Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

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

Project Report

Media Bias and Voting

Ivan Jovanović
Federico Perazzi
Alexandros Vouzas
Christos Lataniotis

Zürich
16.12.2011

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of Originality

This sheet must be signed and enclosed with every piece of written work submitted at ETH.

I hereby declare that the written work I have submitted entitled

_____

is original work which I alone have authored and which is written in my own words.*

Author(s)

Last name _____ First name _____

Supervising lecturer

Last name _____ First name _____

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (http://www.ethz.ch/students/exams/plagiarism_s_en.pdf). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

_____ _____

Place and date Signature

*Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Print form

Agreement for free-download

We hereby agree to make our source code of 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.

Ivan Jovanović

Federico Perazzi

Alexandros Vouzas

Christos Lataniotis

We cannot measure [the media's] power and influence by traditional democratic standards, for these men can create national issues overnight... They can reward some politicians with national exposure, and ignore others. For millions of Americans, the network reporter who covers a continuing issue, like ABM or Civil Rights, becomes, in effect, the presiding judge in a national trial by jury.

Spiro Agnew

Table of contents

Abstract

The phenomenon of electoral result manipulation by media is investigated. Based on the assumption that the biggest portion of a society is being informed about a political candidate's (*incumbent - the existing holder of a political office*) actions through media, it follows that media plays a vital role on the electoral result. In order to investigate this information flow and how electoral results could be altered, a Bayesian network is realized where a political incumbent performs good or bad actions (in terms of being in favour or against the common interest) and then this actions are populated to different groups of voters (societies), with each society having a dominant media channel which broadcasts these actions more or less distorted based on the amount of bias that media has in favour or against the politician. The citizens of each society (voters) receive this information that media sends, and based on the amount of good and bad actions that they receive, they decide about whether they will vote for the politician or not. Moreover, a portion of citizens in each society are aware of the true nature of each political action and try to affect their "surrounding" citizens. The goal is to investigate the effect of media bias in electoral results, and how easy or difficult it is for this "socially aware" portion of citizens to alter this result.

Individual contributions

The whole project was done in a cooperative manner.

1. Introduction and Motivations

There is a widespread belief that in most democracies, the media play a powerful role on electoral outcomes and -as a result- policy making. The phenomenon of media bias is not necessarily correlated with the degree of a country's press freedom since even in countries with high press freedom the media is often seen as biased towards a particular ideology, political party, class or cultural group. Also in today's societies media coverage can have a significant impact in shaping public opinion through the use of distorted information, selective reporting, innuendo, even outright lies in some cases.

In 2009, a survey by Pew Research Center (in the US; for more information see [5]) shows that 74% of the voting public believe that media is biased and one-sided while 18% believe it to be fair. Another interesting figure, concerning the political impact of media bias can be found in [3], where it was shown that in US towns where Fox News was introduced into cable programming, Republicans gained 0.4 to 0.7 percentage point vote share in the Presidential Elections of 2000.

Every politician pays a lot of attention and effort on forming the best possible image to the electorate through the media gates. Normally one could argue that the control of the information is the key to winning the elections. On the other hand reality often proves the exact opposite, great amounts of money are spent to pre-election campaigns with negative election results. The question that becomes apparent is why someone cannot guarantee the success of the final electoral result through media manipulation? Why even with the most structured media strategies (political propaganda) there always exists a portion of population that actively resists and does not follow the political lines formed by mass media? What are the characteristics of this group of people?

In this point, without loss of generality, and in order to simplify our investigated problem and illustrate it at a programming level, we can proceed with the assumption that there exists a certain group of people as a percentage of each population (the exact number of the percentage is not known) that gathers all these characteristics that create the hard to overcome diversity of the election procedure (from a systematic point of view one can consider and model it as noise). At our implementation this group is assigned by the label "*Intelligent*". Here we must make clear that the term "*Intelligent*" is not connected with the mental characteristics of a certain portion of the population. Instead, the group consists of people who appear to be highly resistant to bias/distorted information, have a general feeling of responsibility, education and the will to contribute to the total of their country and population.

"*Intelligent*" people will be our tool to investigate whether there exists a threshold in the portion of this special citizen group that guarantees sufficient resilience of the entire society against biased/distorted media information concerning an incumbent's actions.

In conclusion, the topics for investigation are the following:

- How can media bias affect the electoral result?
- How can "*Intelligent*" people alter the electoral result?
- What is the relative increase of need for "*Intelligent*" people versus increased media bias in order to achieve a good electoral result?

2. Description of the Model

The social network consists of 3 types of entities:

- Politicians
- Media
- Voters

Assume two types of politicians:

- "*Opportunistic*", who pursues his/hers own interest.
- "*Ethical"*, who pursues the public interest.

A strong assumption is silently made here: The "public interest" is unique and applies to all voters, i.e. a politician's actions either represent all voters or none of them.

Media entities receive the actions of a politician entity and broadcast it to the voters. More precisely every media entity broadcasts the news of a politician's actions only to the respective local community. The most important characteristic of this information flow between the media and the voters is that news are being distorted based on how much biased each media entity is in favour, or against the politician.

A general schematic of the communication between the fore mentioned entities can be seen on Figure 1.

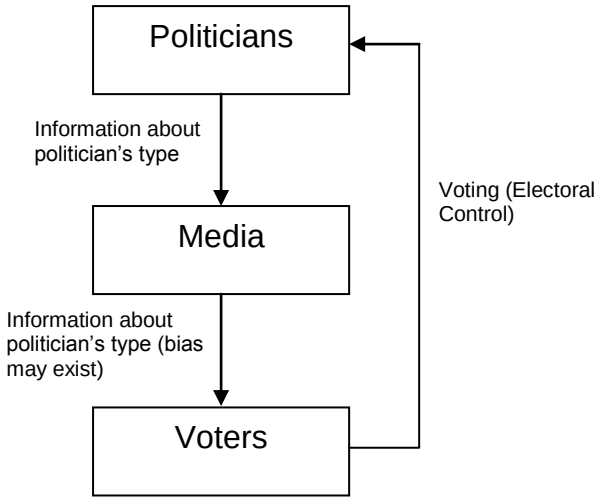

Figure 2.1 : The interaction between the entities of the system

Voter entities can be one of the following types:

- Type 1: The biggest portion of the total population of voters. They can be influenced from the media and from other voters as well.
- Type 2 ("*Intelligent*"): A small portion of the total population of voters. They cannot be influenced from the media, and they know the true identity of each politician's action.

Type 1 voters are divided into "*Influenced voters*" in case they are "sufficiently" near an "*Intelligent*" voter entity or "*Uninfluenced*" voters otherwise. No interaction is assumed to take place between voters of type 1. Each local community is therefore organised, as illustrated in Figure 2.

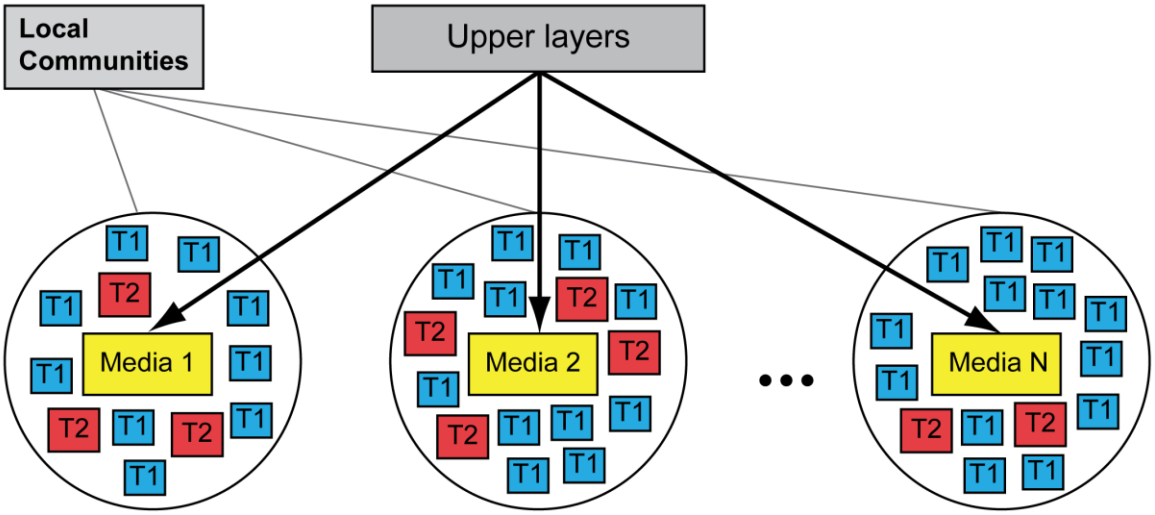

Figure 2.2 : *How the media network is organized*

3. Implementation

3.1. Bayesian Networks

Bayesian networks (BNs) ([6], [7]), also known as belief networks (or directed acyclic graphical models, or Bayes nets for short), belong to the family of probabilistic graphical models (GMs). A Bayesian network represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). In the present project we are going to use the implementation of a Bayesian network in order to represent the probabilistic relationships among certain groups (Politicians - Biased Media - Voters) and investigate all the interactions that conclude to the final result of a voting procedure.

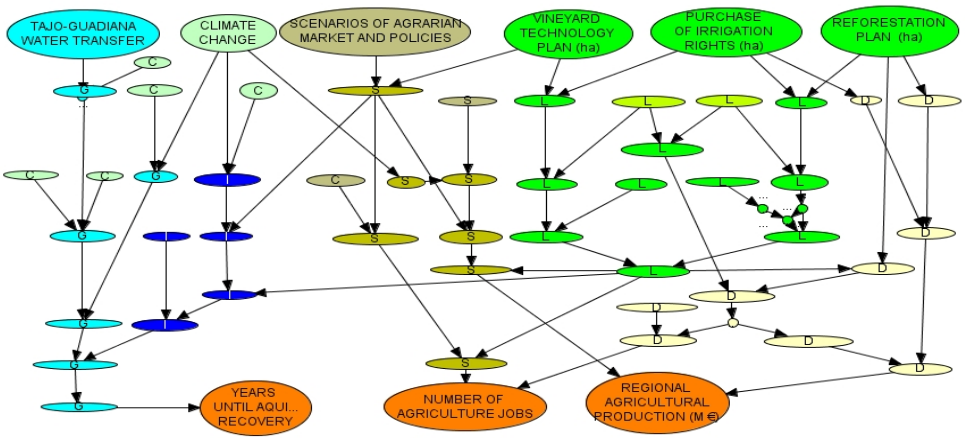

Figure3.1.1 : Bayesian model example

At a more general point of view, graphical structures are used to represent knowledge about an uncertain domain. Each node in the graph represents a random variable, while the edges between the nodes represent probabilistic dependencies among the corresponding random variables. These conditional dependencies in the graph are often estimated by using known statistical and computational methods. Hence, BNs (Bayesian Networks) combine principles from graph theory, probability theory, computer science, and statistics. GMs with undirected edges are generally called *Markov random fields* or *Markov networks*. These networks provide a simple definition of independence between any two distinct nodes based on the concept of a *Markov blanket*[1]. Bayesian networks also correspond to another graph model structure known as a *directed acyclic graph (DAG)* that is popular in the statistics, the machine learning, and the artificial intelligence societies. Bayesian networks are both mathematically rigorous and intuitively understandable. They enable an effective representation and computation of the *joint probability distribution* (JPD) over a set of random variables.

A directed acyclic graph (DAG) is formed by a collection of *vertices* and *directed edges.* The nodes represent random variables and are drawn as circles labeled by the variable names. The edges represent direct dependence among the variables

[1]The Markov blanket for a node in a Bayesian Network is the set of nodes composed of the node's parents, children and its children other parents.

and are drawn by arrows between nodes. In particular, an edge from node *a* to node *b* represents a statistical dependence between the corresponding variables. Thus, the arrow indicates that a value taken by variable *b* depends on the value taken by variable *a*, or roughly speaking that variable *a "*influences*" b*. Node *a* is then referred to as a parent of *b* and, similarly, *b* is referred to as the child of *a*. DAGs may be used to model several different kinds of structure in mathematics and computer science. A collection of tasks that needed to be ordered into a sequence, subject to constraints that certain tasks must be performed earlier than others, may be represented as a DAG with a vertex for each task and an edge for each constraint; algorithms for topological ordering may be used to generate a valid sequence. Another possible usage of DAGs is to model processes in which information flows in a consistent direction through a network of processors.

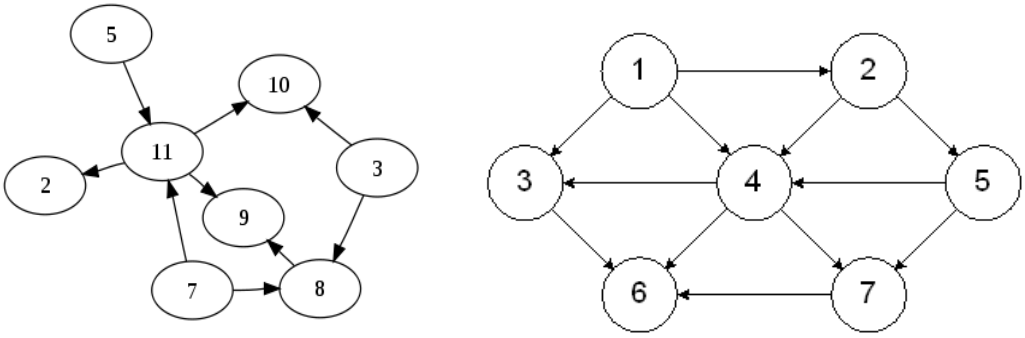

Figure 3.1.2 : Acyclic graphs

In the following lines we are going to present some examples of Bayesian networks implementation:

EXAMPLE 1: ([8]):

Suppose that there are two events which could cause grass to be wet: either the sprinkler is on, or it's raining. Also, suppose that the rain has a direct effect on the use of the sprinkler (namely that when it rains, the sprinkler is usually not turned on). Then the situation can be modeled with a Bayesian network (shown). All three variables have two possible values, *T* (for true) and *F* (for false).

The joint probability function is:

$$P(G, S, R) = P(G \mid S, R)P(S \mid R)P(R)$$

where the names of the variables have been abbreviated to *G* = *Grass wet*, *S* = *Sprinkler*, and *R* = *Rain.*

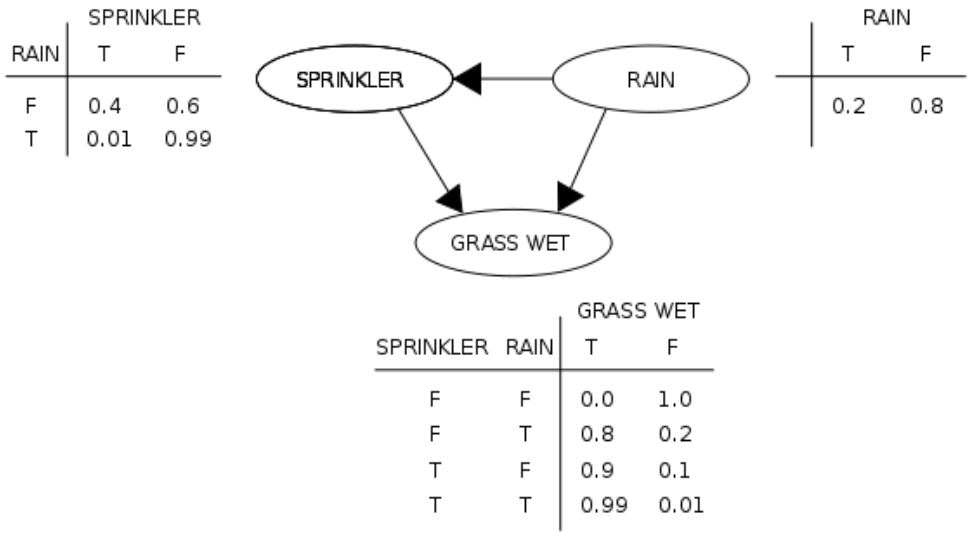

Figure 3.1.3 : Example of a Bayesian network

The model can answer questions like "What is the probability that it is raining, given the grass is wet?" by using the conditional probability formula and summing over all variables:

$$P(R = T \mid G = T) = \frac{P(R = T \mid G = T)}{P(G = T)} = \frac{\Sigma_{s \in T,F} P(G = T, S, R = T)}{\Sigma_{s,R \in T,F} P(G = T, S, R)}$$

$$\frac{(0.99\text{x}0.01\text{x}0.2 = 0.00198_{TTT}) \mid (0.8\text{x}0.99\text{x}0.2 = 0.1584_{TFT})}{0.00198_{TTT} + 0.288_{TTF} + 0.1584_{TFT} + 0_{TFF}} \approx 0.3577$$

As in the example, numerator is pointed out explicitly; the joint probability function is used to calculate each iteration of the summation function. In the enumerator marginalizing over *S* and in the denominator marginalizing over *S* and *R*.

If, on the other hand, we wish to answer an interventional question: "What is the likelihood that it would rain, given that we wet the grass?" the answer would be governed by the post-intervention joint distribution function

$$P(S, R \mid do(G = T)) = P(S \mid R)P(R)$$

obtained by removing the factor $P(G \mid S, R)$ from the pre-intervention distribution. As expected, the likelihood of rain is unaffected by the action: $P\big(R \mid do(G = T)\big) = P(R)$.

If, moreover, we wish to predict the impact of turning the sprinkler on, we have $P\big(R, G \mid do(S = T)\big) = P(R)P(G \mid R, S = T)$ with the term $P(S = T \mid R)$ removed, showing that the action has an effect on the grass, but not on the rain.

One advantage of Bayesian networks is that it is intuitively easier for a human to understand (a sparse set of) direct dependencies and local distributions than complete joint distribution.

EXAMPLE 2: ([6]):

Consider the following example that illustrates some of the characteristics of BNs. It considers a person who might suffer from a back injury, an event represented by the variable *Back* (denoted by *B*). Such an injury can cause a backache, an event represented by the variable *Ache* (denoted by *A*). The back injury might result from a wrong sport activity, represented by the variable *Sport* (denoted by *S*) or from new uncomfortable chairs installed at the person's office, represented by the variable *Chair* (denoted by *C*). In the latter case, it is reasonable to assume that a coworker will suffer and report a similar backache syndrome, an event represented by the variable *Worker* (denoted by *W*). All variables are binary; thus, they are either true (denoted by *T*) or false (denoted by *F*). The CPT (conditional probability table) of each node is listed besides the node. In this example the parents of the variable *Back* are the nodes *Chair* and *Sport*.

The child of *Back* is *Ache*, and the parent of *Worker* is *Chair*. Following the BN independence assumption, several independence statements can be observed in this case. For example, the variables *Chair* and *Sport* are marginally independent, but when *Back* is given they are conditionally dependent. This relation is often called "*explaining away*". When *Chair* is given, *Worker* and *Back* are conditionally independent. When *Back* is given, *Ache* is conditionally independent of its ancestors *Chair* and *Sport*. The conditional independence statement of the BN provides a compact factorization of the JPDs. Instead of factorizing the joint distribution of all the variables by the chain rule, i.e.:

$$P(C,S,W,B,A) = P(C)P(S \mid C)P(W \mid S,C)P(B \mid W,S,C)P(A \mid B,W,S,C)$$

the BN defines a unique JPD in a factored form, i.e.

$$P(C,S,W,B,A) = P(C)P(S)P(W \mid C)P(B \mid S,C)P(A \mid B)$$

Note that the BN form reduces the number of the model parameters, which belong to a multinomial distribution in this case. Such a reduction provides great benefits from inference, learning (parameter estimation), and computational perspective. (combination results see picture below):

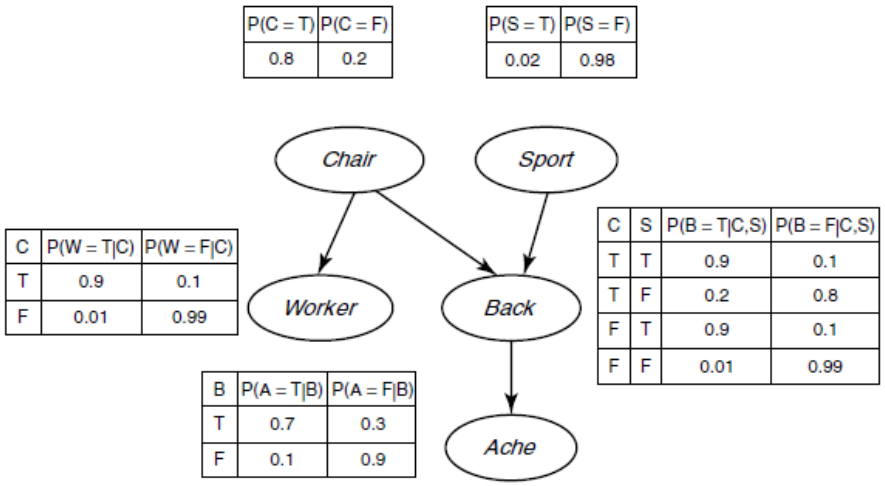

Figure 3.1.4 : Bayesian network example

The probabilistic relationships between the entities of the model (Politicians, Media and Voters) were modeled as connections of nodes in a Bayesian Network. Therefore each entity represents a node in this network.

In order to derive the conditional dependencies between the nodes of the Bayesian network, a set of Conditional Probability Tables (CPT) was derived. CPTs contain the conditional probabilities corresponding to the observed state of each node's parents in the Bayesian Network.

In case of voters of Type 1 (which is the majority of nodes in the network), exact inference of the joint probability distributions is possible, since all the nodes of their Markov blanket are fully observed. Therefore the conditional probability distribution for these nodes can be determined by using the corresponding probabilities from CPTs.

On the other hand, conditional probabilities related to voters of Type 2 cannot be computed in the same way. The difference in this case is that there are many nodes in their Markov blanket and marginalizing over these nodes would be computationally expensive. In order to overcome this problem, approximate inference is done by using Gibbs sampling.

3.2. Gibbs Sampling

Gibbs sampling ([7]) is a method of approximating inference in Bayesian Networks, when the joint distribution is not known explicitly, but the conditional distribution of each variable is known. This method is particularly well-adapted to sampling the posterior distribution of a Bayesian network, since Bayesian networks are typically specified as a collection of conditional distributions.

The Gibbs sampler runs a Markov chain on a set of random variables $(X_1, \dots, X_n)$ conditioned on a set of evidence variables $\boldsymbol{e} \triangleq (e_1, \dots, e_n)$. Its algorithm is the following:

- Initialize
  - Assign to every X_i one of its possible values $x_i, i = 1, \dots, n$
  - Let $\boldsymbol{x}^{(0)} = (x_1, \dots, x_n)$
- For every iteration $t = 1, 2, \dots$
  - Pick an index i, $1 \leq i \leq n$ at random (uniform)[2]
  - Sample x_i from $P(X_i | \boldsymbol{x}_{(-i)}^{(t-1)}, \boldsymbol{e})$, with the (-*i*) operator defined as $\boldsymbol{x}_{(-i)} \triangleq (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$
  - Set $\boldsymbol{x}^{(t)} = \left(\boldsymbol{x}_{(-i)} x_i\right)$

The sampler generates a sequence of samples $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(t)}, \dots$ from the Markov chain over all possible states. The stationary distribution of the Markov chain is the joint distribution

$$P(X_1, \dots, X_n | \boldsymbol{e})$$

[2]Different types of index selection can be chosen. The simplest case would be to go cyclically over all the indices.

Thus, drawing samples from the Markov chain at "long enough intervals"(i.e. allowing the chain to converge to the stationary distribution), gives independent samples from the distribution $P(X_1, \dots, X_n|\boldsymbol{e})$.

In order to make the fore mentioned algorithm clearer, as an example, the Gibbs sampler is going to be applied in the rain network (Figure 3.2.1)

Consider the inference problem of estimating $P(Rain|GrassWet = T, Sprinkler = T)$.

Since *GrassWet* and *Sprinkler* are set to true, the Gibbs sampler draws samples from

$P(Rain, Cloudy|GrassWet = T, Sprinkler = T)$

The Gibbs sampler for the rain network works as follows:

- Initialize
  - Let *Rain=T*, *Sprinkler = T*
  - Then $\boldsymbol{x}^{(0)} = (Rain = T, Sprinkler = T)$

- For $t = 1, 2, \dots$
  - Randomly pick variable to update from $\{Rain, Sprinkler\}$
  - If *Rain* was picked
    - Sample *Rain* from $P(Rain|Cloudy = [value]_{t-1}, Sprinkler = T, GrassWet = T)$
    - Set $\boldsymbol{x}^{(t)} = (Rain = [value]_t, Cloudy = [value]_{t-1})$

  - If *Cloudy* was picked
    - Sample *Cloudy* from $P(Cloudy|Rain = [value]_{t-1}, Sprinkler = T, GrassWet = T)$
    - Set $\boldsymbol{x}^{(t)} = (Rain = [value]_{t-1}, Cloudy = [value]_t)$

Therefore, we need conditional probabilities of *Rain* and *Cloudy* given values for all other variables. Such conditional probabilities can be obtained using the conditional probability tables (given on the figure) and the Markov blanket of the area of interest.
As an example the probability that *Cloudy* = *T* is going to be calculated, given that *Rain* = *True*, *Sprinkler* = *T* and *GrassWet* = *T*.

$$P(Cloudy = T|Rain = T, Sprinkler = T, GrassWet = T) \triangleq P(C|R, S, G)$$
$$= P(C|S, R) = \frac{P(C)P(S, R|C)}{P(S, R)} = \frac{P(C)P(S|C)P(R|C)}{P(C)P(S|C)P(R|C) + P(\neg C)P(S|\neg C)P(R|\neg C)}$$[3]
$$= \frac{0.5 \cdot 0.1 \cdot 0.8}{0.5 \cdot 0.1 \cdot 0.8 + 0.5 \cdot 0.5 \cdot 0.2} = 0.444$$

Clearly, $P(\neg C|R, S, G) = 1 - P(C|R, S, G) = 0.556$

Similarly, the probabilities $P(C|\neg R, S, G)$, $P(\neg C|\neg R, S, G)$ can be computed as well as $P(R|C, S, G)$, $P(\neg R|C, S, G)$, $P(R|\neg C, S, G)$, $P(\neg R|\neg C, S, G)$.

[3]Notation: $\neg C \overset{\triangleq}{\Leftrightarrow} C = False$

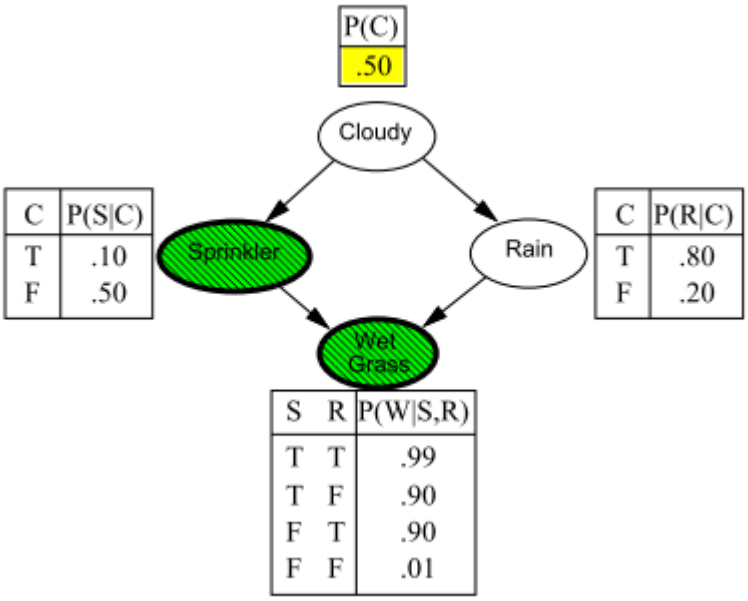

Figure 3.2.1: Gibbs sampling example

The above analysis gives the exact conditional probabilities needed by the Gibbs sampler. Samples are drawn by running the Markov chain implemented by the Gibbs sampler until convergence (to the stationary distribution).

3.3. The Simulation Algorithm

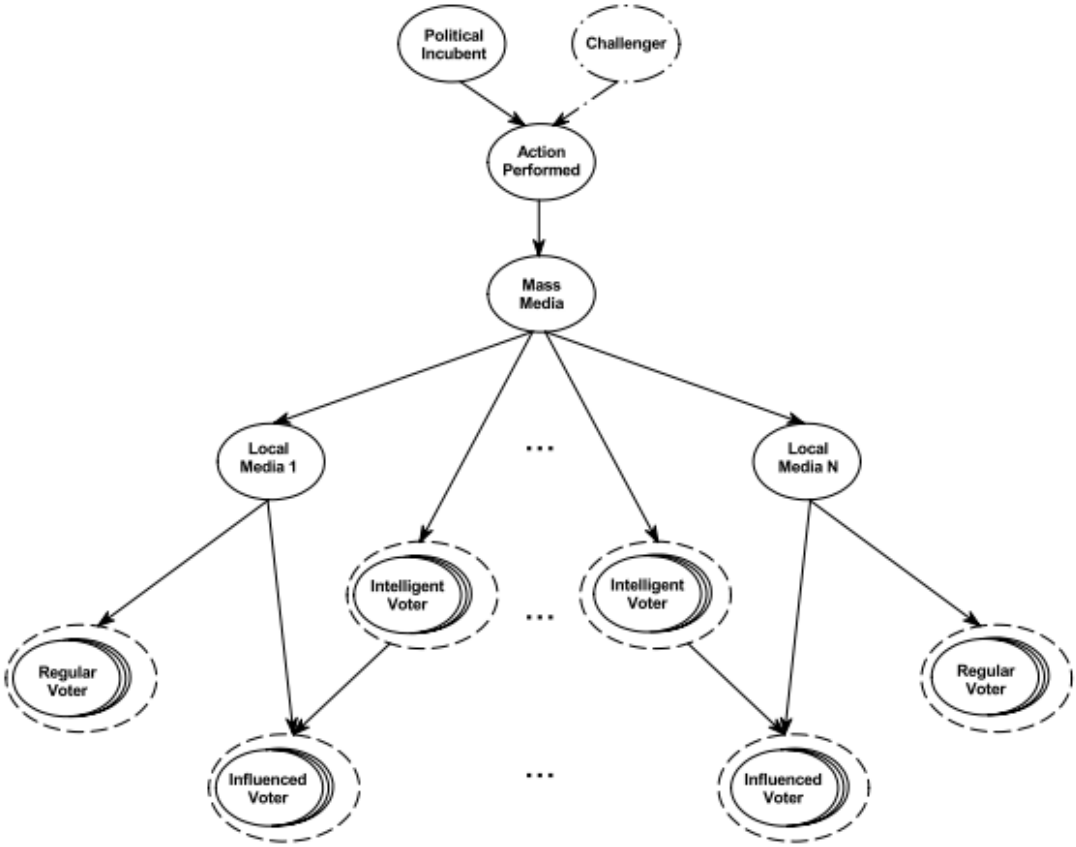

Figure 3.3.1: The Bayesian network used for the simulation

At the beginning of the algorithm a certain probability about the identity of the political person is given. One political entity exists and a dual entity, the "*Challenger"* is also created which has the opposite identity. The reason for creating this dual entity will become clear a bit later.

The politician, which is either "*Ethical*" or "*Opportunistic*" performs an action on every time step of the simulation algorithm. This action is observed by the "*Mass Media*" and follows the information flow path towards the lower layers (voters).*The Mass media* entity is an artificial entity that just makes the Bayesian network structure simpler.

At every iteration the action of the politician is "investigated" by the system. The "*Regular*" voters are influenced only by the "*Local Media*", on the other hand we have a more complicated interaction: the "*Intelligent*" voters receive unbiased information from the upper levels and then influence the "*Influenced*" voters (which are the "*Regular*" voters inside the social circle of the "*Intelligent*" voters) and give them the opportunity to access high level unbiased information and form a more accurate image, evaluating each political action independently (at the same time they are being influenced by the "*Local Media*"). At the end of each iteration the final decision of every voter is kept, and it is weighted with the decisions of the future iterations. Finally, every voter shapes their voting opinion which is counted at the very end as the result of the voting procedure.

Finally the quality of the elections result is judged based on the winner of the elections (*Politician Opportunistic* vs. *Challenger Ethical,* or the opposite).

Note: In order to avoid some unrealistic election results, like 0% 100% votes for an incumbent, a small portion of the population always votes in favour of him and another always votes against him.

3.4. Model Parameters

- $\beta \in [0,1]$, Probability that incumbent is "*Opportunistic*". The parameter is initially set to $\beta = \frac{1}{2}$. If observed $\beta \leq \frac{1}{2}$ then incumbent is said to be "*Opportunistic*".
- $\alpha \in [0,1]$, Probability that challenger is "*Ethical*". The parameter is initially set to $\alpha = \frac{1}{2}$. If observed $\alpha \geq \frac{1}{2}$ then challenger is said to be "*Ethical*".
- $\pi \in (\frac{1}{2}, 1)$, Probability of *Mass media* observing the true nature of incumbent's behavior. If honest behavior is done, mass media observe it as honest behavior with probability π, and as dishonest behavior with probability $1 - \pi$. Similarly, if dishonest behavior is done, *Mass media* observe it as dishonest behavior with probability π, and as honest behavior with probability $1 - \pi$. Model imposes that information structure is accurate enough, i.e. $\alpha \geq \frac{1}{2} \rightarrow \pi > \alpha, \alpha < \frac{1}{2} \rightarrow \pi > 1 - \alpha$
- $\eta_a, \eta_p \in [0,1]$, "*anti-incumbent*" media bias and "*pro-incumbent*" media bias, respectfully. If *Local media* observe honest behavior, they misreport it as "*bad*" news with probability η_a, and report it as "*good*" news with probability $1 - \eta_a$. Similarly, if *Local media* observe dishonest behavior they misreport it as "*good*" news with probability η_p, and report it as "*bad*" news with probability $1 - \eta_p$. *Local media* can be more or less biased in favor of or against the incumbent ($\eta_a < \eta_p$or $\eta_a > \eta_p$). Model imposes natural assumption that local media cannot be too much biased in both directions, i.e. $\eta_a + \eta_p \leq 1$.
- $\sigma \in [0,1]$, Probability of incumbent choosing a dishonest behavior. If incumbent is "*Ethical*" in nature then $\sigma = 0$.

$$\alpha \geq \frac{1}{2} \rightarrow \sigma = \frac{(2\alpha - 1)(1 - \eta_a)\pi + \eta_p(1 - \pi)}{[1 - (\eta_a + \eta_p)](2\pi - 1)\alpha}, \alpha < \frac{1}{2} \rightarrow \sigma = \frac{(1 - 2\alpha)[(1 - \eta_p)(1 - \pi) + \eta_a\pi)]}{[1 - (\eta_a + \eta_p)](2\pi - 1)\alpha}$$

- *R*, Private rent if dishonest behavior is chosen
- *s*, Incumbent's salary
- $\gamma_g, \gamma_b \in [0,1]$, Probability of voter deciding to vote for incumbent based on nature of news he/she receives ("*good*" or "*bad*" news, respectfully).

3.5. Conditional Probability Tables

- **Honest behavior by Incumbent**

$\alpha \geq \frac{1}{2}$, $R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

O	$P(H\|O)$
T	$1-\sigma$
F	1

$\alpha \geq \frac{1}{2}$, $R > [1-(\eta_a+\eta_p)](2\pi-1)s$

O	$P(H\|O)$
T	0
F	1

$\alpha < \frac{1}{2}$, $R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

O	$P(H\|O)$
T	$1-\sigma$
F	1

$\alpha < \frac{1}{2}$, $R > [1-(\eta_a+\eta_p)](2\pi-1)s$

O	$P(H\|O)$
T	0
F	1

- **Honest behavior observed by *Mass media***

O	H	$P(OH\|O,H)$
T	T	π
T	F	$1-\pi$
F	T	π
F	F	0

- **Local media delivers "*good*" news**

OB	EB	HB	$P(G\|OB,EB,HB)$
T	T	T	$1-\eta_a$
T	T	F	η_p
T	F	T	1
T	F	F	η_p
F	T	T	$1-\eta_a$
F	T	F	0
F	F	T	1
F	F	F	0

- **Regular voter votes for incumbent**

$\alpha \geq \frac{1}{2}$, $R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

G	$P(Reg\|G)$
T	$\frac{R}{[1-(\eta_a+\eta_p)](2\pi-1)s}$
F	0

$\alpha \geq \frac{1}{2}$, $R > [1-(\eta_a+\eta_p)](2\pi-1)s$

G	$P(Reg\|G)$
T	1
F	0

$\alpha < \frac{1}{2}$, $R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

G	$P(Reg\|G)$
T	1
F	$1-\frac{R}{[1-(\eta_a+\eta_p)](2\pi-1)s}$

$\alpha < \frac{1}{2}$, $R > [1-(\eta_a+\eta_p)](2\pi-1)s$

G	$P(Reg\|G)$
T	1
F	0

- **"*Intelligent*" voter votes for incumbent**

$\alpha \geq \frac{1}{2} \rightarrow R \in (0, (2\pi-1)s)$

HB	$P(Int\|HB)$
T	$\frac{R}{(2\pi-1)s}$
F	0

$\alpha \geq \frac{1}{2} \rightarrow R > (2\pi-1)s$

HB	$P(Int\|HB)$
T	1
F	0

$\alpha < \frac{1}{2} \rightarrow R \in (0, (2\pi-1)s)$

HB	$P(Int\|HB)$
T	1
F	$1-\frac{R}{(2\pi-1)s}$

$\alpha < \frac{1}{2} \rightarrow R > (2\pi-1)s$

HB	$P(Int\|HB)$
T	1
F	0

- **"*Influenced*" voter voting for incumbent**

$\alpha \geq \frac{1}{2} \rightarrow R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

G	$P(Inf\|G)$
T	$w_{personal}\frac{R}{[1-(\eta_a+\eta_p)](2\pi-1)s}+\sum w_{parent}$
F	0

$\alpha \geq \frac{1}{2} \rightarrow R > [1-(\eta_a+\eta_p)](2\pi-1)s$

G	$P(Inf\|G)$
T	$w_{personal}+\sum w_{parent}$
F	0

$\alpha < \frac{1}{2} \rightarrow R \in (0, [1-(\eta_a+\eta_p)](2\pi-1)s)$

G	$P(Inf\|G)$
T	$w_{personal}+\sum w_{parent}$
F	$w_{personal}(1-\frac{R}{[1-(\eta_a+\eta_p)](2\pi-1)s})+\sum w_{parent}$

$\alpha < \frac{1}{2} \rightarrow R > [1-(\eta_a+\eta_p)](2\pi-1)s$

G	$P(Inf\|G)$
T	$w_{personal}+\sum w_{parent}$
F	0

where $\boldsymbol{w_{personal}}$ **is a weight of personal vote, and** w_{parent} measures the influence of "*Intelligent*" voter

4. Simulation Results and Discussion

As one can imagine, the parameter space of the model is very big, and therefore only the ones of particular interest are going to be investigated, namely media bias and portion of "*Intelligent*" voters.

Important Note:
Instead of looking into the entire space $[0,100]$ for the portion of media bias and intelligent people, for simplicity reasons we only considered five different cases, varying from high to low percentage. Here it should be noted that the algorithm can last for quite some time in case of large population number in each community. In order to deal with this problem the notion of local communities helped. In each local community a different media bias is applied. More precisely, 5 communities are considered, having the same population structure and having values of media bias that start from 10% (pro- or anti- incumbent, depending on the case) in the first community, and reach 30% on fifth.

Therefore we decided to perform 5 simulations. On each simulation the number of "*Intelligent*" voters is fixed and each community has different amount of media bias as described above. The amount of "*Intelligent*" voters varies from 5% (1st simulation) to 30% (5th Simulation). The politician is considered to be ethical and the media bias has varying values but always anti-incumbent.

The *x-axis* of the plots is *time*, therefore we can see the popularity of the political candidate over time. If at the end of the elections the candidate gets more than 50% popularity he wins the elections otherwise he loses.

Next, the plot results of each simulation are shown:

CASE A: 30% of "*Intelligent*" voters

In this set of simulations the slightly decrease on the "*Intelligent*" voters population does not affect the voting results. The difference is that the "*Ethical*" politician wins

marginally (50.11%) at the cases of 25% and 30% anti-incumbent bias. In addition we can see the increase of the fluctuation until we converge to the final results. This could be considered as a sign of an equilibrium that is about to get unstable if the media bias gets increased.

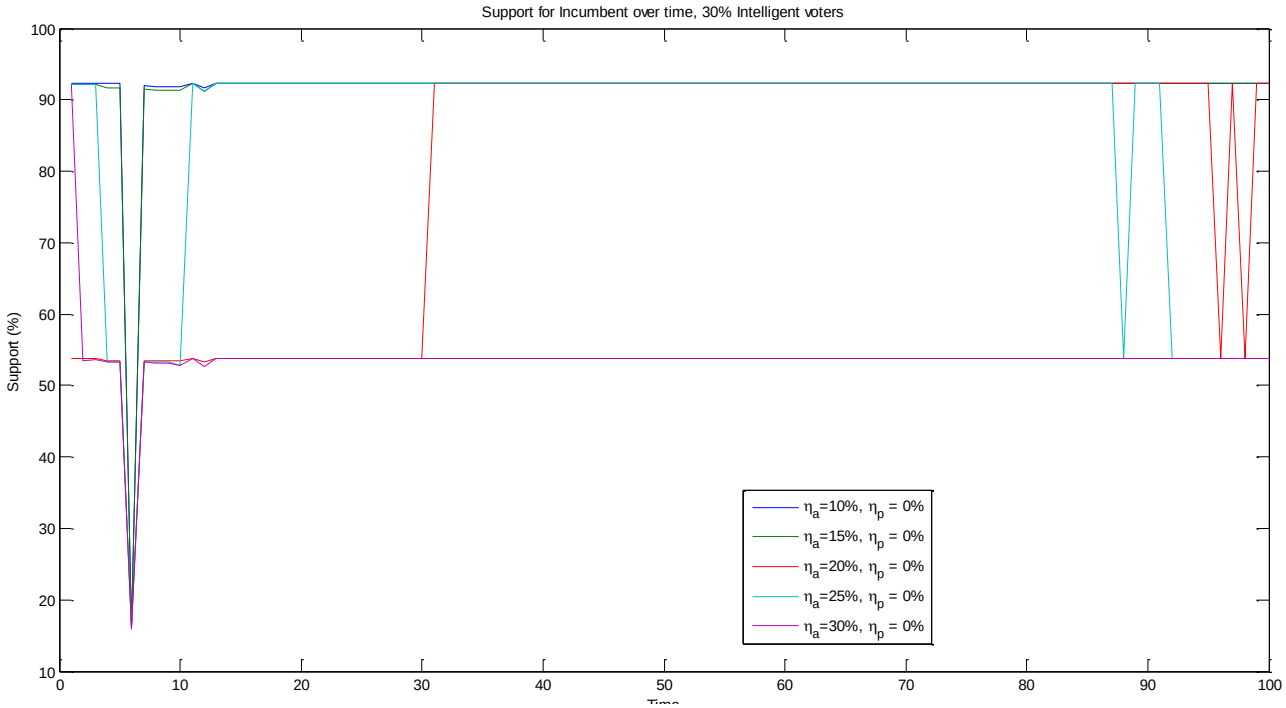

Figure 4.1: Simulation results, 30% "Intelligent" voters

As we can see from the Figure 1, in that extreme case the "*Ethical*" politician wins at all 5 occasions. In particular he achieves a clear victory when the amount of bias is under 25%. At the case that the bias is raised to 30% we can see that we still have a victory for the "*Ethical*" politician, but now the percentage is much lower (54%).

As a conclusion we can say that independently of the amount of bias, a percentage of 30% of "*Intelligent*" voters is securing the clean outcome of the procedure.

CASE B:25% of "*Intelligent*" voters

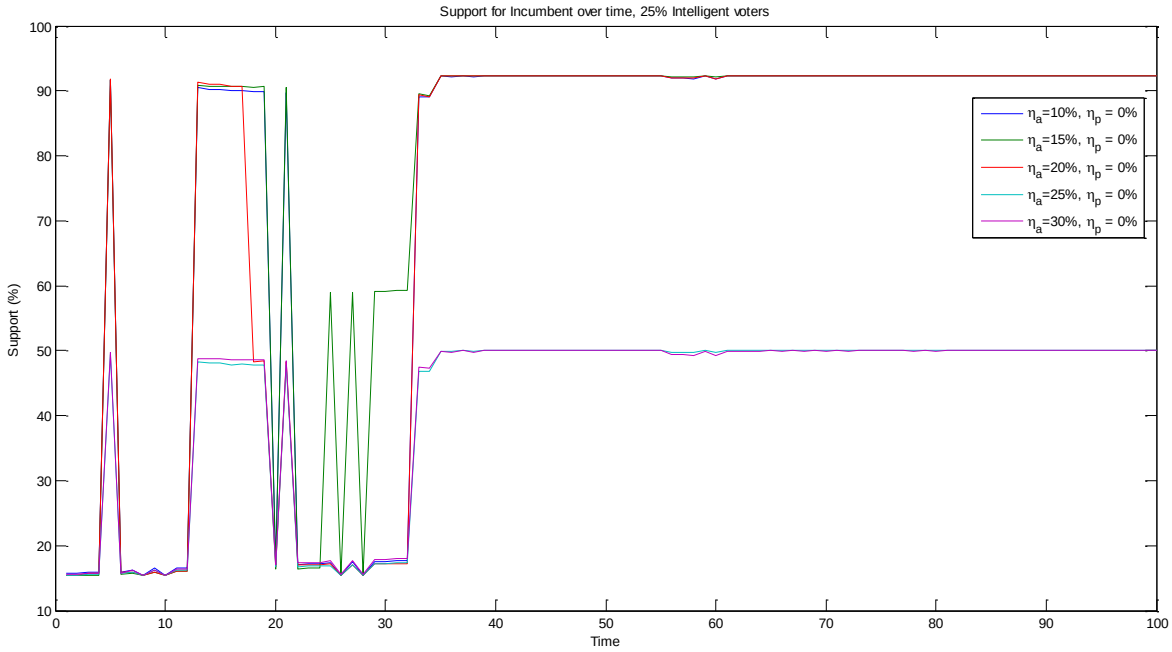

Figure 4.2: Simulation results, 25% "Intelligent" voters

CASE C: 20% of "*Intelligent*" voters

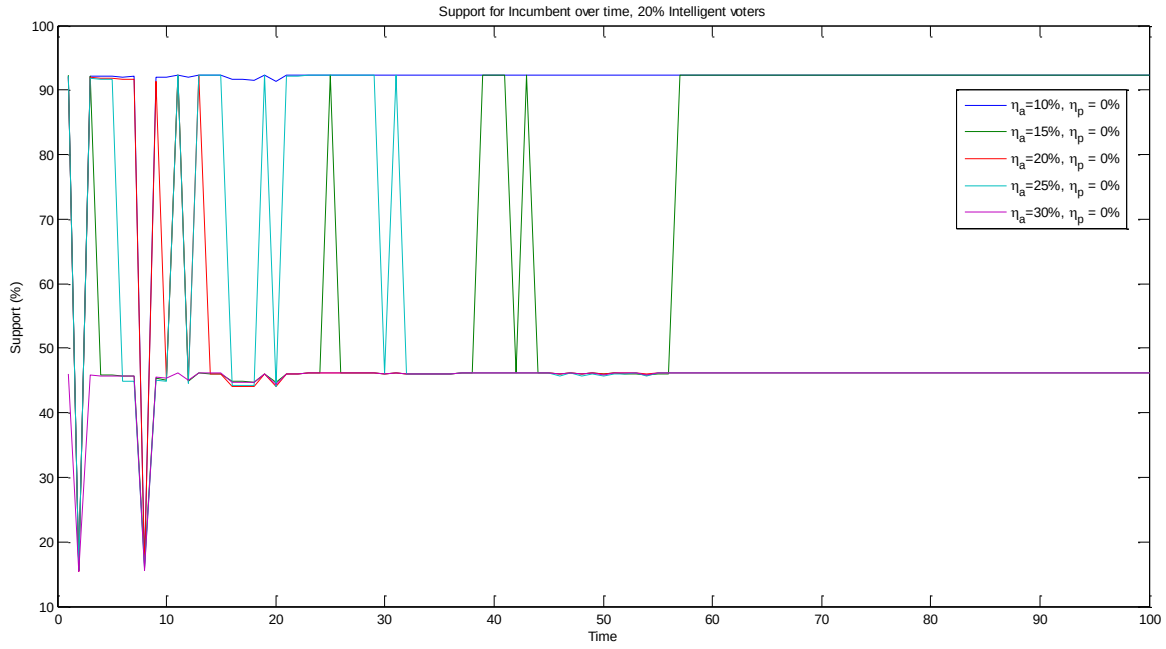

Figure 4.3 : Simulation results, 20% "Intelligent" voters

In this case it is the first time that we witness alteration of the results of the voting procedure by the media bias effect. At two simulations (25% and 30% of bias) the "*Ethical*" politician loses the political battle by 46%. The time needed to end to a conclusion is greater than before (55 iterations).

CASE D: 15% of "Intelligent" voters

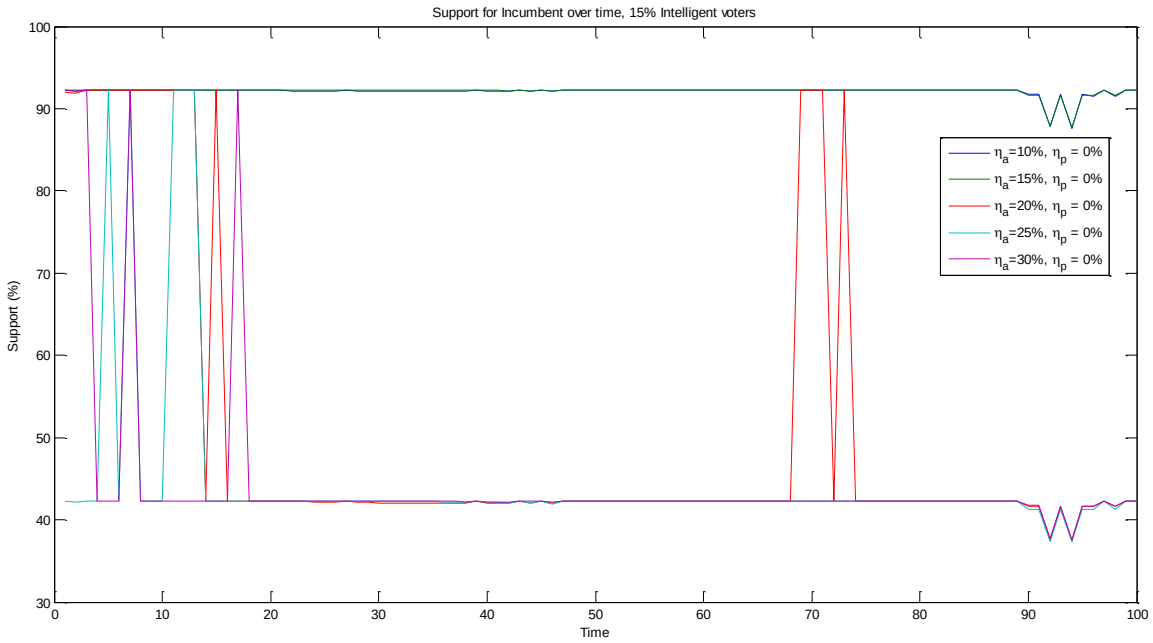

Figure 4.4 Simulation results, 15% "Intelligent" voters

At this simulation set we can see that the influence of bias is becoming more drastic as the population of "*Intelligent*" voters decreases. In these simulations the three out of five elections the "*Ethical*" loses the battle from anti-incumbent biased media (or from his challenger) and the "judgment moment" is increased.

CASE E: 10% of "*Intelligent*" voters

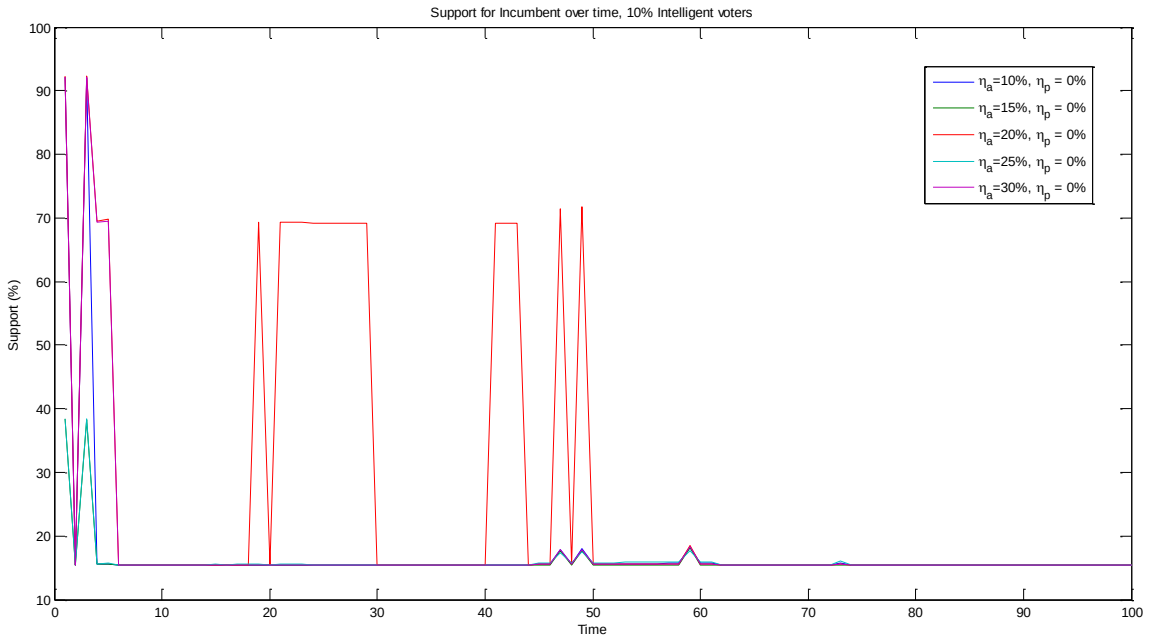

Figure 4.5 : Simulation results, 10% "Intelligent" voters

In figure 4 we can see that the media bias has taken the control of the voting outcome. The "*Ethical*" politician loses all battles and the time needed from the electoral in order to reach their decisions is suppressed (50 iterations). We can further argue that no matter the actions of the good politician, he will never be able to overcome the boundaries that the bias has set.

CASE F: 5% of "Intelligent" voters

NOTE: Here, we must note the uncertainty of the "*Intelligent*" voters influence. The influence of the "*Intelligent*" voters is set randomly at each simulation (i.e. the number of normal voters that can be affected by each "*Intelligent*" voter entity) and thus we must always keep in mind the possibility of variance in the results.

This is an example of this variance, larger media bias seems to be less effective for fewer amount of "*Intelligent*" voters, compared to the previous case. Rerunning the simulation leads to the expected result of full manipulation of the electoral results by the media, since the opposing force in this case is significantly low.

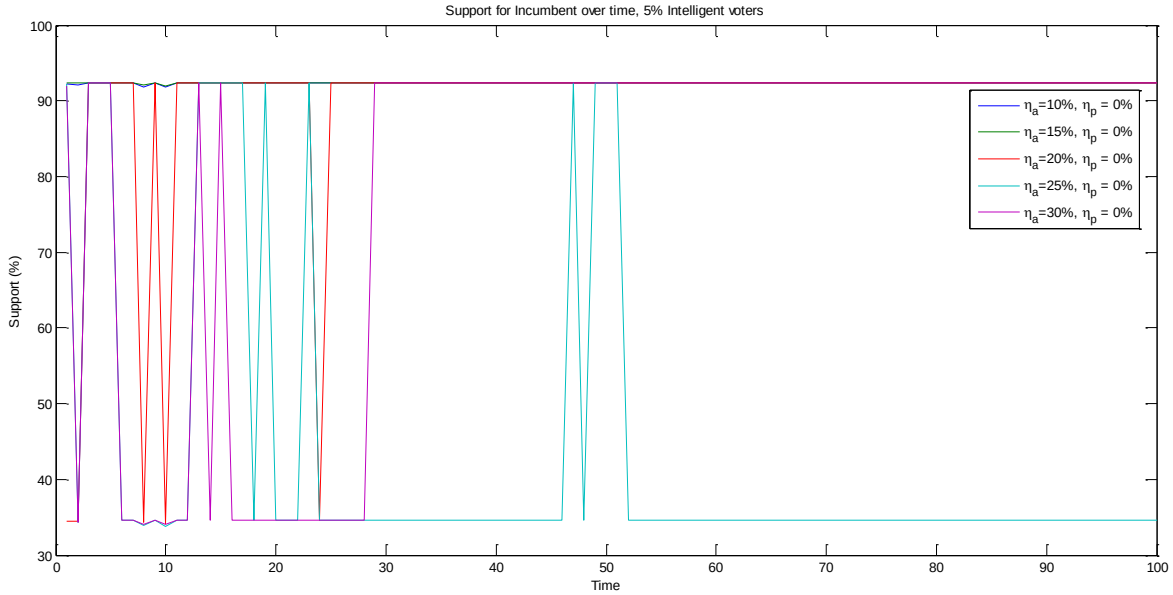

Figure 4.6 : Simulation results, 5% "Intelligent" voters

5. Summary and Outlook

On this point we have to revisit the imposed questions and see whether modelling/simulation done, can give some answer.

- How can media bias affect the electoral result?

We saw the case of "*Ethical*" political incumbent and how easily a distorted broadcast of his actions can totally alter the electoral result.

- How can "*Intelligent*" voters alter the electoral result?

We saw that the lower the portion of "Intelligent" voters, the easier it is for media to manipulate the electoral result.

- What is the relative increase of need for "*Intelligent*" voters versus increased media bias in order to achieve a good electoral result?

In order to answer to this question we have to do a quick review on the 5 simulations that were shown in the previous section.

Media bias %	**% of "*Intelligent*" voters**					
	5	10	15	20	25	30
10	X	X	√	√	√	√
15	X	X	√	√	√	√
20	X	X	X	√	√	√
25	X	X	X	X	~√	√
30	X	X	X	X	~√	√

We can see that for low levels of social awareness media manipulation is quite easy even with a very low amount of media bias. On the other hand we should keep in mind that firstly 30% of media bias as maximum examined value is probably quite low since, higher values could probably be found in real life media channels. On the other hand 30% of "*Intelligent*" voters – as were defined in previous chapters- is probably a very high amount.

Several improvements/further steps can be considered:

1. Enhancement of the parameters that encapsulate the influence of the "*Intelligent*" voter population.
2. Different implementation of the procedure with which the voters make their final decision. One could track the evolution of the political progress in order to produce decision probabilities based on the "history" of the events and not independently on every iteration. In this way the polarization observed at the result figures could be avoided.
3. Draw information from real statistical problems and increase the population numbers in order to achieve results that are closer to reality.
4. Different theoretical approach to the stated problem (i.e. use a different modeling tool instead of Bayesian networks).

Appendix : Simulation Code

```
display('------------------------------------------------------------');
display('MEDIA BIAS AND VOTING');
display('by I. Jovanovic, C. Lataniotis, F. Perazzi, A. Vouzas');
display('------------------------------------------------------------');

clear;
again = 'y';

while strcmp(again, 'y') || strcmp(again, 'Y')
    clear;
    display(' ');
    % Reading the number of Local communities
    reply = input('Insert the number of Local communities (2-5)? ', 's');
    numLocalCommunities = str2double(reply);
    while isnan(numLocalCommunities) || numLocalCommunities< 2 || ...
            numLocalCommunities> 5
        reply = input('Insert the number of Local communities (2-5)? ', 's');
        numLocalCommunities = str2double(reply);
    end

    % Reading the upper bound on number of voters in Local community
    reply = input(['Insert the upper bound of voters in of Local communities',...
        '(300<bound<4000)? '], 's');
    upperBound = str2double(reply);
    while isnan(upperBound) || upperBound< 300 || upperBound> 4000
         reply = input(['Insert the upper bound of voters in of Local communities',...
        '(300<bound<4000)? '], 's');
        upperBound = str2double(reply);
    end

    % Reading the Incumbent's nature
    reply = input('Insert the nature of Incumbent (O/E)? ', 's');
    while ~strcmp(reply, 'e') && ~strcmp(reply, 'E') && ~strcmp(reply, 'o') ...
            && ~strcmp(reply, 'O')
        reply = input('Insert the nature of Incumbent (O/E)? ', 's');
    end
    if strcmp(reply, 'e') || strcmp(reply, 'E')
       incumbent = 0.5 + 0.5 * rand(1);
       alpha = 0.5 * rand(1);
    else
        incumbent = 0.5 * rand(1);
        alpha = 0.5 + 0.5 * rand(1);
    end

    % Probability that mass media observing true nature of incumbent's action
    pi = piInitialization(alpha);

    % Reading the nature of media bias
    % Probabilities that media misreport news as "good" or "bad"
    % na - anti-incumbent bias
    % np - pro-incumbent bias
    display('Insert the nature of media bias');
    reply = input(['Anti-Incumbent, Pro-Incumbent, Media bias in both ', ...
        'directions (A/P/B)? '], 's');
    while ~strcmp(reply, 'a') && ~strcmp(reply, 'A') && ~strcmp(reply, 'p') && ...
            ~strcmp(reply, 'P') && ~strcmp(reply, 'b') && ~strcmp(reply, 'B')
        reply = input(['Anti-Incumbent, Pro-Incumbent, Media bias in both ', ...
        'directions (A/P/B)? '], 's');
    end
    if strcmp(reply, 'a') || strcmp(reply, 'A')
       type = 1;
    elseif strcmp(reply, 'p') || strcmp(reply, 'P')
        type = 2;
    else
        type = 3;
```

```
    end

    amount = input('Insert the amount of media bias (H/L)? ', 's');
    while ~strcmp(amount, 'h') && ~strcmp(amount, 'H') && ~strcmp(amount, 'l') ...
            && ~strcmp(amount, 'L')
        amount = input('Insert the amount of media bias (H/L)? ', 's');
    end
    if strcmp(reply, 'l') || strcmp(reply, 'L')
      amount = 1;
    else
        amount = 2;
    end

    [na, np] = mediaBiasInitialization(numLocalCommunities, type, amount);

    % Reading the number of Incumbent's actions
    reply = input('Insert the number of Incumbent''s actions? ', 's');
    numActions = str2double(reply);
    while isnan(numActions) || numActions<= 0
        reply = input('Insert the number of Incumbent''s actions? ', 's');
        numActions = str2double(reply);
    end

    % STARTING THE ELECTION PROCESS
    result = elections(numLocalCommunities, upperBound, incumbent, ...
    alpha, pi, na, np, numActions);

    % Another simulation?
    again = input('Another simulation (Y/N) [enter:N]? ', 's');
    if isempty(again)
        again = 'n';
    end
    while ~strcmp(again, 'y') && ~strcmp(again, 'Y') && ~strcmp(again, 'n') ...
            && ~strcmp(again, 'N')
        again = input('Another simulation (Y/N) [enter:N]? ', 's');
    end
end


function pi = piInitialization(alpha)

% Probability that mass media observing true nature of incumbent's action
% pi e [0.5, 1]
% alpha >= 0.5 -> pi > alpha
% alpha < 0.5 -> pi > 1 - alpha

pi = 0.5 + rand(1) * 0.5;
if alpha >= 0.5
    while pi <= alpha
        pi = 0.5 + rand(1) * 0.5;
    end
else
    while pi <= 1 - alpha
        pi = 0.5 + rand(1) * 0.5;
    end
end

end


function [na, np] = mediaBiasInitialization(numLocalCommunities, type, amount)

% Probabilities that media misreport news as "good" or "bad"
% na - anti-incumbent bias
% np - pro-incumbent bias
% na + np<= 1

% type = 1, Anti-incumbent bias
% type = 2, Pro-incumbent bias
```

```
% type = 3, Media bias in both directions

na = rand(1, numLocalCommunities);
np = rand(1, numLocalCommunities);

if amount == 1
    na = 0.2 * na;
    np = 0.2 * np;
elseif amount == 2
    na = 0.5 * na + 0.2;
    np = 0.5 * np + 0.2;
end

if type == 1
    np = zeros(1, numLocalCommunities);
elseif type == 2
    na = zeros(1, numLocalCommunities);
end

% SATISFYING CONSTRAINTS
for i = 1 : numLocalCommunities
      while na(i) + np(i) > 1
          na(i) = rand(1);
          np(i) = rand(1);

          if amount == 1
              na(i) = 0.2 * na(i);
              np(i) = 0.2 * np(i);
          elseif amount == 2
              na(i) = 0.5 * na(i) + 0.2;
              np(i) = 0.5 * np(i) + 0.2;
          end

          if type == 1
              np(i) = 0;
          elseif type == 2
              na(i) = 0;
          end
       end
end

end


function result = elections(numLocalCommunities, upperBound, incumbent, ...
    alpha, pi, na, np, numActions)
%% INITIALIZATIONS

% Number of Voters per community without members of political parties (1-10000)
numVoters = 1 + floor(rand(1, numLocalCommunities) * upperBound);
% Number of Intelligent voters per community (~<10%)
numIntelligentVoters = 1 + floor(numVoters .* (rand(1, numLocalCommunities) * 0.1));
% Number of Influenced voters per community (~<30%)
numInfluencedVoters = 1 + floor(numVoters .* (rand(1, numLocalCommunities) * 0.3));
% Number of Regular voters per community (the difference)
numRegularVoters = numVoters - numIntelligentVoters - numInfluencedVoters;

% Number of members of Incumbent's political party
incumbentMembers = 1 + floor(numVoters .* (rand(1, numLocalCommunities) * 0.2));
% Number of members of Challenger's political party
challengerMembers = 1 + floor(numVoters .* (rand(1, numLocalCommunities) * 0.1));

% Total number of voters
% Number of voters + Members of political parties + Incumbent + Challenger
totalVoters = sum(numVoters) + sum(incumbentMembers) + sum(challengerMembers) + 2;

% Displaying the input parameters
display(['There are ' num2str(numLocalCommunities) ' local communities.']);
display(['There are ' num2str(totalVoters) ' voters.']);
```

```
display(['There are ' num2str(sum(numIntelligentVoters))' intelligent voters.']);
display(['There are ' num2str(sum(numInfluencedVoters))' influenced voters.']);
display(['There are ' num2str(sum(numRegularVoters)) ' regular voters.']);
display(['There are ' num2str(sum(incumbentMembers)) ' members of Incumbent''s party.']);
display(['There are ' num2str(sum(challengerMembers)) ' members of Challenger''s party.']);
display(' ');

% Number of nodes in Bayesian network
% Incumbent + his behaviour + Observed Action +
% Number of media/local communities + 2 * Bias of media + numVoters
N = 3 + 3 * sum(numLocalCommunities) + sum(numVoters);

% Probability that incumbent is opportunistic
beta = 0.5;

% Mean values of media bias
naMean = mean(na);
npMean = mean(np);

% Incumbent's salary in CHF (20000-40000)
salary = 20000 + floor(rand(1) * 300) * 100;

% Weight for personal vote of Regular voter
personalW = 0.8;

% Number of actions done by incumbent
display(['The number of actions is ' num2str(numActions)'.']);
display(' ');

% Number of honest and dishonest actions done by Incumbent
honestActions = 0;
dishonestActions = 0;

% Number of Incumbent actions observed as honest or dishonest
honestActionsObserved = 0;
dishonestActionsObserved = 0;

% Number of good and bad news reported by each media
goodNewsReported = zeros(1, numLocalCommunities);
badNewsReported = zeros(1, numLocalCommunities);

%% GRAPH INITIALIZATION
[adj, par, wei] = graphInitialization(numLocalCommunities, numVoters, ...
    numIntelligentVoters, numInfluencedVoters, numRegularVoters);

%% CONDITIONAL PROBABILITY TABLES - BAYESIAN VARIABLE ENUMERATION
O = 1; % Incumbent is opportunistic
H = 2; % Incumbent choosing honest behaviour
HOA = 3; % Honest action observed by mass media
G = 4; % Media delivers good news


%% CONDITIONAL PROBABILITY TABLES
% Independent from the incumbent's behaviour
% 1 corresponds to TRUE, 2 corresponds to FALSE

% Conditional Probability Table
% Honest action observed by mass media given the nature of incumbent and
% nature of action
% CPT{HOA}(O, H)
CPT{HOA} = CPT_HonestActionObserved(pi);

% Dishonest action observed by mass media given the nature of incumbent and
% nature of action
% 1 - CPT{HOA}

% Conditional Probability Table
% Media delivers good news given the nature of action, existance of
% anti-incumbent media bias and pro-incumbent media bias
% CPT{i}(HOA, AB, OB}
```

```
% i - ordinal number of media
offset = 3;
for i = 1 : numLocalCommunities
    CPT{offset + 1} = CPT_GoodNewsDelivered(na(i),np(i));
    offset = offset + 3 + numVoters(i);
end

% Media delivers bad news given the nature of action, existance of
% anti-incumbent media bias and pro-incumbent media bias
% 1 - CPT{i}(HOA, AB, OB}
% i - ordinal number of media

%% OBSERVING VARIABLES
evidence{N} = [];

% Observing the incumbent's nature
if incumbent < beta
    evidence{1} = 1;
    display('Incumbent''s nature: OPPORTUNISTIC.');
else
    evidence{1} = 2;
    display('Incumbent''s nature: ETHICAL.');
end

% Observing the challenger's nature
if alpha >= 0.5
    display('Challenger''s nature: ETHICAL.');
else
    display('Challenger''s nature: OPPORTUNISTIC.');
end
display(' ');

% Observing the Anti-incumbent bias and Pro-incumbent bias
offset = 3;
for i = 1 : numLocalCommunities
    % Observe anti-incumbent bias
    if na(i) > 0
        evidence{offset + 2} = 1;
        display(['Media ' num2str(i) ' has Anti-incumbent bias in amount ' ...
            num2str(na(i) * 100) '%.']);
    else
        evidence{offset + 2} = 2;
        display(['Media ' num2str(i) ' does not have Anti-incumbent bias.']);
    end

    % Observe pro-incumbent bias
    if np(i) > 0
        evidence{offset + 3} = 1;
        display(['Media ' num2str(i) ' has Pro-incumbent bias in amount ' ...
            num2str(np(i) * 100) '%.']);
    else
        evidence{offset + 3} = 2;
        display(['Media ' num2str(i) ' does not have Pro-incumbent bias.']);
    end

    offset = offset + 3 + numVoters(i);
end
display(' ');

% Conditional Probability Table
% Initializing the number of Conditional probability tables for Regular
% voters
CPTreg{numLocalCommunities} = [];
CPTinf{numLocalCommunities} = [];

%% THE SIMULATION LOOP

% Counting the number of times a voter decided to vote for incumbent after
% receiving a news from media
individualVotes = int32(zeros(1, N));
```

```
for t = 1 : numActions
    % Private rent obtained from action
    rent = rentFromAction(evidence{1}, naMean, npMean, pi, salary);

    % Probability of opportunistic incumbent chosing a dishonest action
    sigma = sigmaCalculation(evidence{1}, alpha, rent, naMean, npMean, pi, salary);

    % Conditional Probability Table
    % Incumbent chooses honest behaviour given the nature of incumbent
    % CPT{H}(O)
    CPT{H} = CPT_HonestBehaviour(rent, naMean, npMean, pi, salary, sigma);

    % Incumbent chooses dishonest behaviour given the nature of incumbent
    % 1 - CPT{H}(O)

    % Observing the incumbents behaviour
    sample = rand(1);
    if sample < CPT{H}(evidence{O})
        evidence{H} = 1;
        % display(['Incumbent''s behavior ' num2str(t) ' is: HONEST.']);
        honestActions = honestActions + 1;
    else
        evidence{H} = 2;
        % display(['Incumbent''s behavior ' num2str(t) ' is: DISHONEST.']);
        dishonestActions = dishonestActions + 1;
    end

    % Observing the nature of observed action
    sample = rand(1);
    if sample < CPT{HOA}(evidence{O}, evidence{H})
        evidence{HOA} = 1;
        % display(['Incumbent''s behavior ' num2str(t) ' is observed as: HONEST.']);
        honestActionsObserved = honestActionsObserved + 1;
    else
        evidence{HOA} = 2;
        % display(['Incumbent''s behavior ' num2str(t) ' is observed as: DISHONEST.']);
        dishonestActionsObserved = dishonestActionsObserved + 1;
    end

    % Observing the Media delivers good news
    offset = 3;
    for i = 1 : numLocalCommunities
        sample = rand(1);
        if sample < CPT{offset + 1}(evidence{HOA}, evidence{offset + 2}, ...
                evidence{offset + 3})
            evidence{offset + 1} = 1;
            % display(['After Incumbent''s behavior ' num2str(t) ' Media ' ...
                % num2str(i) ' delivers GOOD NEWS.']);
            goodNewsReported(i) = goodNewsReported(i) + 1;
        else
            evidence{offset + 1} = 2;
            % display(['After Incumbent''s behavior ' num2str(t) ' Media ' ...
                % num2str(i) ' delivers BAD NEWS.']);
            badNewsReported(i) = badNewsReported(i) + 1;
        end

        offset = offset + 3 + numVoters(i);
    end

    % Conditional Probability Table
    % Intelligent voter votes for incumbent given the nature of action observed
    % by mass media - unbiased information
    % CPTint(OHA)
    CPTint = CPT_IntelligentVoter(alpha, rent, pi, salary);

    % Intelligent voter votes for challenger given the nature of action observed
    % by mass media - unbiased information
    % 1 - CPTint{j}(OHA), j - ordinal number of Intelligent voter
```

```
% Conditional Probability Table
% Influenced voter votes for incumbent given the nature of news reported by
% media without influence of Intelligent voters
% CPTinf{i}(G)
% i - ordinal number of Media
for i = 1 : numLocalCommunities
   CPTinf{i} = CPT_InfluencedVoter(alpha, rent, na(i), np(i), pi, salary, personalW);
end

% Influenced voter votes for challenger given the nature of news reported by
% media without influence of Intelligent voters
% 1 - CPTinf{i}(G)
% i - ordinal number of Media

% Conditional probability table
% Regular voter votes for incumbent given the nature of news reported by
% media
% CPTreg{i}
% i - ordinal number of Media
for i = 1 : numLocalCommunities
   CPTreg{i} = CPT_RegularVoter(alpha, rent, na(i), np(i), pi, salary);
end

% Regular voter votes for challenger given the nature of news reported by
% media
% 1 - CPTreg{i}
% j - ordinal number of Media

%%  VOTING
offset = 6;
for i = 1 : numLocalCommunities

   % GIBBS SAMPLING FOR INFLUENCED AND INTELLIGENT
   distribution = gibbsSampling(adj, par, wei, numIntelligentVoters(i), ...
   numInfluencedVoters(i), offset, CPTint, CPTinf{i}, evidence{HOA}, evidence{G});

   % Observing the votes from Intelligent and Influenced voters
   for j = 1 : numIntelligentVoters(i) + numInfluencedVoters(i)
     sample = rand(1);
     if sample < distribution(j)
        evidence{offset + j} = 1;
        individualVotes(offset + j) = individualVotes(offset + j) + 1;
     else            evidence{offset + j} = 2;
     end
   end

   % Voting for regular voters
   offset = offset + numIntelligentVoters(i) + numInfluencedVoters(i);
   for j = 1 : numRegularVoters(i)
      sample = rand(1);
      if sample <CPTreg{i}(evidence{offset - 2 - numIntelligentVoters(i) ...
            - numInfluencedVoters(i)})
         evidence{offset + j} = 1;
         individualVotes(offset + j) = individualVotes(offset + j) + 1;
      else
         evidence{offset + j} = 2;
      end
   end

   offset = offset + numRegularVoters(i) + 3;
end

%%  SIMULATION
% Calculating pie data
supportForIncumbent = sum(individualVotes> 0.5 * t);
supportForIncumbent = 100 * (supportForIncumbent + sum(incumbentMembers))...
      / totalVoters;
supportForChallenger = 100 - supportForIncumbent;
pie = [supportForIncumbent, supportForChallenger];
```

```
    % Calculating bar data
    honestActionsUntilNow = 100 * honestActions / t;
    dishonestActionsUntilNow = 100 - honestActionsUntilNow;
    honestActionsObservedUntilNow = 100 * honestActionsObserved / t;
    dishonestActionsObservedUntilNow = 100 - honestActionsObservedUntilNow;
    goodNewsReportedUntilNow = 100 * sum(goodNewsReported) / (t * numLocalCommunities);
    badNewsReportedUntilNow = 100 - goodNewsReportedUntilNow;
    bars = [honestActionsUntilNow, dishonestActionsUntilNow; ...
        honestActionsObservedUntilNow, dishonestActionsObservedUntilNow; ...
        goodNewsReportedUntilNow, badNewsReportedUntilNow];

    % Displaying the simulation
    figure(1)
    subplot(2,1,1), makePie(pie,{['Voting for Incumbent', 10]; ...
        ['Voting for Challenger', 10]}, ...
        ['SUPPORT FOR INCUMBENT IN LOCAL COMMUNITIES', 10, 'AT TIME ', num2str(t)]),
    subplot(2,1,2), makeBars(bars, {'Incumbent''s behavior (Honest / Dishonest)'; ...
        'Behaviors observed by Mass Media (Honest / Dishonest)'; ...
        'News reported by Local media (Good / Bad)'},...
        ['INCUMBENT''S MANDATE STATISTICS, \mu(\eta_a)=', num2str(100 * naMean), ...
        '%, \mu(\eta_p)=', num2str(100 *npMean), '%'])
    set(gcf, 'Position', get(0,'Screensize'));
end

%% DISPLAYING THE ELECTION RESULTS
display(' ');

% Displaying the statistics about Incumbent's mandate
display(['During his mandate Incumbent had 'num2str(honestActions) ...
    ' honest behaviors.']);
display(['During his mandate Incumbent had 'num2str(dishonestActions) ...
    ' dishonest behaviors.']);
display(' ');

% Displaying the statistics about number of observed actions
display(['Media observed 'num2str(honestActionsObserved) ...
    ' Incumbent''s behaviors as honest.']);
display(['Media observed 'num2str(dishonestActionsObserved) ...
    ' Incumbent''s behaviors as dishonest.']);
display(' ');

% Displaying the number of good and bad news delivered by local media
for i = 1 : numLocalCommunities
    display(['Local media ' num2str(i) ' delivered 'num2str(goodNewsReported(i)) ...
        ' good news, and ' num2str(badNewsReported(i)) ' bad news.']);
end
display(' ');

% COUNT THE VOTES, CALCULATE THE RESULT
% Summing all the votes
votingForIncumbent = (individualVotes> 0.5 * numActions);
votesForIncumbent = sum(votingForIncumbent) + sum(incumbentMembers) + 1;

% Displaying the number of votes for Incumbent
display(['After elections Incumbent got 'num2str(votesForIncumbent) ...
    ' votes or ' num2str(100 * votesForIncumbent/totalVoters) '%.']);
display(' ');

% Displaying the final election results
if votesForIncumbent/totalVoters> 0.5
    display('INCUMBENT GOT REELECTED!');
elseif votesForIncumbent/totalVoters == 0.5
    display('IT WAS A TIE!');
else
    display('INCUMBENT LOST THE ELECTIONS!');
end
display(' ');

result = votesForIncumbent/totalVoters;
```

```
end


function [adj, par, wei] = graphInitialization(numLocalCommunities, numVoters, ...
    numIntelligentVoters, numInfluencedVoters, numRegularVoters)
% GRAPH INITIALIZATION
% Adjacency list

% Number of nodes in Bayesian network
% Incumbent + his behaviour + Observed Action +
% Number of media/local communities + 2 * Bias of media + numVoters
N = 3 + 3 * sum(numLocalCommunities) + sum(numVoters);

% Initializing and connecting first two nodes
adj{N} = [];
adj{1} = [2, 3];
adj{2} = 3;

% Variable index offset
offset = 3;

% Initialization of local community by local community
for i = 1 : numLocalCommunities
    % Connecting Observed action with Media and Intelligent voters
    adj{3} = [adj{3}, offset + 1, offset + 4 : offset + 3 + numIntelligentVoters(i)];
    % Connecting Media with Influenced and Regular voters
    adj{offset + 1} = offset + 4 + numIntelligentVoters(i) : offset + 3 + numVoters(i);
    % ConnectingEthycal bias with Media
    adj{offset + 2} = offset + 1;
    % Connecting Opportunistic bias with Media
    adj{offset + 3} = offset + 1;

    % Randomizing connections of Intelligent voters to Influenced voters
    % while making sure that there every Influenced voters gets connected
    % to at least 1 Intelligent voter
    influences = rand(numIntelligentVoters(i), numInfluencedVoters(i));
    influences = influences > 0.9;
    while ~isempty(find(sum(influences) == 0, 1))
        correction = find(sum(influences) == 0, 1);
        influences(:, correction) = rand(numIntelligentVoters(i), 1) > 0.9;
    end

    % Connecting Intelligent voters to Influenced voters
    for j = 1 : numIntelligentVoters(i)
        adj{offset + 3 + j} = offset + 3 + numIntelligentVoters(i) + ...
            find(influences(j, :) ~= 0);
    end

    % Shifting the offset
    offset = offset + 3 + numVoters(i);
end

% Determining the parenting relationships between nodes in Bayesian network
par{N} = [];
for i = 1 : N
    if ~isempty(adj{i})
        for j = 1 : size(adj{i}, 2)
            par{adj{i}(j)} = [par{adj{i}(j)} i];
        end
    end
end

% Initialize weights for Influenced voters
% Normalize weights to sum to 0.2
offset = 6;
wei{N} = [];
for i = 1 : size(numInfluencedVoters, 2)
    offset = offset + numIntelligentVoters(i);
    for j = 1 : numInfluencedVoters(i)
```

```
        wei{offset + j} = rand(1, size(par{offset + j}, 2));
        wei{offset + j}(1) = 0;
        wei{offset + j} = 0.2 * wei{offset + j} ./ sum(wei{offset + j});
    end
    offset = offset + numInfluencedVoters(i) + numRegularVoters(i) + 3;
end


function HOA = CPT_HonestActionObserved(pi)

% Conditional Probability Table
% Honest action observed by mass media given the nature of incumbent and
% nature of action
% CPT{HOA}(O, H)

HOA = reshape([pi, pi, 1 - pi, 0], [2 2]);

end




function CPT = CPT_GoodNewsDelivered(na, np)

% Conditional Probability Table
% Media delivers good news given the nature of action, existance of
% anti-incumbent media bias and pro-incumbent media bias
% CPT{i}(HOA, AB, OB}
% i - ordinal number of media

CPT = reshape([1 - na, np, 1, np, 1 - na, 0, 1, 0], [2 2 2]);

end


function rent = rentFromAction(incumbent, naMean, npMean, pi, salary)

% Private rent obtained from action

if incumbent == 1
    rent = 2 * rand(1) * (1 - (naMean + npMean)) * (2 * pi - 1) * salary;
else
    rent = 0;
end

end


function sigma = sigmaCalculation(incumbent, alpha, rent, naMean, npMean, pi, salary)

% Probability of opportunistic incumbent chosing a dishonest action
if incumbent == 1
    if alpha >= 0.5
        if rent < (1 - (naMean + npMean)) * (2 * pi - 1) * salary
            sigma = (2 * alpha - 1) * (1 - naMean) * pi + npMean * (1 - pi);
            sigma = sigma / ((1 - (naMean + npMean)) * (2 * pi - 1) * alpha);
            sigma = sigma * 0.3;
        else
            sigma = 1;
        end
    else
        if rent < (1 - (naMean + npMean)) * (2 * pi - 1) * salary
            sigma = (1 - 2 * alpha) * ((1 - npMean) * (1 - pi) + naMean * pi);
            sigma = sigma / ((1 - (naMean + npMean)) * (2 * pi - 1) * alpha);
            sigma = sigma * 0.3;
        else
            sigma = 1;
        end
```

```
    end
else
    sigma = 0;
end

end


function CPT = CPT_HonestBehaviour(rent, naMean, npMean, pi, salary, sigma)

% Conditional Probability Table
% Incumbent chooses honest behaviour given the nature of incumbent

if rent < (1 - (naMean + npMean)) * (2 * pi - 1) * salary
        CPT = [1 - sigma, 1];
    else
        CPT = [0, 1];
end

end


function CPT = CPT_IntelligentVoter(alpha, rent, pi, salary)

% Conditional Probability Table
% Intelligent voter votes for incumbent given the nature of action observed
% by mass media - unbiased information

% Probability of Intelligent voter voting for incumbent
omega = rent / ((2 * pi - 1) * salary);

% Conditional probability table

if rent >= (2 * pi - 1) * salary
    CPT = [1, 0];
elseif rent == 0
    CPT = [1, 0];
else
    if alpha >= 0.5
        CPT = [omega, 0];
    else
        CPT = [1, 1 - omega];
    end
end

end


function CPT = CPT_InfluencedVoter(alpha, rent, na, np, pi, salary, personalW)

% Probability of Influenced voter voting for incumbent without influence of
% Intelligent voters
gamma = rent / ((1 - (na + np)) * (2 * pi - 1) * salary);

% Conditional probability table
% Influenced voter votes for incumbent given the nature of news reported by
% media
if rent >= (1 - (na + np)) * (2 * pi - 1) * salary
    CPT = [personalW, 0];
elseif rent == 0
        CPT = [1, 0];
else
    if alpha >= 0.5
        CPT = [personalW * gamma, 0];
    else
        CPT = [personalW, personalW * (1 - gamma)];
    end
end
```

```
end



function CPT = CPT_RegularVoter(alpha, rent, na, np, pi, salary)

% Probability of Regular voter voting for incumbent
gamma = rent / ((1 - (na + np)) * (2 * pi - 1) * salary);

% Conditional probability table
% Regular voter votes for incumbent given the nature of news reported by
% media
if rent >= (1 - (na + np)) * (2 * pi - 1) * salary
    CPT = [1, 0];
elseif rent == 0
    CPT = [1, 0];
else
    if alpha >= 0.5
        CPT = [gamma, 0];
    else
        CPT = [1, 1 - gamma];
    end
end

end



function distribution = gibbsSampling(adj, par, wei, numIntelligentVoters, ...
    numInfluencedVoters, offset, CPTint, CPTinf, HOA, G)

% Number of samples for Gibbs sampling
numSamples = 100;

% Determining the opposite decision
if G == 1
    NG = 2;
else
    NG = 1;
end

if HOA == 1
    DOA = 2;
else
    DOA = 1;
end

% Probability of each node
nodeProbability = 0.5;

% Randomizing the initial values for Intelligent and Influenced voters
votes = 1 + floor(rand(1, numIntelligentVoters + numInfluencedVoters) * 2);
votes = [votes ; zeros(numSamples - 1, numIntelligentVoters + numInfluencedVoters)];
votes = int8(votes);

% GIBBS SAMPLING
for i = 2 : numSamples

    for j = 1 : numIntelligentVoters
        % For each intelligent voter read it's children nodes
        influenced = adj{offset + j};
        % Initializing the arrays for conditional probabilities of child
        % nodes
        influencedForIncumbent = ones(1, size(influenced, 2));
        influencedNotIncumbent = ones(1, size(influenced, 2));

        % For each child
        for k = 1 : size(influenced, 2)
            % Observe it's parent node - Intelligent voters
            influences = par{influenced(k)};
```

```
        % Observe influence weights
        weights = wei{influenced(k)};

        % Read initial probability from Conditional probability table
        % for influenced voters
        probabilityIncumbent = CPTinf(G);
        probabilityNotIncumbent = CPTinf(NG);

        % For each Influence if decision is to vote for Incumbent then
        % add the influence. Otherwise, subtract it
        for l = 2 : size(influences, 2)
          if votes(i - 1, influences(l) - offset) == 1
             probabilityIncumbent = probabilityIncumbent + weights(l);

             if influences(l) - offset == j
                probabilityNotIncumbent = probabilityNotIncumbent + weights(l);
             else
                probabilityNotIncumbent = probabilityNotIncumbent - weights(l);
             end
          else
             probabilityIncumbent = probabilityIncumbent - weights(l);

             if influences(l) - offset == j
                probabilityNotIncumbent = probabilityNotIncumbent - weights(l);
             else
                probabilityNotIncumbent = probabilityNotIncumbent + weights(l);
             end
          end
        end

        % Corrections
        if probabilityIncumbent< 0
           probabilityIncumbent = 0;
        end

        if probabilityIncumbent> 1
           probabilityIncumbent = 1;
        end

        if probabilityNotIncumbent< 0
           probabilityNotIncumbent = 0;
        end

        if probabilityNotIncumbent> 1
           probabilityNotIncumbent = 1;
        end

        % Put the corresponding probabilities into array for
        % conditional probabilities
        influencedForIncumbent(k) = probabilityIncumbent;
        influencedNotIncumbent(k) = probabilityNotIncumbent;
     end

     % Calculating the approximated probability of Intelligent voter voting for
     % incumbent
     probability = nodeProbability * CPTint(HOA) *...
        prod(influencedForIncumbent);
     marginalization = nodeProbability * CPTint(HOA) * ...
        prod(influencedForIncumbent) + (1 - nodeProbability) * CPTint(DOA) * ...
        prod(influencedNotIncumbent);
     probability = probability / marginalization;

     % Observing the Intelligent voter's vote
     sample = rand(1);
     if sample < probability
        votes(i, j) = 1;
     else
        votes(i, j) = 2;
     end
  end
```

```
    % Shifting the offset
    infOffset = offset + numIntelligentVoters;

    % For each Influenced voter
    for j = 1 : numInfluencedVoters
        % Observe it's parent node - Intelligent voters
        influences = par{infOffset + j};
        % Observe influence weights
        weights = wei{infOffset + j};
        % Read initial probability from Conditional probability table
        % for influenced voters
        probabilityIncumbent = CPTinf(G);

        % For each Influence if decision is to vote for Incumbent then
        % add the influence. Otherwise, subtract it
        for l = 2 : size(influences, 2)
            if votes(i, influences(l) - offset) == 1
                probabilityIncumbent = probabilityIncumbent + weights(l);
            else
                probabilityIncumbent = probabilityIncumbent - weights(l);
            end
        end

        % Observing the Influenced voter's vote
        sample = rand(1);
        if sample <probabilityIncumbent
            votes(i, j + numIntelligentVoters) = 1;
        else
            votes(i, j + numIntelligentVoters) = 2;
        end
    end

end

distribution = sum(votes == 1) / 100;

end


function  makePie(X, labels, ptitle)
% Draws a pie for displaying Incumbent support in Local communities

% Drawing a pie
explode = zeros(size(X));
[~, offset] = max(X);
explode(offset) = 1;
h = pie(X, explode);

colormap winter

textObjs = findobj(h, 'Type','text');
oldStr = get(textObjs, {'String'});
val = get(textObjs, {'Extent'});
oldExt = cat(1, val{:});

Names = labels;
newStr = strcat(Names, oldStr);
set(textObjs, {'String'}, newStr);

val1 = get(textObjs, {'Extent'});
newExt = cat(1, val1{:});
offset = sign(oldExt(:, 1)) .* (newExt(:, 3) - oldExt(:, 3)) / 2;
pos = get(textObjs, {'Position'});
textPos =  cat(1, pos{:});
textPos(:,1) = textPos(:, 1) + offset;
set(textObjs, {'Position'}, num2cell(textPos, [3, 2]));

title(ptitle, 'FontWeight', 'bold');
```

```
end


function  makeBars( X, labels, ptitle)
% Draws a bars for displaying the statistics about Incumbent's mandate

% Drawing bars
bar(X);
title(ptitle, 'FontWeight','bold');
set(gca, 'XTickLabel', labels);
ylabel('%');

end
```

References

[1] ‘The rise and decline of nations: economic growth, stagflation and social rigidities ’, M. Olson, Yale University Press 1982.

[2] ‘Political Accountability, Electoral Control and Media Bias’, T. Adachi, Y. Hizen, November 2010.

[3] ‘The Fox News Effect: Media bias and Voting’, S. Della Vigna, E. Kaplan, The Quarterly Journal of Economics, August 2007.

[4] ‘Media Bias and Electoral Competition’, A. Chakraborty, P. Gosh, Not yet published, November 2010..

[5] ‘Press Accuracy Rating Hits Two Decade Low: Public Evaluations of the News Media: 1985-2009’, Pew Research Center for the People & the Press, 2009. http://peoplepress.org/report/543/

[6] ‘Bayesian Networks’, Ruggeri F., Faltin F., Kenett R., Encyclopedia of Statistics in Quality & Reliability, Wiley & Sons, 2007.

[7] ‘Pattern Recognition and Machine Learning’, C. Bishop, 1st edition, Springer, 2006.

[8] ‘Bayesian Network’, Wikipedia: The Free Encyclopaedia, Wikimedia Foundation Inc (http://en.wikipedia.org/wiki/Bayesian_network).