BANA 8083: MS-BANA Capstone

District Configuration Analysis through Evolutionary Simulation

Matt Policastro
Professor Peng Wang, First Reader
Professor Michael Magazine, Second Reader

2017-07-22

Abstract

This capstone replicates a methodology for identifying biased electoral district schemes on a smaller scale in a new context. Rather than electoral districts this project examines three of the five Cincinnati Police Department (CPD) districts containing twenty-four of the City of Cincinnati's fifty neighbourhoods. It should be noted that this project does not constitute a rigorous analysis of potentially biased districting practices on the part of the city; instead, this project identifies advantages, trade-offs, and other challenges related to implementation and analysis.

Contents

Introduction

Background

The practice of manipulating district configurations in order to effect specific outcomes—gerrymandering, as it is commonly known in politics and law—is well-established and pernicious. Courts have struggled a great deal with establishing robust tests for bias or malign intent, as legal precedent is too broad to cover the highly specific techniques used to manipulate voting effects. Furthermore, quantitative measures (e.g., compactness, contiguity) have largely failed to make an impact as gerrymandered districts can be manifested in myriad ways (Cho and Liu [2016], p.351–353). In short, specific quantities rarely demonstrate biased districting.

As such, alternative approaches of developing legal and analytical context have been proposed. Full enumeration is in some ways the ideal solution, as it provides perfect and complete context for evaluating an existing plan. All solutions to the problem of assigning districts become visible, and remote or unlikely configurations are consequently apparent. However, as is typically the case of problems with meaningful size, the computational effort is simply infeasible. In the case of geographic district maps, which are essentially set assignment problems, the total number of configurations is Stirling number of the second kind ($S(n, k)$), where n is the number of units and k the number of districts. As Cho and Liu [2016] state:

> Even with a modest number of units, the scale of the unconstrained map-making problem is awesome. If one wanted to divide n=55 units into k=6 districts, the number of possibilities is 8.7×10^{39}, a formidable number. There have been fewer than 10^{18} seconds since the beginning of the universe. Of course, the number becomes significantly smaller with relevant legal constraints, such as contiguity, in place. However, it does not become manageable smaller. The problem remains massive.

In response, there have been ongoing efforts to leverage simulation in exploring the solution space. As computing power has become radically inexpensive over time, the real costs of complex simulations have dropped accordingly. Today, a simulation with fairly simple constraints is a viable strategy for examining the potential solution space for a districting problem. At minimum, Cho and Liu [2016] (p. 358) identified three basic types of constraints for simulating electorally-pertinent districting problems:

1. Each unit must be assigned to exactly one district.
2. The maximum population deviation between districts cannot exceed a given threshold.
3. The units in each district must form a geographically continuous and -contiguous set.

These constraints can be augmented with additional constraints or by objectives such as electoral responsiveness or geographic compactness. With this structure in place, it then becomes straightforward to develop potential solutions through an evolutionary algorithm-driven simulation. The work undertaken by Cho and Liu [2016] and Liu et al. [2016] is one such effort, and resulted in the Parallel Evolutionary Algorithm for Redistricting (PEAR) which leverages asynchronous communication to minimise overlap between evolutionary processes in a supercomputing environment.

Overview

Examining an electoral problem at a state scale would have simply been too large for the scope of this project. PEAR, for example, required tens of thousands of processors (131,072 in the case of the Blue Waters supercomputer) to conduct a simulation in a reasonable time frame (Liu et al. [2016], p. 87). Furthermore, local or regional elections do not have the same data availability or reliability, particularly with respect to geospatial data. However, the methods developed to handle geospatial relationships and contiguity were of particular interest and were chosen as the foundation for the project.

Local police districts were chosen as an alternative subject of simulation and analysis. The City of Cincinnati provides a significant amount of data to the public through their open data portal (Cincinnati [a]), including anonymised crime reports by street address and neighbourhood. And as was stressed in a review of other methods (Cho and Liu [2016], p. 354), populations and districts must remain constant to keep analysis valid. To satisfy that requirement, data from the 2010 Census and statistical neighbourhood approximations for population and geography were paired with Cincinnati Police Department crime data for the same period (Cincinnati [b]).

Finally, as this project required a naïve implementation of the evolutionary algorithm used by Liu et al. [2016], all code and data was housed in a version control system (VCS). Git, a popular VCS, maintains a history of files and their changes over time by tracking "commits" from a project repository's maintainers and users. All code and files used throughout this project—including notes and comments on process—can be found in the author's personal repository at the following URL: github.com/mattpolicastro/BANA-8083-Capstone.

Methods & Process

Sources

Data on crime reports between the beginning of 2008 and the end of 2012 was requested from Cincinnati's open data portal, ignoring any reports without explicit neighbourhood references. Exploratory data analysis found the crime report data to be virtually non-existent before July of 2010 (Fig. 1). To infer the missing crime report counts, a naïve linear regression was chosen over a time series model for the sake of simplicity. The regression was developed on the following three years of crime data (the ratio of crimes committed in the first and second halves of the year, 2011–2013), yielding an estimate of (r) of 0.915. As the simulation was only concerned with annual totals, crime report rates were considered constant and equal within each neighbourhood throughout 2010. The total crime report counts were calculated using Eq. 1.

$$C_{\text{total}} = C_{\text{2nd half}} \times [1 + r] \tag{1}$$

Neighbourhoods, police districts, and population counts were extracted and checked by hand from the city's website (neighbourhood-specific approximations from the 2010 census were only available as PDFs). Two consolidations were made to match the city's official neighbour-

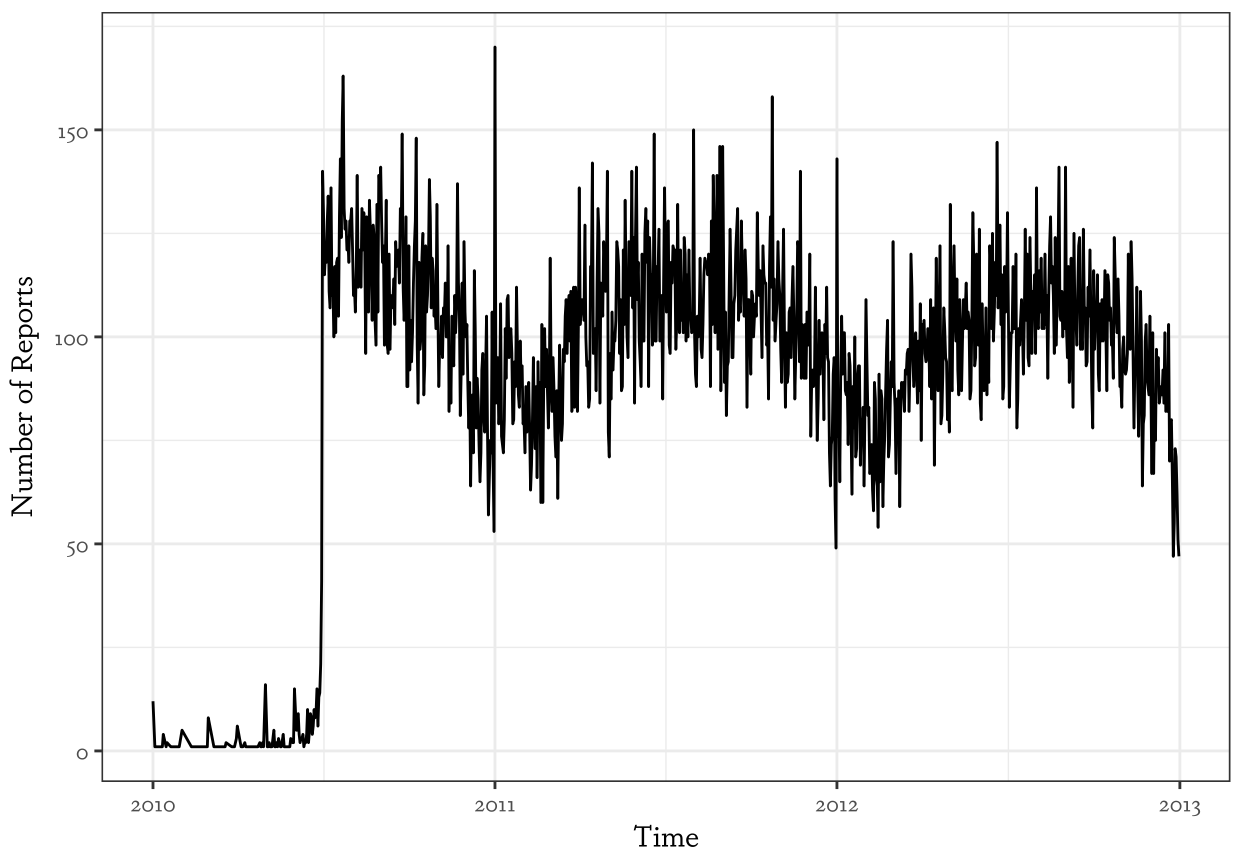


Figure 1: Crime report daily totals

hood designations: Clifton Heights, University Heights, and Fairview were consolidated into one neighbourhood, "CUF"; the Fay Apartments were re-designated as "The Villages of Roll Hill" as part of a public redevelopment project (Simes [2010]). To clean and join the data sets for crime and population counts, mappings between the city's standard nomenclature and those used by the police department were developed.

The other major data component was geospatial data (commonly known as GIS or shapefile data) for the neighbourhoods, which was somewhat challenging to acquire. Geospatial data consists of encoded shapes, lines, attributes, etc. in a coordinate system, allowing for complex geometric calculations and operations. Hamilton County and the city's Enterprise Technology Solutions department provide and update these files at regular intervals (County), but they were found to be extremely unreliable. Thousands of intersecting points prevented accurate calculations of area and perimeter, and the data was ultimately discarded.

An alternative geospatial data set was found and extracted as GeoJSON from the Cincinnati Police Department (CPD) using an open-source Python package (OpenAddresses). (The same name mappings used above were used to clean this dataset's names.) This data was then edited slightly to prevent a false negative space or "hole" in the map, but cursory analysis indicated the maps were a good match for the city's official records in terms of perimeters and areas. Using another open-source Python package (GeoPandas), the data was then transformed to remove three additional negative spaces created by three independent communities surrounded by the city of Cincinnati (Edgemont, Elmwood Place, St. Bernard). Finally, a shape representing the exterior boundaries of the city was formed by subtracting each neighbourhood's geometry from a regional overlay.

To restrict the size of the solution space, the final data sets were filtered to only include the three districts under consideration—districts one, four, and five—giving k=3. Those three districts then contained n=24 neighbourhoods. Districts two and three (the districts furthest east and west in Fig. 2) introduced several problems and were excluded for the following reasons:

- Many of the neighbourhoods on the east side of the city are eccentrically shaped, adding compactness penalties, and introduce "holes" in the area, adding computational complexity.
- Both contain a number of neighbourhoods which only have one or two neighbours and are on the city's edges, meaning they would have little opportunity to change district memberships during the simulation. For example: if Oakley changes district membership, Madisonville must also change membership in order to maintain contiguity, thereby forming something of a super-neighbourhood.
- The number of possible solutions for set assignments is $S(50, 5)$, or 7.4×10^{32}. Removing these two districts brings the solution set size down to $S(24, 3)$, or 1.4×10^{11}. As this project was carried out on consumer-grade computing hardware, the reduction in problem size was meaningful when attempting to generate a representative set.

Preparation

The terminology used with PEAR is also applied here: *units* are indivisible sub-regions, which in this case are the city's neighbourhoods; police districts are *zones*, and are composed of units;

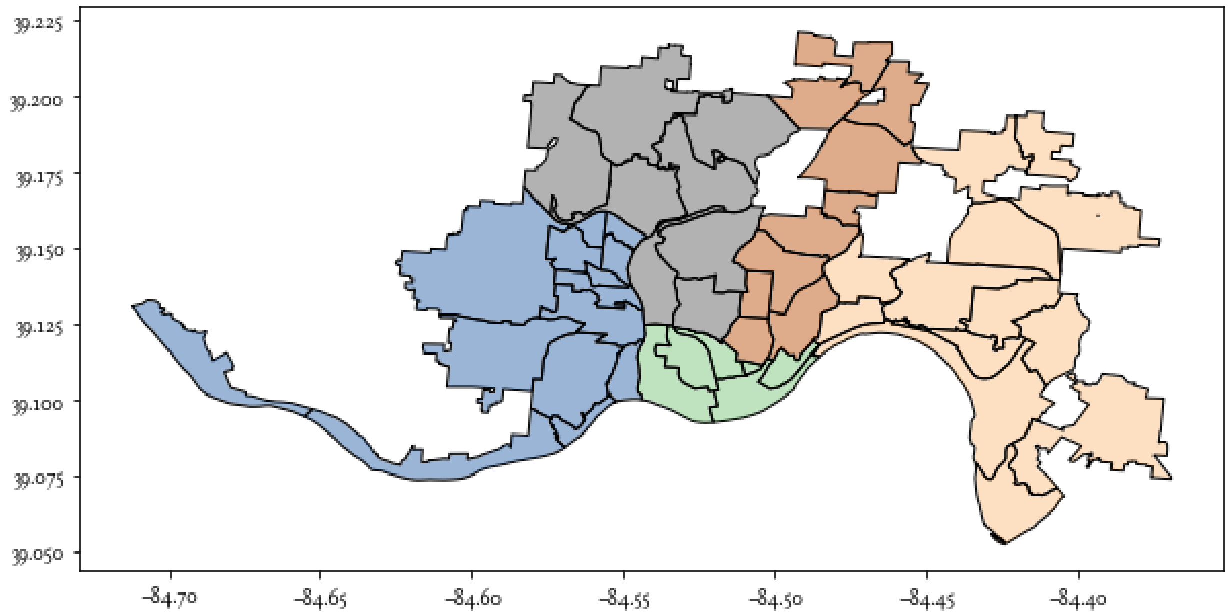


Figure 2: City of Cincinnati neighbourhoods by police district

the city (or at least the districts under consideration) is the *region*, which is the entirety of the study area. Traditional geospatial analysis is undoubtedly powerful, but it is relatively expensive in terms of computational operations. As such, the data sets required transformation into new data structures, per the methods outlined by Liu et al. [2016] (p.82).

The most important primary data structure used in the evolutionary algorithm was the chromosome: an array of values indicating the zone membership of each unit, where the index position indicates the current unit and the value that unit's assigned zone membership. Almost equally important was the adjacency graph. Using another open-source Python package for geospatial analysis (PySAL), a network graph was extracted from the geospatial data detailing the neighbours of each unit in the region. However, rather than "queen" adjacency (units are considered adjacent if they share a corner or a border), this project only considered "rook" adjacency (units are considered adjacent only if they share a border).

Where electoral law has established precedence for queen adjacency, this project was concerned with the physical relation of units as traversable areas, and accordingly opted for shared borders. Another network graph was developed for the zones and their neighbours. This zone adjacency graph was generated and revised in each mutation executed by the algorithm to determine the order of operations and to check for violations of contiguity. (If a zone has only one neighbour in the graph, it has been surrounded by that neighbour and is considered invalid.) Finally, a geo-data frame (a table-like data structure with embedded geographic/geometric information) was used as a canonical reference containing the geometries, crime report counts, and populations for each unit in the simulation.

An initial population for the simulation was generated with PySAL, which includes a `Random_Regions()` method for generating random zone assignments from a provided adjacency graph. One thousand of these random assignments were generated, converted into chromosomes, and checked for violations of contiguity. Of the generated chromosomes, 94.1% were considered valid, and one hundred of those chromosomes were randomly sampled to create the initial population.

Simulation

The open-source framework Distributed Evolutionary Algorithms in Python (DEAP) was used directly and as a design reference in implementing the evolutionary algorithm (Fortin et al. [2012]). Individuals (the chromosomes representing zone memberships) constitute a population, which is grown over a succession of generations in the following process (beginning with the randomly generated initial population):

1. From a starting population, a generation of clones is created (the "offspring").
2. Each individual in the offspring is mutated and evaluated for fitness, yielding a new population (the "mutants").
3. The mutants and their fitnesses are stored in an external data structure and passed to the next generation as the starting population.

This process was repeated for one hundred generations in a semi-parallel fashion. Each population as subdivided, dispatched to multiple child processes using multiprocessing pool, and then re-combined before proceeding to the next step. Fitness checking consisted of three values with a [0, 1] range: compactness, maximum inter-zone population deviation, and maximum inter-zone crime report count deviation. Compactness quantifies the spatial efficiency of a zone, where one is the maximum value (e.g., a circle, which is the most efficient ratio of area to perimeter). The maximum observed deviation, however, makes pairwise comparisons between zones to identify the most imbalanced assignments, where any sufficiently large difference is capped to a value of *1*. The same calculations used by Cho and Liu [2016] (p. 358) for compactness (Eq. 2, where A is the area and P is the perimeter) and maximum deviation (Eq. 3, where P_k is the population of zone k) were used here.

$$C_{IPQ} = \frac{4\pi A}{P^2} \tag{2}$$

$$p = \frac{\max_k(P_k) - \min_k(P_k)}{\sum_{k=1}^{K} P_k} \tag{3}$$

Similarly, the implementation of the mutation operation closely followed the algorithm reference provided in Liu et al. [2016] (p.84). While DEAP provides a wealth of efficient evolutionary operators, none were able to perform contiguity and validity checks on geospatial data. The mutation operator implemented for this project followed a general procedure:

1. Generate a random sequence of source zones, finding a neighbouring destination zone for each.
2. If the source zone contains more than one unit, continue the mutation; else, move to the next source zone.

3. Beginning with a unit on the border between the source and destination zones, select a random number of neighbouring units without selecting the entire source zone.
4. If the "donation" of the selected units does not violate contiguity or validity, shift those units into the destination zone; else, if the mutation has been attempted x times, move to the next zone; else, shift the selected units to the destination zone and begin 2. with the next source zone until that sequence is complete.

This operator only makes one significant departure from the referenced algorithm, which is the arbitrary cut-off of x for indefinitely cycling solutions. The contiguity and validity checks included in this procedure avoid some unnecessary validation checks later on in the simulation algorithm.

It is also worth noting that this simulation did not leverage any selection criteria to create a "hall of fame" of best performers. As this project leveraged approximations for quality rather than more tangible or meaningful criteria (e.g., time to respond to emergency call) and was only run for one hundred generations, all valid results were recorded for the analysis.

Analysis

District Comparison

Of the ten thousand variations generated by the simulation, 84.51% of them were found to be unique variations. As this simulation did not implement inter-process messaging to prevent duplication of efforts, this was expected. The set used for primary analysis was filtered to ignore duplicates to focus on the realm of potential solutions rather than their frequency. Table 1 contains the summary statistics for each of the three fitness criteria. For each fitness criteria, beta distributions with fixed location and scale parameters (0 and 1)were fitted using maximum likelihood estimates for and . To approximate goodness of fit, Kolmogorov-Smirnov (K-S) tests were conducted using those estimated parameters at 95% significance; the null hypothesis being that the observed and theoretical distributions are identical, and the alternative being that the observed distribution is significantly greater or less than the theoretical.

As can be seen in Fig. 3, the real-world CPD districts are not particularly compact, scoring in the 99th percentile of the simulated data. Presumably due to the large spikes seen there, the null hypothesis was soundly rejected with a p-value of $\sim 6.0350 \times 10^{-16}$. The crime count deviation criteria also rejected goodness of fit, albeit in a relatively less severe fashion with a p-value of $\sim 4.9917 \times 10^{-7}$. Fig. 4 shows departures from the fitted distribution, but much less significantly than the previous criteria. That said, the actual CPD districting outperformed many of the simulations, scoring in the 6th percentile. Contrary to the other two fitness criteria, the population scores fell fairly neatly around the real-world maximum deviation (the 44th percentile) and the K-S test rejected the null but at an even less-severe p-value of ~ 0.0017. Fig. 5, showing the simulation against the baseline and fitted distribution, supports that conclusion.

However, another criteria was considered in the post-simulation analysis: the Adjusted Rand Index (ARI). Typically used in cluster analysis, the ARI quantifies agreement between two sets of group assignments through pairwise group membership comparison. The traditional Rand

Index ranges from zero to one—zero meaning no agreement, one meaning full agreement—and the adjusted version accounts for random chance and allows for negative values, indicating the score for the given assignment falls beneath the expected value. Crucially, the ARI is agnostic toward the actual group labels; it is only concerned with pairwise membership across the entire set. (i.e., if plan A assigns units 4 and 17 to the same zone, are they also in the same zone in plan B?)

As was done with the previous criteria, a beta distribution was fitted to the ARI scores and was tested for goodness of fit. As seen in Fig 6, the ARI seemed to stop abruptly around-0.10 in agreement/disagreement, yielding a p-value of $\sim 2.6896 \times 1^{-58}$ in the K-S test. However, as a score of 1 indicates total agreement, the relative lack of agreement between the simulation results and the actual memberships is striking and suggests many of the alternative plans were a significant departure from their real-world counterpart.

Table 1: Descriptive Statistics for Fitness Criteria

Criteria	Actual	Max.	Mean	Median	Min.	Var.
Compactness	0.0823	0.0921	0.0564	0.0567	0.0243	0.0001
Crime Dev.	0.3214	0.9154	0.3417	0.3365	0.0036	0.0243
Pop. Dev.	0.3214	0.9359	0.3521	0.3438	0.0005	0.0264
Adj. Rand Ind.	N/A	0.7331	0.1128	0.0978	-0.0829	0.0124

Cycling

Of the duplicate solutions, four configurations (Fig. 7) occurred at least one hundred times each over the course of the simulation, suggesting cycling was an issue. (The configurations in Fig. 7(a) are identical, only using different district labels for the same geometries). Although some logs were kept during the simulation runtime, detailed data about the relative frequency of these solutions was not available. However, the small number of neighbours and "bottlenecks" to those units may have prevented the mutation operator from "leaping" out of those solutions over successive generations. Furthermore, the spikes seen in Figs. 3–6 might suggest these duplicates were part of a larger trend of "plateaus" in the simulation results.

Conclusions

Using a few simple criteria for evaluating fitness, the simulation approach seems to theoretically satisfactory. However, the current implementation could be improved. Each simulation typically approximately required an hour and twelve minutes. Admittedly, the hardware used was not appropriate for computation-intensive projects (a 1.7 GHz Intel Core i7 processor with 8GB of DDR3 RAM), but 16.49% of the generated plans were duplicated efforts, and each generation required approximately forty-three seconds of running time. Improvements to the evolutionary operators are certainly possible, and fall into three categories:

- **Cycle avoidance.** Thresholds could be adjusted or additional checks implemented to prevent cycling (and the resulting expensive geospatial operations),
- **Parallelisation.** The current implementation is straightforward in dividing and recombining populations, but each step requires all individuals to finish before moving on, meaning those needing additional mutations halt progress for the entire generation. True parallel processing or asynchronous operations would remedy some of this inefficiency.
- **Messaging.** Similarly, a message-passing framework to update each process' knowledge of best solutions and potentially unexplored areas (thereby preventing duplicated efforts) is possible, but a significant implementation undertaking.

That said, the general methodology appears to be sound. Each of the fitness criteria demonstrated reasonable spread across the simulation, and the ARI scores indicate significant departures from the current districting plan being used by CPD. Although there is casual evidence of plateaus and duplicated efforts, experimentation with mutation thresholds may be sufficient to address that issue. And, perhaps most importantly, designing more complex fitness criteria or selection operators to develop the analysis is rather straightforward meaning this implementation forms a solid foundation for improvement and expansion.

References

Wendy K. Tam Cho and Yan Y. Liu. Toward a talismanic redistricting tool: A computational method for identifying extreme redistricting plans. *Election Law Journal*, 15(4):351–366, 2016. doi: http://doi.org/10.1089/elj.2016.0384.

Cincinnati. Cincinnati open data portal | open data, a. URL https://data.cincinnati-oh.gov/.

Cincinnati. Census & demographics, b. URL http://www.cincinnati-oh.gov/planning/reports-data/census-demographics/.

Hamilton County. Cincinnati area geographic information system (cagis). URL http://cagismaps.hamilton-co.org/cagisportal/mapdata/.

CPD. Cpd neighborhoods. URL https://services.arcgis.com/JyZag7oO4NteHGiq/arcgis/rest/services/CPD_Neighborhoods/FeatureServer.

Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

GeoPandas. Geopandas. URL https://github.com/geopandas/geopandas.

Git. URL https://git-scm.com/.

Yan Y. Liu, Wendy K. Tam Cho, and Shaowen Wang. Pear: a massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and Evolutionary Computation*, 30:78–92, 2016. doi: http://doi.org/10.1016/j.swevo.2016.04.004.

OpenAddresses. esri-dump. URL https://github.com/openaddresses/pyesridump.

PySAL. Pysal. URL https://github.com/pysal/pysal.

Randy A. Simes. Fay apartments to be renamed the villages of roll hill, undergo $36m renovation, 2010. URL http://www.urbancincy.com/2010/10/fay-apartments-to-be-renamed-the-villages-of-roll-hill-undergo-36m-renovation.

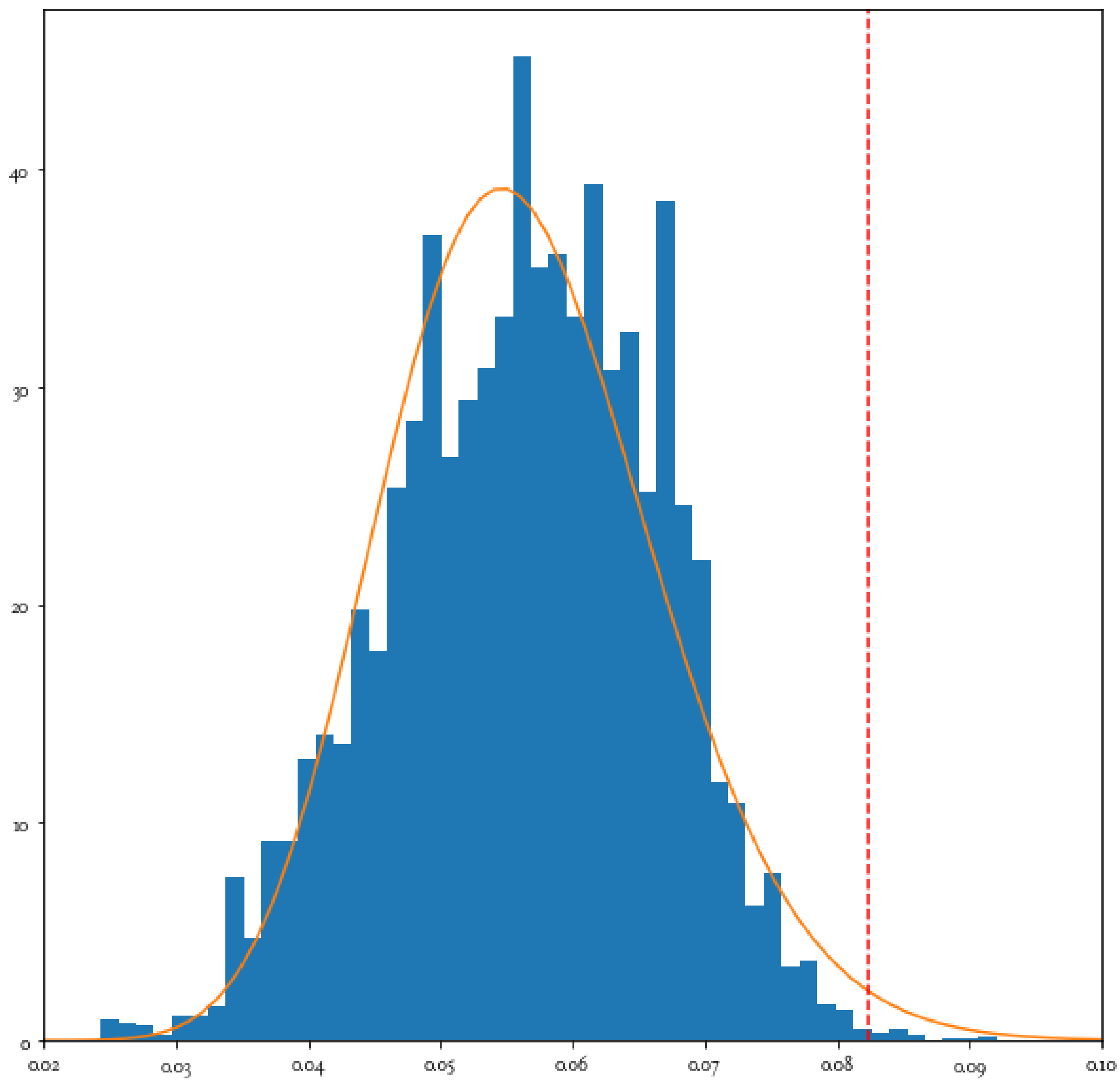


Figure 3: Simulated Compactness scores vs. actual districting score

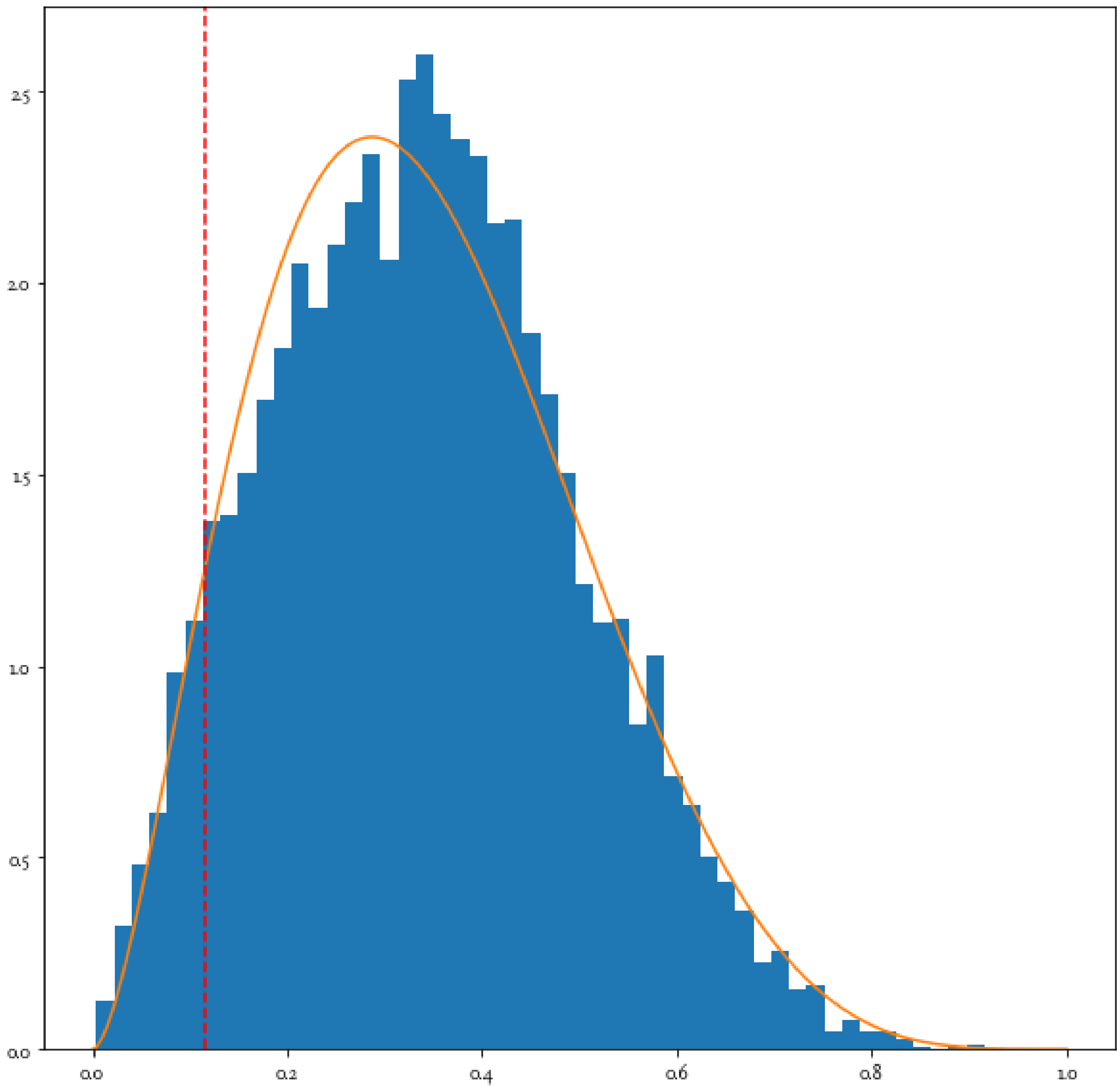


Figure 4: Simulated crime scores vs. actual districting score

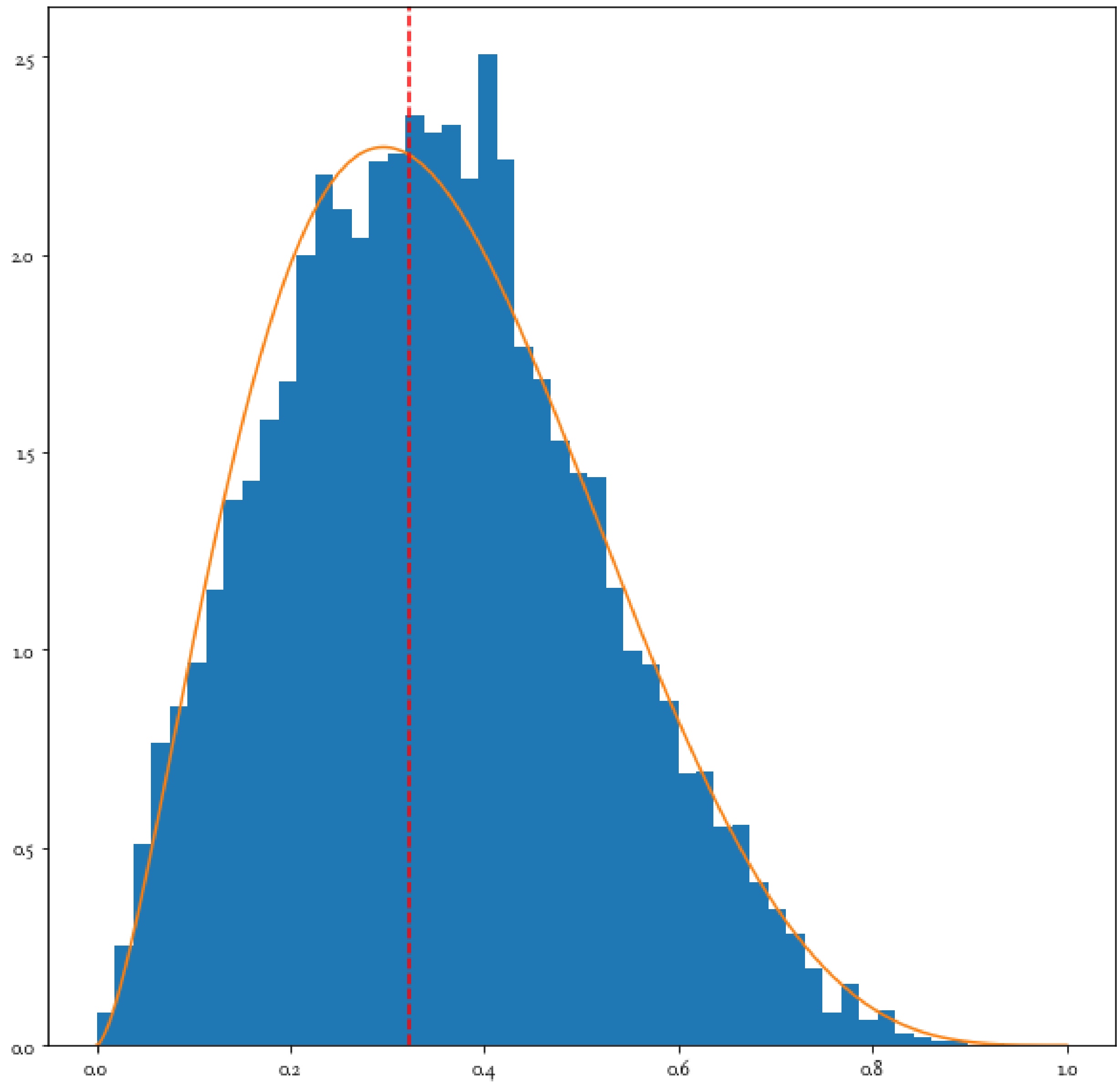


Figure 5: Simulated population scores vs. actual districting score

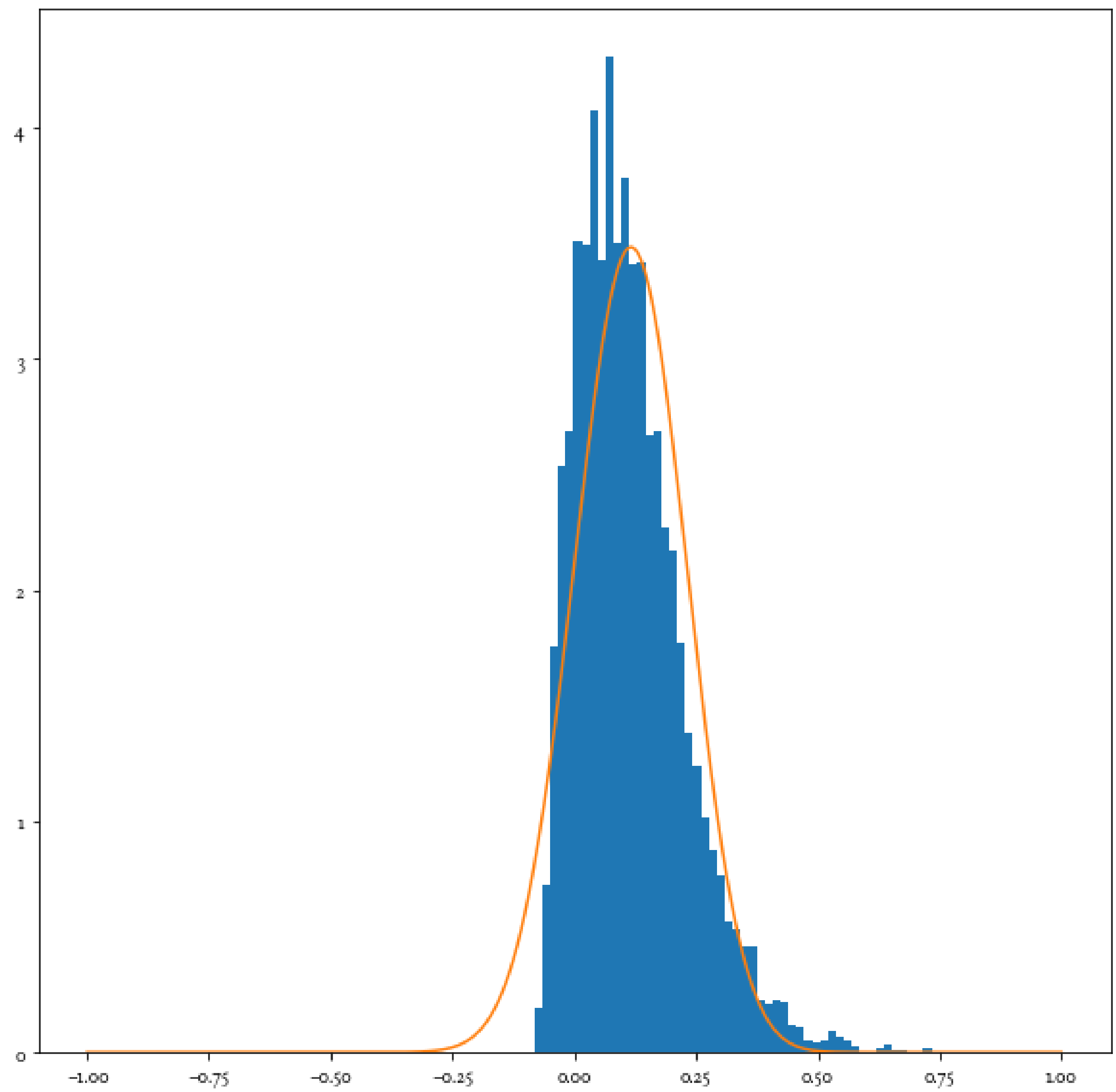
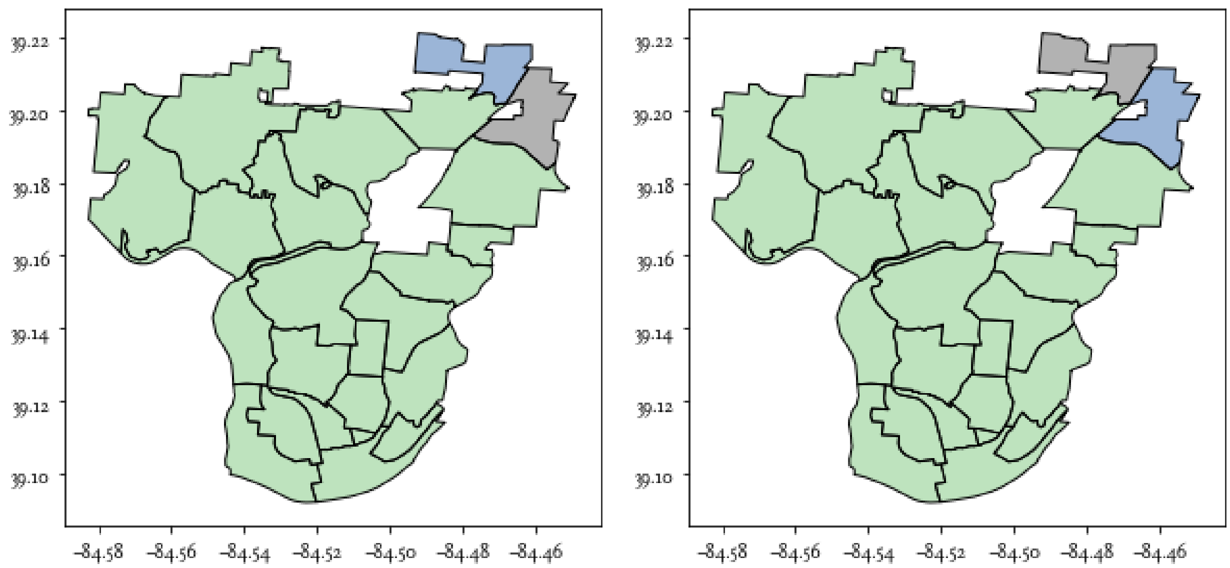
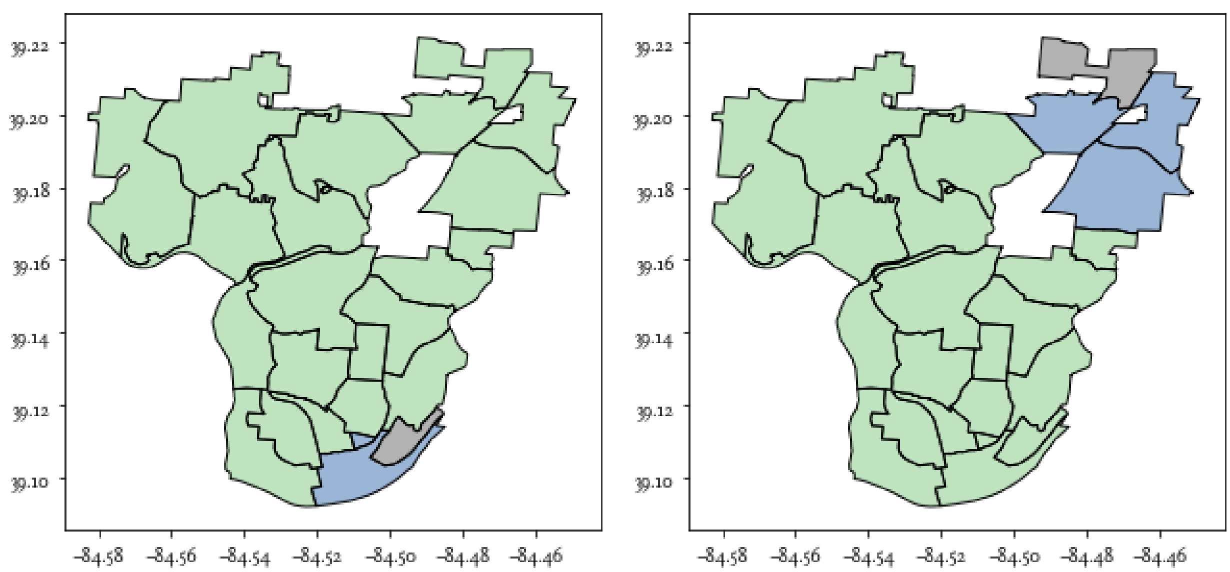


Figure 6: Adjusted Rand Indices for simulations against the actual



(a)



(b)

Figure 7: Recurring district configurations