AA228 Final Project Report - Police Officer Distribution

Somrita Banerjee[*], Keiko Nagami [†], Daniel Thomlinson[‡]

This project analyzes police department allocation of personnel throughout the city of San Francisco in order to address crime. This problem will be modeled as a Markov Decision Process (MDP) with states, actions, and rewards. Crime report data from 2018 will be used to find crime levels throughout the city for a given hour in any given day. Using this data, an optimal policy of police officer movement for each hour in a day will be computed, that best addresses the expected crime throughout the city.

I. Nomenclature

C	=	Crime value matrix
$C_{lat,long,h}$	=	Crime value for a given latitude, longitude and hour
MDP	=	Markov Decision Process
P	=	Police presence matrix
S	=	Matrix relating expected crime and police presence = $P - C$

II. Introduction

The distribution of police officers throughout a city should be based on the probability of crime occurring in a given area and time.This project looks at finding an optimal policy of moving San Francisco's police force based on previous crime data. Several factors come into play when modeling this problem. For one, we must take into account how police officers will move from hour to hour. With this movement, we must consider how to decide which police officers should move where for a given hour depending on the crime levels that are present for that hour. Taking these considerations into account, it is sensible to model this problem as an MDP with states, actions, and rewards.

III. Related Work

One recent paper has approached a very similar problem to the one shown here. Their focus was in determining the optimal solution to the multiple units assignment problem. Some notable differences with our approach are that they determined a ϵ-optimal horizon approximation solution, while our approach attempts to find the best policy from a subset of all policies, which is unlikely to be optimal. They incorporate actions and reward functions across each adjacent cell their agents move. Our approach has the police officers effectively "jump" from one district to another. Their rewards correspond to how well-patrolled the district is, whereas rewards here correspond to the cost of moving officers. Lastly, their approach quite a bit of randomized exploration into the patrol routes, while the policies here were formulated carefully and inflexibly from past crime data. [1]

Other studies on this problem investigated the affect of police presence on criminal activity. From these studies, it was noted that an increased probability of a criminal being convicted reduced the probability of a person committing a crime. This assumes that the criminal is risk averse, and that their utility of committing an unlawful act would be reduced. [2] In order to simplify the problem, this study was applied to our model by making the assumption that an increased police presence will reduce crime.

The decision of which police officers to move when, in reality, will depend on numerous factors, however two large factors are the response time and availability of police force. [3] In a study evaluating optimal dispatch decisions for response units, these two factors were largely focused on, and will therefore be the metrics for optimization in our project.

[*]Stanford University
[†]Stanford University
[‡]Stanford University

Crime analysis studies have implemented various methods of crime weighting in order to compare different crimes to one another. The dataset used in this project contains crimes ranging anywhere from property damage to homicide. In evaluating the severity of crime in a given area, these two types of crime should not be weighted equally. Standardized crime reports based on crime type categorize crimes as Part I and Part II offenses as seen in the table below. In determining weighting factors for crime labels in the dataset, Part I crimes were weighted by a factor of 4 and Part II crimes weighted by a factor of 2. Any crime present in the dataset that was not in either category had a factor of 1. This factored weighting system was modeled after The Cincinnati Police Department's factoring approach. [4]

Part I	Part II	
Aggravated assault	Simple assault	Liquor offenses
Forcible rape	Curfew offenses	Offenses against the family
Murder	Forgery and counterfeiting	Prostitution
Robbery	Driving under the influence	Public drunkenness
Arson	Drug offenses	Runaways
Burglary	Disorderly conduct	Sex offenses
Larceny-theft	Embezzlement	Stolen property
Motor vehicle theft	Loitering	Vandalism
	Fraud	Vagrancy
	Gambling	Weapons offenses

IV. Problem Formulation

A. Model

The data will be modeled with states, actions, and a reward function. In order to develop a policy that will effectively place police officers throughout the city of San Francisco, we will conduct policy evaluation, and preemptively prune the number of possible policies.

1. Data

The crime dataset contains longitude, latitude, type of crime, time, and day of recorded incidences. This dataset spans from January 1, 2018 until October 25, 2018, and contains 137,877 data points or crime incidences. The dates of each of these crimes were neglected, as this project focuses on finding the hourly policy for an average day. The city was binned into a 40 x 40 grid space, each grid cell having the same dimensions. The longitude and latitude ranges of -122.516 to -122.363 and 37.83 and 37.708 were each split into 40 segments. The data points from the dataset were then binned into these grid cells. The resulting simplified data was represented as a 3-Dimensional array of size 40 x 40 x 24 to represent the latitude bin, longitude bin, and hour. The values in the array correspond to a crime weight for that location and time index, which depends both on the number of crime occurrences and the weight of each occurrence. This crime level value will be referenced as $C_{lat,long,h}$.

2. States

The states of our model will be defined by a Cartesian grid over the city of San Francisco. Each cell in this grid will represent a discrete state. This state space will be similar to grid world problems analyzed in class.[5] In this project, we will have a 3-Dimensional crime matrix from our data, C, and we will have a police matrix, P, for the fraction of the police force that is currently at each grid location. The first hour will begin with a uniform P matrix, with each value corresponding to $0.000625 = \frac{1}{1600 Gridcells}$. The S matrix will be evaluated as follows:

$$S = P - C$$

This way, our state matrix S will encode a relationship between crime level and police presence. The values in the S matrix will be interpreted as the amount of unattended crime in each grid cell location.

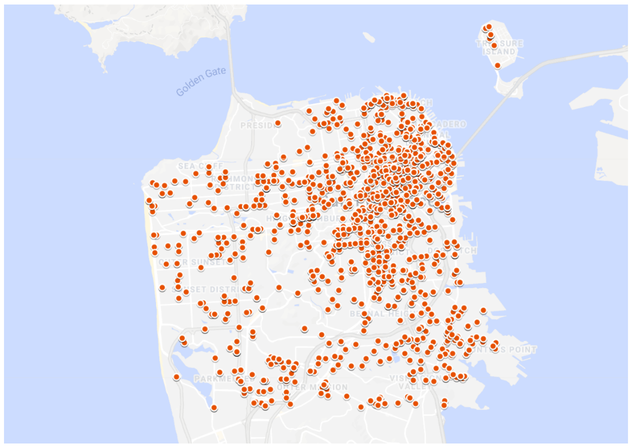

Fig. 1 Map of all crime incidences from 2018 dataset

3. Actions

The actions of our model will be the movement of a fraction of police officers to another cell. This consists of a quantity, a departure cell, and an arrival cell.

Because there would be a very large amount of possible next actions to take, this action space will be pruned by evaluating the "neediness" of grid locations, the surplus of police force presence in surrounding grid cells, and distances between needy and surplus grids cells.

4. Rewards

The reward associated with each action will be computed, and the action that produces the maximum reward will be chosen and added to the policy for that hour. This reward function produces a negative value, which encodes the cost of moving police officers. The reward function used in this project is as follows:

$$R(s, a) = -d * p_{move},$$

where d is the total distance between a needy state and a surplus state, and p_{move} is the fraction of police officers to be moved. In using this reward function, we are able to see that moving a large fraction of police officers from far away to address a needy state will be unfavorable to moving the same fraction of police officers from a grid cell that is very close.

V. Methodology

The implementation of finding the actions that maximize reward are explained through each of the helper functions and the main code that was used in producing our optimal policies. The full code can be obtained at the following link: https://github.com/somritabanerjee/AA228PoliceOfficersOnDemand.

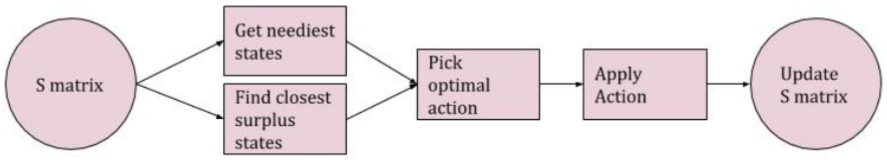

Fig. 2 Methodology of implementing actions and updating S matrix

In this project, we evaluated policies for different crime thresholds, as outlined in our test matrix below:

Table 1   Test matrix

Threshold Values	-0.001	-0.005	-0.01	-0.03

A. Neediest States

This helper function finds the states within our S matrix that are considered to be the neediest. This function had an input of the S matrix as well as a threshold that indicated the value under which states would be considered needy. This threshold value was tuned, and sets of policies were evaluated for different thresholds. A higher threshold would allow for higher levels of potential crime, where as a lower, more negative value would better restrict the amount of potential crime. After finding which states were below this threshold value, their location values and unattended crime value were stored in a queue. As actions are taken, actions fall out of this queue once they are under the threshold value. The code will iterate until this queue is empty.

B. Find Closest Surplus Cells

The function to find the closest surplus cells looks at the cells closest to the needy cell and adds the cell to a list if it contains a surplus. The code searches at increasing distance until it finds a specified number of cells with surplus.

C. Distance

The distance function returns the rectilinear distance (sum of the absolute differences in Cartesian coordinates) between two specified grid cells.

D. Pick Optimal Action

Given the current S matrix, a needy cell, and the list of closest surplus states, this function returns the optimal action to take. To save computation power, it only considers the 5 most promising cells of the list. For each action, it computes the individual isolated cost of each action. The chosen optimal action corresponds to the one with the smallest cost.

E. Apply Action

This function takes in an action to apply and updates the S matrix accordingly. It subtracts the moving surplus from the departure cell and adds this amount to the arrival cell.

1. Algorithm Example

An example of our action-picking and applying algorithm is shown in Figure 3. Initially, cell (3,3) is the neediest (largest negative value). To address this cell, it looks at a combined list of highest surplus states and closest surplus states and moves the surplus with the least cost. In this case, the closest and smallest movable amount is in (4,3). The moved amount will be the minimum of the need and the available surplus magnitudes- in this case, the latter. The result of applying this action is shown in the second panel. This procedure then repeats. Panel 3 shows that the surplus in (2,4) was deemed least costly and subsequently moved. If the algorithm continued, the neediest cell would now shift to (1,3) and would likely consider moving from (1,1) as the best action.

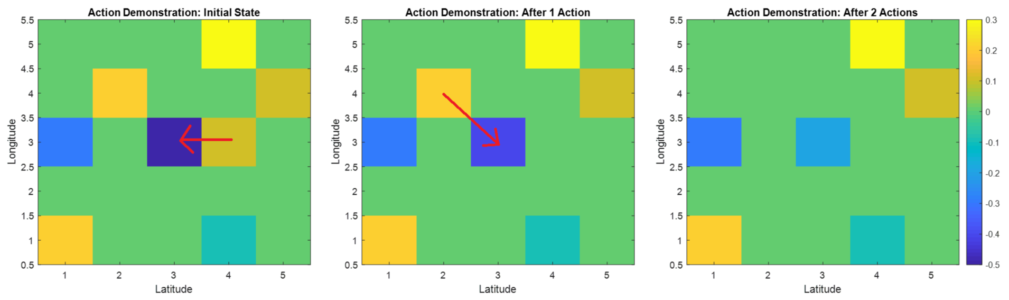

Fig. 3   Example of our algorithm on fictitious data.

F. Main

The main script initializes S based on P and C for a given hour and subsequently, finds the best policy available and updates the P matrix accordingly. The exact procedure here is shown in Figure 3.

G. Find Hourly Policy

For each hour of the day (1-24), this function finds the prescribed best policy. At each hour, it calls main with that hour's computed C matrix as well as the initial P matrix which carries over from the previous hour. At the start of the 1st hour, the P matrix is initialized to a uniform distribution. This means that each cell has a value $= 1/(\text{\# of lat cells} * \text{\# of long cells}) = 1/(40 * 40) = 1/1600 = 0.000625$. At the end of hour 1, the actions that are suggested by the policy are used to determine the transformation from the initial uniform P to the P at the end of the hour. This new P is stored and used as the initial P for the next hour. P_0 = uniform, $P_{1\,\text{start}} = P_0$, $P_{1\,\text{end}} = P_{1\,\text{start}} + \text{actions}_{\text{hour }1}$, $P_{2\,\text{start}} = P_{1\,\text{end}}$ etc.

H. Evaluate Policy

Once we have a compiled policy for a given threshold, we will evaluate its performance against others with different thresholds. The higher we set the threshold, the more discrepancy we are willing to accept between our P and C matrices. Setting a lower threshold implies that we want the police officer distribution to very closely match the crime distribution.

Table 2 Sample policy/action list with times to complete each action

Policy for hour h	Time to complete action in policy (fraction of an hour)
a_1	0.9
a_2	0.7
a_3	0.7
a_4	0.1
⋮	⋮
a_{100}	0.1

Table 3 Sort actions in order to increasing time of completion

0.1	0.7	0.9
a_4	a_2	a_1
a_{100}	a_3	

Note that in a policy for each hour, all the actions are independent of each other and can be executed simultaneously. To evaluate the policy, we take the distance of each action in the list and compute the time it would take to complete that particular action, by estimating an average velocity. An example of this procedure is outlined in Table 2.

$$\text{Time to complete an action } a_i = \frac{\text{distance to move police officers in that action}}{\text{maximum distance we could move in this grid}} = \frac{\text{distance}(a_i)}{\text{2*gridLength}}$$

We then sort the actions in order of increasing completion time, as seen in Table 3. For each unique completion time, we apply all actions that terminate at that time to generate "intermediate timestep" S matrices within each hour. In the example above, we generate an S matrix for time 0.1 (by applying actions a_4 and a_{100}), time 0.7 (by applying actions a_2 and a_3), and 0.9 (by applying action a_1). This allows us to look at how the negative values in the S matrix (intermediate unattended crime, which we want to minimize) are resolved within each hour. Summing up all the negative values in the S matrix becomes the policy score at that timestep.

VI. Results

A. S Matrices

Running the program on the weighted crime data has yielded successful policies that meet the desired S-matrix thresholds. Applying the policies generates before and after S matrices for each hour, 48 explicit matrices total for each threshold. For 3 main thresholds (-0.001, -0.005, -0.01), this means 144 total state matrices. A very small sample of these 144 is shown in Figure 4, which is the before and after of just one specific hour for each threshold. Notice that as the threshold increases, there are fewer cells that require addressing which reduces the differences between the states. These observations hold across all other hours as well.

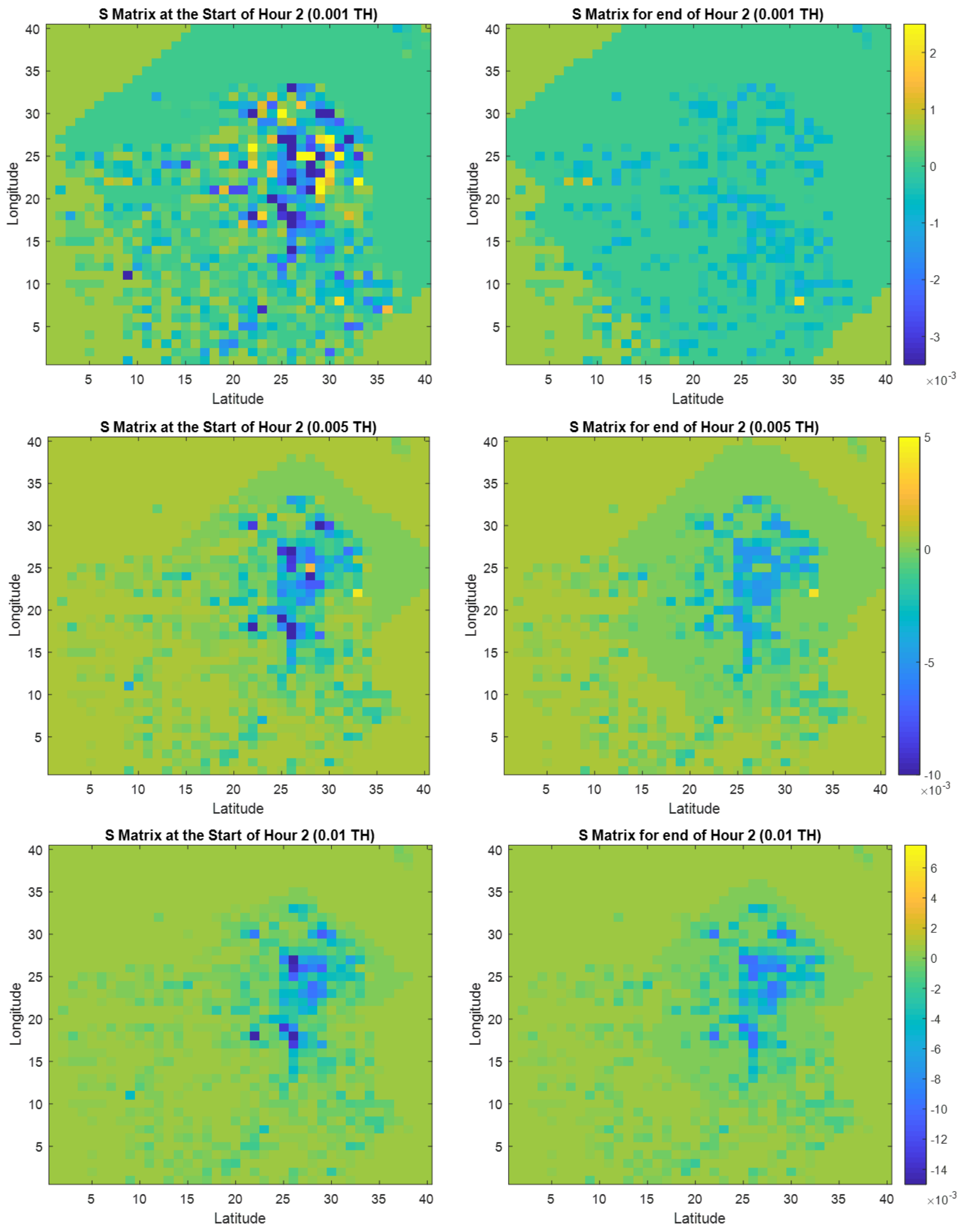

Fig. 4 S matrices before and after executing the policy for hour 2 for each of the 3 main threshold magnitudes (TH).

VII. Discussion & Conclusion

A. Choosing a policy

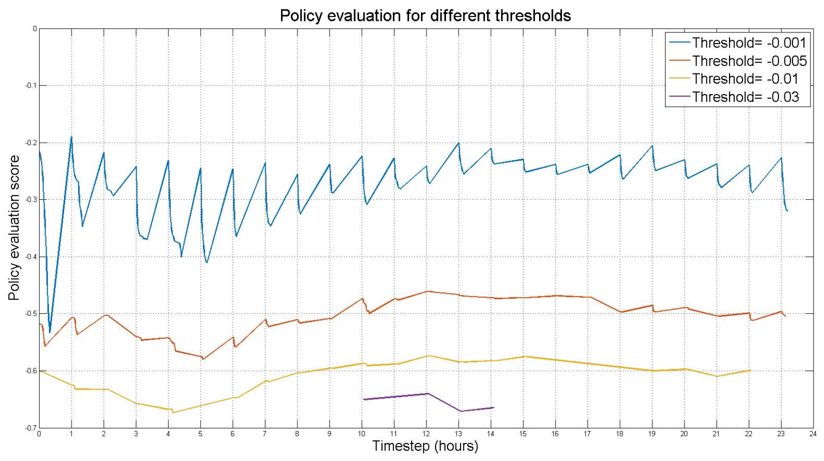

Fig. 5 Scores for different policies

We see in Figure 5 that when the threshold is set too low, e.g. -0.001, the S matrix "overfits" to the data. A large number of actions are taken in each hour and the police officer distribution is forced to match the crime distribution very closely. At the start of each new hour, there has to be a large reshuffling of police officers again to match the new crime distribution matrix. This reshuffling can be seen in the jagged spikes in the threshold line for -0.001 in Figure 5. On the flip side, setting the threshold too high, e.g. -0.03 means that the police officer and crime distribution difference in most grid cells is within our acceptable threshold and we only take a small number of actions when it becomes absolutely necessary, i.e. only between hours 10-14.

Evaluating between which policy to use for any given day will be at the discretion of the police department, depending on what factors are considered desireable give the output from this project. While the policy generated with a threshold of -0.001 had the highest overall score, the jumps between each hour suggests that there would be a lot of movement of police officers throughout the day. If this is unfavorable, the department may choose to use the policy generated at a threshold of -0.005 where the officers are required to move less overall, but the policy score would be generally lower than that of the threshold at -0.001.

Acknowledgments and Contributions

Thank you to Dr. Kochenderfer and all of the AA228 TA's for a great quarter. Contributions are as follows:

- Somrita Banerjee: Coding, policy evaluation, report writing
- Keiko Nagami: Data manipulation, map overlay, report writing
- Daniel Thomlinson: Map creation, methodology, editing, report writing

References

[1] Chen, X., "Patrol Districting and Routing with Security Level Functions," 2010.

[2] Becker, G., "Crime and Punishment: An Economic Approach," *Journal of Political Economy*, Vol. 76, No. 2, 1968, pp. 169, 218.

[3] Dunnett, J. L., Sarah, and Jackson, L., "Optimising Police Dispatch for Incident Response in Real Time," *Journal of the Operational Research Society*, 2018.

[4] Walton, F. E., "Selective Distribution of Police Patrol Force: History, Current Pracitces, Recommendations," *Journal of Criminal Law and Criminology*, Vol. 49, No. 2, 1958, p. 49.

[5] Kochenderfer, M., "Decision Making Under Uncertainty," 2015.