ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

Civil Violence
An agent-based computational model

Florian Besser & Fabian Mönkeberg & David Schwarz

Zürich
December 2011

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Florian Besser

Fabian Mönkeberg

David Schwarz

Contents

1 Abstract

This project presents an agent-based computational model of civil violence. Therein goes the central authority for it to suppress decentralized rebellion. In general we build on an existing model of J.M.Epstein and try to make it more realistic by changing and adding certain properties of the "agent" and "cop". The focus of our work lies on the dynamics of the system and not on the political or social order. We are going to compare the results of our model, with those of J.M.Epstein.

2 Individual contributions

Abstract: Fabian and David
Introduction and Motivations: Florian
Description of the Model: David
Implementation: Florian
Simulation Results and Discussion: Fabian
Summary: Fabian and David

Original Code: Florian
Simulation run: David, with additional tweaks from Fabian

3 Introduction and Motivations

During the course of human history, regime changes have often occurred by civilians taking up arms. This civil violence also happens right now in Syria, and a similar movement has just been successful in Libya and Egypt. These uprisings or rebellions have a big impact on our world, and an ever bigger impact on the people actually executing them. So it is imperative to be able to predict such changes, and that's exactly what we strife for: A clear model for civil violence,so that in the future we can better predict outcomes and aftermaths of these events.

Real world examples: Syria, Libya, Egypt, Russia, United States of America, even Switzerland etc.
(The canton Wallis had various unsuccessful uprisings until it was granted autonomy from Bern.)

Today, as taught in our schools, revolutions are formed by normalcivilians, who are unhappy with their current situation. Somehow, this unhappy (but non-active) mob produces/elects a leader, who will then lead them into action. This now active mob will then have to fulfill certain tasks, such as taking up weapons against an oppressive military force, or a demonstration of force in form of a peaceful protest. On completion of these tasks, they can then form their own government according to their liking.
While the very first step (unhappy civilians) is often a requirement, the idea of a revolution forming and a revolution marching against the current system is not sophisticated enough. There are many factors which can catalyze or hinder the formation of a rebellion, as we are going to show.

The tactics which can be employed by a system that is under attack are also somewhat left out in our education, take the revolution in Egypt as an example: What would have happened if the government went all out on the protesters? Would the military rule have held another decade?

4 Description of the Model (Cop/Agent)

This is an agent-based model about civil violence. We look at a model were the central authority goes for it to suppress decentralized rebellion. We don't take respect to political or social order, it is just a general case. It is based on the model of J.M.Epstein. This model differs between two main actors. "Agents" are members of the general population and may be actively rebellious or not. "Cops" are the token of the central authority, which arrest actively rebellious agents. Later we will make some more sophisticated characterizations, trying to do a more precise and realistic simulation. J.M.Epstein tried to get recognizable macroscopic revolutionary dynamics of fundamental interest with a model, that is as idealized as possible. We try to improve this model. It is now interesting, if we can make remarkable refinements, such that this model is more realistic. First I will describe the agents. The original part is always from the idea of J.M.Epstein and then the additional specifications are our own ideas and extensions.

4.1 The original agent specification

To simulate a rebellion each agent needs some properties, on which we can measure the feelings of them. Therefore there must be some representation of political grievance. The treatment of this model will be really simple and is denoted by two idealized components, which are called hardship (**H**) and legitimacy (**L**). Their definitions are as follows:
H is the agent's perceived hardship (i.e., physical or economic privation). It is defined exogenous and it is assumed to be heterogeneous across agents. It is simply presumed to be uniformly distributed on the interval (0,1). An other important factor is **L**, the perceived legitimacy of the regime. This is exogenous and is equal across agents. In the different runs, it will be varied over the interval (0,1).
This two properties can be combined to the level of grievance any agent feels towards the regime. It is suggested, because of the many functional relationships as:

$$G = H(1 - L) \tag{1}$$

Grievance is the product of perceived hardship (**H**) and perceived "illegitimacy" (**1-L**). The intuition behind this formula is simple. If you think it is legitimate how the government acts (high **L**) then you will not revolt, do not matter how bad things are going on. And if the perceived hardship is pretend to be low, then the grievance towards the regime will be low, too. It does not matter what the perceived illegitimacy is. Because the decision to rebel depends on more than one's grievance, there is defined **R** as the agent's level of risk aversion. Heterogeneous across agents, this is assumed to be uniformly distributed on (0,1) and it is fixed for the agent's lifetime.
Everyone thinks before getting active, what the current risk is to get arrested and this is assumed to be proportional with the cop-to-active ratio $(\mathbf{C/A})_{\mathrm{v}}$ within vision **v**. Where the vision **v** is defined as the agent's vision, this is the number of lattice positions (north, south, east and west of the agent's current position) that the agent is able to inspect. Now is the agent's estimated arrest probability **P** defined by

$$P = 1 - exp[-k(C/A)v] \tag{2}$$

The constant k is set to ensure a plausible estimate (of $\mathbf{P} = 0.9$) when $\mathbf{C}$=1 and $\mathbf{A} = 1$. Notice that $(\mathbf{C/A})_{\mathrm{v}}$ is always defined, because the agent counts always with himself

as active, so **A** is at least 1. Finally we define the agent's net risk **N** = **RP**, what is the product of his risk aversion and estimated arrest probability. This is the special case of **N** = $\mathbf{RPJ}^{\alpha}$, where **J** is the jail term, as discussed in Epstein. A quiescent agent is called an agent in state Q and an active agent is called an agent in state A. Now we can write down the agent's rule:

$$Agent\ rule\ A : If\ G - N > T\ be\ active; otherwise, be\ quiet. \tag{3}$$

Typically , T is set at some small positive value. So we consider, that the agents weigh expected costs and benefits rationally.

4.2 Additional agent specification

Our aim was now, to get a better simulation for revolutions. In the real world, there is a point on which the people try to kill the cops and vice versa. Therefore we implemented a murder threshold **T_murder** and if (**G** - **N**) > **T_murder**, then he tries to kill cops. Therefore exists a **murder success rate** and a **murder assassin death rate**, which define his success by murdering cops and the probability, that he dies during this action, himself. This means, that if a certain level is reached, the agents can now murder cops, too. During an arrest, there can occur some trouble. Therefor, there exists the **agent death chance**, the probability, that the agent dies during this conflict. And the **cop death chance** is the probability, that the cop dies. If we have the first case, that an agent dies, then the legitimacy drops by the **agent death legitimacy penalty**. And if a cop dies, then the legitimacy increases by the **cop death legitimacy boost**.
With this new attribute, we hope to get a new dynamic. Because if cops are murdered, then the $(\mathbf{C/A})_{\mathrm{v}}$ will drop and so there can go more agents active. Otherwise if a agent dies, there is the **agent death legitimacy penalty** for balancing the raised $(\mathbf{C/A})_{\mathrm{v}}$.
Again a unreal setting of the original version is that everyone has the same legitimacy. For that we define the **legitimacy_mod**, so that the personal legitimacy equals $L + L\_personal$, where L_personal is a value out of U(-legitimacy_mod,legitimacy_mod). This means, that we have a certain variance of personal legitimacy. Now we may have agents with an higher personal legitimacy and go earlier active than the others. This means,that they are a trigger for other agents to go active and so we have a starting point of a revolution.

4.3 The original cop specification

The cops are much simpler than the agents. The properties of them are:
$\mathbf{v}^*$ is the cops vision and it is like **v** the number of lattice positions that the cop is able to inspect. It is exogenous and equal across cops. But it does not have to be equal to **v**. The rule, with which the cops operate is:

$$Cop\ rule\ C :\ Inspect\ all\ sites\ within\ v^*\ and\ arrest\ a random\ active\ agent. \tag{4}$$

In the basic model of J.M.Epstein cops never defect to the revolution.

4.4 Additional cop specification

Because cops are human, too, the government can not do whatever they want to do with them. After a certain point (see rule S), they realize, that they do not want to work for the government anymore and stop their work. If it gets even worse, then they can defect (go to the agents).
Then we thought about this, how to model. And we have got the result, that we have to introduce a legitimacy for the cops, too. Therefore they get the same basic legitimacy like the agents, but with an other interval for the personal legitimacy change. And then the personal legitimacy equals $L + L\_personal$, where L_personal is now a value out of U(-legitimacy_mod_cop,legitimacy_mod_cop).
Therefor we define two new rules:

$$Stop\ rule\ S:\ If\ L + L\_personal \leq legitimacy\_threshold\_stop,\\ then\ he\ will\ not\ arrest\ agents\ anymore.$$

$$Defect\ rule\ D:\ If\ L + L\_personal \leq legitimacy\_threshold\_defect,\\ then\ the\ cop\ transforms\ to\ an\ agent.$$

It is clear, that legitimacy_threshold_defect $\leq$ legitimacy_threshold_stop.

4.5 Movement and Jail Terms

One fundamental property of people, is the movement. This gives the whole model a certain dynamic, because the agent information (the number of cops and actives they see) is heterogeneous. And this dynamic will change, by changing $\mathbf{v}$ and $\mathbf{v}^*$. The movement rule is the same for both agents and cops:

$$Movement\ rule\ M:\ Move\ to\ a\ random\ site\ within\ your\ vision. \tag{5}$$

For the jail term, the user set a a value for the maximum jail term, **J_max**. Then, any arrested active is assigned a jail term drawn randomly from U(0,**J_max**). This is an important term in the model, which handles the duration of removing actives from circulation. If you set alpha to zero, there will be no deterrent effect of increasing the jail term.

4.6 General changes of the model

To anlyse some scenarios we implemented the **each_round_legitimacy_change**. If you set it not equal to zero, the legitimacy (**L**) will change every round by this factor. And so you can see what happens if the legitimacy of the government drops constantly. To get even more heterogeneous agents and cops, we let the vision$\mathbf{v}$ and $\mathbf{v}^*$, of the agents and

cops, be various and uniformly distributed in (vision_Min_agent,vision_Max_agent) and in (vision_Min_cops,vision_Max_cops).
One last diversification is, that in the beginning all the cops start in the middle of the space and than move from there on. This should be the scenario, that the agent can communicate, where they will meet and the cops get the information not until they met!

5 Implementation

The implementation of our model brings many features, as well as some added complexity. A short overview about the core functionality:

5.1 Problem Generation

According to user-set constants, the playfield is filled with Agents and Cops. These players are generated with alle the required functionality to later perform their actions. Agents, for example, are already equipped with Hardship.

5.2 Step

Since the problem is now specified, we can loop one step at a time, until we reach some final state. In each step, there are a few things that must be done:

5.2.1 Reset Moves

Every Cop and Agent can move exactly once per step, so we must ensure that they can now move properly.

5.2.2 Release imprisoned

All the imprisoned Agents have their sentence reduced by one. All Agents with a sentence of zero or less are set free. They are randomly put onto empty fields on the playfield.

5.2.3 Move Cops and Agents

Agents scout out the empty fields in their vision, and choose one at random. In case there are no empty fields, they stay put.
Cops scout for active Agents, as well as empty fields. If they spot an Agent, they move to the Agent's field and arrest him. Should no Agent be inside the Cop's vision, they randomly move to an empty field. In case they can'tfind an Agent or an empty field, they stay put.

5.2.4 Set Status of Agents

With all the Agents and Cops moved, the Agents need to re-evaluate whether they want to be active. Their activeness is set for the next round, as well as for display purposes. Cops as well as empty squares are also saved for display.

5.2.5 Display current Model

With the saved values from the step above, a graph can be plotted, showing all Agents and Cops, as well as some info about them.

5.3 Display summarizing plots

At the very end, plot some graphs of the most important values during the computation. Current graphs include active Agents, inactive Agents, imprisoned Agents, active Cops, Legitimacy.

5.4 Tweaks

This core model has been enhanced by many tweaks and options, listed here:

5.4.1 Size

The size of the playfield can now be changed.

5.4.2 Occupied to Unoccupied Ratio

This ratio regulates Occupied Spaces vs. Unoccupied Spaces. In the original paper, the author started with a full playfield, but that might not always be desired.

5.4.3 Cop to Agent Ratio

This ratio sets the global amount of Cops vs. Agents. Especially useful when simulating low/heavy police presence with varying playfield size.

5.4.4 Jailterm Min/Max

Minimum and Maximum amout of steps an Agent will spend in jail if arrested.

5.4.5 Jail sentence

In the original paper, the jail sentence was chosen uniformly from zero to some maximum. In reality, jail sentences are not chosen uniformly, and so this function can be altered in whatever way desired. It uses the Jailterm Min/Max discussed above.

5.4.6 Jail Aggreviance

In the original paper, released Agents left the jail just as aggrieved as when they were captured. Modeling different criminal justice systems is nearly impossible that way.Therefor, we implemented this variable. If Agents are badly treated in jail, they now will be released in an angry state.

5.4.7 Activism Threshold

The attitude of an agent against the regime is computed by subtracting the net risk of arrest for that agent from his personal grievance. But what attitude is needed for an Agent to actually go active? In the original paper, this was assumed to be zero, but we implemented a variable to model different people and their resistance to regimes.

5.4.8 Murder Threshold

In order to model militant uprisings we needed some functionality for Agents to actually kill or at least attack Cops. This murder threshold is the outcome. Normally much higher than the activism threshold, if fulfilled, the Agent will no longer protest but instead go and hunt down Cops.

5.4.9 Murder success rate

Depending on the tools available to murdering Agents, they may or may not succeed in killing a cop. This is used to differentiate between countries with easy/difficult access to weapons for civilians.

5.4.10 Murder assassin death rate

Sometimes Agents might get killed while trying to kill a Cop. If the Cops are taken by suprise the revolution might go much more smoothly than if all Cops are heavily armed and have orders to kill every attacker.

5.4.11 Vision Min/Max

These variables, separate for Agents and Cops, govern the visible field. In every step, players can only gauge what happens in their fieldof vision, and act upon it. Large fields of vision will allow Cops to become more effective, while Agents can better interpret their chances of getting arrested.

5.4.12 Death

Whenever a Cop tries to arrest an Agent, either one of them might die. As seen in 2008 in Greece, the death of Alexandros Grigoropoulos sparked violence. So whenever an Agent dies while being arrested, the Legitimacy of the regime suffers. To somewhat even out the effect, in case a Cop dies, the Legitimacy receives a boost. These variables are especially useful when gauging if a regime should use many Cops to quickly crack down on Agents, or whether they should use a moderate force to slowly curb the unrest.

5.4.13 Important Places

In the original paper, Cops as well as Agents wander pretty much randomly. When modeling Egypt's revolution in and around Tahir square, the idea of randomly moving doesn't cut it. Both Agents and Cops marched to one important place, and then clashed there. To model such an important point, this feature was added.

5.4.14 Important people

Nerly every revolution had it's heroes. While they were not so powerful by themselves, they would often tip the balance in their favor. We modeled important people with an extended Vision, as well as enhanced effectiveness: When an Agent tries to find out his chances of arrest, he will measure a low chance when standing next to an important Agent, but a very high chance when seeing an important Cop.

5.4.15 Defective Cops

Normally, Cops won't defect to the Agents side. In special cases, such as Egypt and Tunesia, the Cops decided to join the Agents since the Legitimacy of the regime had fallen too low. We modeled this using two variables. Should the Legitimacy fall below some threshold, Cops will stop arresting Agents. If the Legitimacy falls further, they will abandon their post and join the Agents.

5.4.16 Better Legitimacy

Legitimacy is normally not globally percieved. Some people will always agree with a certain regime, while others will always disagree. To model such cases, we assigned every Agent and Cop his own impression of the regime. This allows for some more believable outcomes, such as Agents with low Hardship but a distinct hatred of the regime going active, as well as Agents with high Hardship keeping silent because they love the regime.

5.4.17 Legitimacy Change each step

In certain cases a regime might loose supporters as time passes. For example the people of Egypt are pressing for reforms, and the longer they have to wait, the less satisfied they will become with the regime. This variable can be used to model such cases, and help decide when the breaking point is reached, by which a reform needs to be introduced, or the protests will start again.

5.4.18 Number of Frames

This variable does not have any influence on the computation itself,but governs how many steps are computed and archived.

5.4.19 Plot every X Frames

In case the number of frames ishigher than five, one might want to consider notplotting each and every step, and this variable governs that.

6 Simulation Results and Discussion

USE THE FUNDAMENTAL QUESTIONS POSED IN THE PROPOSAL

7 Summary and Outlook

8 References

Modeling civil violence: An agent-based computational approach, Eppstein 2002

9 Appendix A: Proposal

Document Version: 4

Group Name: creative!

Group participants names : Florian Besser, Fabian Mönkeberg, David Schwarz

General Introduction

During the course of human history, regime changes have often occurred by civilians taking up arms. This “civil violence” also happens right now in Syria, and a similar movement has just been successful in Libya and Egypt. These uprisings or rebellions have a big impact on our world, and an ever bigger impact on the people actually executing them. So it is imperative to be able to predict such changes, and that's exactly what we strife for: A clear model for civil violence,so that in the future we can better predict outcomes and aftermaths of these events.

Real world examples would be: Syria, Libya, Egypt, Russia, United States of America, even Switzerland etc.

(The canton Wallis had various unsuccessful uprisings until it was granted autonomy from Bern.)

Today, as taught in our schools, revolutions are formed by “normal”civilians, who are unhappy with their current situation. Somehow, this unhappy (but non-active) mob produces/elects a leader, who will then lead them into action. This now active mob will then have to fulfill certain tasks, such as taking up weapons against an oppressive military force, or a demonstration of force in form of a peaceful protest. On completion of these tasks, they can then form their own government according to their liking.

While the very first step (unhappy civilians) is often a requirement, the idea of a revolution forming and a revolution marching against the current system is not sophisticated enough. There are many factors which can catalyze or hinder the formation of a rebellion, as we are going to show.

The tactics which can be employed by a system that is under attack are also somewhat left out in our education, take the revolution in Egypt as an example: What would have happened if the government went “all out” on the protesters? Would the military rule have held another decade?

Fundamental Questions

Modifications of the model from Eppstein:

1. Different start conditions: some Cops are positioned in a casern (Soldier-cops) in the middle of the field. The rest is equidistributed (Police-cops).
2. Different distributions: the equidistribution for the level of hardship H or risk aversion R and over variables is not satisfying. To model the reality better, we want to use a distribution on [0,1] similar to the Gaussian distribution centred at ½.
3. Jail experience: The level of Risk aversion will rise during jailtime.
4. Agent leader: A new typ of agent. He will „lead“ the revolution. He’s able to increas the probability of Agents in an area around him to become active (by decreasing the level of Legitimacy L and decreasing the risk aversion R)
5. Copleader : New typ of cop. Modified by the ability of increasing the Level of Legitimacy L and the level of risk aversion R around him.
6. Intelligente Agents: active agents also move around. They have a higher change of moving towards a Leader and/or other agents (inside their visible area). This models the forming of a demonstration . This will change the cop /agent ratio and thereby decreas the risk for an active agent to be arrested.
7. Intelligente cops: Cops will have a higher change of moving towards active agents (inside their visible area). If they can`t see any active agents the Soldier-cops will have a higher chance of moving towards the casern in the middle of the field. The police-cops jus twill move randomly.

Questions:

- How does the results of the Eppstein Model differ from the result of theModel modified by 1 and 2?
- Does the autcome changes under a combination of Modifications 3-7? What influence do the modifications have ? Are there interesting cases ?
- How does the success rat of a revolution depend of the different Modifications ? Success of a revolution means that where are a certain number of active agents. The total sucess will be measured by the amount of succesful revolutions in a certain time period. This correlats with the average waiting time between two successful revolutions.
- How does the amount of cops influence the total success in the different models ? What is the influence of the amount of military-cops and Police cops ? So are equidistributed cops better then centrally based cops? Will the probability for a revolution increas with the distance from the casern i.e. will active agents be likely far away from the casern ?
- Is a revolution successful because of it`s people or it`s leaders : Is there a significant higher probability for a sucessful revolution if there is an Agent Leader or not ?

Expected Results

A model that offers an improved algorithm to model uprisings and produce similar patterns as in real life.

References

Modeling civil violence: An agent-based computational approach,, Eppstein 2002

Project about « Civil Violance » from 2010

Research Methods

Agent-Based Model (Continuous Modeling would be possible

10 Appendix B: Code

Listing from source file `Agent.m`

Listing 1: Agent

```
classdef Agent
    %Person Summary of AGENT
    %   Detailed explanation goes here

    properties
        Hardship;
        Risk_Aversion;
        Vision;
        Jailt=0;
        Can_Move = true;
        Pers_Legitimacy;
        Effectiveness=1;
        Is_Active = false;
    end

    methods
            function obj = Agent(H, R, V, Leg, E)
            % class constructor
                obj.Hardship = H;
                obj.Risk_Aversion = R;
                obj.Vision = V;
                obj.Jailt=0;
                obj.Can_Move=true;
                obj.Pers_Legitimacy = Leg;
                obj.Effectiveness = E;
                obj.Is_Active = false;
            end

            function G = Grievance (this, L)
                G = this.Hardship*(1-min(1, max(0, L+this.Pers_Legitimacy)));
            end
            function P = Chance_of_Arrest (this, k, C, A)
                P = 1 - exp(-k*C/A);
            end
            function N = Net_Risk (this, P, J, alpha)
                N = this.Risk_Aversion*P*(J^alpha);
            end
            function N = Get_Type (this)
                N = 'Agent';
            end
            function N = Jailtime (this)
                N = this.Jailt;
```

```
        end
        function this = Set_Jailtime (this, J)
            this.Jailt = J;
        end
        function this = Set_Hardship (this, H)
            this.Hardship = H;
        end
        function b = Moveable (this)
            b = this.Can_Move;
        end
        function this = Set_Move (this, b)
            this.Can_Move = b;
        end
        function b = Active (this)
            b = this.Is_Active;
        end
        function this = Set_Active (this, b)
            this.Is_Active = b;
        end
    end

end
```

Listing from source file `Cop.m`

Listing 2: Cop

```
classdef Cop
    %Person Summary of AGENT
    %   Detailed explanation goes here

    properties
        Vision;
        Can_Move = true;
        Pers_Legitimacy;
        Effectiveness=1;
    end

    methods
        function obj = Cop(V, Leg, E)
        % class constructor
            obj.Vision = V;
            obj.Can_Move=true;
            obj.Pers_Legitimacy = Leg;
            obj.Effectiveness = E;
        end
        function N = Get_Type (this)
            N = 'Cop';
```

```
            end
            function b = Moveable (this)
                b = this.Can_Move;
            end
            function this = Set_Move (this, b)
                this.Can_Move = b;
            end
    end

end
```

Listing from source file `Display.m`

Listing 3: Display

```
%   FIX VALUES, CHANGE THESE IF NECESSARY

clear all;

side = 40;
%Side of the playfield

L = 0.9;
%Legitimacy of the Regime

O_to_U_Ratio = 0.5;
%Occupied Spaces to Unoccupied Spaces

C_to_A_Ratio = 0.1;
%Cops to Agents Ratio

Jailterm_Min = 3;
%Jailterm in Years

Jailterm_Max = 5;
%Jailterm in Years

Jailsentence = @() random('Uniform',Jailterm_Min,Jailterm_Max,1,1);
%Jailsentence = @(x, y) random('Uniform',x,y,1,1);

Better_Deterrent_of_Jailtime = 0;
%In Paper alpha=0

Better_Aggreviance_when_Agent_leaves_Jail = false;
Aggreviance = @() random('Uniform',-0.5,0.5,1,1);

Activism_Threshold = 0.1;
%If Grievance - Net Risk > Activism_Threshold -> Agent is active
```

```
Murder_threshold =1;
Murder_success_rate = 0.0;
Murder_assasin_death_rate = 0.0;
%If Grievance - Net Risk > Murder_threshold -> Agent kills Cops

Vision_Min_Cops = 6;
Vision_Max_Cops = 10;
Vision_Min_Agent = 6;
Vision_Max_Agent = 10;

Death = false;
Agent_Death_Chance = 0.0;
Agent_Death_Legitimacy_Penalty = 0.02;
Cop_Death_Chance = 0.00;
Cop_Death_Legitimacy_Boost = 0.03;
%Agents and Cops can die in Confrontations

Important_Land = false;
%Some spots of land are important
ID = 210;
Importance = 0;
ImportantFields{1} = Field(ID, Importance);

Important_People = false;
Important_Chance_Agents = 0.0;
Important_Effectiveness_Agents = 5;
Important_Chance_Cops = 0.0;
Important_Effectiveness_Cops = 5;
%Some people are important
%Important Agents count as "more" Agents, effectively making others rebell
%Important Cops count as "more" Cops, effectively silencing protest.
%Important People have extended Vision

Defective_Cops = false;
Legitimacy_Mod_Cop = 0.0;
Legitimacy_Threshold_Stop = 0.0;
Legitimacy_Threshold_Defect = 0.0;
%If a Cop's view of the Legitimacy drops too low, he may stop arresting, or de

Better_Legitimacy = false;
Legitimacy_Mod = 0.0;
%Agents all have a personal Legitimacy Modifier, which modifies the global
%Legitimacy.

Each_Round_Legitimacy_Change = 0;

One_Time_Change = true;
```

```
One_Time_Legitimacy_Change = -0.3;
round_for_One_Time_Legitimacy_Change = 10;

Number_of_Frames = 30;
%Number of Rounds

Plot_every_X_Frames = 5;
%Only plot some of the rounds (1=Plot every Round)

k = 2.3;
%Constant so that 0.9 = 1 - exp(-k), see Paper

%StatisticalAnalysis
%AllReleases = [];
%AllAgentMoves = [];
%AllCopMoves = [];

    %   First, we need to generate a Problem
    %   OY, KEEP HANDS AWAY FROM CODE!

x = zeros(1, side*side+3);
y = zeros(1, side*side+3);

%Playfield;                  %Fuck you, MATLAB
%Prison;                     %Fuck you, MATLAB
Prison{1} = Empty();

Colors_Attitude = zeros(1, side*side+3);
Colors_Activism = zeros(1, side*side+3);

for i=1:1:side

    for j=1:1:side

        x((i-1)*side + j) = i;
        y((i-1)*side + j) = j;



        if random('Uniform',0,1,1,1) > 1/(O_to_U_Ratio+1)

            if random('Uniform',0,1,1,1) > 1/(C_to_A_Ratio+1)
                %Cop
                V = floor(random('Uniform',Vision_Min_Cops,Vision_Max_Cops,1,1
                if Defective_Cops
                    Leg = random('Uniform',-Legitimacy_Mod_Cop,Legitimacy_Mod_C
                else
                    Leg = 0;
```

```
                end
                if random('Uniform',0,1,1,1) < Important_Chance_Cops
                    E = Important_Effectiveness_Cops;
                    V = V*Important_Effectiveness_Cops;
                else
                    E = 1;
                end
                Playfield{(i-1)*side + j} = Cop(V, Leg, E);
            else
                %Agent
                H = random('Uniform',0,1,1,1);
                R = random('Uniform',0,1,1,1);
                V = floor(random('Uniform',Vision_Min_Agent,Vision_Max_Agent,1
                if Better_Legitimacy
                    Leg = random('Uniform',-Legitimacy_Mod,Legitimacy_Mod,1,1);
                else
                    Leg = 0;
                end
                if random('Uniform',0,1,1,1) < Important_Chance_Agents
                    E = Important_Effectiveness_Agents;
                    V = V*Important_Effectiveness_Agents;
                else
                    E = 1;
                end

                Playfield{(i-1)*side + j} = Agent(H, R, V, Leg, E);
            end
        else
            Playfield{(i-1)*side + j} = Empty();
        end
    end

end


%Dirty Hack for correct Colors
x(side*side+1) = -5;
x(side*side+2) = -5;
x(side*side+3) = -5;
y(side*side+1) = -5;
y(side*side+2) = -5;
y(side*side+3) = -5;
Colors_Attitude(side*side+1) = -5;
Colors_Attitude(side*side+2) = -3;
Colors_Attitude(side*side+3) = 1;
Colors_Activism(side*side+1) = -5;
Colors_Activism(side*side+2) = -3;
Colors_Activism(side*side+3) = 1;
```

```
    %    Then, wesolve it!
    %    OY, KEEP HANDS AWAY FROM CODE!

%fig1=figure(1);
%winsize = get(fig1,'Position');
%winsize(1:2) = [0 0];

%Movie =moviein(Number_of_Frames,fig1,winsize);
%set(fig1,'NextPlot','replacechildren');

Active_Agents_Array = zeros(1, Number_of_Frames);
Inactive_Agents_Array = zeros(1, Number_of_Frames);
Active_Cops_Array = zeros(1, Number_of_Frames);
Inactive_Cops_Array = zeros(1, Number_of_Frames);
Defected_Cops_Array = zeros(1, Number_of_Frames);
Murdered_Cops_Array = zeros(1, Number_of_Frames);

Prison_Population_Array = zeros(1, Number_of_Frames);
Legitimacy_Array = zeros(1, Number_of_Frames);

g = 0;
first=true;
figurecounter=1;

for h=1:Number_of_Frames

    %legitimicy-drop

        if (One_Time_Change && h==round_for_One_Time_Legitimacy_Change)
        L =L + One_Time_Legitimacy_Change;
        end


    %Reset Moves

    for j=1:1:side*side
        A = Playfield{j};

        if strcmp(A.Get_Type(), 'Agent') || strcmp(A.Get_Type(), 'Cop')

            A = A.Set_Move(true);

            Playfield{j} = A;

        end
    end

    % Release imprisoned
```

```
Prison_Population = 0;

for i=2:1:length(Prison)

    A = Prison{i};

    if strcmp(A.Get_Type(), 'Agent')

        A = A.Set_Jailtime(A.Jailtime() - 1);

        if A.Jailtime() <= 0
            A = A.Set_Jailtime(0);
            A = A.Set_Hardship(max(0, min(1, A.Hardship + Aggreviance())));

            EmptyFields = [];

            for j=1:1:side*side
                % Get all Empty Places
                if strcmp(Playfield{j}.Get_Type(), 'Empty')
                    EmptyFields(length(EmptyFields)+1) = j;
                end
            end

            EmptyFields = EmptyFields( randperm(length(EmptyFields)) );

            Release = select_field( EmptyFields, ImportantFields, side );
            %AllReleases(length(AllReleases)+1) = Release;

            Playfield{Release} = A;          %Actual Move
            Prison{i} = Empty();

        else

            Prison{i} = A;
            Prison_Population = Prison_Population + 1;

        end

    end
end

Prison_Population_Array(h) = Prison_Population;

%Move Cops / Agents

for i=1:1:side
```

```
        for j=1:1:side

            A = Playfield{(i-1)*side + j};

            if Defective_Cops && strcmp(A.Get_Type(), 'Cop') && A.Moveable(
                %Check if Cop,and if he defects now
                H = random('Uniform',0,1,1,1);
                R = random('Uniform',0,1,1,1);
                A = Agent(H, R, A.Vision, A.Pers_Legitimacy, A.Effectivenes
                Playfield{(i-1)*side + j} = A;

                Defected_Cops_Array(h) = Defected_Cops_Array(h)+1;
            end

            if (strcmp(A.Get_Type(), 'Agent') || strcmp(A.Get_Type(), 'Cop'

                %Move to unoccupied space, check if active

                AgentFields = [];

                if strcmp(A.Get_Type(), 'Cop') && (L + A.Pers_Legitimacy >
                    %We have an active Cop, search for active Agents
                    %for Arrest
                    for k=1:1:A.Vision
                        % Get all agent Places

                        if i > k && strcmp(Playfield{(i-1-k)*side + j}.Get_
                            AgentFields(length(AgentFields)+1) = (i-1-k)*si
                        end
                        if i <= side-k && strcmp(Playfield{(i-1+k)*side + j
                            AgentFields(length(AgentFields)+1) = (i-1+k)*si
                        end
                        if j > k && strcmp(Playfield{(i-1)*side + j-k}.Get_
                            AgentFields(length(AgentFields)+1) = (i-1)*side
                        end
                        if j <= side-k && strcmp(Playfield{(i-1)*side + j+k
                            AgentFields(length(AgentFields)+1) = (i-1)*side
                        end
                    end
                    AgentFields = AgentFields( randperm(length(AgentFields)
                end

                EmptyFields = [];

                if strcmp(A.Get_Type(), 'Agent') || (strcmp(A.Get_Type(),
                    %We have an Agent or a Cop who can't find Agents to
                    %Arrest -> wander aimlessly
                    for k=1:1:A.Vision
```

```
                %Get all Empty Places
                if i > k && strcmp(Playfield{(i-1-k)*side + j}.Get
                    EmptyFields(length(EmptyFields)+1) = (i-1-k)*si
                end
                if i <= side-k && strcmp(Playfield{(i-1+k)*side + j
                    EmptyFields(length(EmptyFields)+1) = (i-1+k)*si
                end
                if j > k && strcmp(Playfield{(i-1)*side + j-k}.Get
                    EmptyFields(length(EmptyFields)+1) = (i-1)*side
                end
                if j <= side-k && strcmp(Playfield{(i-1)*side + j+k
                    EmptyFields(length(EmptyFields)+1) = (i-1)*side
                end
            end
            EmptyFields = EmptyFields( randperm(length(EmptyFields)
        end

        CopFields = [];

        if strcmp(A.Get_Type(), 'Agent') && A.Grievance(L) > Murde
            %We have an Agent that is ready to kill, search
            %Assassination targets
            for k=1:1:A.Vision
                %Get all Empty Places
                if i > k && strcmp(Playfield{(i-1-k)*side + j}.Get
                    CopFields(length(CopFields)+1) = (i-1-k)*side +
                end
                if i <= side-k && strcmp(Playfield{(i-1+k)*side + j
                    CopFields(length(CopFields)+1) = (i-1+k)*side +
                end
                if j > k && strcmp(Playfield{(i-1)*side + j-k}.Get
                    CopFields(length(CopFields)+1) = (i-1)*side + j
                end
                if j <= side-k && strcmp(Playfield{(i-1)*side + j+k
                    CopFields(length(CopFields)+1) = (i-1)*side + j
                end
            end

            CopFields = CopFields( randperm(length(CopFields)) );
        end

        if strcmp(A.Get_Type(), 'Cop') && length(AgentFields) >= 1
            %Our Cop has found some Agents to Arrest
            if random('Uniform',0,1,1,1) < Agent_Death_Chance
                %HE tried to arrest an Agent, but the Agent
                %died.
                L = L - Agent_Death_Legitimacy_Penalty;
                Arrest = select_field(AgentFields, ImportantFields,
```

```
                        A = A.Set_Move(false);

                        Playfield{(i-1)*side + j} = Empty();
                        Playfield{Arrest} = A;
%Free Cell
                    elseif random('Uniform',0,1,1,1) < Cop_Death_Chance
                        %He tried to arrest an Agent, but the Cop died.
                        L = L + Cop_Death_Legitimacy_Boost;
                        Playfield{(i-1)*side + j} = Empty();
                    else
                        %Arrest successful
                        Arrest = select_field( AgentFields, ImportantFields
                        ArrestedAgent = Playfield{Arrest};
                        ArrestedAgent = ArrestedAgent.Set_Jailtime(Jailsent
                        Prison{length(Prison)+1} = ArrestedAgent;
%Jail

                        A = A.Set_Move(false);

                        Playfield{(i-1)*side + j} = Empty();
                        Playfield{Arrest} = A;
%Free Cell
                    end

                    %AllCopMoves(length(AllCopMoves)+1) = Arrest;

                elseif length(CopFields) >= 1 && A.Grievance(L) - A.Net_Ris
                    %Our Agent successfully murders a Cop
                    GoTo = select_field( CopFields, ImportantFields, side

                    A = A.Set_Move(false);

                    %AllAgentMoves(length(AllAgentMoves)+1) = GoTo;

                    Playfield{(i-1)*side + j} = Empty();
                    Playfield{GoTo} = A;          %Actual Move

                    Murdered_Cops_Array(h) = Murdered_Cops_Array(h)+1;
                elseif length(CopFields) >= 1 && random('Uniform',0,1,1,1)
                    %Our Agent tries to murder a Cop, but dies in the
                    %process
                    Playfield{(i-1)*side + j} = Empty();
                elseif length(EmptyFields) >= 1
                    %Wander aimlessly
                    GoTo = select_field(EmptyFields, ImportantFields, side

                    A = A.Set_Move(false);
```

```
                    %AllAgentMoves(length(AllAgentMoves)+1) = GoTo;

                    Playfield{(i-1)*side + j} = Empty();
                    Playfield{GoTo} = A;          %Actual Move
                end

            else
                %Empty Square, do nothing
            end

    end


end

% Accomplish Actions forAgents & Cops
Active_Agents= 0;
Inactive_Agents = 0;
Active_Cops = 0;
Inactive_Cops = 0;

for i=1:1:side

    for j=1:1:side

        A = Playfield{(i-1)*side + j};

        if strcmp(A.Get_Type(), 'Agent')

            Cops = 0;
            Agents = 1;

            for k=1:1:A.Vision

                if i > k && strcmp(Playfield{(i-1)*side + j - k*side}.Get_T
                    Agents = Agents+Playfield{(i-1)*side + j - k*side}.Effe
                elseif i > k && strcmp(Playfield{(i-1)*side + j - k*side}.G
                    Cops = Cops+Playfield{(i-1)*side + j - k*side}.Effectiv
                end
                if i <= side-k && strcmp(Playfield{(i-1)*side + j + k*side
                    Agents = Agents+Playfield{(i-1)*side + j + k*side}.Effe
                elseif i <= side-k && strcmp(Playfield{(i-1)*side + j + k*s
                    Cops = Cops+Playfield{(i-1)*side + j + k*side}.Effectiv
                end
                if j > k && strcmp(Playfield{(i-1)*side + j - k}.Get_Type()
                    Agents = Agents+Playfield{(i-1)*side + j - k}.Effective
                elseif j > k && strcmp(Playfield{(i-1)*side + j - k}.Get_Ty
```

```
                    Cops = Cops+Playfield{(i-1)*side + j - k}.Effectiveness
                end
                if j <= side-k && strcmp(Playfield{(i-1)*side + j + k}.Get_
                    Agents = Agents+Playfield{(i-1)*side + j + k}.Effective
                elseif j <= side-k && strcmp(Playfield{(i-1)*side + j + k}.
                    Cops = Cops+Playfield{(i-1)*side + j + k}.Effectiveness
                end

            end

            Colors_Attitude((i-1)*side + j) = max(-1, A.Grievance(L) - A.N
            if A.Grievance(L) - A.Net_Risk(A.Chance_of_Arrest(k, Cops, Age
                Colors_Activism((i-1)*side + j) = 1;
                A = A.Set_Active(true);
                Playfield{(i-1)*side + j} = A;

                Active_Agents = Active_Agents + 1;
            else
                Colors_Activism((i-1)*side + j) = 0;

                Inactive_Agents = Inactive_Agents + 1;
            end


        elseif strcmp(A.Get_Type(), 'Cop')

            Colors_Attitude((i-1)*side + j) = -5;
            Colors_Activism((i-1)*side + j) = -5;

            if (L + A.Pers_Legitimacy > Legitimacy_Threshold_Stop || Defect
                Active_Cops = Active_Cops + 1;
            else
                Inactive_Cops = Inactive_Cops + 1;
            end

        else

            Colors_Attitude((i-1)*side + j) = -3;
            Colors_Activism((i-1)*side + j) = -3;

        end
    end

end

Active_Agents_Array(h) = Active_Agents;
Inactive_Agents_Array(h) = Inactive_Agents;
Active_Cops_Array(h) = Active_Cops;
```

```
    Inactive_Cops_Array(h) = Inactive_Cops;
    Legitimacy_Array(h) = L;

    L = max(0, min(1, L + Each_Round_Legitimacy_Change));

    %Provide Feedback
    h

    if g == 0

        figx=figure(figurecounter);
        figurecounter = figurecounter+1;
        axes;
        scatter(x, y, 150, Colors_Attitude, 's', 'filled');
        legend([ 'Attitude Round ', num2str(h), 10, 'Dark Blue = Cops', 10, 'L
        axis([0 side+1 0 side+1]);

        figx=figure(figurecounter);
        figurecounter = figurecounter+1;
        axes;
        scatter(x, y, 150, Colors_Activism, 's', 'filled');
        legend([ 'Activism Round ', num2str(h), 10, 'Dark Blue = Cops', 10, 'L
        axis([0 side+1 0 side+1]);
        %Movie(:,i)=getframe(fig1,winsize);

        if h == 1 && Plot_every_X_Frames > 1
            g=Plot_every_X_Frames-1;
        else
           g=Plot_every_X_Frames;
        end
    end

    g=g-1;

end

figx=figure(figurecounter);
figurecounter = figurecounter+1;
plot(1:Number_of_Frames, Active_Agents_Array);
axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))]);
legend('Active Agents','Location','SouthOutside');

figx=figure(figurecounter);
figurecounter = figurecounter+1;
plot(1:Number_of_Frames, Inactive_Agents_Array);
axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))]);
legend('Inactive Agents','Location','SouthOutside');
```

```
figx=figure(figurecounter);
figurecounter = figurecounter+1;
plot(1:Number_of_Frames, Prison_Population_Array);
axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))]);
legend('Imprisoned_Agents' ,'Location','SouthOutside');

%figx=figure(figurecounter);
%figurecounter = figurecounter+1;
%plot(1:Number_of_Frames, Active_Cops_Array);
%axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))]);
%legend('Active Cops' ,'Location','SouthOutside');

%figx=figure(figurecounter);
%figurecounter = figurecounter+1;
%plot(1:Number_of_Frames, Inactive_Cops_Array);
%axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))]);
%legend('Inactive Cops' ,'Location','SouthOutside');

%figx=figure(figurecounter);
%figurecounter = figurecounter+1;
%plot(1:Number_of_Frames, Defected_Cops_Array);
%axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))/25]);
%legend('Defected Cops' ,'Location','SouthOutside');

%figx=figure(figurecounter);
%figurecounter = figurecounter+1;
%plot(1:Number_of_Frames, Murdered_Cops_Array);
%axis([1 Number_of_Frames 0 side^2*(1-1/(O_to_U_Ratio+1))/100]);
%legend('Murdered Cops' ,'Location','SouthOutside');

figx=figure(figurecounter);
figurecounter = figurecounter+1;
plot(1:Number_of_Frames, Legitimacy_Array);
axis([1 Number_of_Frames 0 1]);
legend('Legitimacy' ,'Location','SouthOutside');
```

Listing from source file `Empty.m`

Listing 4: Empty

```
classdef Empty
    %Person Summary of EMPTY
    %   Detailed explanation goes here

    properties
        Vision=0;
    end
```

```
    methods
            function obj = Empty()
                % class constructor
                Vision = 0;
            end

            function N = Get_Type (this)
                N = 'Empty';
            end
    end

end
```

Listing from source file `Field.m`

Listing 5: Field

```
classdef Field
    %Person Summary of FIELD
    %   Detailed explanation goes here

    properties
        Importance = 0;
        ID;
    end

    methods
        function obj = Field(d, I)
            % class constructor
                obj.ID = d;
                obj.Importance = I;
            end
    end

end
```

Listing from source file `select`$_{f}$*ield.m*

Listing 6: select$_{f}$*ield*

```
function field = select_field( fields, important_places, side )
% SelectField Summary of this function goes here
%   Detailed explanation goes here

    %field = fields( round(random('Uniform',0.5,length(fields)+0.499,1,1))
    maxfield = 0;
    maximportance = 0;
```

```
    for i=1:length(fields)

        i_y = ceil(fields(i)/side);
        i_x = fields(i) - ((i_y-1) * side);

        for j=1:length(important_places)
            j_y = ceil(important_places{j}.ID/side);
            j_x = important_places{j}.ID - ((j_y-1) * side);

            importance = 1/sqrt((i_y - j_y)^2 + (i_x - j_x)^2 + 1) * import

            if (importance >= maximportance)
                maximportance = importance;
                maxfield = i;
            end
        end
    end

    field = fields(maxfield);

end
```