Using Crime Statistics to Recommend Police Precinct Locations

Chuck Cartledge
chuck@clc-ent.com

John Sinues
walkandride@hotmail.com

November 5, 2017

Contents

List of Algorithms

List of Figures

1 Introduction

Cities and governments are responsible for the safety and security of their citizens. They are also responsible to use their money responsibly. From a city planner's perspective, one question that combines both of these needs is: where should new police precints be located? This simply stated problem is complicated because some police precincts are already built and operational, so the placement of new precincts has to keep the existing ones in mind.

In this short report, we look at police incident reports and existing precinct locations for the City of Virginia Beach, VA to derive optimal locations for a small number of movable/new precincts.[1] The basic ideas about how to compute the optimal location are applicable to a large number of areas, including: rezoning of school districts, size and shape of congressional precincts, and unconstrained placement of manufacuturing facilites.

In this investigation, we will look at the optimal placement of police precincts based on historical crime data in the city of Virginia Beach, VA. We will present our results, our algorithm, and identify specific areas where the implementation could be improved.

2 Problem definition

Police indicent reports (also sometimes call crime reports) are a tabulation of the cimes within an area. Government is responsible for the safety and security of its citizens. Combining these to semi-related data and requirements, means that a government should place police precints (or places from which peace keepers can come from) at locations that are closest to the reported crimes.

At first glance, it would appear that determining the optimal location of the precincts is a geo-spatial clustering problem. But it isn't. Classical spatial clustering attempts to build clusters from spatial data points based on: the closeness of adjacent points, and a minimum number of points that defines a cluster. Once these clusters are identified, then a centroid can be computed. Clusters in a spatial context can have very complex and irregular shapes.

Identfying the optimal centroid location depends on the minimizing the distance between data points and the centroid. When locating these optimal locations, the problem is complicated by the fact that some of the locations are fixed and cannot move. These fixed locations affect the location of non-fixed precincts.

The problem is where to locate new precincts, especially when there are already precincts built.

[1]The idea for this report came from converstations between the authors after a Big Data Data Analysis boot camp sponsered by Old Dominion University, Norfolk, VA.

3 Algorithms

From an algorthmic perspective, the program is very simple, just a set of nested loops (see Algorithm 1 on the following page). Updating the precinct locations is easy as well (see Algorithm 2 on page 4). Adding a unused precinct to the system is also easy (see Algorithm 3 on page 4).

4 Data Sources

The goal of the program is to find optimal placement of movable police precincts (sometimes called kiosks) in a "field" fixed precincts and historical crime location data.

4.1 Crime statistics

For Virginia Beach, VA, historical crime data (aka, police incident reports) is available online as a comma separated value (CSV) file from `https://data.vbgov.com/Public-Safety/Police-Incident-Reports/iqkq-gr5p`. The csv file has a header line identifying the various data fields:

1. Police Case Number
2. Date Reported
3. Date Occurred
4. Date Found
5. Offense Code
6. Offense Description
7. Subdivision
8. Zone ID
9. Case Status
10. Location

Location data (latitude and longitude in decimal degrees, +North and +East) were used to locate the crime in the data "field." Crime reports ranged from January 2015 through September 2017 as 350,261 physical records, and approximately 113,664 logical records. In the data file, there was one header record, and three physical records per logical records. The location records were constrained to a bounding box between -76 and -75 degrees longitude and 36 to 37.5 degrees latitude, essentially the bounds of the city of Virginia Beach. Approximately 3,089, or 0.9% of the crime locations were outside this bounding box.

4.2 Precinct locations

The street address for each of the current police precincts is available online from `https://www.vbgov.com/government/departments/police/opsdiv/Pages/pct.aspx`. A street address can be converted to a latitude and longitude online using this URL `https://www.`

```
Data: PrecinctData ← initial data
Data: CrimeData ← inital data
begin
    MaxIterations ← 100
    CrimeData ← precinct #1
    for p ∈ PrecinctData do
        PrecinctData ← updatedLocationBasedOnCrimeAssignedData
        DoWork ← TRUE
        IterationCounter ← 0
        while ((IterationCounter ≤ MaxIterations)AND(DoWork == TRUE)) do
            IterationCounter ← IterationCounter + 1
            DoWork ← FALSE
            for i ∈ CrimeData do
                for j ∈ p do
                    Dist ← DistanceFromCrimeToJ
                    if Dist < CrimeDataDistance then
                        CrimeDataDistance ← Dist
                        CrimeDataPrecinct ← j
                        DoWork ← TRUE
                    end
                end
            end
            SumOfDistances ← ∑ CrimeDataDistance
            NunberOfUsedPrecincts ← uniqueCrimeDataPrecinct
            if ((SumOfDistances ≠
             OldSumOfDistances)OR(NunberOfUsedPrecincts ≠ p)) then
                DoWork ← TRUE
                if NunberOfUsedPrecincts ≠ p then
                    PrecinctData ← addMissingPrecinct()
                end
            end
            if DoWork == TRUE then
                PrecinctData ← updatedLocationBasedOnCrimeAssignedData
            end
        end
    end
end
```

Algorithm 1: Overall system algorithm.

```
begin
    for p ∈ PrecinctData do
        if crimes assigned to p then
            if p is moveable then
                p location data ← median crime location data
            end
        end
    end
end
return Updated PrecinctData
```

Algorithm 2: Updating precinct location based on assigned crimes.

```
begin
    ListOfPrecinctsUsed ← CrimeData
    for p ∈ PrecinctData do
        if p ∈ ListOfPrecinctsUsed then
            Find p with greatest crime distance σ
            pMax ← p
        end
    end
    if PrecinctData[pMax] is movable then
        Find crime location that is furthest from pMax
        Update PrecinctData with location midway between pMax and crime location
    end
end
return Updated PrecinctData
```

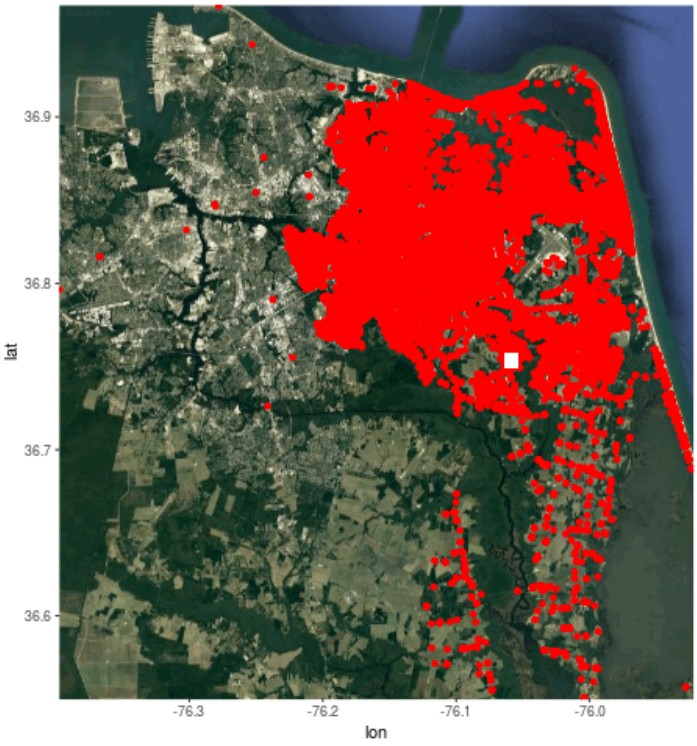
Algorithm 3: Adding a missing precinct.

latlong.net/convert-address-to-lat-long.html.[2] Locations of fixed precincts will not be adjusted or moved by the software. Movable precincts are shifted around the data field in an effort to achieve the optimal solution.

5 Results

Data was collected after a precinct was added to the system, and the system reached a current optimal solution. The following figures show the solution based on the number of fixed and movable precincts, boxplots of the distances from each crime to its assigned precinct, and a plot showing the rate of each system towards closure.

[2]The small number of fixed precincts did not justify the time and effort to use a RESTfull, or HTTP API to convert them pragmatically.



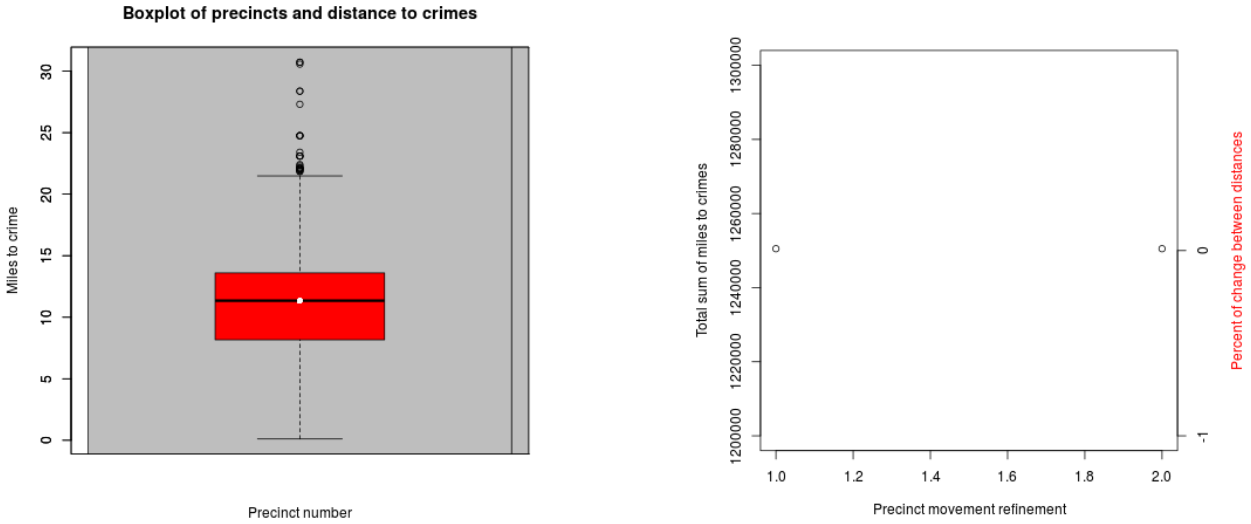


Figure 1: Satellite imagery showing the location of 1 precinct and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.

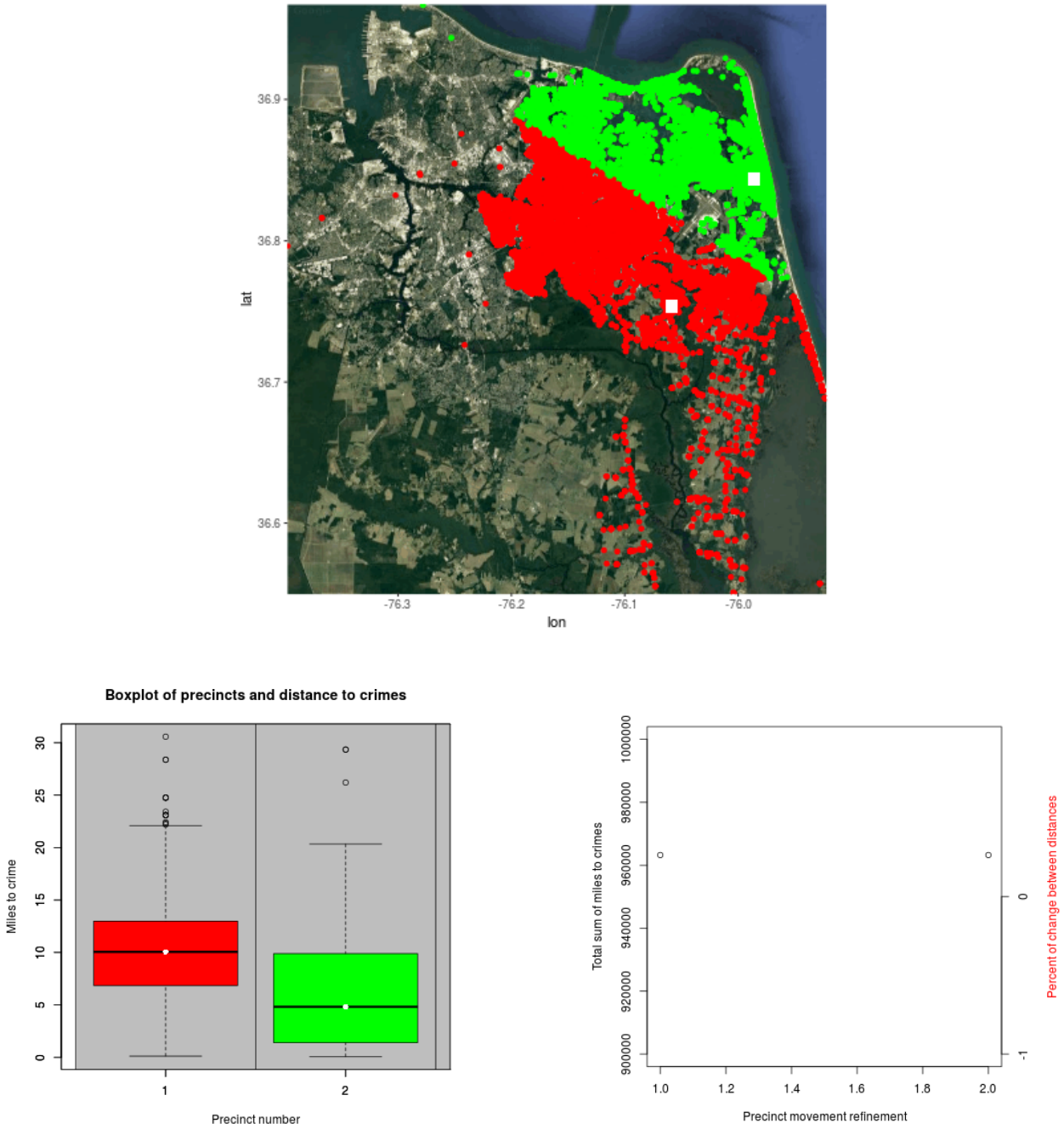
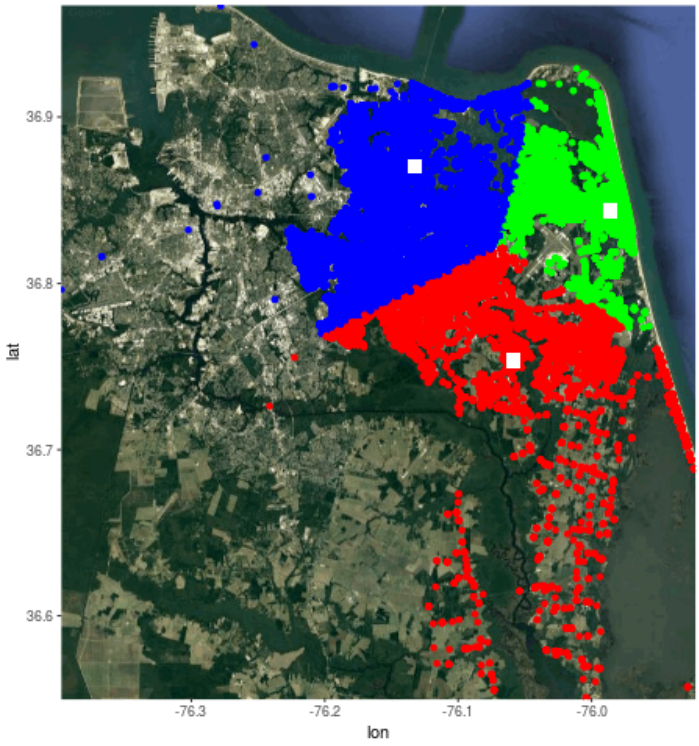


Figure 2: Satellite imagery showing the location of 2 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



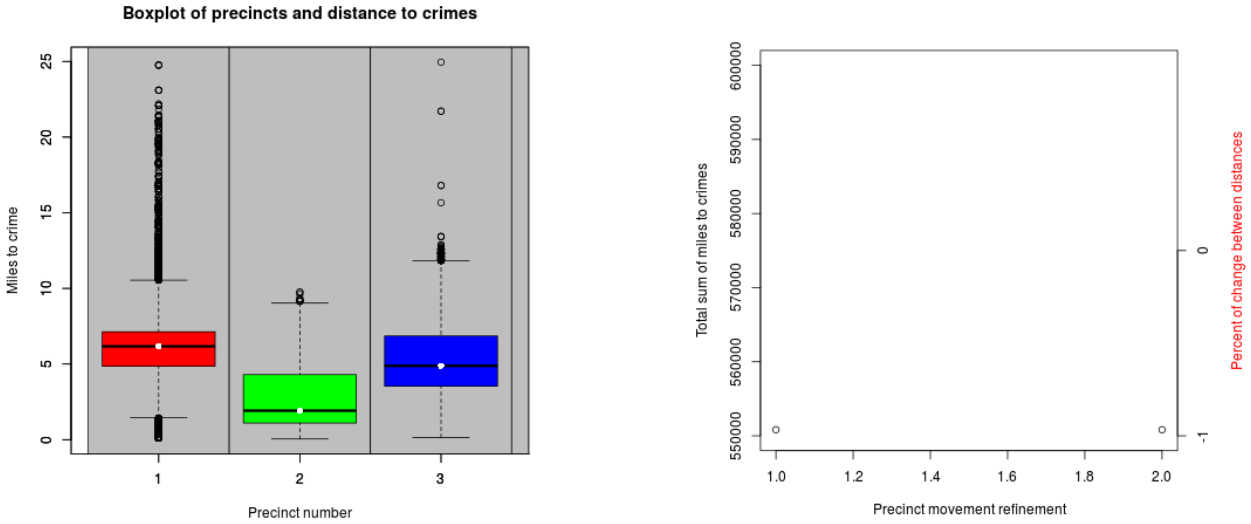
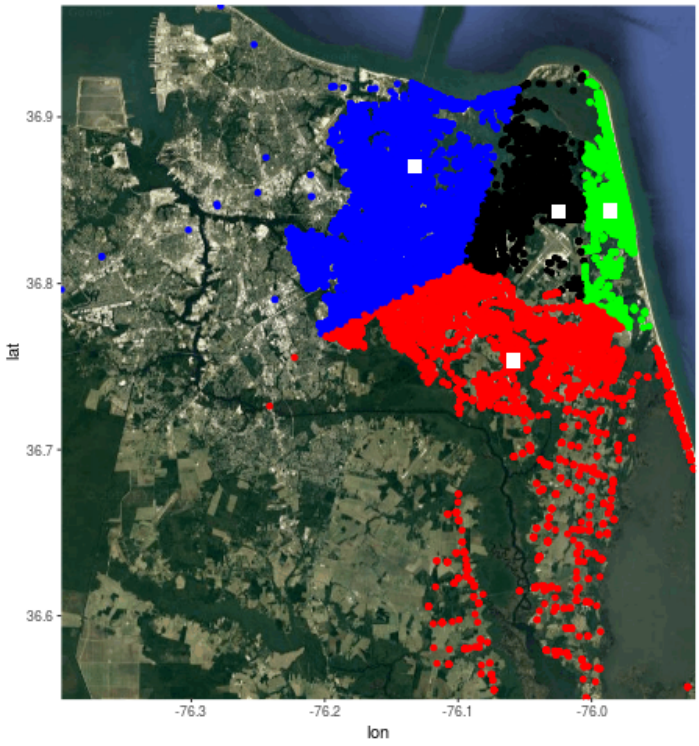


Figure 3: Satellite imagery showing the location of 3 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



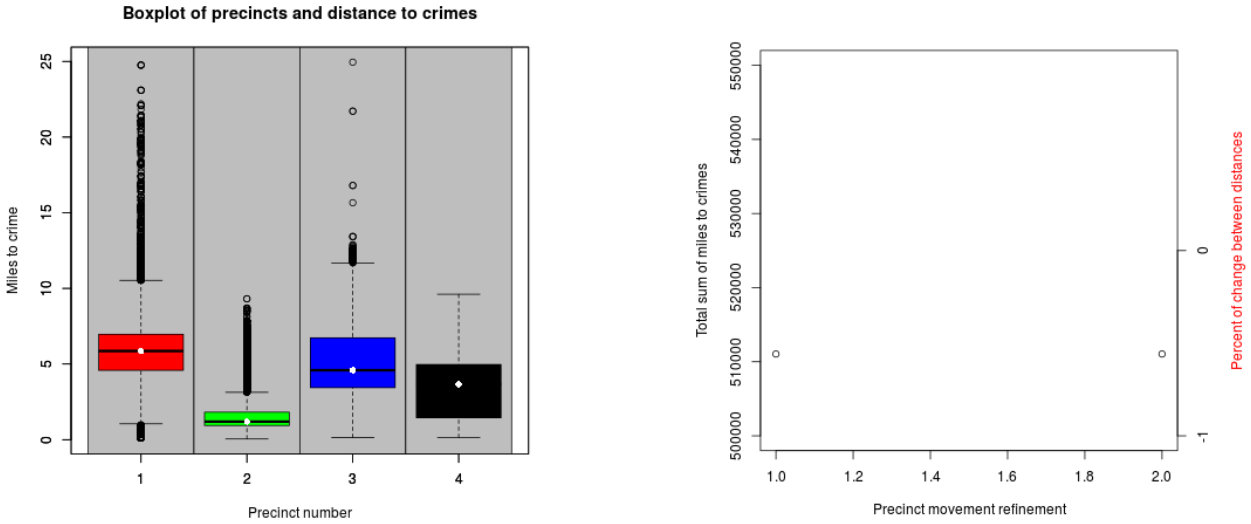
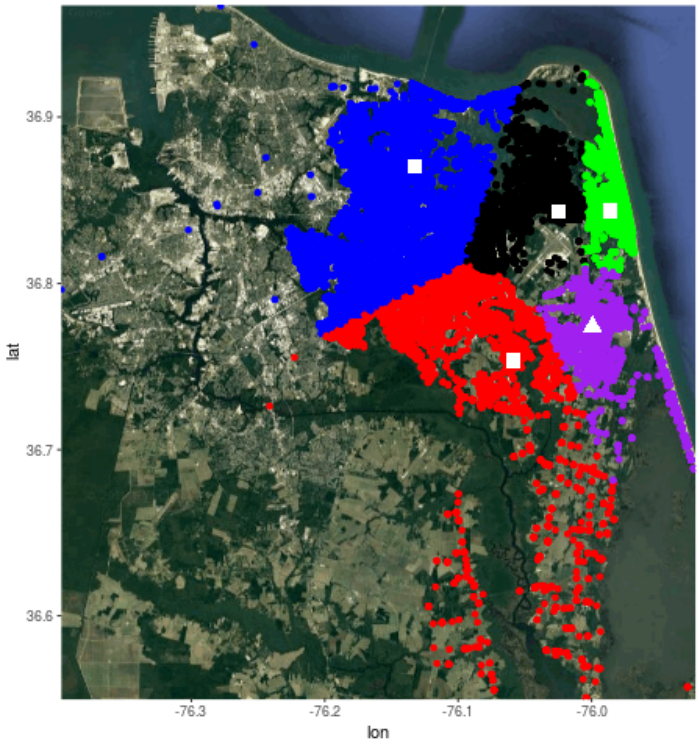


Figure 4: Satellite imagery showing the location of 4 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



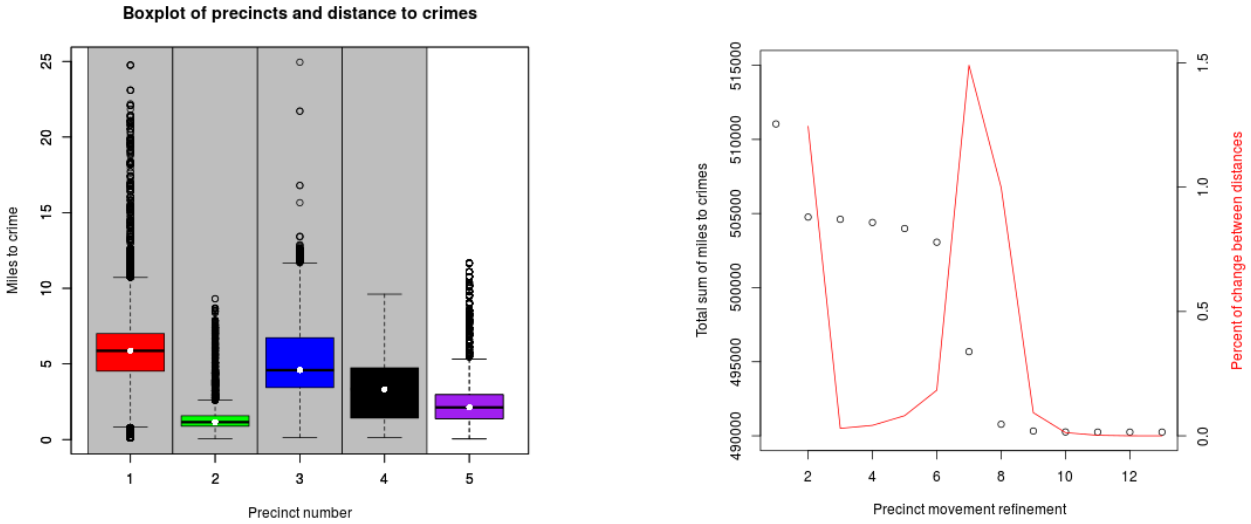
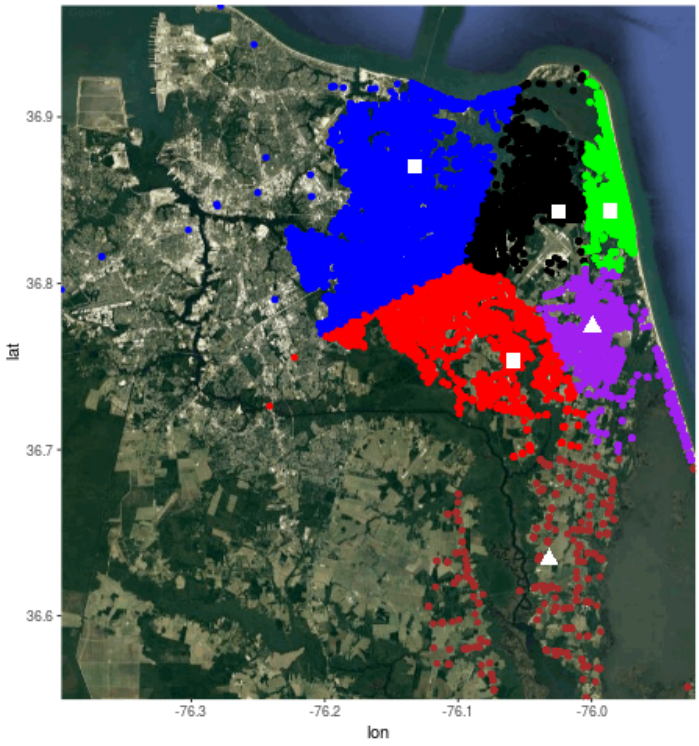


Figure 5: Satellite imagery showing the location of 5 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



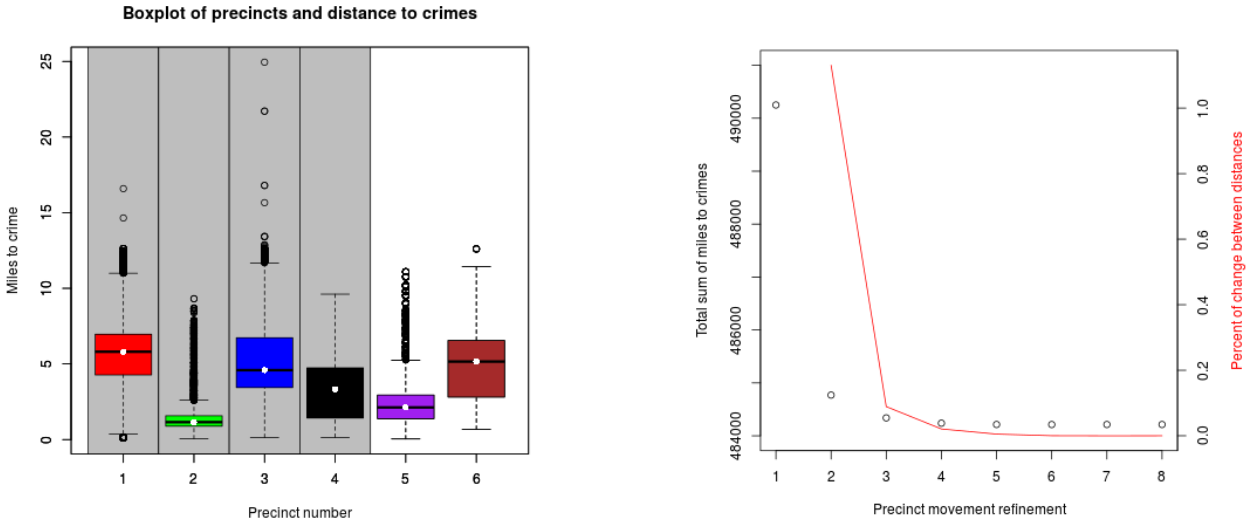
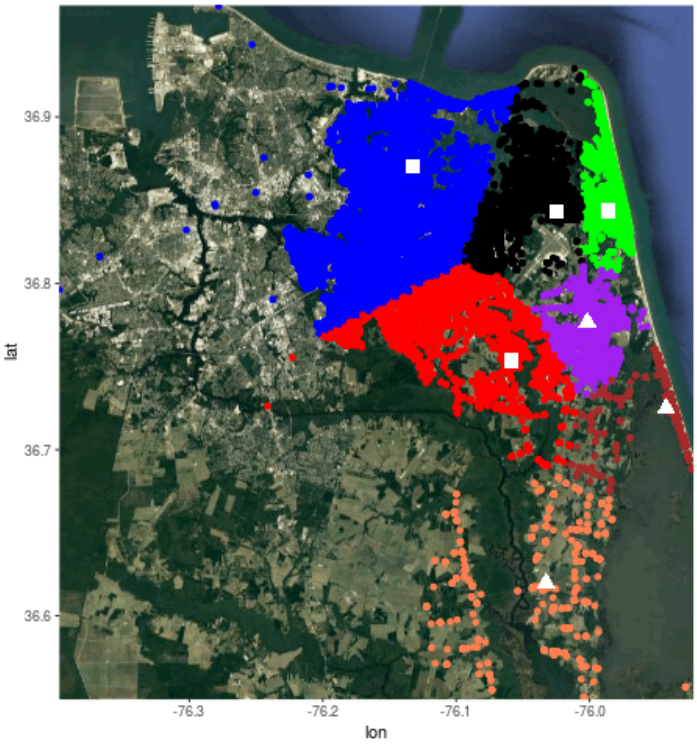


Figure 6: Satellite imagery showing the location of 6 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



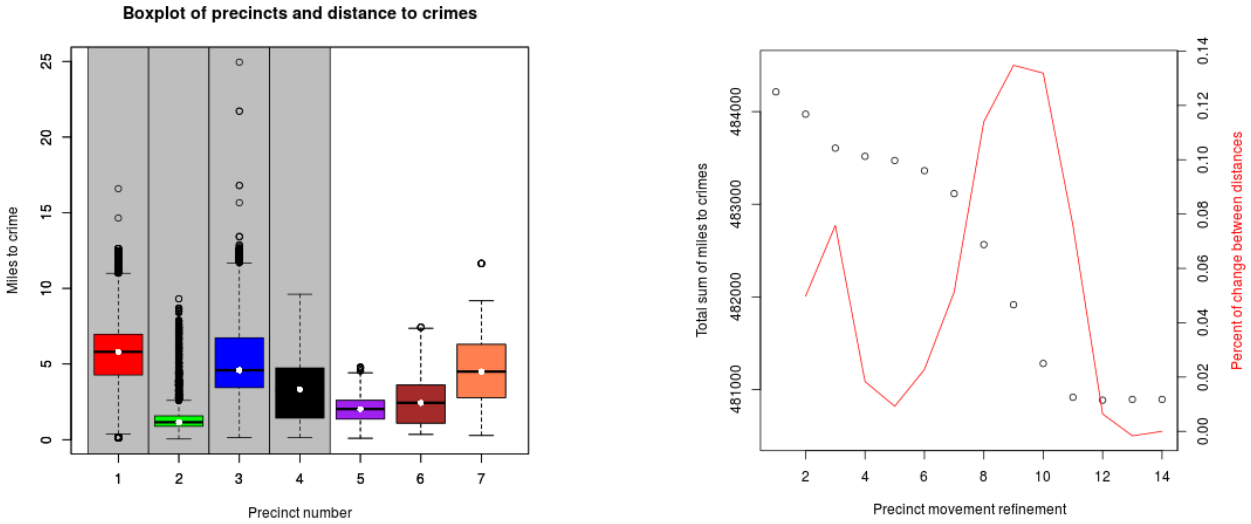
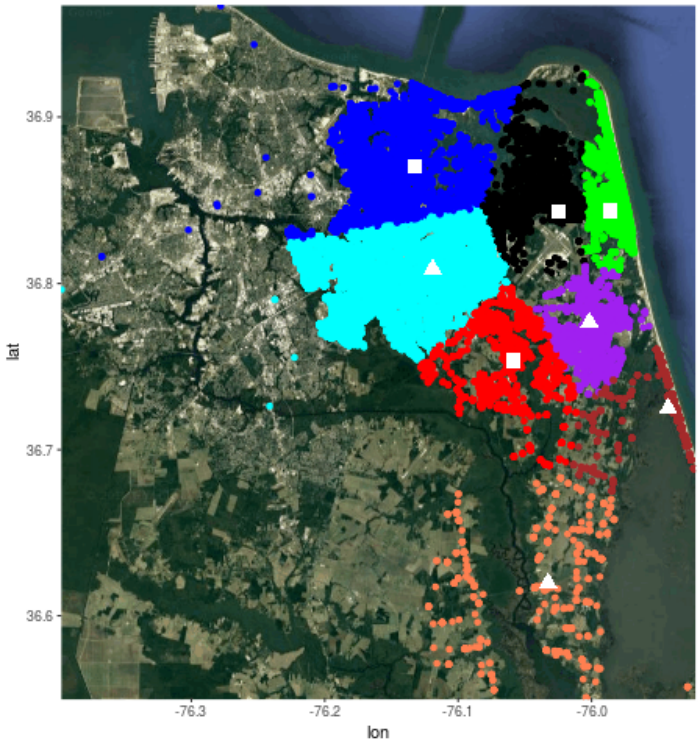


Figure 7: Satellite imagery showing the location of 7 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



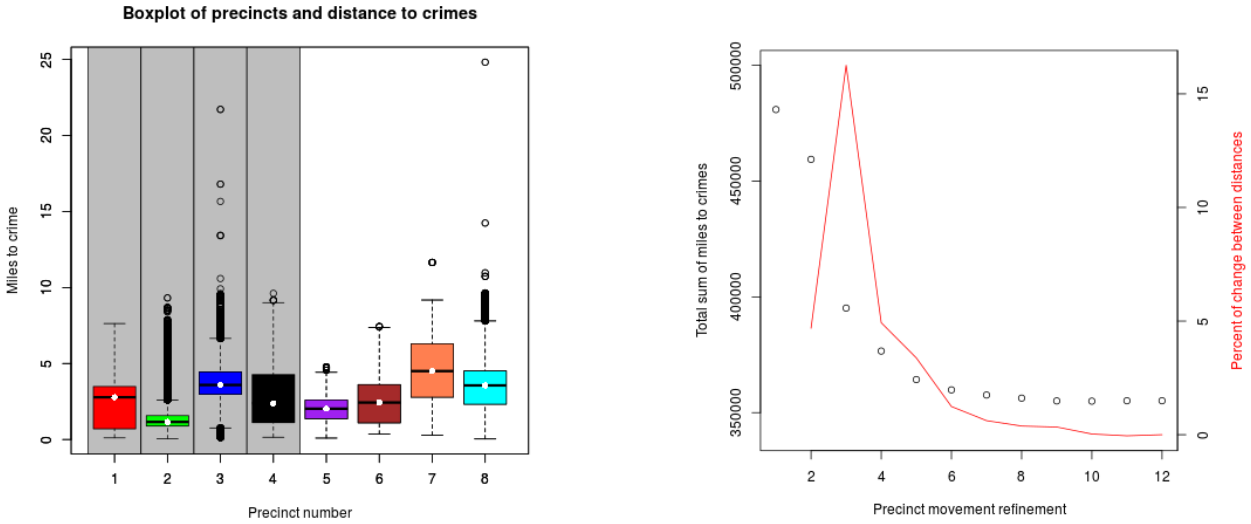
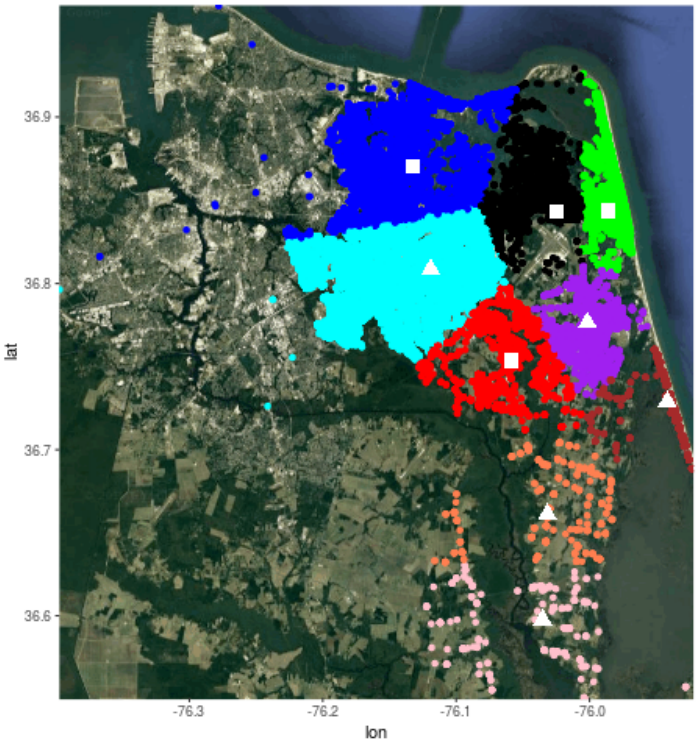


Figure 8: Satellite imagery showing the location of 8 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



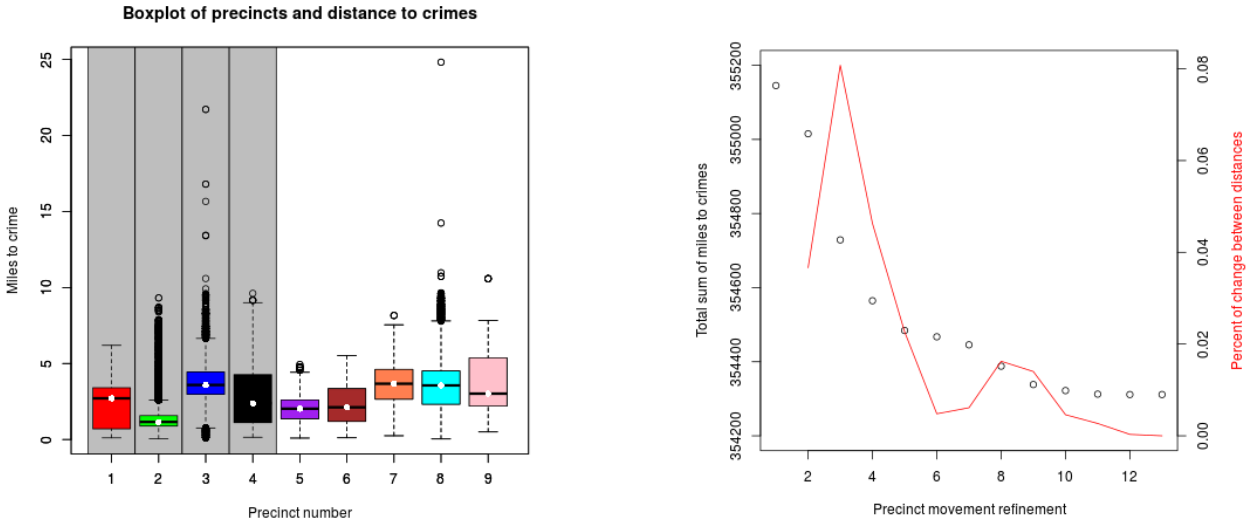
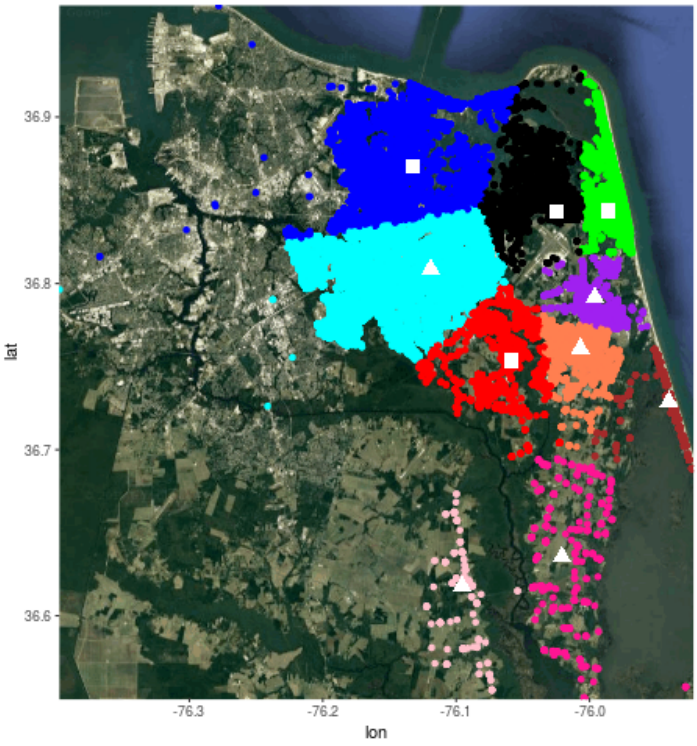


Figure 9: Satellite imagery showing the location of 9 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



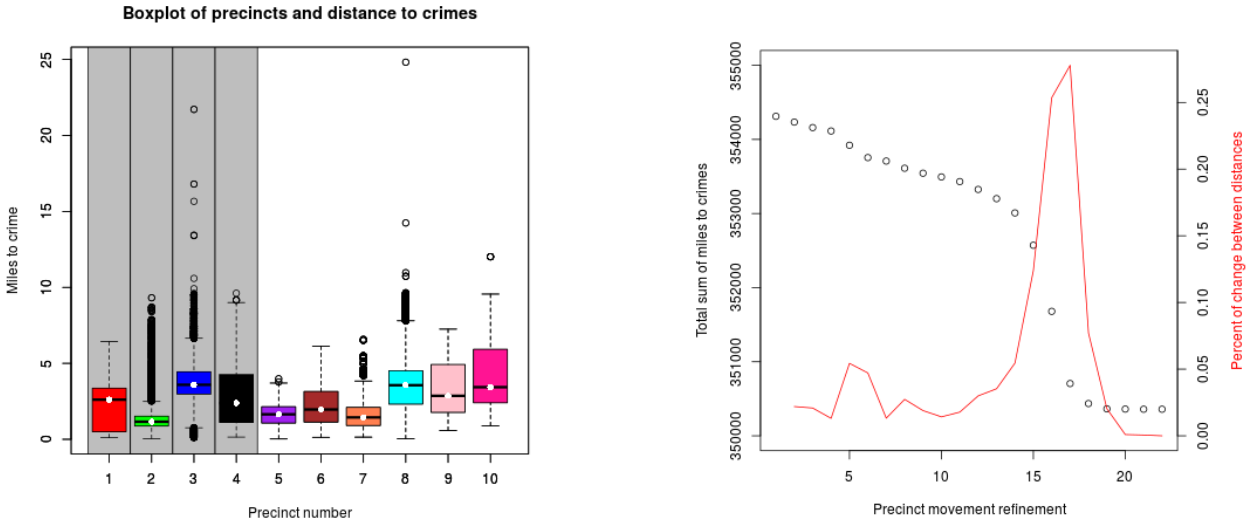
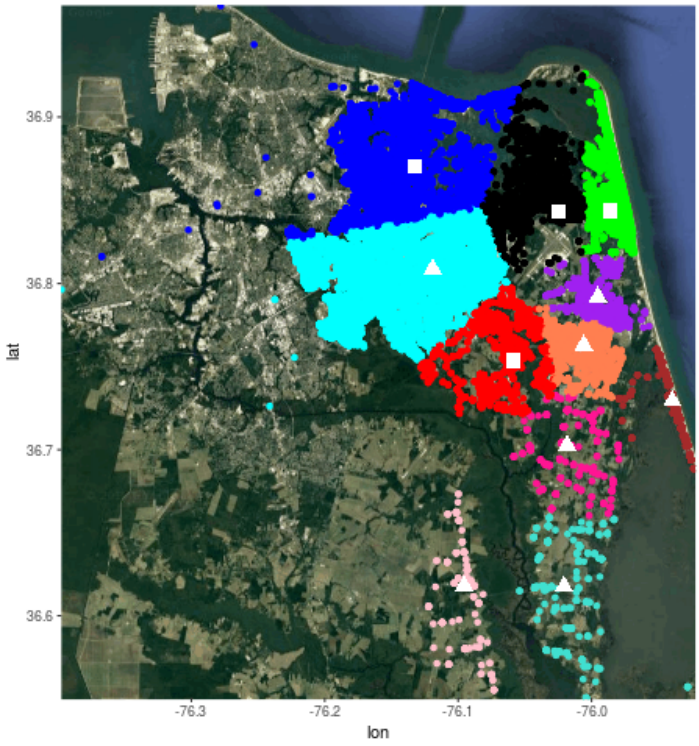


Figure 10: Satellite imagery showing the location of 10 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



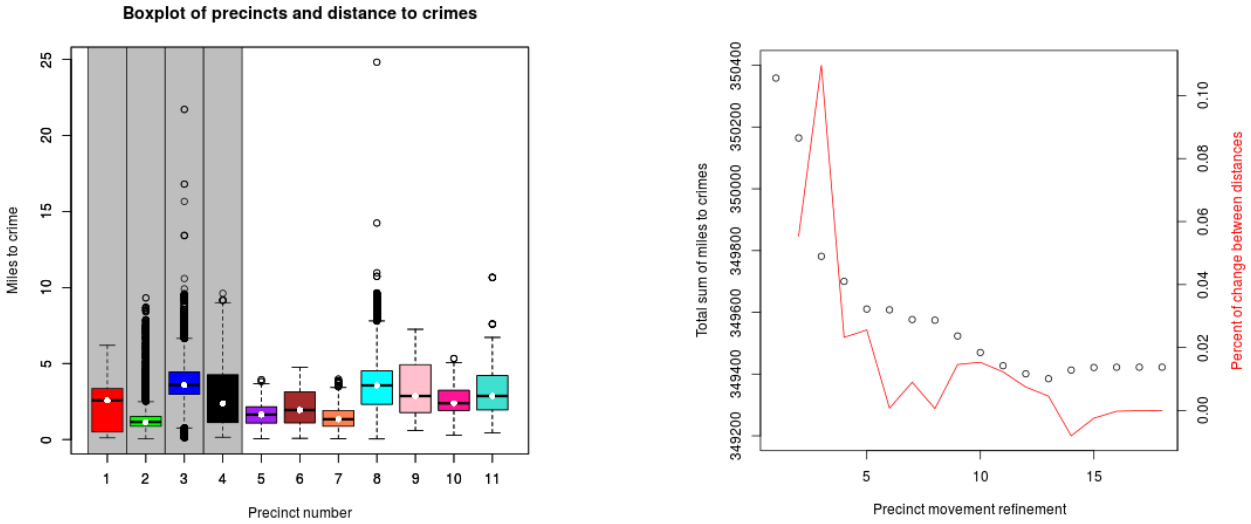
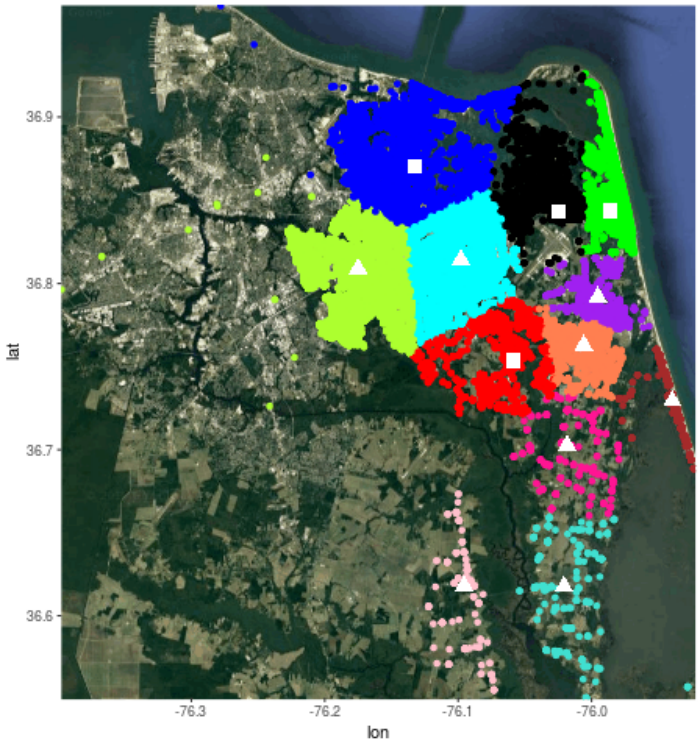


Figure 11: Satellite imagery showing the location of 11 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.



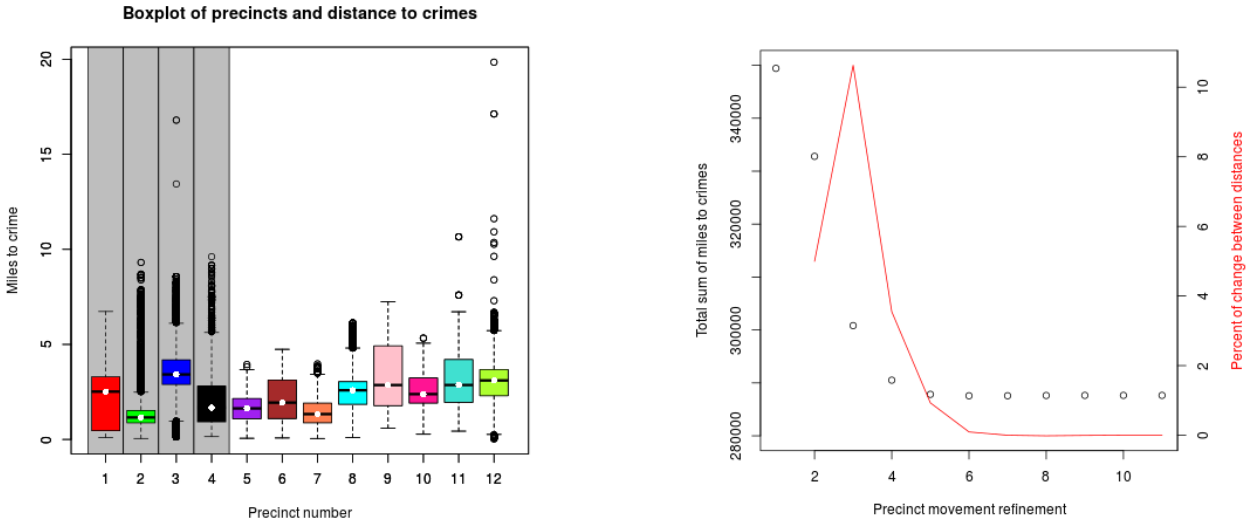


Figure 12: Satellite imagery showing the location of 12 precincts and assigned crimes. Fixed precincts are shown as white squares in the satellite imagery. Movable precints are shown as white triangles. The "greyed" precincts in the boxplots are fixed (aka, immovable). The system iterates towards stability, and is shown as decreasigng sum of distances, and percentage change per iteration.

The performance of the system (as measured by sum of distances from assigned precincts) is best illustrated by (see Figure 13 on the next page).

While the system is attempting to obtain an optimal solution; precincts that are movable, move. The movements of precinct 5 (the first movable precinct) are presented (see Figure 14 on page 20).

6 Areas for improvement

As the program stands; it finds an optimal position for the relocatable precincts, where optimal means the sum of the distances between a crime location and it "assigned" precinct location is minimized across all crimes and precincts. There are several implied attributes in this minimization, including:

1. The correct distance measurement is being used. At its surface, how to measure distance can seem a trivial problem, but it is very interesting because because to measure the distance between two (or more) things, a common set of reference points has to be established. The "distance" between two things, can also be called their "dissimilarity"[1]. In the current R script, a haversine (or great circle) distance is used because both the crime and precinct data have latitude and longitude values. Alternatively, a Euclidean formula could be used, because in general the range distances is small and the associated errors are probably acceptable. A possibly better distance measurement would be the Manhattan distance because it takes into account streets and directionality, vice being "as the crow flies." Haversine was chosen because it was felt that making HTTP API calls for each crime and each precinct would take too long.

2. Any crime can be assigned to any precinct. It is easy to envision that crimes of different classifications could be assigned to precincts with expertise in handling that classification. This "specialization" would very likely affect the location of the movable precincts.

3. It is assumed that the precinct can exist at its optimal location. The computed location does not address whether or not a precinct could be constructed at the optimal location. If the optimal location is in a lake, or river, the placement program has no knowledge of those types of restrictions.

4. The optimal location is computed based on the positional crime data used. Each of the crime reports has a time and date component. The current application does not use these data. It is possible that the optimal location could change over time. This is an aspect of the data that has not been addressed.

5. The optimal location is based on crime locations, but looking at the crime data on a geographic plot, seems to indicate that there is more crime in densely populated

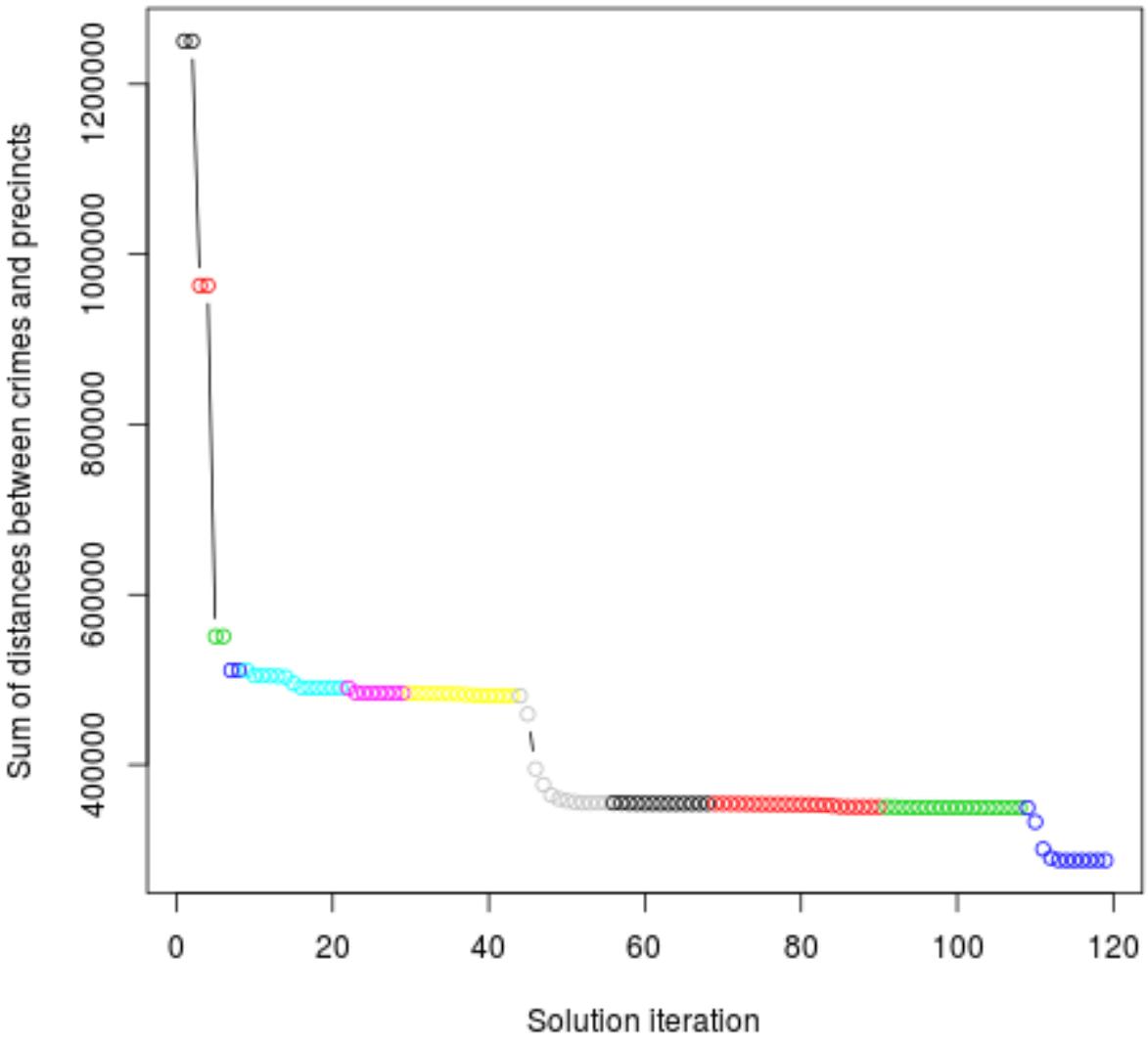


Figure 13: Overall system optimization performance. Each color change means that a new precinct has been added to the system. Each marker of the same color show how the system has reduced the sum of distances for that number of precincts. In general, when a new precinct is added, there is a relatively significant improvement. As more precincts are added, it takes longer for the system to optimize. The width of the plateaus is interesting as well. The first plateau is from 8 to about 42, and the second is from about 44 to 110. It is almost as if they are doubling in width. Meaning there are wide ranges of program iteration where minimal system improvement is obtained.

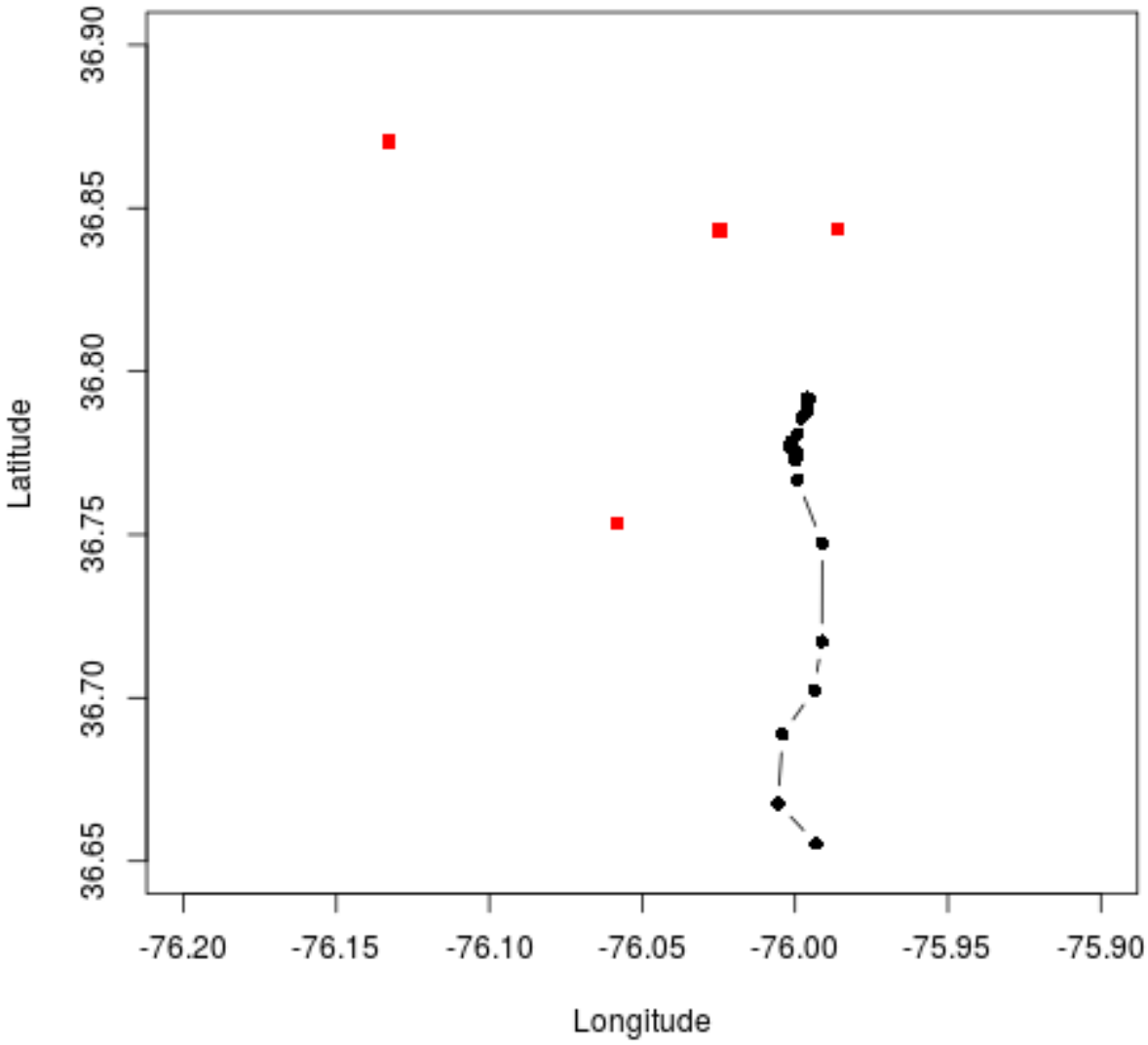


Figure 14: Movement of precinct 5. The four fixed precinct locations are shown as red squares. Precinct 5 (in black circles) starts in the south and "migrates" towards the North until it becomes nearly fixed.

areas. It may be that there is a relatively constant crime rate, and therefore precincts should be placed in the more densely populated areas because there will be more crimes nearby.

6. Using a more refined way to detect when a solution has been detected. Currently the search for a solution terminates when either one of two conditions are met. Either the maximum number of iterations has been reached, or the current sum of distances is EXACTLY the same as the previous sum of distances. The maximum number of iterations threshold is an escape if the system becomes unstable for some reason. A change in how the sum of distances is computed, or the change in the sum is detected could terminate the iterations earlier. One possible approach is to use an exponential average of the sum of distances to detect stability, where the current sum is compared to the exponentially average sum to detect 0 change. Another approach is to establish a percentage change between the previous value and the current value that would signify stability. These ideas come from looking at the long tail of the closure rate plot.

7. During the crime to precinct allocation process, occasionally all crimes assigned to a precinct will be reassigned to a nearer precinct. When this happens, the precinct is in a sense "orphaned." Because the goal is to place a preset number of precincts, when a precinct is orphaned, another attempt is made to place it in a "good" location. the standard deviation of all crimes from their assigned precincts is computed, and the precinct with the highest standard deviation is identified. Using the location of this precinct, the crime that is furthest away is identified (if there is more than one crime at the same distance, the first one is used). A point midway between the precinct with the highest standard deviation, and that precinct's furthest crime is the candidate location for the unused precinct. An improvement could be to use a midpoint of all existing precincts (a sort of spring force position), the argument that this new location would have the greatest benefit to the entire system vice just one precinct. Another variation would be to use the precinct with the highest standard deviation, its furthest crime, and the location of the nearest precinct that it was not assigned to as the spring locations. This approach would provide the highest benefit to a portion of the system. The current system does not handle the situation where precincts are "orphaned" more than once.

8. Currently the program focuses on placing the precincts at locations that minimizes the distances between all crimes and all precincts. An alternative goal could be to minimize the cost of the system. A precinct could have a fixed cost, and each assigned crime could be assigned a cost (representing personnel, material, and other direct and indirect expenses). The goal could then become to minimize the cost of the total system.

9. Precinct locations/movements could be monitored to determine system stability. If the

changes in precinct location across the board were to fall below some threshold (say 0.002 degrees, or approximately 0.1 miles), then it could be assumed that the system had reached a nearly stable condition.

10. The bulk of the current program's execution time is spent computing the distance between each crime and each precinct. The crime location data is invariant, while at least some of the precincts' location data will vary. A possible way to improve performance would be to have a collection of servers that would return crime to precinct assignments when given an updated set of precinct locations. In theory, if the precinct data were spread across 4 servers, then execution time would be reduced by approximately a factor of two thirds (allowing for overhead time to marshal and unmarshal results). This might be a really big boost as the number of crimes, or precincts increases.

11. The current implementation injects new precincts one at a time. A variation would be to inject more than one at a time and see how the system performs. Perhaps all precincts would be used, and there wouldn't be any "orphans."

12. Consolidate police reports. Cursory examination of some of the reports seems to indicate that multiple reports were done at the same location simultaneously. Therefore they could be considered to be the same incident, vice separate incidents. The threshold for consolidation should consider time within some small tolerance, location within some small tolerance, and possibly the type of crime.

13. Remove context switches to improve execution time. It appears that R uses a "pass by value" model when passing arguments to functions. This is supported by the fact that changes made in a function to data it was passed, are not visible in the calling function. The marshalling of arguments to pass to a function has to take time, this time could be eliminated if there was only one context. Rewriting the script to eliminate functions would decrease its maintainability, but should reduce execution time.

14. Allow the program to be restarted after successful conclusion. Currently most of the program's "state" data is persisted to support post run analysis. The program could be restructured to take advantage of this data so that runs with higher number of precincts could use previous data as a starting point.

7 Conclusion

We downloaded over 100,000 real-life police incident reports from the City of Virginia Beach, VA, USA covering the period from January 2015 to mid-August 2017. We wrote, tested and debugged an R script used to find the optimal location for new police precinct headquarters taking into account the four current precinct locations and up to eight new precinct locations. Program execution was not fast. It took 168,284 seconds, or 46.75 hours for a complete run.

For our purposes, an optimal location for a precinct was determined when the sum of the distances from all precincts to all crimes assigned to that precinct remained constant. The addition of more precincts provided incremental improvements that were relatively constant for long periods of time, and number of precincts. There were step-wise improvements in the system behavior, but the execution time was so long as to make exploration of these steps impractical.

The system was able to reduce the distance from one precinct to all crimes from approximately 1,250,000 to less than 390,000 and 12 precincts.

8 References

References

[1] Michel Marie Deza and Elena Deza, *Encyclopedia of Distances*, Encyclopedia of Distances, Springer, 2009, pp. 1–583.

A R script files

The following files are part of this report:

1. `Police_Incident_Reports.csv` crime data for Virginia Beach, VA for 2015 - 2017. Data was downloaded from:
   `https://data.vbgov.com/Public-Safety/Police-Incident-Reports/iqkq-gr5p`
2. `policeBadge02.jpg` a police badge meaning that the precinct's location is computed
3. `policeBadge01.jpg` a police badge meaning that the precinct's location is fixed
4. `library.R` an R script file that is included by other R files
5. `explorer.R` an R script file used to "explore" crime and precinct data without analysis "overhead" that is useful for setting latitude and longitude limits on crime data, and for identifiying the correct zoom factor for satellite maps
6. `assignCrimes03.R` the R script used to create images and analyze data for this report