Summary

In an August, 2006 article entitled "Leveled Colorado Playing Field Creates an Election Laboratory," the New York Times expressed incredulity at finding a Congressional District *not* gerrymandered to guarantee easy victory. Colorado's seventh district was called "remarkable" and "unlike the vast majority of House districts around the country." A survey of districting plans around the U.S. confirms this commentary: most House elections are not competitive, because contorted, irregularly shaped districts ensure solid majorities for one political party or another.

Is it possible to draw fair and simple districting plans for states? In this paper, we answer this question in the affirmative, and apply our methods to New York State as a case study of our methods.

We treat districting as a transportation problem, and solve it using the efficient *transportation algorithm*. Using U.S. Census Bureau data, we create districts by grouping the following discrete population units:

- counties
- county subdivisions
- census tracts.

We consider only the center of each unit, rather than information about borders and area.

At the same time, we choose optimization functions that ensure contiguity and compactness of districts. At each step, the algorithm aims to maximize population equality between districts while optimizing the compactness measure. As a result,

- district population deviations from the ideal are all less than 0.6% and in most cases less than 0.3%.

Using open source Geographic Information Systems mapping software, we then generate accurate color-coded districting maps.

This paper includes a detailed and complete districting map of New York State, with an inset showing the districting of the areas around New York City. The district shapes are simple and contiguous. We also include results for Iowa and Kentucky as illustrative cases.

Using our algorithm, we generate districting plans that vastly improve upon the tortuous versions seen in most states today.

Compacting Salamanders: An Algorithmic Approach to Creating Fair and Simple Congressional Districts

MCM Contest Question A

Team # 2084

February 12, 2007

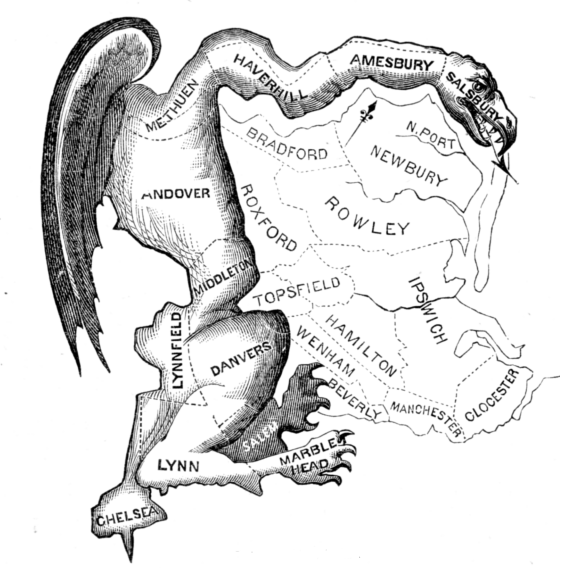

Figure 1: Essex County, Massachusetts, 1812

Contents

1 Introduction: Gerrymandering **4**

2 What We Require of a Districting Plan **5**

3 Our Model: Linear Transport Model **6**
3.1 Basic Algorithm 7
3.2 Data and Population Units 8
3.3 Limitations of Data and Algorithm 11
3.4 Compactness 12
3.5 Enhancements 12

4 New York **14**
4.1 Complete Districting Map for New York State 16
4.2 Bronx 18
4.3 Central 19
4.4 Kings 20
4.5 Long Island 21
4.6 Upstate 22
4.7 Western 23

5 How 'Fair' is Our Districting Plan? **24**

6 Conclusion **25**

List of Figures

1 Essex County, Massachusetts, 1812 1
2 Districting plans for Kentucky 10
3 The algorithm's view of districts and population units. . . . 11
4 Example of postprocessing algorithm 13
5 New York State. 16
6 Inset of New York City. 17
7 The Bronx: Districts 1 - 5 18
8 Central: Districts 6 - 11 19
9 Kings: Districts 12 - 15 20
10 Long Island: Districts 16 - 19 21
11 Upstate: Districts 20 - 24 22
12 Western: Districts 25 - 29 23

List of Tables

1 Population Equality: Bronx 18
2 Population Equality: Central 19
3 Population Equality: Kings 20
4 Population Equality: Long Island 21
5 Population Equality: Upstate 22
6 Population Equality: Western 23

1 Introduction: Gerrymandering

In 1812, Essex County, Massachusetts became the classic result of contorted senatorial district boundaries when governor Elbridge Gerry signed a remap into law[Sch65]. Because the shape of the new Massachusetts voting district resembled a salamander, the editors of the Boston Gazette March 26th, 1812, dubbed it the "gerrymander" and printed the term with the cartoon depicted in Figure 1. Throughout American history, gerrymandering has been used to advantage or disadvantage political parties, racial minorities, and other groups. The process of redistricting is governed by the Constitution, as it is ineluctably intertwined with partisan politics and racial discrimination.

After each diennial census, seats in the House of Representatives are reapportioned among the states. *Districting* then, is the establishment of the geographical boundaries that define representational districts within a respective state. Redistricting must occur after each census. Since 2006, a Supreme Court decision that congressional redistricting may occur every year has raised the stakes of the districting problem even higher.

Currently, states employ a variety of techniques to redistrict, all with mixed success. Some states employ independent redistricting commissions, or require that district plans achieve consensus, but this often leads to deadlock. The Center for Range Voting has proposed a simple algorithm for "objective" districting called the Shortest Splitline algorithm, but it does not deal with detailed population data, nor does it take geographic or political boundaries into account [The07]. In short, despite a variety of efforts, many of today's districts are gerrymandered, irregular shapes.

To design an algorithm that draws fair and simple political districts, we model the districting problem as a linear transportation problem. By taking this approach, we are able to make use of the efficient *transportation algorithm*. Our algorithm employs U.S. Census Bureau data and achieves a high standard of population equality among districts. Our districts are compact, contiguous, and preserve geographical and political boundaries.

Districting in New York State poses a particular problem. Currently, 29 seats in the House of Representatives belong to New York, which means the state must be divided into 29 districts. Upstate New York contains large swaths of rural areas with low population density, dotted by occasional cities such as Buffalo and Rochester, while extremely high population densities predominate New York City and areas immediately surrounding. In fact, nearly half the population of New York State lives in New York City. Because of these vast discrepancies of scale, maps of New York districts

need an inset to display districts in and around Manhattan. Any districting algorithm must deal with not only an unusual population distribution but also equally unwieldy configurations of islands and peninsulas. Our algorithm accomplishes this task.

2 What We Require of a Districting Plan

- *Districts must have equal populations. The maximum acceptable deviation between the largest district and the smallest district is 1%.*

  This strict condition aims to uphold the "one person, one vote" principle to the highest standard. Our 1% requirement is consistent with Supreme Court decisions: the Court has never approved a districting plan with a deviation of greater than 1%. In fact, in Karcher v. Daggett (1983), it struck down a New Jersey plan in which the largest district's population exceeded that of the smallest by .7%. The Court has declared that no population deviation greater than the lowest possible deviation is technically acceptable.

  If anything, though, our 1% condition is stricter than necessary, because only districting plans already suspected of gerrymandering are challenged before the Court. Unchallenged districting plans with sufficiently compact districts (such as the ones we aim to draw) often have population deviations greater than 1%.

- *Districts must be as "simple" as possible.*

  We evaluate simplicity by following three criteria:

  - **Contiguity.** A district must be geographically contiguous - that is, it must be possible to travel from one point in the district to every other point in the same district without passing through another district. We require substantive contiguity, not just technical contiguity: districts which are technically contiguous but are connected by very narrow sections, sometimes called "telephone line"-contiguous districts, do not qualify [Alt98].
  - **Compactness.** Districts should be as compact as possible.

    Qualitatively speaking, a district is compact if its shape is regular and confined. Because scholars and legal analysts associate non-compactness with evident gerrymandering, and compactness with fairness, this measure is crucial. However, because there are at least 24 distinct ways to measure compactness, it

is also contentious. All the known measures of compactness are deficient in some way or another; all are relative measures, rather than absolute ones; none is a sufficient replacement for a human evaluation [Alt98]. Rather than evaluate our districting plans in terms of these faulty measures, we build a compactness requirement into our algorithm.

- **Visual Evaluation.** A district must satisfy visual evaluation.

  No quantitative measure yet proposed can substitute for the human eye, which is especially proficient in dealing with unusual geographic configurations. For example, in terms of both contiguity and compactness, configurations involving islands are difficult to quantify. We subject all districting plans to visual evaluation in order to check for anomalies that may elude our quantitative requirements.

- *Districts should respect extant geographical and political boundaries.*

  Specifically, districts should respect county lines whenever possible, and, as a next priority, city and town boundaries.

3 Our Model: Linear Transport Model

We approach the problem of congressional districting as an instance of the classic *transportation problem*. In the transportation problem, goods from m warehouses must be efficiently distributed to n locations. Translating these ideas into the language of districts, we need to determine m district centers so that the distances to each of their constituent population units are minimized. Creating the districts in this manner ensures the "simplicity" of the districts.

We define the problem in formal terms. Let p_i be the population of an indivisible population unit i, and let x_{ij} be 1 if population unit i is in district j. Let c_{ij} be the "cost" of including population unit i in district j. This will typically be the distance from the center of population unit i to the center of the district j or that distance weighted by population. Then, our goal is to determine the optimum x_{ij}'s to minimize the *objective function*

$$\sum_{j=1}^{m} \sum_{i=1}^{n} x_{ij} c_{ij}$$

subject to the constraint that each population unit is in exactly one district,

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \text{for } i = 1, 2, ..., n,$$

and that district j contains exactly g_j population units

$$\sum_{i=1}^{m} x_{ij} = g_j \qquad \text{for } j = 1, 2, ..., m.$$

The transportation problem is well studied in the field of linear programming and operations research. A known, efficient algorithm known as the *transportation algorithm* computes the solution to transportation problems [Mun57]. As stated above, we need to know the number of population units that are going to appear in each of the districts. Since there is no way to know this a priori, we iteratively update the g_j's to approach population equality between districts.

3.1 Basic Algorithm

Below, we give an outline for our basic algorithm which closely follows "Political Redistricting by Computer" by Helbig, et. al. [HOR72]. A more detailed description can be found in that article.

Step 1a: Make an initial guess for the centers of the districts. This can be done in a number of ways such as a uniform distribution of the area under consideration, randomly selected centers of population units, or the centers of the current districts. In our experimentation, we found that the initial choices do not influence the final solution too heavily.

Step 1b: Make an initial guess for the number of population units that will appear in each of the districts. These are the g_j's that were discussed above. In our experimentation, we found that just uniformly distributing the populations evenly among the districts is sufficient.

Step 2: Compute the distances d_{ij} from the the centers of all of the population units i to the centers for all the districts j. We use the formulas at `http://williams.best.vwh.net/avform.htm` to compute the distance in miles between two points given their longitude and latitude. Thus, this distance is not a planar approximation; it corresponds to a distance measured along the Earth's spherical surface.

Step 3: Using the transportation algorithm, solve the subproblem for the x_{ij}'s to obtain a new districting plan. In the transportation algorithm,

the costs c_{ij} can either be the distances d_{ij} or the population-weighted distances $d_{ij} * p_i$. If the objective function with these costs is minimized, then reasonably compact districts will result.

Step 4: Based on the new districting plan from Step 3, compute the centers of the new districts as either the average of the centers of all the constituent districts or as the population-weighted average of the constituent centers.

Step 5: Check to see if the new solution is significantly different from any one previously computed. This can be done by checking if the new centers differ from the old centers by more than a prespecified tolerance. If they are significantly different, go to Step 2 using the new district centers to refine the solution futher. If they are not significantly different, then go to Step 6.

Step 6: Calculate the ratio of the population of the largest district to the population of the smallest district. If this is smaller than any ratio previously seen, then save the districting plan. If Step 2 has been run more than a set number of times (say 200) without improving the ratio, the the algorithm terminates. Otherwise, proceed to Step 7.

Step 7: In this step we modify the g_j's in an attempt to improve population equality. In [HOR72], it is suggested that 1 is added to the g_j corresponding to the smallest district and 1 is subtracted from the g_j corresponding to the largest district. For problems with large numbers of population units, this updating procedure is often too conservative. Instead, we use the average population of each of the population units to determine approximately how many "extra" population units the largest district contains. This number is then subtracted from the largest district's g_j and added to the smallest districts g_j. Once the g_j's have been updated, proceed to Step 2.

3.2 Data and Population Units

In order to use the linear transport model, we need data that provides us with population units, their populations, and the coordinates (longitude and latitude) of their centers. The United States Census Bureau's Gazetteer Files for the 2000 Census document this data [Uni00a].

The Gazetteer Files provide census data according to five different divisions: counties, county subdivisions, census designated places, census tracts, and zip code tabulation areas. We do not consider zip code tabulation areas since they do not necessarily respect local governmental or geographical features. Similarly, we do not consider the census designated

places since they do not cover the entire state.

Each of the other population units have advantages and disadvantages:

Counties are the largest division of a state for which the gazetteer files report data. The voting process is usually handled individually by the counties. Therefore, districting a state using counties as the fundamental population units makes the voting process easier to conduct. Since counties are the largest subdivision, they make the algorithm much faster. On the other hand, becuase counties are so large, it is often more difficult to ensure population equality when using them as the fundamental population unit. In New York, for example, some counties require more than one congressional representative; thus, it is not feasible to use counties as population units.

Another option for the fundamental population unit is the county subdivision. Because it is an intermediate size, it provides a balance of the advantages from the counties and census tracts, while mitigating the disadvantages in either of the other two options. County subdivisions respect county lines like census tracts. Computing time is faster using county subdivisions than it is using census tracts, but county subdivisions provide a more detailed base unit for the algorithm than counties.

Census tracts are the finest divisions that are reported. They are designed to respect county boundaries, geographical boundaries, and to have a population of approximately 4,000 people. They appear to be the smallest population units that are feasible to use in constructing congressional disctricts. Because the census tracts are so fine, it is much easier to obtain congressional districts with population equality. However, it is more difficult logistically to run the elections using census tracts since counties and cities may be split among multiple districts.

When choosing which divisions to use for population units, one must population equality, ease of computation, and preservation of existing boundaries. These decisions need to be made on a state by state basis since a state population distributions vary.

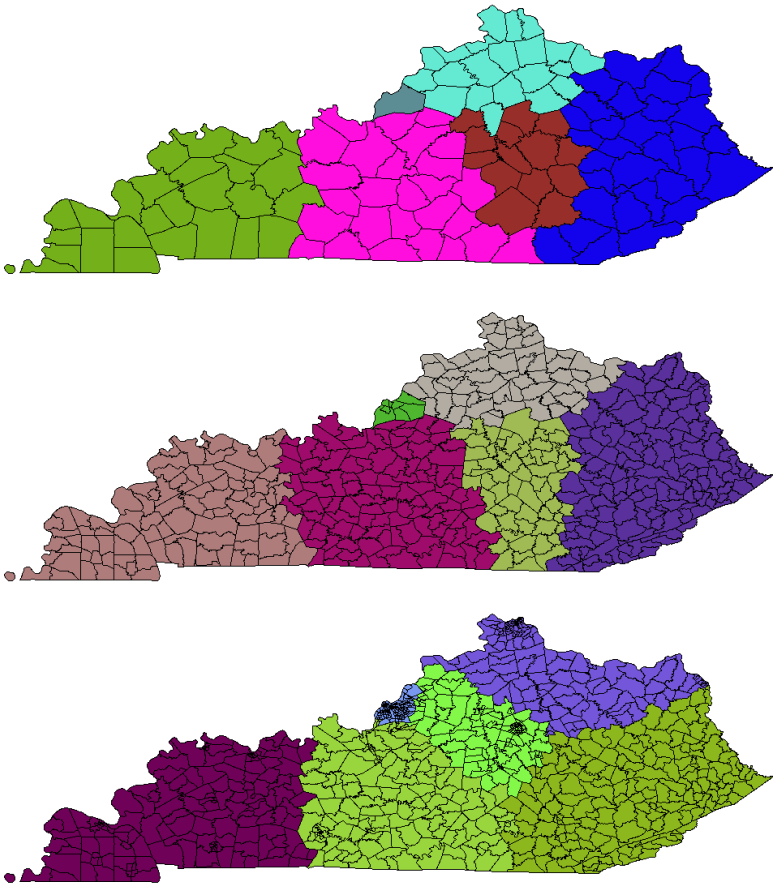

Figure 2: Districting plans for Kentucky

From top to bottom, the plans were produced using decreasing population unit size: counties, county subdivisions, and tracts.

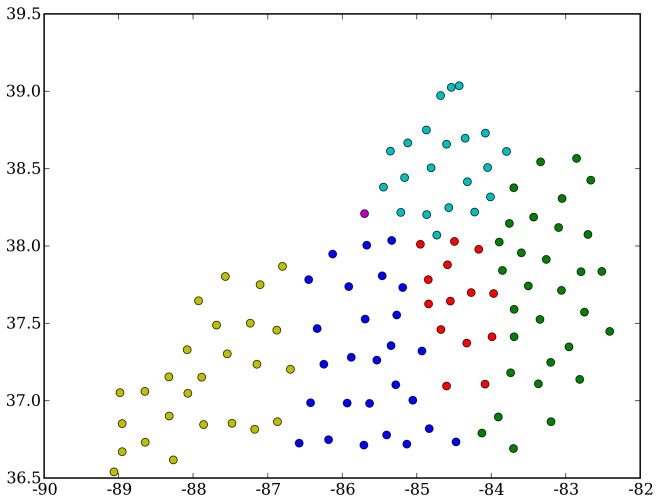

Figure 3: The algorithm's view of districts and population units.

3.3 Limitations of Data and Algorithm

One drawback to our technique is that the algorithm and the data do not give a detailed specification of the shapes of the population units nor do they provide information about how the units border one another. In addition, the algorithm cannot make considerations about islands or other unique geographic features. Any algorithm or technique relying solely on this data will suffer from these drawbacks. Figure 3 shows an example of how the algorithm views the data when creating a district plan. The centroids of Kentucky counties are shown as dots colored according to district.

Another possible drawback is that the algorithm places much more empahasis on having population equality as opposed to compactness. That is, if the algorithm has to choose between a district layout with poor compactness but good population equality versus a different district layout with much better compactness and only slightly worse population equality, then it will select the one with better population equality. In future work, it would be interesting to explore incorporating a tradeoff between compactness and population equality into the algorithm.

When computing the centroids of the districts, we do not take into account that the population units are distributed upon a spherical surface.

This does not hamper the algorithm too much since the areas under consideration are small enough that they are practically planar. However, further investigation is needed to confirm this, especially if districting areas are very close to the poles.

3.4 Compactness

As mentioned in section 2, we aim to draw compact districts. We might seek to quantify how successful our districting plans are in this regard. However, many compactness measures involve area:perimeter ratios, and we cannot compute district perimeters using the available data. More importantly, a thorough survey of districting literature and legal standards reveals that no widely accepted, reliable measure for compactness exists. Of the 24 measures of geographic compactness surveyed by Altman, all violate basic axioms concerning dispersion, dissection, and indentation. This rigorous classification confirms Young's work, in which he showed that shapes which appear noncompact to a viewer can achieve high ratings from certain compactness measures [You88]. In short, all measures of geographic compactness are deficient. Population compactness, which has been less extensively studied, is equally unwieldy. Known measures of population compactness are also deficient in terms of Altman's axioms.

As evidenced throughout this paper, the objective functions used in the transportation algorithm produce compact districts that withstand a visual evaluation. Rather than consider a proposed compactness measure in our analysis or algorithm, we employ these objective functions, and recognize their success in producing simple districting plans.

3.5 Enhancements

We supplement the basic algorithm with two major enhancements:

- **Postprocessing.** Once our algorithm has produced a district layout, our algorithm could be supplemented by a postprocessing algorithm to ensure contiguity. As discussed above, the data only specifies the center of a population unit and nothing about the shape. When the algorithm minimizes the objective function to obtain compactness, unconnected districts may arise due the population unit shapes. An example of this can be seen in Figure 4. Here we generate a districting plan for Iowa using census tracts for population units and geographical distance for the objective function.

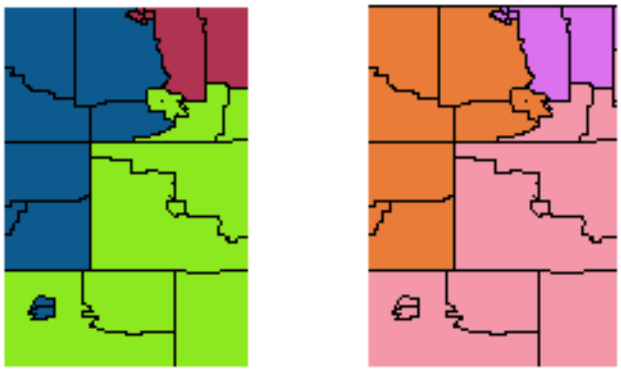

Figure 4: Example of postprocessing algorithm

The above example is taken from the output of our algorithm for the districting of Iowa using tracts as the population units. The left hand image is shown before postprocessing while the right one is after it. Notice that the two disconnected tracts in the lower right corner change districts from belonging to the one in the upper left to belonging to the district lower right. To balance out this change in district populations, a postprocessing step identifies the census tract on the border whose population most closely matches the population of the disconnected tracts. It then swaps the border tract with the disconnected tracts. In this manner, the postprocessing algorithm ensures contiguity of the districts while maintaining approximate population equality. In Figure 4, the census tract marked with the dot is chosen for swapping.

- **Iterative Approach.** In order to make the problem computationally easier, our algorithm could first be run using counties as population units, to divide the state up into a smaller number of "substates." This problem is relatively easy due to the small number of counties. Once the substates are produced, the algorithm runs on each of the substates using the desired population units. We use this method to divide New York state into 6 substates each with a population that is approximately a multiple of the ideal number of residents per district.

4 New York

The state of New York poses some interesting challenges to constructing its 29 congressional districts. It has a both extremely urban and extremely rural areas. It is these dense, urban areas that present the greatest problem in determining congressional districts. For example, Manhattan alone contains over 1.5 million people – enough for more than 2 congressional districts. Thus, census tracts must be used as population units.

Due to time and computational constraints, we were not able to have our algorithm consider New York State as a whole with its 4,907 tracts. Instead, we used the iterative approach described above to split the state into 6 separate substates by county. Then, we ran the basic algorithm on each of the substates using census tracts as the population units. In all of the substates, the postprocessing algorithm was used to ensure district contiguity.

- **Bronx:** Orange, Richmond, Rockland, Westchester, and Bronx Counties. (5 districts)
- **Long Island:** Suffolk and Nassau Counties. (4 districts)
- **Central:** New York and Queens Counties. (6 districts)
- **Kings:** Kings County. (4 districts)
- **Western:** Allegany, Cattaraugus, Chautauqua, Steuben, Wyoming, Livingston, Erie, Niagara, Genesee, Monroe, Seneca, Wayne, Yates, Shuyler, Tioga, Tompkins, Ontario, Broome, Cayuga, Cortland, Chemung, and Orleans Counties. (5 districts)
- **Upstate:** Chenango, Oswego, Oneida, Onondaga, Madison, Lewis, Jefferson, Herkimer, Hamilton, St. Lawrence, Franklin, Essex, Clinton, Warren, Washington, Saratoga, Fulton, Montgomery, Schenectady, Otsego, Schoharie, Albany Rensselaer, Columbia, Delaware, Greene, Dutchess, Sullivan, Ulster, and Putnam Counties. (5 districts)

Note that the ideal number of residents per district will vary from substate to substate (although these variations are smaller than the ones found from state to state). This is an artifact of having to subdivide the state into substates because of the time constraints of the contest. Instead, a better insight into our algorithm is gained by looking at the districting within the

substates. With more time and computational power, all of New York could be considered at once eliminating these substate differences.

In Appendix B, the tract makeup for District 1 can be found.

4.1 Complete Districting Map for New York State

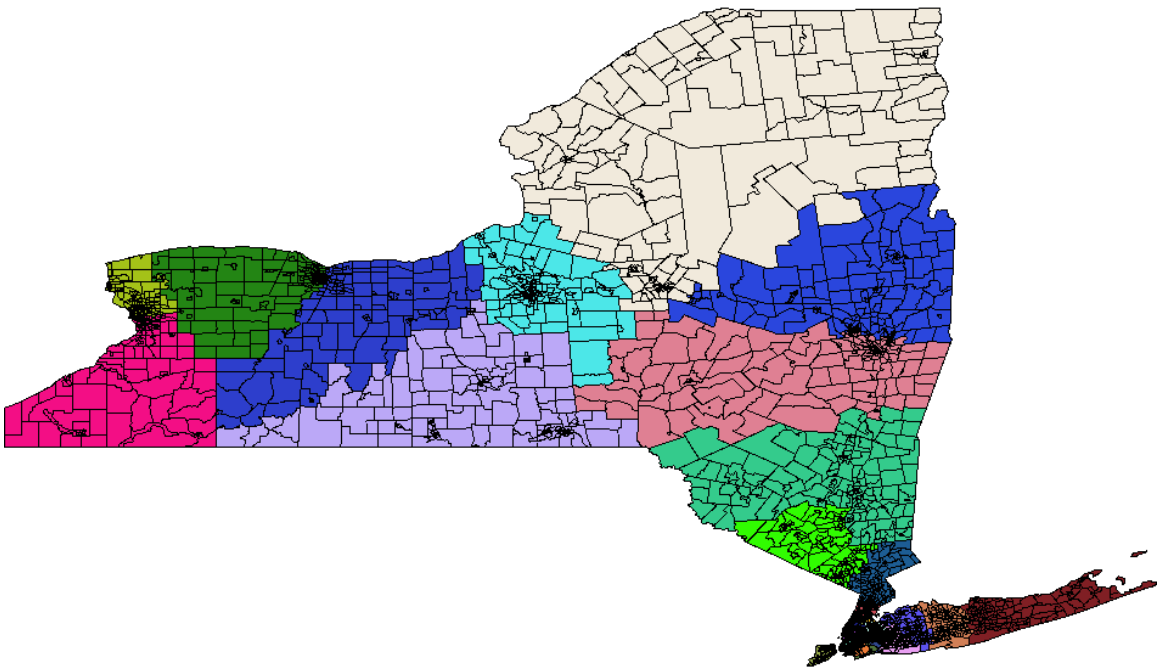

Figure 5: New York State.

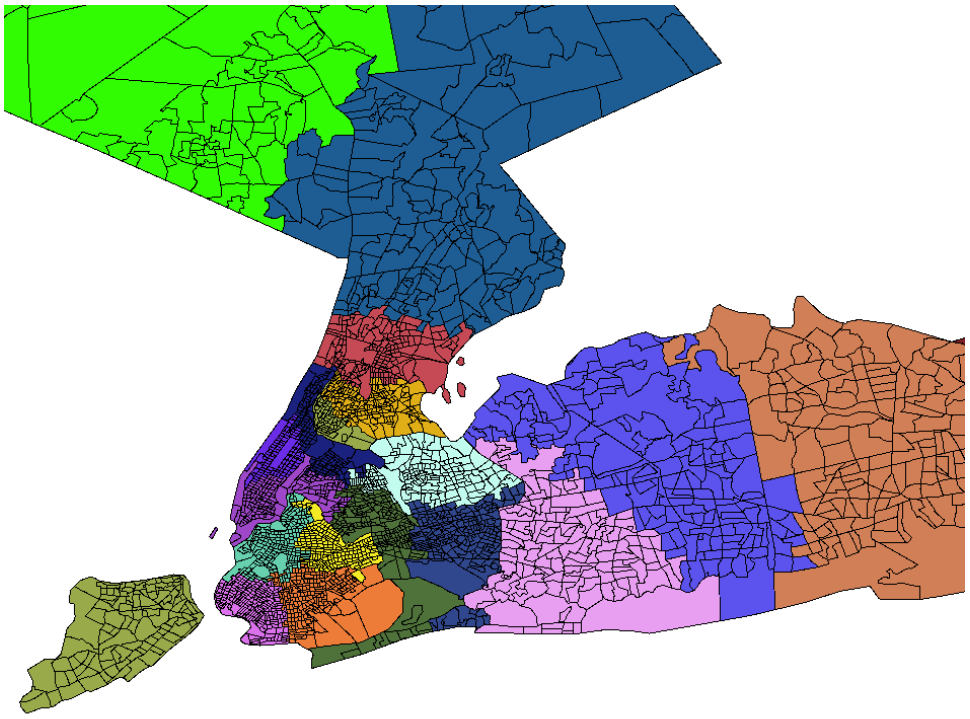

Figure 6: Inset of New York City.

4.2 Bronx

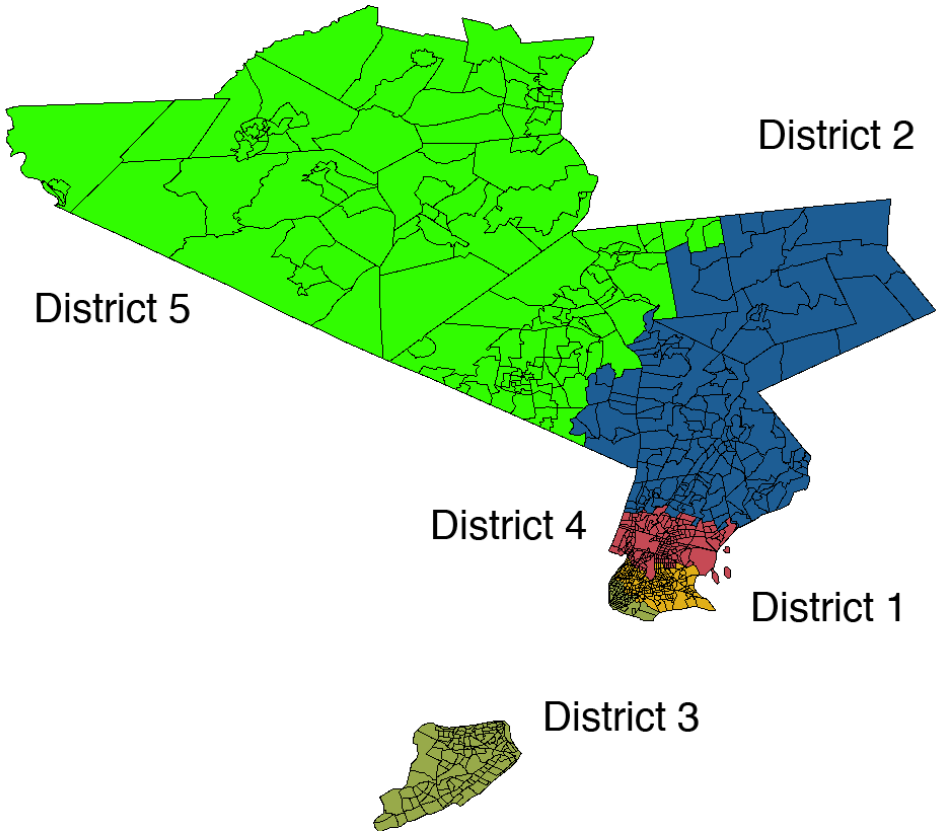

Figure 7: The Bronx: Districts 1 - 5

	Population	Deviation	Deviation
Optimal:	666495		
District 1:	668571	+2076	0.3115
District 2:	664771	-1724	-0.2587
District 3:	665371	-1124	-0.1686
District 4:	669851	+3356	0.5033
District 5:	663913	-2582	-0.3874
Average Deviation:			0.33 %

Table 1: Population Equality: Bronx

4.3 Central

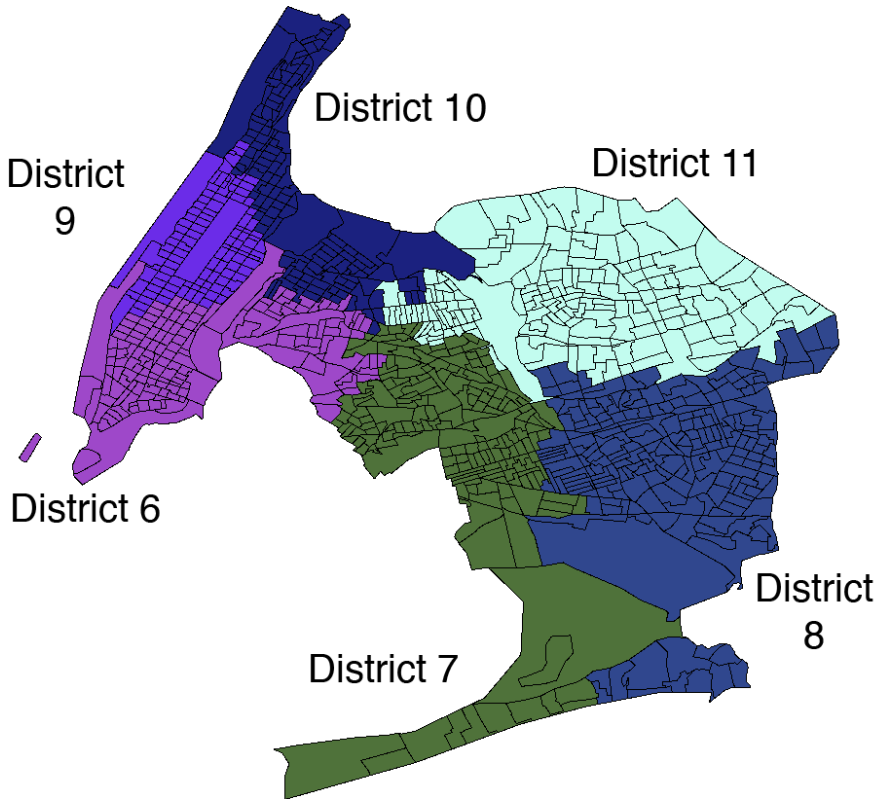

Figure 8: Central: Districts 6 - 11

	Population	Deviation	Deviation %
Optimal:	627818		
District 6:	628869	+1051	0.1674
District 7:	626374	-1444	-0.2300
District 8:	627826	+8	0.0013
District 9:	628182	+364	0.0580
District 10:	627881	+63	0.0100
District 11:	627774	-44	-0.0070
Average Deviation:			0.079 %

Table 2: Population Equality: Central

4.4 Kings

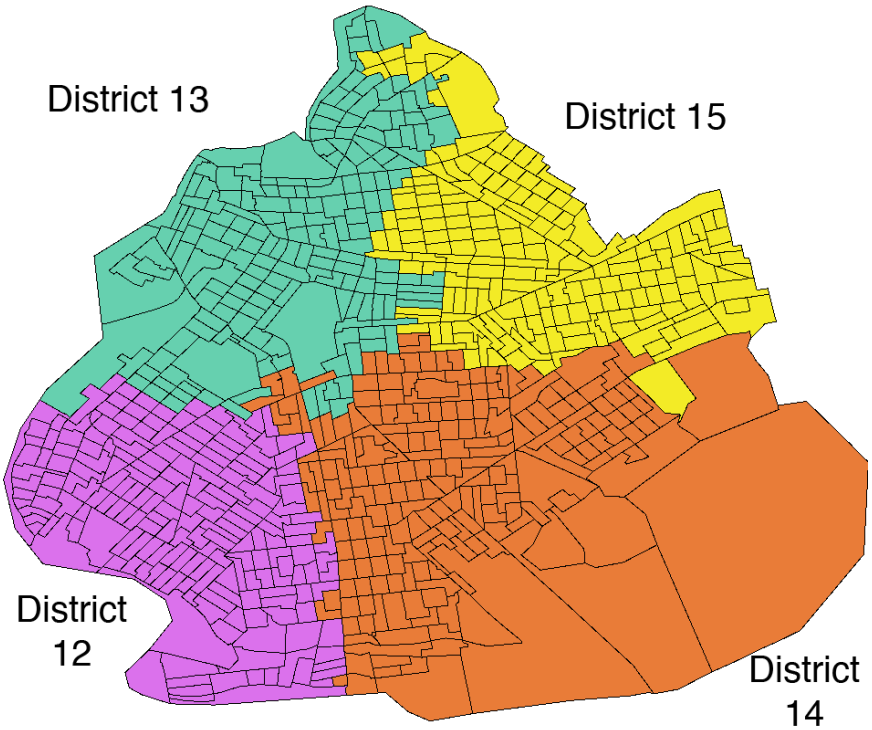

Figure 9: Kings: Districts 12 - 15

	Population	Deviation	Deviation %
Optimal:	616332		
District 12:	616062	-270	-0.0438
District 13:	616598	+266	0.0432
District 14:	615184	-1148	-0.1863
District 15:	617482	+1150	0.1866
Average Deviation:			0.11%

Table 3: Population Equality: Kings

4.5 Long Island

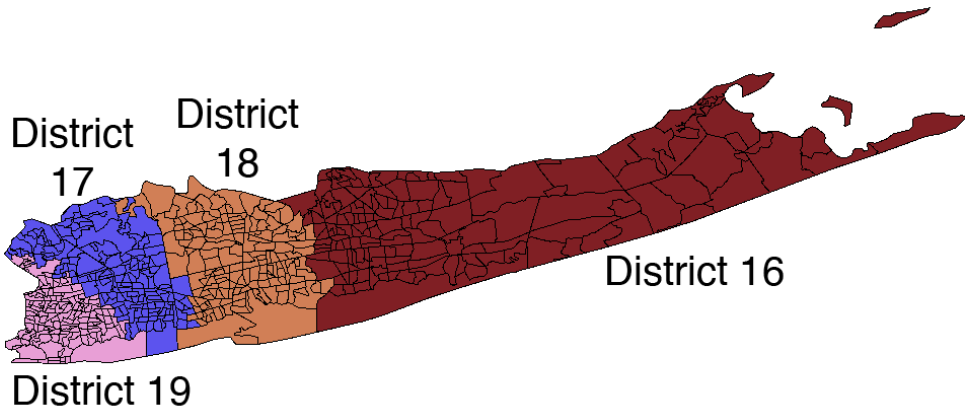

Figure 10: Long Island: Districts 16 - 19

	Population	Deviation	Deviation %
Optimal:	691349		
District 16:	694877	+3528	0.5103
District 17:	691674	+325	0.0470
District 18:	688252	-3097	-0.4480
District 19:	690592	-757	-0.1095
Average Deviation:			0.28%

Table 4: Population Equality: Long Island

4.6 Upstate

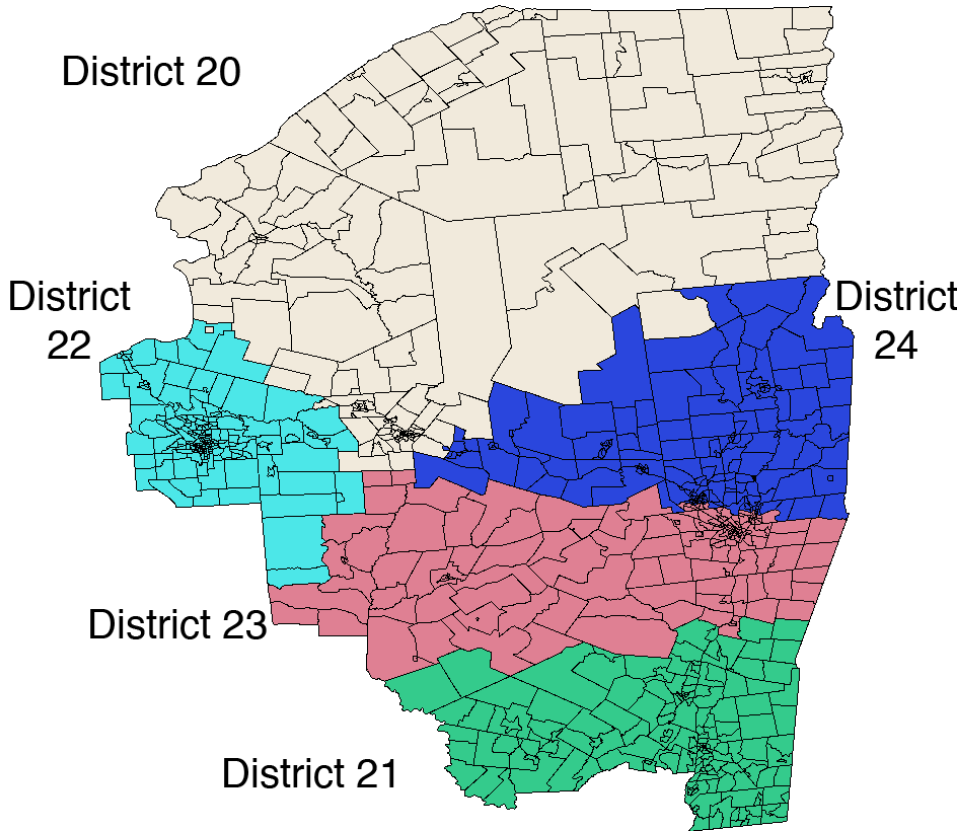

Figure 11: Upstate: Districts 20 - 24

	Population	Deviation	Deviation %
Optimal:	666190		
District 20:	664515	-1675	-0.2510
District 21:	667671	1481	0.2220
District 22:	665824	-366	-0.0550
District 23:	668078	1888	0.2830
District 24:	664865	-1325	-0.1990
Average Deviation:			0.20%

Table 5: Population Equality: Upstate

4.7 Western

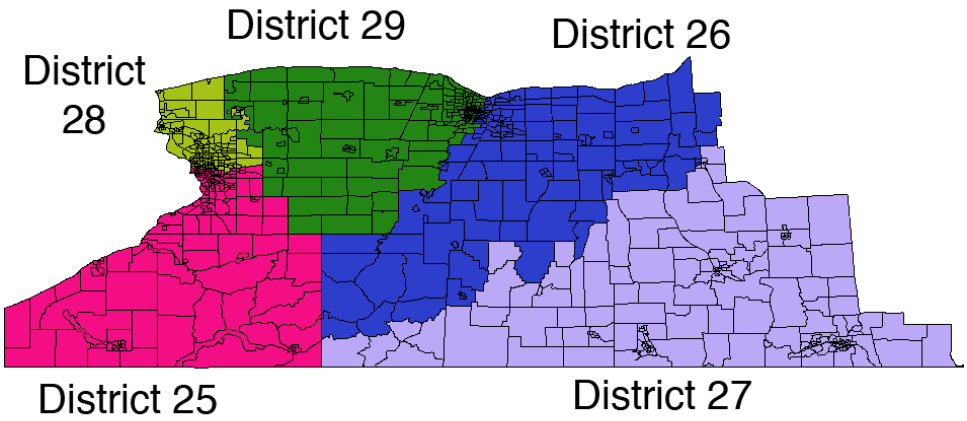

Figure 12: Western: Districts 25 - 29

	Population	Deviation	Deviation %
Optimal:	667882		
District 25:	666490	-1392	-0.2084
District 26:	669374	1492	0.2234
District 27:	668854	972	0.1455
District 28:	666957	-925	-0.1385
District 29:	667735	-147	-0.0220
Average Deviation:			0.15%

Table 6: Population Equality: Western

5 How 'Fair' is Our Districting Plan?

To evaluate whether our districting plan is an improvement over the ones currently in practice, we examine a set of criteria on the "fairness" of a district [LS85].

1. *Equal population*: The district should comply with the Supreme Court's ruling patterns for "One man, one vote."
2. *Compact*: The district will appear intuitively compact.
3. *Contiguous*: The district will be visually well connected, with no meandering boundaries or isthmus-like separations.
4. *Border Preservation*: The district should not cross county borders, in particular we seek to minimize the number of counties crossed by each district.
5. *Communities of Interest*: If there is a special interest community within a district, this community should not be divided.
6. *Competitive*: A competitive district is one in which the partisan outcome is uncertain.
7. *Ethnic Balance*: The racial, ethnic, and linguistic groups of a state should not be intentionally sequestered to diminish the weight of their votes, neither should their communities be excessivly dispersed so that they have no leverage against the majority.
8. *Party Symmetry*: The percentage of the vote obtained by a given party in the district should be proportional to the number of seats the party obtians in the House of Representatives
9. *Incumbent Blind*: The district should not be drawn with respect to the residencies of officeholders.

The districting model presented in this paper does not make reference to a political landscape, as that would necessarily imply a biased approach. Fairness criteria 5-8 acknowledge political interests and are consequently not handled by this model. By the same token, we certainly satisfy criterion 9.

In [LS85], the authors discuss several arguments against compactness as a politically neutral districting criterion. These arguments claim that a

compactness requirement advantages Democrats, who tend to live in urban areas with high population density. However, there have been no studies to offer real evidence that this is the case. None of the arguments define a metric for evaluating compactness, and all base their claims on the vague notion that compactness is the abscence of oddly shaped districts.

Another concern about compactness is that it may clump special interest groups together in general, which might be considered a from of gerrymandering. Altman argues, however, that the effects of compactness on a group's political power depend upon the population distribution of that group. If a group is evenly distributed throughout a region, then no district will contain a majority of a minority group, whether the district is considered compact or not. On the other hand, uniform and normal distribution models show that special interest groups are parceled evenly throughout a region. Thus, any compact districting would be equally likely to give one group the political advantage over another. He asserts that it is not compactness that violates political neutrality, but rather that geographic districting itself advantages the majority.

If it is possible, using counties as the population unit will ensure that we satisfy 4. However, counties are often too coarse to satisfy the population requirement, and in this case using county subdivisions or census tracts respects geographical and political borders as much as possible. One drawback of our model is that it does not take city and town borders into account. For future work, one might also attempt to preserve these borders.

In summary, because our districting plan considers only population distributions, and not partisan voting trends or other factors, it is fair.

6 Conclusion

To combat the problem of gerrymandering in the United States, we have proposed a non-partisan algorithm for redistricting states. By treating redistricting as a transport problem, we preserve population equality, draw simple shapes, and respect existing boundaries. This algorithm was successful when applied to several states. A full redistricting of New York based on census tracts shows that it is possible to achieve a logical state redistricting.

Appendix A: Implementation

Our basic algorithm as well as the transportation algorithm was implmented in Python. We used the boundary files provided by the U.S. Census Bureau to produce the maps [Uni00b]. The shape files were loaded into a geometry-aware PostGIS database [Pos07]. The district maps were produced by using the Quatum Geographical Information System [Qua07].

```
#!/usr/bin/python
import scipy, sys, yaml, math
import random
from transportation import *
from goto import *

rad = math.pi / 180.0
def calcDistance(lat1, lon1, lat2, lon2):
    """
    Calculate distance between two lat lons in NM
    """
    yDistance = (lat2 - lat1) * 60.00721
    xDistance = (math.cos(lat1 * rad) + math.cos(lat2 * rad)) * (lon2 - lon1) * ( 60.10793 / 2)
    distance = math.sqrt( yDistance**2 + xDistance**2 )
    return distance * 1.15078

def districtInfo(x):
    NUM_DISTRICTS = x.shape[1]
    di = []
    for j in range(NUM_DISTRICTS):
        nl = []
        l = scipy.where(x[:,j]>0)[0]
        for num in l:
            nl.append(num)
        nl.sort()
        if nl != []:
            di.append(nl)
    return di

def dumpDistrictInfo(di, filename):
    file = open(filename, "w")
    for district in di:
        file.write( " ".join([str(x) for x in district]) + "\n" )
    file.close()

try:
    filename = sys.argv[1]
    file = open(filename)
    parameters = yaml.load(file)
    file.close()
except:
    print "You must specify a YAML input file!"
    sys.exit(0)

NUM_DISTRICTS = int(parameters['districts'])
POP_UNITS     = int(parameters['popUnits'])
```

```
H = 200
oldMax = (0,0)
refineCount = 0
lastChanged = [-1,1]

try:
    EPSILON = float(parameters['epsilon'])
except:
    EPSILON = 0.4


#Load the data from the file
popUnitInfo = []
dataLayoutFile = open(parameters['dataType'])
dataLayout = yaml.load(dataLayoutFile)
dataLayoutFile.close()

inputFile = open(parameters['inputFile'])
for line in inputFile:
    info = {}
    for key in dataLayout.keys():
        (startPos, endPos, type) = dataLayout[key]
        info[key] = line[startPos-1:endPos]
        if type == "i":
            info[key] = int(info[key])
        elif type == "f":
            info[key] = float(info[key])

    popUnitInfo.append(info)

#The coordinates of the population units
u = scipy.zeros(POP_UNITS)
v = scipy.zeros(POP_UNITS)
i = 0
for info in popUnitInfo:
    u[i] = info['longitude']
    v[i] = info['latitude']
    i += 1

#The coordinates of the disctricts
a = scipy.zeros(NUM_DISTRICTS)
b = scipy.zeros(NUM_DISTRICTS)
aPrime = scipy.zeros(NUM_DISTRICTS)
bPrime = scipy.zeros(NUM_DISTRICTS)
try:
    k = 0
    for (long, lat) in parameters['initial']:
        a[k] = long
        b[k] = lat
        k += 1
except:
    k = 0
    for i in random.sample(xrange(POP_UNITS), NUM_DISTRICTS):
        a[k] = u[i]
        b[k] = v[i]
        k += 1
```

```
#The populations of the populations units
p = scipy.zeros(POP_UNITS, dtype="int32")
s = scipy.ones(POP_UNITS)
i = 0
for info in popUnitInfo:
    p[i] =  info['population']
    i += 1
POPULATION = sum(p)
MAX_UNIT = max(p)
MIN_UNIT = min(p)
AVG_UNIT = scipy.average(p)
optLargePop = POPULATION/float(NUM_DISTRICTS)


#Make the initial guess for size of the districts
g = scipy.zeros(NUM_DISTRICTS, dtype="int32")
g += POP_UNITS/NUM_DISTRICTS
g[0] += POP_UNITS - (NUM_DISTRICTS*(POP_UNITS/NUM_DISTRICTS))

#Make the row constraints
r = scipy.zeros(POP_UNITS, dtype="int16")
r += 1

#The array that tells us whether or not a
#population unit is in a given district
x = scipy.zeros((POP_UNITS, NUM_DISTRICTS), dtype="int16")

#This matrix tells us the distance from the
#population units to the centers of the districts
#population unit is in a given district
d = scipy.zeros((POP_UNITS, NUM_DISTRICTS), dtype="int32")

lowestRatio = 100000
q = 0

#Step 1: Guess the centroids (a,b)
label .step1
print "Running Transportation Algorithm... ", q

#Step 2: Compute the distances
label .step2
for i in range(POP_UNITS):
    for j in range(NUM_DISTRICTS):
        d[i,j] = int(calcDistance(v[i], u[i], b[j], a[j]))


#Step 3: Using the transportation algorithm
#solve the subproblem for the x_{ij}'s
label .step3

cost  = scipy.zeros((POP_UNITS, NUM_DISTRICTS))
for j in range(NUM_DISTRICTS):
    #cost[:,j] = p*d[:,j]
    cost[:,j] = d[:,j]
```

```
x = transportationAlgorithm(r, g, cost)


#Step 4: Calculate the centroids of the new districts
label .step4
px = scipy.dot(p,x)
sx = scipy.dot(s,x)

#Population Weigted Centers
#aPrime = scipy.dot((u*p),x) / px
#bPrime = scipy.dot((v*p),x) / px

#Unweighted Centers
aPrime = scipy.dot(u,x) / sx
bPrime = scipy.dot(v,x) / sx

#Step 5: Check to see if the new solution is significantly
#        different
label .step5
if max(abs(a-aPrime)) < EPSILON and max(abs(b-bPrime)) < EPSILON:
    #Go to Step 6
    print max(abs(a-aPrime)), max(abs(b-bPrime))
    goto .step6
else:
    #Repeat the above process
    if ( max(abs(a-aPrime)), max(abs(b-bPrime))) == oldMax or refineCount == 20:
        goto .step6

    oldMax = ( max(abs(a-aPrime)), max(abs(b-bPrime)))
    refineCount += 1
    print oldMax

    for j in xrange(NUM_DISTRICTS):
        a[j] = aPrime[j]
        b[j] = bPrime[j]

    goto .step2


#Step 6: Adjust the g's
label .step6
print "Adjusting g's"
refineCount = 0
##a. Find the district with the largest pop
largestDistrict = scipy.argmax(px)
smallestDistrict = scipy.argmin(px)
##b. Calculate the ratio
newRatio = float(px[largestDistrict])/px[smallestDistrict]
##c. Compare
print "Lowest ratio: ", lowestRatio, "newRatio: ", newRatio
if newRatio < lowestRatio:
    g[largestDistrict] -= 1
    g[smallestDistrict] += 1
    lowestRatio = newRatio
    best = districtInfo(x)
```

```
    dumpDistrictInfo(best, parameters["outputFile"])
    q = 0
    print g, sum(g)
    print "Running Transportation Algorithm... ", q
    goto .step2
else:
    q += 1
    if q == H:
        goto .solution

    #Conservative Update
    #g[largestDistrict] -= 1
    #g[smallestDistrict] += 1

    #Aggressive Update
    largeDiff = px[largestDistrict] - optLargePop
    change = max(int(largeDiff/AVG_UNIT),1)
    g[largestDistrict] -= change
    g[smallestDistrict] += change

    lastChanged = [min(largestDistrict, smallestDistrict), max(largestDistrict, smallestDistrict)]

    #Go to step 2
    print g, sum(g)
    print "Running Transportation Algorithm... ", q
    goto .step2

label .solution
print lowestRatio
print best
```

Appendix B: Sample District Breakdown by Tract

The census tracts that make up New York's District 1 are shown in the table below. Each tract within the state is uniquely identified by a 9 digit number; the first three digits correspond to the county the tract resides in, and the final 6 digits are a tract number.

The tract tables for the rest of the districts can be obtained from the authors upon request.

009940000	009940200	009960100	009960200
009960400	009960500	009960600	009960700
009960900	009961000	009961100	009961200
009961400	009961500	009961600	009961700
009962100	013000000	013030100	013030200
013030400	013030500	013030600	013030700
013035100	013035200	013035300	013035400
013035600	013035700	013035800	013035901
013036000	013036100	013036300	013036400
013036600	013036700	013036800	013036900
013037100	013037200	013037300	013037400
013940000	029000000	029000100	029000200
029000400	029000500	029000600	029000700
029000900	029001000	029001100	029001200
029001302	029001401	029001402	029001500
029001700	029001800	029001900	029002000
029002200	029002300	029002400	029002501
029002600	029002701	029002702	029002800
029003100	029006702	029006800	029006900
029007101	029007102	029007201	029007202
029009900	029010804	029010805	029010806
029010901	029010902	029011000	029011100
029011300	029011400	029011500	029011600
029011800	029011901	029011902	029012001
029012003	029012100	029012200	029012300
029012501	029012502	029012600	029012700
029012901	029012902	029013001	029013002
029013102	029013201	029013202	029013300
029013501	029013502	029013600	029013701
029013800	029013900	029014000	029014101
029014204	029014205	029014300	029014400
029015002	029015003	029015101	029015102
029015202	029015301	029015302	029015401
029015501	029015502	029015600	029015700
029015900	029016001	029016002	029016100

References

[Alt98] Micah Altman. *Districting Principles and Democratic Representation*. Doctoral Thesis, California Institute of Technology, 1998.

[HOR72] Robert E. Helbig, Patrick K. Orr, and Robert R. Roediger. Political redistricting by computer. *Communications of the ACM*, 15(8):735–741, 1972.

[LS85] Daniel H. Lowenstein and Jonathan Steinburg. The quest for legislative districting in the public interest: Elusive or illusory? *UCLA Law Review*, October, 1985.

[Mun57] James Munkers. Algorithms for the assignment and transportation problems. *Journal of the Scociety for Industrial and Applied Mathematics*, 5, 1957.

[Pos07] Postgis. Available online, 2007.

[Qua07] Quantum gis. Available online, 2007.

[Sch65] Glendon Schubert. *Reapportionment*. Chales Scribner's Sons, New York, 1965.

[The07] The center for range voting. Available online, 2007.

[Uni00a] United states census bureau 2000 gazetteer files. Available online, U.S. Census Bureau, 2000.

[Uni00b] United states census bureau boundary files. Available online, U.S. Census Bureau, 2000.

[You88] H.P. Young. Measuring the compactness of legislative districts. *Legislative Studies Quarterly*, 13(1), 1988.