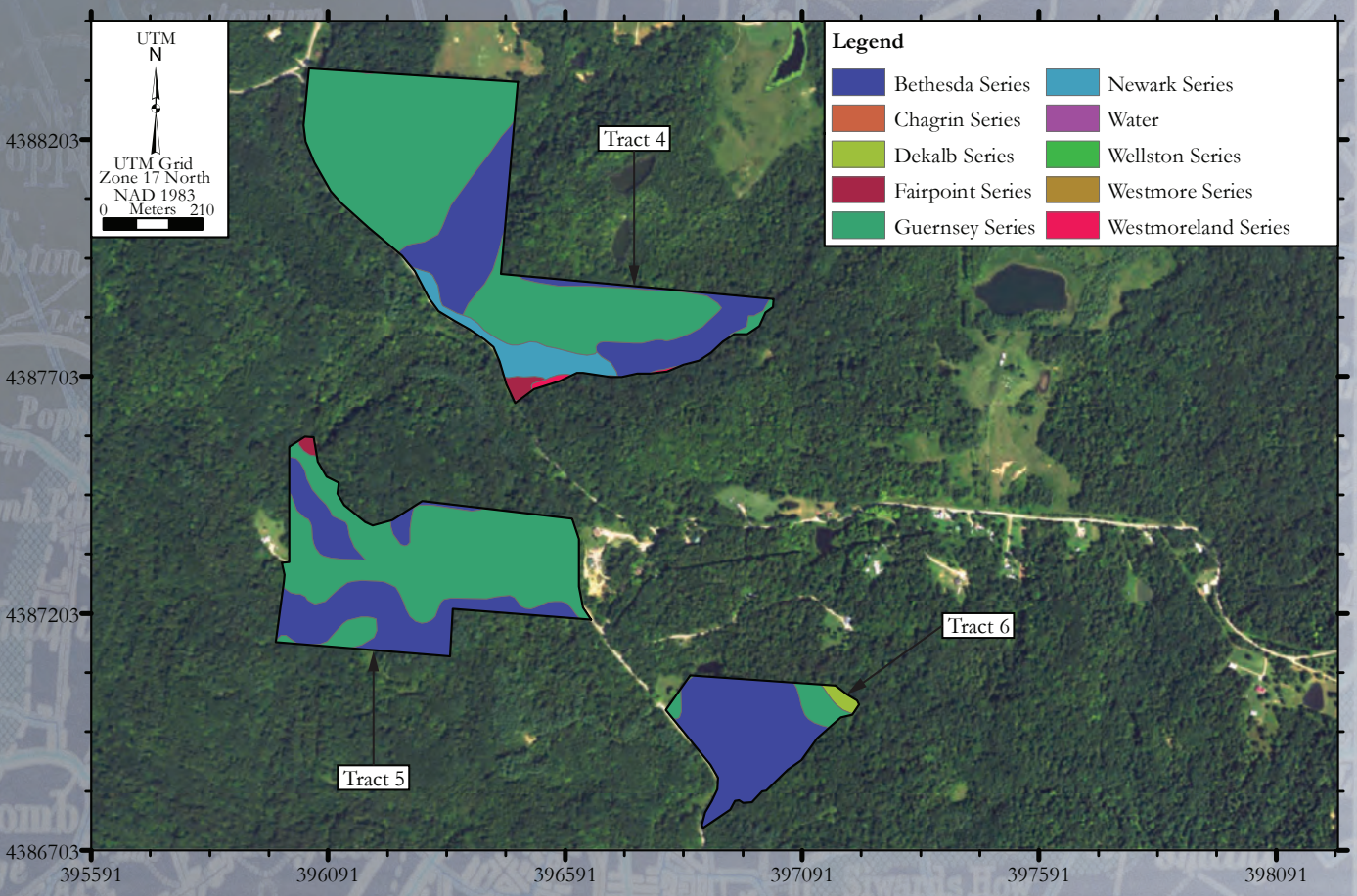
Model Builder

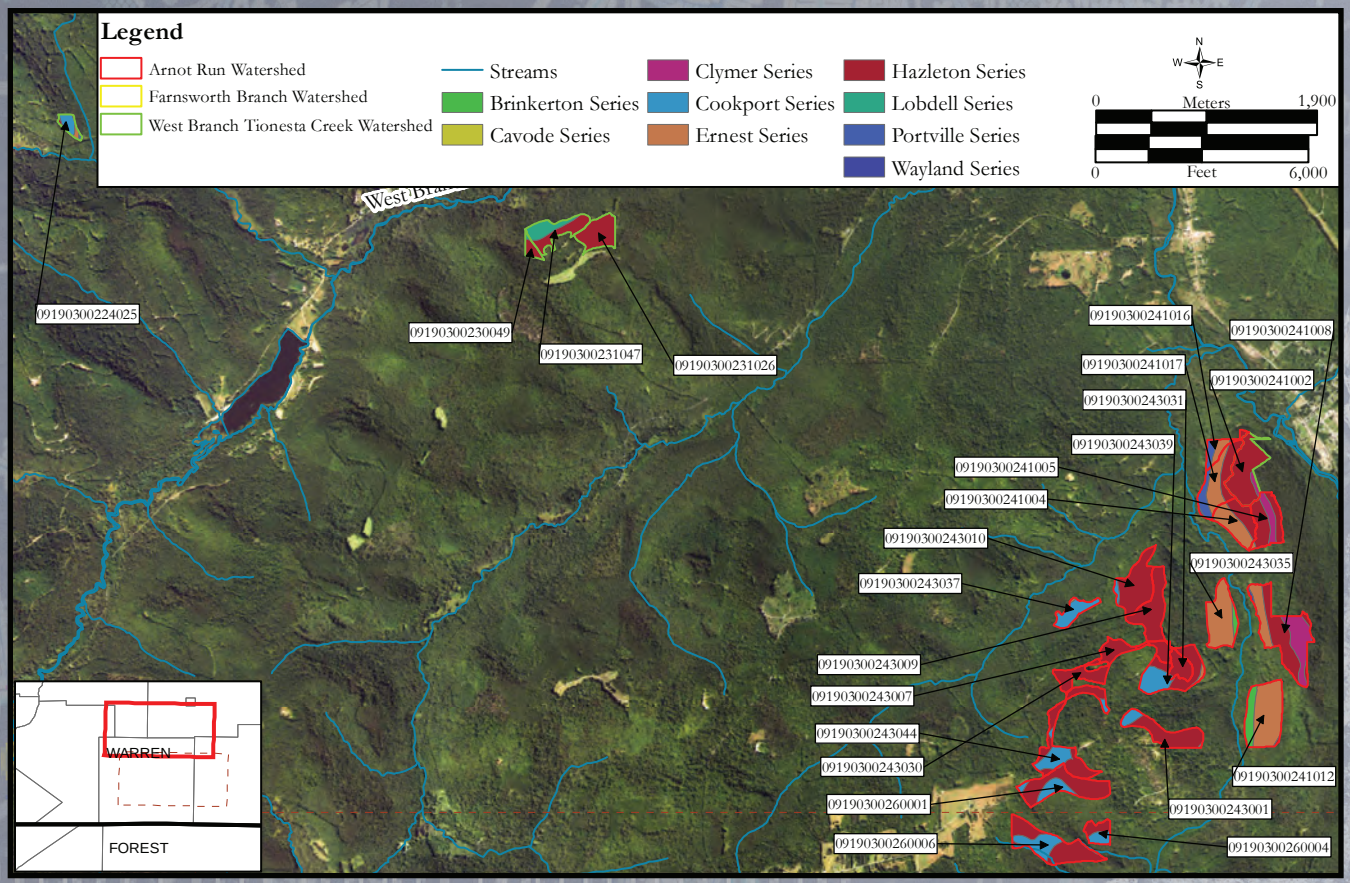




Data Joins

Joining publicly provided data, such as soil data from SSURGO, to local project data.

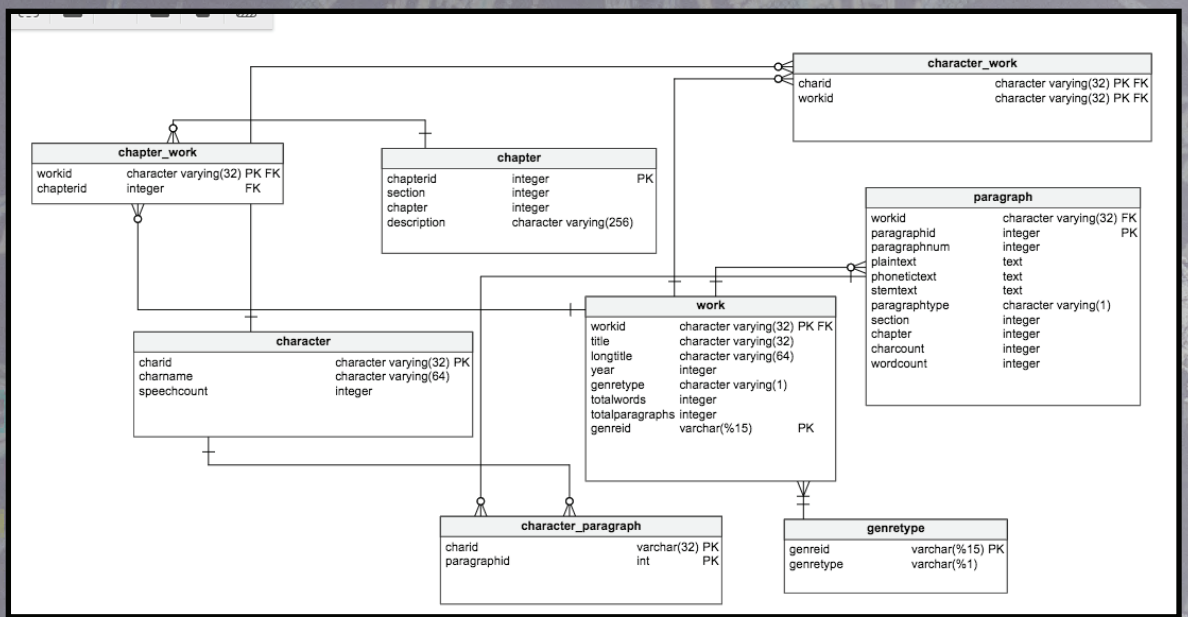




SQL/RDBMS

```
CREATE TABLE complete_crime_data.tract_thefts AS
(With st as (SELECT tracts.object_id, count(thefts.geom) AS theft_count
FROM complete_crime_data.theft_from_vehicle as thefts
LEFT JOIN complete_crime_data.census_age_2010 as tracts
ON st_within(thefts.geom,tracts.wkt)
GROUP BY tracts.object_id)
Select bt.*,theft_count
FROM complete_crime_data.census_age_2010 as bt
RIGHT Join st
on bt.object_id= st.object_id);
```

```
 /*Theft From Vehicle*/

create table complete_crime_data.theft_from_vehicle
        (wkt geometry(point, 0),district text, psa varchar(1),
        dispatch_date date, dispatch_time text, hour numeric,
        location_block text, ucr_Code text,
        general_crime_category text, police_districts text);

COPY complete_crime_data.theft_from_vehicle
        (wkt, district, psa,
        dispatch_date, dispatch_time, hour,
        location_block, ucr_Code,
        general_crime_category, police_districts)
FROM 'C:\Users\davestrohmeier\Desktop\Spatial Database Design\CSV\theft_from_vehicle.csv'
WITH CSV HEADER;

UPDATE complete_crime_data.theft_from_vehicle
 SET wkt = ST_SetSRID(ST_GeomFromwkb(wkt),2272);

 CREATE INDEX theft_from_vehicle_geom_idx
  ON complete_crime_data.theft_from_vehicle
  USING GIST (wkt);

 ALTER TABLE complete_crime_data.theft_from_vehicle
 ADD COLUMN pk BIGSERIAL PRIMARY KEY;

 ALTER TABLE complete_crime_data.theft_from_vehicle
 RENAME COLUMN wkt TO geom;
```



Network Analysis