

SIMULATION OF AGENT BASED DISEASE SPREAD MODELS

by

Ayyüce Begüm Bektaş

B.S. Industrial Engineering, Sabancı University, 2013

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2017

SIMULATION OF AGENT BASED DISEASE SPREAD MODELS

APPROVED BY:

Assoc. Prof. Wolfgang Hörmann

(Thesis Supervisor)

Assist. Prof. Kemal Dinçer Dengeç

Prof. Refik Güllü

DATE OF APPROVAL: 17.05.2017

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisor Assoc. Prof. Wolfgang Hormann of the Faculty of Engineering at Boğaziçi University. He systematically allowed this paper to be my own work, and guided me whenever I needed it.

I must express my sincere gratitude to my family for providing me with unfailing support and continuous encouragement throughout my years of study. My father Kemalettin Bektaş, my mother Şaziye Taciser Bektaş and my sister Hande Mefkûre Özkaya, this accomplishment would not have been possible without you. My dear friends Gizem Öztürk, Mehmet Özyürek, Gizem Zorlu, thank you for supporting, correcting and motivating me all the time with your comments and critics, I am very lucky to have you.

Thank you.

ABSTRACT

SIMULATION OF AGENT BASED DISEASE SPREAD MODELS

Disease spread models can be used to see how different parameters might affect the spread of a particular disease and to know how a disease can be controlled before it becomes common in public.

By simulating agent based disease spread models, it is aimed, in case of an outbreak, to predict the number of people that will get affected and to be able to take important precautions such as antiviral agents as a preventive mechanism before people become infected. In this thesis, agent based simulation of disease spread models is studied and Turkish population structure is considered. Possible data structures to be included in the simulation, possible infection parameter values and contact structures are analyzed. A synthetic population structure is formed and the simulation is done using this population structure. A function that calculates approximate R_0 value for a given population to calibrate infection probabilities is built. The relation between the R_0 value and the expected outbreak behavior is analyzed in detail. The reliability of R_0 value described in the literature is discussed. The simulation model is an improvable model in case that new parts are added to the population structure or that any small detail is required to be changed. The model and the population structure are highly flexible since they are developed using R mathematical programming language starting from the beginning.

ÖZET

HASTALIK YAYILMA MODELLERİNİN BİREY TABANLI SİMÜLASYONU

Hastalık yayılma modelleri farklı parametrelerin bir hastalığın yayılımını nasıl etkilediğini görmek ve toplumda bir hastalığın artış göstermesinden önce nasıl kontrol edilebileceğini bilmek için kullanılabilir.

Hastalık yayılma modellerinin birey tabanlı simülasyonu (benzetimi) yapılarak, bir salgın anında bu salgından etkilenecek kişi sayısını tahmin etmek, enfeksiyon öncesi antiviral ajan kullanımı gibi önemli önleyici tedbirler alabilmek hedeflenmektedir. Bu tezde de hastalık yayılımının birey tabanlı simülasyonu çalışılmış ve bu çalışmada Türk toplum yapısı öncelikle göz önünde bulundurulmuştur. Benzetimde kullanılması muhtemel veri yapıları incelenmiş, muhtemel enfeksiyon bulaşma olasılık değerleri ve toplumdaki temas yapıları analiz edilmiştir. Sentetik bir toplum yapısı oluşturulmuş ve bu toplum ile simülasyon yapılmıştır. Simülasyon modeli üzerinden senaryo analizleri yapılmış, aşı ile hastalığa bağışıklık kazanmanın ve/veya hastalık anında evde dinlenmenin salgın hastalığın genel karakterini nasıl etkileyeceği tahmin edilmeye çalışılmıştır. Yaklaşık R_0 değerini toplumun temas yapılarını göz önünde bulundurarak hesaplayan bir fonksiyon tasarlanmıştır. Literatürde tanımlanan R_0 değerinin homojen olmayan topluluklardaki salgın hastalık karakterini yorumlamada ne denli güvenilir olduğu tartışılmış, R_0 değeri ve beklenen salgın hastalık karakteristiği arasındaki ilişki detaylı bir şekilde incelenmiştir. Simülasyon modeli başından itibaren R programlama diliyle yapıldığından geliştirilebilir ve esnektir. Veri yapısında değişiklikler olduğunda ya da herhangi bir detayın eklenmesi gerekliliğinde model kolaylıkla değiştirilebilecektir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT.....	iv
ÖZET	v
LIST OF FIGURES	viii
LIST OF TABLES.....	xiii
1. INTRODUCTION	1
2. DISEASE SPREAD MODELS AND AGENT BASED MODELS FOR DISEASE SPREAD	3
2.1. Epidemic Models and the SIR Model	3
2.2. Agent Based Models for Disease Spread	4
2.3. Literature Research	6
2.3.1. International Literature Research	6
2.3.2. Turkish Literature Research	8
3. A GENERIC IMPLEMENTATION OF SIR MODEL WITH MIXING GROUPS IN R.....	12
3.1. Definition of the Discrete Time Stochastic Model.....	12
3.2. Inputs for the Simulation Program	13
3.3. The General Idea of the Code and Implementation in R.....	16
3.4. Controlling the Correctness of the Implementation	23
3.5. Code Performance and an Experiment with a Synthetic Population Structure.....	27

4. SELECTING MODEL PARAMETERS	31
4.1. Creating the Population Matrix and a Synthetic Population Structure.....	31
4.1.1. Use of Population Structure Code to Generate Different Population Matrices	37
4.2. The Information that the Simulation Model is Built on and Infection Probability Calibration.....	38
5. SCENARIOS	47
5.1. Scenario 1: Vaccination	47
5.2. Scenario 2: Stay Home.....	48
5.2.1. Debugging of Scenario 2	51
6. EXPERIMENTS AND RESULTS	55
6.1. The Baseline Model Results.....	55
6.2. Results of the Scenarios	59
6.2.1. Results with probHome 0.2	59
6.2.2. Results with probHome 0.5	63
6.2.3. Results with probImm=0.15 and probHome=0.2	65
6.2.4. Results of Simulation with probImm=0.4 and probHome=0.2	68
6.2.5. Results of Simulation with probImm=0.15 and probHome=0.5	72
6.2.6. Results of Simulation with probImm=0.4 and probHome=0.5	75
7. CONCLUSION.....	78
REFERENCES	80
APPENDIX A: POPULATION GENERATING CODE	83

LIST OF FIGURES

Figure 3.1.	Screenshot of an example population matrix with 5 families, 2 neighborhoods, 2 schools and 1 workplace.....	15
Figure 3.2.	An example of a state matrix containing state and endState columns added to the population matrix.....	16
Figure 3.3.	The tables created by “table” command. The tables in the figure refer to neighborhood mixing groups, school/workplace mixing groups and family mixing groups, respectively.	17
Figure 3.4.	Table showing the number of infected and susceptible in community...	18
Figure 3.5.	The main section of the code.	19
Figure 3.6.	A section from $n \times d$ matrix.	22
Figure 3.7.	A section from the array storing the attack rate of each repetition.	23
Figure 3.8.	A screenshot of the population matrix that is used to trace the simulation program.	24
Figure 3.9.	Outputs generated for controlling the correctness of the implementation.....	25
Figure 3.10.	Outputs generated for controlling the correctness of the implementation.....	26
Figure 3.11.	Outputs generated for controlling the correctness of the implementation.....	27
Figure 3.12.	An example part from one of the primitive synthetic population matrices.	28

Figure 4.1. Family, neighborhood and community structure.	32
Figure 4.2. An example of fsizeTable. The column fsize refers to family size and the column fquantity indicates how many families of each size there are, in one neighborhood.	33
Figure 4.3. An example from matrix that is used to get the output population matrix.	36
Figure 4.4. An example from the population matrix formed to be used in the simulation model.	36
Figure 4.5. Example of population generating function (popdat) with all its inputs..	38
Figure 4.6. Output of the function.	40
Figure 4.7. Function to calculate approximate R_0 value.	41
Figure 4.8. Trial with factor value=1 and the function output.	45
Figure 4.9. Trial with factor value=0.4 and the function output.	45
Figure 4.10. Trial with factor value=0.3317 and the function output.	46
Figure 5.1. Code part that shows passing directly from S to R.	48
Figure 5.2. A section from the code.	49
Figure 5.3. A section from the code that lets all people with state "IH" pass to state "H".	49
Figure 5.4. A section from the code.	50
Figure 5.5. Code showing exiting the loop with break command.	51
Figure 5.6. Screenshot of the primitive population matrix formed for debugging....	51
Figure 5.7. Tracing output for for days 1 and 2.....	52
Figure 5.8. Tracing output for days 3 and 4.	53
Figure 5.9. Tracing output for days 5-9.....	53

Figure 6.1.	A plot showing the average number of infected people per day and number of infected people per day for random repetitions.	56
Figure 6.2.	Histogram of attack rates of the baseline model, mean value is 0.551. .	57
Figure 6.3.	Histogram of maximum number of infected on a single day of each repetition.	57
Figure 6.4.	Histogram of disease peaking day of each independent repetition.	58
Figure 6.5.	Histogram of time when the epidemic dies out.....	58
Figure 6.6.	First step of calculating the approximate R_0 value.....	59
Figure 6.7.	Second step of calculating the approximate R_0 value.	60
Figure 6.8.	A plot showing the average number of infected people per day and the number of infected people per day for random repetitions with probHome=0.2 and probImm=0.	61
Figure 6.9.	Histogram of attack rates with probHome=0.2 and probImm=0.	61
Figure 6.10.	Histogram of maximum number of infected of each independent repetition with probHome=0.2 and probImm=0.	62
Figure 6.11.	Histogram of disease peaking days of each repetition.	62
Figure 6.12.	A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.	63
Figure 6.13.	Histogram of attack rates with probHome=0.5.	64
Figure 6.14.	Histogram of maximum number of infected on a single day of each repetition.	64
Figure 6.15.	Histogram of disease peaking days of each repetition.	65
Figure 6.16.	A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.2 and probImm=0.15.	66
Figure 6.17.	Histogram of attack rates of combined scenarios with probImm=0.15 and probHome=0.2.....	67

Figure 6.18. Histogram of maximum number of infected on a single day of each repetition.	67
Figure 6.19. Histogram of disease peaking days.	68
Figure 6.20. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.2 and probImm=0.4.	69
Figure 6.21. Histogram showing the attack rates of simulations with probHome=0.2 and probImm=0.4.	71
Figure 6.22. Histogram of maximum number of infected on a single day of each repetition.	71
Figure 6.23. Histogram of disease peaking days.	72
Figure 6.24. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.15.	73
Figure 6.25. Histogram of attack rates with probHome=0.5 and probImm=0.15.	73
Figure 6.26. Histogram of maximum number of infected on a single day of each repetition.	74
Figure 6.27. Histogram of peaking days.	74
Figure 6.28. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.4.	75
Figure 6.29. Histogram showing the attack rates of simulations with probHome=0.5 and probImm=0.4.	76
Figure 6.30. Histogram of maximum number of infected on a single day of each repetition.	76
Figure 6.31. Histogram of disease peaking days.	77

Figure A.1. Population generating code.	83
--	----



LIST OF TABLES

Table 3.1. The input parameters of the simulation model and their explanations.....	14
Table 3.2. Runtime durations(s) of the experiments with an approximate R_0 value of 1.4. Size refers to population size.....	29
Table 4.1. Inputs to form the population matrix and their explanations.	35
Table 4.2. Inputs of the function.	40
Table 6.1. Simulation outputs of the baseline model.	56
Table 6.2. Simulation outputs of Scenario 2 with probHome=0.2.	60
Table 6.3. Simulation outputs with probHome=0.5, probImm=0.	63
Table 6.4. Simulation outputs of combined scenarios with probImm=0.15 and probHome=0.2.	66
Table 6.5. Simulation outputs of combined scenarios with probImm=0.4 and probHome=0.2.	68
Table 6.6. Average attack rates of 8 schools when the approximate calculated R_0 is 0.84 and when the average total attack rate is 0.164.	70
Table 6.7. Simulation outputs with probHome=0.5 and probImm=0.15.	72
Table 6.8. Simulation outputs of combined scenarios with probImm=0.4 and probHome=0.5.	75

1. INTRODUCTION

Disease spread modelling has become popular recently, not only because it is discussed in many domains such as medicine, biology, engineering, but also as the topics that can be discovered and the models that can be built gained more and more importance through the great development in technology. As for simulating agent based disease spread models, the aim is, in case of an outbreak, to predict the number of people that will be affected and to be able to assess the success of intervention strategies such as vaccination or the use of antiviral drugs.

Modelling a health related issue with engineering information and background can be considered as a great motivation for studying this subject. Agent-based simulation of disease spread models is a promising study domain, since it will be possible with these models to predict the number of infectious, susceptible, recovered and immune people resulting from a pandemic in a known time interval, to take precautions such as using antiviral agents, storing sufficient amount of vaccines, planning hospital usage in emergency or withstanding, surviving a bioterrorist attack.

The objective of this thesis is to build an agent based (individual based) simulation model for disease spread, that is flexible enough to be used everywhere, especially in Turkey. For this objective, possible data for the simulation code and possible infection parameter values are analyzed.

During the literature research I have observed that no one has ever used individual based simulation for disease spread modeling in Turkey. This was one of the motivations to select this subject, which aims to combine theoretical and practical information to arrive at practically useful results.

The simulation code is run with different input sets. First, several primitive population structures are created to check the model correctness and performance; then, a population structure that is not too detailed but carries the main characteristics of the Turkish population is built to be used for the simulation code together with the calibrated infection probabilities. Even though I used the population structure that carries the main characteristics of a middle sized Turkish population, the simulation code is flexible and works for any given population structure.

The program is coded by using R mathematical programming language which is known for its good performance and for being much faster than Matlab. Even though C++ or other object oriented languages are often used for disease spread modeling, I chose R since it is easy to debug the code and interpret the results. R is an intuitive language and everyone that is familiar with programming can use R.

This thesis first introduces agent based models for disease spread and gives a small overview of related literature in Part 2. It then gives in Part 3, the details about the simulation program that has been built by using R mathematical programming language. It continues with the details of the synthetic population structure that has been created to be used in the simulation program and gives the detailed information about selecting model parameters in Part 4. In Part 5, it describes the scenarios that are tried after the baseline model building phase. Part 6, it depicts the results of several simulation cases and interprets the results. Lastly, in Part 7, it concludes with a synthesis of the work that has been done, of its results and of possible future steps.

2. DISEASE SPREAD MODELS AND AGENT BASED MODELS FOR DISEASE SPREAD

This part gives a brief information about disease spread models, agent based models and contains the related literature research.

2.1. Epidemic Models and the SIR Model

Epidemic models can be seen as simplified forms of real world phenomena. They are built to depict the transmission of a communicable disease through humans. They are generally used to find out whether a specific strategy against epidemics would work or not. Thus, having good knowledge about the outcomes of an outbreak, it is possible to take the necessary precautions that could decrease the related damage.

In epidemic modelling, compartments are used to identify the stages of an infection. Compartments refer to the stages of an epidemic. It can also be interpreted as “current state of an individual” or “current stage of the epidemic”. Nature and the rate of the transfer from one subgroup (compartment) to another are to be defined by assumptions. Usually, compartments are named in the literature S (Susceptible), I (Infected), R (Recovered), E (Exposed), etc. An epidemic model that is named as “SIR” model refers to the model including the stages S, I and R. Likewise, if an epidemic model is called as “SEIR model” this means that it includes also the E (Exposed) stage.

These models can be divided into two main categories: Stochastic epidemic models and deterministic epidemic models. The word deterministic means that no randomness takes place in the model. In other words, the behavior of a population is determined completely by its history and by the rules which describe the model [1]. Deterministic models use transition rates (the rate at which individuals from one subgroup pass to another, for example, S to I, I

to R, etc.). Transition from one subgroup to another is done by using these transition rates. Transition rates are derivatives with respect to time of the sizes of the compartments in mathematical terminology so the deterministic model is built using differential equations [1].

Stochastic models contain random variables if so, inherent randomness and allow random variation of one or more inputs over time. They depend on among-individual chance variation in risk of exposure, disease and other factors. They are used when these variations are important, as in small or isolated populations. Stochastic models are good in capturing the uncertainty and variability that is inherent in real-life epidemics due to factors such as the unpredictability of person-to-person contact [2]. These models can become mathematically highly complex.

2.2. Agent Based Models for Disease Spread

In this thesis, the idea of “heterogeneous mixing of individuals is more realistic than homogeneous mixing” is used. After doing literature research, I found some articles that refer to the importance of using population details to form more realistic mixing group structures.

Longini *et al.* [3] used in their study, a discrete-time, stochastic simulation model of Influenza spread in a structured population to determine the effectiveness of targeted antiviral prophylaxis in controlling influenza. They built a model simulating stochastic spread of the epidemic in populations with known contact groups using number of contacts that an individual makes per day. For each person in state “S(Susceptible)”, the probability of infection is calculated using infection transmission probabilities in each mixing group that this person is in. They gave an example of a child in elementary school and basically stated that if child to child transmission probability is p_{hcc} , adult to child transmission probability is p_{hac} , transmission probability in school is p_s , infection transmission probability in neighborhood is p_n and in community is p_c , the number of infected persons that this child is

exposed to is I_{hc} and I_{ha} in his household, I_s in his school, I_n in his neighborhood and I_c in his community then the probability to become infected for this child on that day is:

$P=1-[(1-p_{hcc})^{I_{hc}} (1-p_{hac})^{I_{ha}} (1-p_s)^{I_s} (1-p_n)^{I_n} (1-p_c)^{I_c}]$. The simplified form of the formula that they used to calculate transmission probabilities between people is as the following:

“ $P = 1 - [(1-p_{infection})^{(n_{infected})}]$ ” calculated regarding each mixing group that the person is in. They added randomness to their model by creating a vector consisting of uniformly distributed random numbers and checked for each individual whether the found probability P by the above formula is greater than the corresponding random value for the person or not. If the corresponding random value for the person is less than P , then the person got infected. They began with a baseline case simulation model then added some details. Here, it would be useful to explain what baseline case means. The baseline case means that everybody in the population is susceptible, open and not immune to the infection. In baseline case, vaccination, other immunizing strategies or precautions against the epidemic are not studied. It is important and better to formulate a flexible baseline case model first so that, after building it, it is easier to add any other needed detail to the model. Longini *et al.* [3] created in each simulation the contact structure stochastically based on the age distribution. They gathered approximate household sizes from US Census Bureau. The population structure is consisting of 2000 persons with 4 neighborhoods, one high school, one middle school, and two elementary schools. Briefly, in their publication, the authors probed whether or not, giving antiviral agents prophylactically to close contacts of susceptible influenza indexes is advantageous to vaccination strategies.

This type of modeling which programs explicit representation of individuals, their interaction with the environment and each other is a form of agent-based modelling. These models are generally called as individual based models (IBMs). These are microscale simulations and interactions of individuals are modeled elaborately. IBMs refer to a specific subclass of ABMs where individuals may be simpler than fully autonomous agents of ABMs. Mostly, IBMs are used to model the interactions within the individuals and to their environment, which is useful to explore interesting phenomena, such as population level characteristics [4]. To program these interactions, the important part in these models is to

determine what information an individual needs to have, how it interacts with other individuals and with its environment.

2.3. Literature Research

This section contains the related national and international literature research. It starts with international literature research and then continues with Turkish literature research.

2.3.1. International Literature Research

There exist many simulation models worldwide, for disease spread and epidemic prevention studies. But since the real values of parameters are not known, the models cannot be validated. Also, the parameters can vary in different population structures and social behaviors. So, if a model is built according to a population with certain characteristics, it may not fit to another population with different characteristics. It would be appropriate to say here that, this issue may be considered as a problem of methodological modelling.

In worldwide, specially Longini *et al.* worked on this subject and they built a stochastic epidemic simulation model of disease spread, to determine the effectiveness of antiviral intervention strategies [3]. In their following study, they found a relation between school opening periods and epidemic spread [5]. They also tried to estimate household and community transmission parameters for influenza using an influenza infection dataset by using maximum likelihood method [6].

Longini *et al.* [3], as mentioned in Section 2.2., used in their study a discrete-time stochastic simulation model of Influenza spread in a structured population to determine the effectiveness of targeted antiviral prophylaxis in controlling influenza. This model simulates stochastic spread of the epidemic in populations with known contact groups using the number of contacts that a typical person makes per day. The contact structure of each

simulation is stochastically generated based on the age distribution and approximate household sizes are gathered from US Census Bureau. This structure is consisting of 2000 persons with 4 neighborhoods, one high school, one middle school, and two elementary schools. Preschool children are assumed to attend either small playgroups or larger day-care centers. In their model, newly infected persons enter to a latent period where they do not have any influenza symptoms and after which they recover with immunity. The authors compare the results of simulations with and without antiviral intervention, to determine the effectiveness of antiviral agents. They assume that the probability that a person will be symptomatic given this person is infected is 67% and also an asymptomatic infection is 50% less infectious than a symptomatic one. They use also the term basic reproductive number (R_0) to determine the number of secondary infections produced by a randomly chosen infected person in the population. To calculate this number, they first seed a randomly chosen unvaccinated infected person into a population and then count the mean of secondary infections. Finally, they repeat this procedure 1000 times. To sum up, they probed whether or not giving antiviral agents prophylactically to close contacts of susceptible influenza indexes is advantageous to vaccination strategies. In case antiviral agents are not used, the predicted illness attack rate is claimed to be 33% of the population and death rate is 0.58 deaths per 1000 persons. On the other side, if the antiviral prophylaxis is used with 80% of persons maintaining prophylaxis for 8 weeks, the authors claim that the epidemic would be controlled with illness attack rate of 2% and death rate of 0.04 deaths per 1000 persons, which is similar to vaccinating 80% of population. As a conclusion, targeted antiviral prophylaxis is stated as a potentially effective measure to control the epidemic, until enough vaccines are available. Longini *et al.* [7] also represented in another study, the contact connectivity of a typical Asian population. They used a discrete time, stochastic simulation model within a population of size 500000 to compare the various intervention strategies against the infection. Their model represented the number of close and casual contacts a person makes per day.

As another similar work, Ajelli *et al.* [8] compared results of a structured metapopulation stochastic model and a stochastic agent based model. They used a detailed data of Italian socio demographic structure. The metapopulation simulations use a model based on high resolution census data from all over the world, airline travel flow data and

human mobility patterns. Age structure is also considered. In conclusion, the authors stated that a hybrid model which combines the agent based and the metapopulation approaches could be useful.

In Egypt, Khalil *et al.* [9] developed an agent based simulation model of pandemic influenza. Parameters of the model includes distribution of Egypt population, social attributes of agents and patterns of interactions between agents. The authors tried to interpret the results of the simulations to find characteristics of the virus such as transmission patterns and possible outbreak scenarios. They also find the ways to control influenza by intervening actions.

Perez *et al.* [10] developed an agent based model integrating geographic information systems (GIS). Their aim was to be able to simulate disease spread by using geospatial context interaction of individuals. They chose measles for the simulation.

Milne *et al.* [11] built an agent based model of a small population. They used household structure and demographic information, contact patterns and within community movements of agents, while doing their simulation. Their aim was to determine the effects of social distancing measures such as school closure, for preventing the spread of influenza pandemic. Finally, they concluded that modeling the spread and finding control strategies of pandemic influenza are possible.

2.3.2. Turkish Literature Research

In progress of my thesis, I tried to find possible ways to implement the agent based simulation model to any part of Turkish population and to discover how to build an appropriate synthetic population structure for the simulation model. I researched whether or not there were previous Turkish studies about epidemic models, pandemic, bioterrorist attacks, and agent based simulation of disease spread; also tried to find any possibly related healthcare and epidemiology research. I researched Turkish public statistical reports to find

information like age group, work-school, household, etc., to be used as an input for the simulation model. I found out that there was no epidemic model related study in Turkey that took population structure into consideration. There are only a few mathematicians writing papers about deterministic models.

Pinar Mic [12] investigated in her dissertation how to predict case numbers in the epidemics using basic reproduction number, R_0 and to analyze how sufficient hospital resources are. She built a simulation model and ran it for several R_0 values to analyze whether or not under simulated scenarios, hospital resources would be sufficient.

Ebru Aydin [13] in her dissertation aimed to analyze knowledge level and activities of provincial health directorate managers in Turkey about outbreak and control of contagious diseases. She used a scoring system according to which the mean score of study is reported to be 53.6 ± 20.5 . A score over 60 was noted satisfactory and 43% of participants got satisfactory scores. Throughout the study, outbreak control activities were found insufficient in the provinces where the study participants were located.

Carhan *et al.* [14] state that there exist a national influenza surveillance program launched by Turkish Ministry of Health. The surveillance is conducted by Istanbul Medical Faculty Virology Laboratory and Refik Saydam National Influenza Laboratory. The aim of the program is to determine the virus types and effectiveness of seasonal vaccines. The authors claim that during one year, 1157 people were investigated for the presence of influenza or other viruses. In the article, the authors also give a general view of monthly number of virus related cases.

Cikman *et al.* [15] approached to the inspection of pandemic influenza A(H1N1) cases in their region. He states that 570 pre-diagnosed cases of pandemic influenza A(H1N1) were from five different hospitals in a 3-month period. Samples from these patients were sent to Refik Saydam Hygiene Center to investigate using Real Time PCR (Polymerase Chain Reaction) Method. The author gives a quantitative distribution of positive, intensive care

needing and dead peoples. He also builds an age-influenza infection relationship. According to the author, common risk factors for influenza are having an age less than 5, cardiovascular or lung diseases and use of immunosuppressive drugs. The author indicates that there is a difference between seasonal flu and pandemic influenza which could cause more severe clinical results and that is why taking cautions to pandemic influenza and trying to predict its effects would be of great use.

Kucuk *et al.* [16] tried to evaluate the data from children diagnosed with influenza containing their virus type and age information. The data include laboratory and radiological findings, treatments and clinical examination of 399 patients for an approximately 1-year period. As a result, demographic info about children, their age distribution, whether they have Influenza A or B are given. The most frequent complaint is reported as fever. It is also claimed that Influenza A and B does not differ in terms of laboratory and radiological findings. As a conclusion, the authors state that the number of people with influenza B increases by the time passing.

Sema Ozgur Sakarya [17] in her dissertation researched about registration ratio of tuberculosis patients in İzmir and factors affecting this ratio. She stated that even if tuberculosis was a notifiable infectious disease, there were unregistered people due to some technical and communication problems. Her aim is to determine the ratio of registered tuberculosis patients over all tuberculosis patients, predicting incidence of the infection and finding the technical problems which prevent the patients from being registered. The data consist of tuberculosis cases diagnosed in 2003, 2004 and 2005 in İzmir and the author compares them with electronical records.

Ertem *et al.* [18] investigated fatality rate and hospitalization rate of pandemic influenza H1N1. Their motivation was the influenza outbreak in 2009. Between 36th and 53rd weeks of the pandemic, 731 influenza like illness (ILI) cases were reported from all health centers. At the 36th week, the first case was confirmed to have influenza virus and then the outbreak was seen, peaking at the 43rd week and settled by then. During the outbreak, 31117 cases were investigated as ILI with 2% as hospitalization rate and 11.5/1.000.000 as

mortality rate. Authors stated that symptoms such as coughing, fever, running nose, headache and diarrhea of pandemic influenza were similar to the ones of seasonal flu. The time interval between the onset of symptoms and hospital admission was reported to have a median of 3.5 days and the time interval between the onset of symptoms and death occurrence was reported to a median of 7.5 days. According to the authors, infectivity of the outbreak was high but mortality rate was lower than expected. They claim that mortality rate could be higher (especially dangerous for pregnant women) and hospitals could remain incapable to address the needs so, appropriate actions should be taken.

Kebabci *et al.* [19] analyzed clinical, laboratory and epidemiological findings of 85(44 female and 41 male) confirmed pandemic influenza A(H1N1) cases -during 3 months period- in Uludag University Hospital, Turkey. The median age was reported to be 33, 20% of the patients had pneumonia and 11 cases were in Intensive Care Unit. Mean treatment duration was reported to be 10 days and the mortality rate was 4.7%.

As can be seen from the above studies in Turkey, there is none that is about building an individual based simulation model by considering contact structures. There are only mathematicians who wrote about deterministic models, healthcare professionals who made case studies, or who wrote about the medical side of the matter. In this thesis, contact structures are considered and an individual based simulation of disease spread model that is similar to the one proposed by Longini *et al.* [3] is built.

3. A GENERIC IMPLEMENTATION OF SIR MODEL WITH MIXING GROUPS IN R

In this part, description of the discrete time stochastic simulation model, inputs for the model, the general idea of the code and its implementation, debugging and code performance are explained. In Section 3.1, a brief explanation about why this is a discrete time and stochastic model is given. In Section 3.2, necessary input to make the simulation is explained in detail. In Section 3.3, principles of the simulation and how to implement the problem in R is explained. In Section 3.4, debugging and controlling the correctness of the code is discussed. Finally in Section 3.5, the algorithm is tested with an artificial population.

The model that is considered in this thesis is similar to, but more flexible than the one proposed in Longini *et al.* [3] article. The simulation program itself requires a population structure as one main input. Therefore, I created a synthetic population to be used as an input in the simulation. The detailed information about this population can be found in Section 4. Basically, to create the synthetic population, the number of families of each household size in a neighborhood is required as one input. Other main parameters are the initial number of infected people in a population, the infection probabilities within different mixing groups, the distribution of the disease duration and the number of days to simulate.

3.1. Definition of the Discrete Time Stochastic Model

The simulation model presented in this thesis is a discrete time stochastic model. It is assumed in the model that every day, a certain number of infection takes place. Since the simulation flows day by day, it is a discrete time model. It is also stochastic since, in the model, for each person, there is a state and this state changes from “S(Susceptible)” to “I(Infected)” randomly by generating $[0,1]$ random variates and checking whether random numbers is smaller than the probability of becoming infected calculated by the general formula “ $P = 1 - [(1 - p_{\text{infection}})^{n_{\text{infected}}}]$ ” for each person and for each mixing group (Technical

details are described in detail in Section 3.3). It is important to note here that, these Bernoulli variates are independent but not identically distributed because, the infection probabilities are different according to the mixing groups. Also in the model, the disease duration for each newly infected person is generated randomly following a discrete distribution and seeding of initially infected people to the population is done randomly.

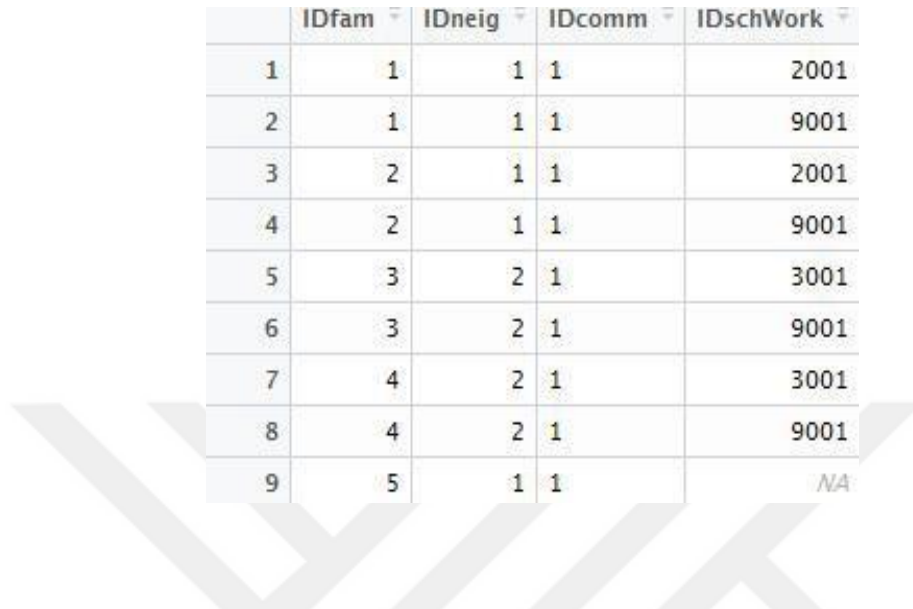
3.2. Inputs for the Simulation Program

For the simulation program, the population structure, the initial number of infected individuals, the transmission probabilities within mixing groups and the cumulative distribution of the disease duration and the number of days to simulate are needed. In the model, initially infectious people are spreading the infection to the population by using infection probabilities within each mixing group. In addition to mixing in schools, family, neighborhood and community as Longini *et al.* [3] proposed in their article, mixing in workplaces is considered for adults in this thesis, so workplaces are considered as a mixing level for adults. Therefore, a separate infection transmission probability within workplace that is different than the infection transmission probability for a child spreading the infection in school is required. The input parameters of the simulation model and their explanations are given in Table 3.1:

Table 3.1. The input parameters of the simulation model and their explanations.

Parameter	Meaning
<i>popDat</i>	Population structure matrix with columns: IDfam: A column holding the family ID of each individual. IDneig: A column holding the neighbourhood ID of each individual. IDcomm: A column holding the community ID of each individual. IDSchWork: A column holding the school/workplace ID of each individual. If the ID for school/work of an individual does not exist, then NA should be written for this entry. NA means the individual does not go to school or work.
<i>pinfFam</i>	Infection transmission probability within family
<i>pinfSchool</i>	Infection transmission probability within school
<i>pinfWork</i>	Infection transmission probability within workplace
<i>pinfNei</i>	Infection transmission probability within neighborhood
<i>pinfComm</i>	Infection transmission probability within community
<i>minDayDisease</i>	Minimum disease duration(days)
<i>P</i>	CDF of disease duration, first entry is CDF value of minimum duration of disease (minDayDisease)
<i>I0</i>	Initial number of infected people

An example population matrix is shown in Figure 3.1:



	IDfam	IDneig	IDcomm	IDschWork
1	1	1	1	2001
2	1	1	1	9001
3	2	1	1	2001
4	2	1	1	9001
5	3	2	1	3001
6	3	2	1	9001
7	4	2	1	3001
8	4	2	1	9001
9	5	1	1	NA

Figure 3.1. Screenshot of an example population matrix with 5 families, 2 neighborhoods, 2 schools and 1 workplace.

The population structure to be used in the simulation is consisting of 4 main mixing groups: School/work mixing group (IDschWork), family mixing group (IDfam), neighborhood mixing group (IDneig), community mixing group (IDcomm).

IDschWork column contains school or work information. If the person does not go to school or to work, it takes as IDschWork “NA”. Babies, seniors and adults who don’t work take “NA”. If a person attends for example the 10th school of code 2 (which is preschool), it has IDschWork 2010. The same rule holds for 3,4,5 age groups, which are related to primary school, middle school and high school, respectively.

Detailed technical information and assumptions about the population structure of a small Turkish town that is used for the simulation can be found in Part 4.

3.3. The General Idea of the Code and Implementation in R

The code simulating the disease spread, first takes as an input the initial number of infected people and is seeding the infected people randomly into the whole population. Individuals who are randomly assigned as infected get the state letter “I” (Infected) in “state” column of the initial population matrix. The rest of the population take the state letter “S” (Susceptible) in the “state” column. For each person that is in state “I”, an end-of-infection time is generated randomly. End-of-infection times are generated randomly by using the minimum disease duration and the vector holding the CDF of disease duration; these times are stored in the endstate column. Using these end-of-infection times, it is determined in the for loop when a person will pass from state “I” (infected) to state “R” (recovered). After state initialization, the number of infected people in each mixing group (community, neighborhoods, school/workplaces, family) are counted by using the “table” command. The tables are used to obtain the number of infected people for each level in each mixing group. An example section of the state matrix (after initialization) with state and endstate columns can be seen in Figure 3.2. From this matrix, the number of infected and susceptible people in each mixing group can be found.

	state	endState	IDfam	IDneig	IDcomm	IDSchWork
1993	R	15	919	4	1	2001
1994	R	20	920	4	1	NA
1995	S	0	920	4	1	9003
1996	R	23	920	4	1	NA
1997	R	25	920	4	1	NA
1998	S	0	920	4	1	NA
1999	R	21	920	4	1	4001
2000	R	12	920	4	1	5002
2001	I	35	921	5	1	9008
2002	S	0	922	5	1	NA
2003	R	31	923	5	1	9010
2004	R	22	924	5	1	9006

Figure 3.2. An example of a state matrix containing state and endState columns added to the population matrix.

Examples of the tables created by table command are shown in Figure 3.3.

	I	S	R
1	1	0	0
2	0	0	0
3	1	0	0
4	0	0	0

Neighborhood

	I	S	R
4006	0	0	0
4007	0	0	0
4008	0	0	0
4009	0	0	0
4010	0	0	0
4011	0	0	0
4012	0	0	0
4013	1	0	0
4014	0	0	0
4015	0	0	0
4016	0	0	0
4017	0	0	0
4018	0	0	0
4019	0	0	0
4020	0	0	0

School-Work

	I	S	R
651	0	0	0
652	0	0	0
653	0	0	0
654	0	0	0
655	0	0	0
656	0	0	0
657	0	0	0
658	0	0	0
659	1	0	0
660	0	0	0
661	0	0	0
662	0	0	0
663	0	0	0
664	0	0	0
665	0	0	0

Family

Figure 3.3. The tables created by “table” command. The tables in the figure refer to neighborhood mixing groups, school/workplace mixing groups and family mixing groups, respectively.

It is seen in Figure 3.3 that, in each one of neighborhood 1 and 3, there is 1 person that is infected, in school with code 4013 there is 1 child that is infected and in the family with ID number 659, there is 1 person that is infected.

After having counted the initial number of infected people in each mixing group by using the “table” command, the main for loop starts with day 1. It first takes the indices of susceptible people and generates a vector with its length equal to the number of susceptible people, consisting of uniform random numbers. It calculates transmission probabilities for each susceptible person and randomly selects from the susceptible who will be infected on this day. For example, the probability that a school child will not become infected is calculated by the formula $P = [(1-p_f)^{I_f} (1-p_{sch})^{I_{sch}} (1-p_n)^{I_n} (1-p_c)^{I_c}]$ where p_f , p_{sch} , p_n , p_c are the transmission probabilities within family, within school, within neighborhood and within community, respectively. I_f , I_{sch} , I_n and I_c are the number of infected people in family mixing group, school mixing group, neighborhood mixing group and community mixing group,

respectively. So, the simplified version of the formula for the probability not to be infected is $P = (1-p_{\text{infection}})^{(n_{\text{infected}})}$ for each mixing group. For each individual, if the generated random number is less than $(1-P)$, the individual becomes infected and passes from state “S” to state “I”. For the people that are newly infected on this day, end-of-infection times are created and stored in the vector “endstate”. After that, a check is required for whether or not there are people who have finished their infection. It is checked in the for loop for each infected, if the current day is equal to the end time of the infection. If this is the case, his state is changed from “I” (Infected) to “R” (Recovered).

At the end of the for loop, the code updates the number of infected individuals in tables, by using the “table” command. The counted number of infected people in each mixing group will be used in the next iteration of the for loop, while calculating the infection probabilities for each person, using the above formula.

In Figure 3.4, the table for the community is shown. It is seen that the number of infected people in the community has increased from 3 to, at the end of day 1.

	I	R
1	3	0

Figure 3.4. Table showing the number of infected and susceptible in community.

A section from the code (simplified form) can be found in the following figure:

```
# IO...the initial number of infected people;
# minDayDisease...minimum disease duration in terms of days
# p ... vector of length 4, holding CDF of the disease duration.
# Disease duration can be 3,4,5,6,7 days.
# pinfFam ... probability of infection within family
# pinfSchool ... probability of infection within school
# pinfWork...probability of infection within workplace
# pinfNei...probability of infection within neighborhood
# pinfComm ... probability of infection within community
# d...number of days
# pdat...population matrix after state initialization, with state and endstate columns added.
# U refers to the vector of uniform random numbers
# The first update, number of infected people is counted in each mixing group:
# upStateGroup is a subfunction that updates the number of I in all groups of gdat
# uses the info of pdat,
# returns the updated group
# gdat ... array fdat(families), ndat(neighborhoods), cdat (communities) or swdat(schools/workplaces)
# gno ... 3(IDfam) 4(IDneig) 5(IDcomm) 6(schWork)
# gno is the column number in the state initialization matrix.
# For example, gno=3 refers to the column that stores the family IDs
```

Figure 3.5. The main section of the code.

```

df <- upStateGroup(pdat,gno=3)  #table holding the number of infected in all families
dn <- upStateGroup(pdat,gno=4)  #table holding the number of infected in all neighborhoods
dc <- upStateGroup(pdat,gno=5)  #table holding the number of infected in community
ds <- upStateGroup(pdat,gno=6)  #table holding the number of infected in all schools/workplaces

numberOfSchools<-length(which(as.numeric(levels(as.factor(popDat$IDSchWork)))<9000))
numberOfWorkplaces<-length(which(as.numeric(levels(as.factor(popDat$IDSchWork)))<9999&
as.numeric(levels(as.factor(popDat$IDSchWork)))>9000))

aveSchSize<-length(which(popDat$IDSchWork>2000&popDat$IDSchWork<9000))/numberOfSchools
aveWorkSize<-length(which(popDat$IDSchWork>9000&popDat$IDSchWork<9999))/numberOfWorkplaces

initialInfNumberSch<-sum(ds$I[which(as.numeric(rownames(ds))<9000)])
initialInfNumberWork<-sum(ds$I[ which(as.numeric(rownames(ds))>9000 &
as.numeric(rownames(ds))<9999)])

#Main for loop (Simplified form):
for(it in 1:d){ # simulate new infections (passing from S to I)

ivS <- (1:npeople)[pdat$state=="S"] #vector holding row numbers of all susceptible people
U <- runif(length(ivS))

#family, here, no. of infected in family mixing group table is used to calculate probability.
prob <- (1-pinfFam)^(df$I[pdat$IDfam[ivS]]+df$H[pdat$IDfam[ivS]])

# neighborhood
prob <- prob*(1-pinfNei)^dn$I[pdat$IDneig[ivS]]

# community ...

# school/work ...

# inewI ... a vector of indexes of newly infected people
inewI <- ivS[U<1-prob]  #inewI<-ivS[prob<U] is also OK.
inewI <- ifelse(is.na(inewI),array(0,length(prob)),inewI)

```

Figure 3.5. A section from the code (cont.).

```

# set endState for the "new I"s pdat$state[inewI] <- "I"
pdat$endState[inewI]<-endOfInfection(length(inewI),minDayDisease=minDayDisease,p=p,t0=it)

# end all I of now

ivIend <- (1:npeople)[pdat$state=="I" & pdat$endState==it] pdat$state[ivIend] <- "R"
if(sum(pdat$state=="I")==0) break #If there is no more infected, stop!


# update at the end of the for loop.

# update to use the infected people number in the prob calculation section of the next day.

# update the number of I, R in the groups
df <- upStateGroup(pdat,gno=3) #in Family
dn <- upStateGroup(pdat,gno=4) #in Neighborhood
dc <- upStateGroup(pdat,gno=5) #in Community
ds <- upStateGroup(pdat,gno=6) #in School/Workplace

#store the number of I to be able to return these values
nIv[it]<- dc$I }#Hold the number of infected people in the whole community, for each day.
breakT<-length(which(nIv>0)) #the day when the simulation stopped
nIi_max<-which.max(nIv)
nI_max<-nIv[nIi_max]
schWorkSizes<-table(popDat$IDschWork)
schSizes<-schWorkSizes[ which(rownames(schWorkSizes)<9000)] #school sizes
sch_attackRate<-as.data.frame(ds$R[which(rownames(ds)<9000)]/schSizes)[,2]
nS <- sum(pdat[,1]=="S") #number of people never infected

return(list(nI=nIv,percInfected=(npeople-I0-nS)/(npeople-I0),pdat,breakTime=breakT,
          maxnIi=nIi_max,maxnI=nI_max,initialInfNumberSch=initialInfNumberSch,
          sch_attackRate=sch_attackRate,averageSchoolAttackRate=mean(sch_attackRate)))

```

Figure 3.5. A section from the code (cont.).

The code returns the number of infected people of each day, the percentage of the population that was infected is calculated by the formula $\text{percInfected} = (\text{npeople} - \text{I0} - \text{nS}) / (\text{npeople} - \text{I0})$ where npeople is the total number of people in the community, I0 is the initial number of infected people, and nS is the number of people that has never been infected.

After having built the main function for a single simulation, I have written a code for independent repetitions. The code takes the inputs to be used in the simulation model and repeats it “nrep” time. “nrep” is a new input, used in the code for independent repetitions and refers to “the number of independent repetitions”. As an output, the program gives a matrix of size $n \times d$, with n representing the number of people in the population and d representing the number of days of the simulation. The $n \times d$ matrix contains the number of infected of each day of each repetition. The code returns also the percentage of infected people of each repetition (attack rate). A section from the output generated by the code can be seen in Figure 3.6 and a section of the array storing the attack rate of each repetition can be seen in Figure 3.7:

	nl.1	nl.2	nl.3	nl.4	nl.5	nl.6	nl.7	nl.8	nl.9	nl.10	nl.11	nl.12	nl.13	nl.14	nl.15	nl.16	nl.17	nl.18	nl.19	nl.20
1	17	21	23	24	25	23	24	28	35	44	52	63	83	97	107	125	128	152	161	169
2	14	18	24	35	46	56	69	82	97	120	131	160	182	193	217	231	238	231	242	256
3	17	24	30	31	38	43	57	66	82	93	100	114	117	131	141	149	177	177	200	203
4	15	18	23	22	23	22	23	22	29	35	46	50	59	72	84	98	120	148	161	171
5	14	19	21	26	30	29	33	34	41	46	55	69	75	80	82	94	97	103	120	140
6	15	21	28	37	48	58	68	74	75	87	108	125	130	134	144	155	175	181	192	208
7	15	22	30	33	40	48	58	65	89	108	135	149	171	203	214	232	247	265	280	294
8	14	16	21	21	21	17	24	30	38	46	53	58	70	85	93	102	110	117	131	147
9	14	20	23	24	33	45	49	50	61	76	82	87	96	117	127	140	151	156	168	176
10	15	18	22	26	30	39	48	59	74	91	111	137	157	182	188	199	215	224	232	240
11	17	27	33	39	52	52	61	67	80	89	111	131	153	179	195	229	255	272	298	299
12	14	16	22	21	26	35	37	50	74	90	101	118	132	161	202	228	242	262	288	288
13	13	16	16	17	18	23	25	29	30	33	43	51	68	73	80	89	93	100	119	133
14	16	20	25	32	33	34	35	47	52	73	98	117	142	154	171	189	208	224	244	256
15	15	21	29	42	58	72	88	107	127	146	171	178	209	226	246	246	261	272	257	236

Figure 3.6. A section from $n \times d$ matrix.

	percInfected
1	0.5160966
2	0.5503018
3	0.4959759
4	0.5578471
5	0.5286720
6	0.4949698
7	0.5181087
8	0.5477867
9	0.5075453
10	0.5402414
11	0.5372233
12	0.5407445
13	0.5020121
14	0.5241449
15	0.5105634

Figure 3.7. A section from the array storing the attack rate of each repetition.

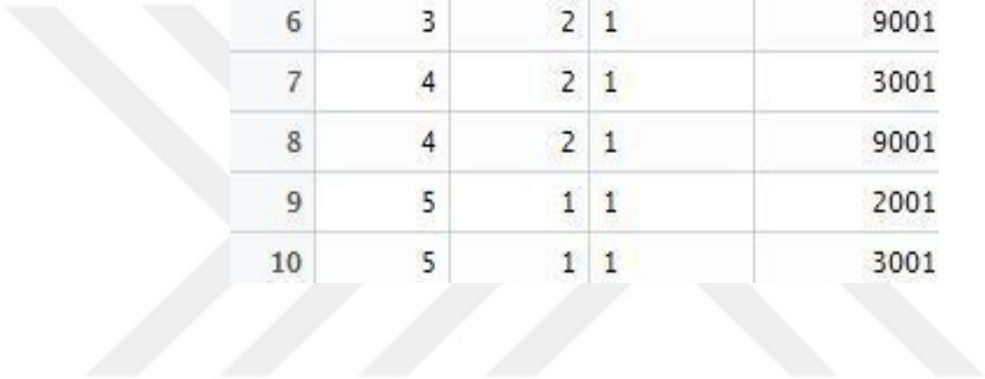
3.4. Controlling the Correctness of the Implementation

Debugging with trace is crucial to control the correctness of the implementation. So, to control the flow of the program, the code uses a TRACE object. This object gives the desired outputs that are generated at the desired part of the code. If TRACE is equal to TRUE in the function parameter list, the outputs of each iteration of the for loop are generated. If TRACE is equal to FALSE, which is the default, the function executes without tracing.

Since in the simulation function, the flow of the main loop should carefully be checked, printing the parts of the generated output using the TRACE object is significant to test that in each iteration, the code flows as it should. In the program, for each iteration of the for loop, it is controlled if the results are as expected and fit to the used infection parameters.

I formed a small population matrix to test the correctness of the simulation program. The population structure consists of 5 families, 2 neighborhoods, 1 workplace, 1 primary

and 1 middle school. In Figure 3.8, a screenshot of the population matrix that is used to trace the simulation program can be seen:



	IDfam ↕	IDneig ↕	IDcomm ↕	IDSchWork ↕
1	1	1	1	3001
2	1	1	1	9001
3	2	1	1	2001
4	2	1	1	9001
5	3	2	1	2001
6	3	2	1	9001
7	4	2	1	3001
8	4	2	1	9001
9	5	1	1	2001
10	5	1	1	3001

Figure 3.8. A screenshot of the population matrix that is used to trace the simulation program.

```

[1] "it= 5"
[1] 0.6549390 0.7633887 0.9404978 0.3818887 0.9202248 0.4217849 0.4449650 0.7549025

[1] 0.0002199859 0.0002199859 0.0000199999 0.0000199999 0.0000199999 0.0000199999
0.0002199859 0.0002199859

      state  endState  IDfam IDneig  IDcomm  IDSchWork
1      S          0      1      1        1        3001
2      S          0      1      1        1        9001
3      R          5      2      1        1        2001
4      I          6      2      1        1        9001
5      S          0      3      2        1        2001
6      S          0      3      2        1        9001
7      S          0      4      2        1        3001
8      S          0      4      2        1        9001
9      S          0      5      1        1        2001
10     S          0      5      1        1        3001

[1] "it= 6"
[1] 9.025334e-01 5.245861e-05 1.011548e-02 7.505868e-01 6.112033e-01 7.658450e-01
4.681629e-01 2.156379e-01

[1] 0.000109999 0.000109999 0.000010000 0.000010000 0.000010000 0.000010000 0.000109999
0.000109999

      State  endState  IDfam IDneig  IDcomm  IDSchWork
1      S          0      1      1        1        3001
2      I          9      1      1        1        9001
3      R          5      2      1        1        2001
4      R          6      2      1        1        9001
5      S          0      3      2        1        2001
6      S          0      3      2        1        9001
7      S          0      4      2        1        3001
8      S          0      4      2        1        9001
9      S          0      5      1        1        2001
10     S          0      5      1        1        3001

```

Figure 3.9. Outputs generated for controlling the correctness of the implementation.

The rule for infection was $U < 1 - \text{prob}$. From the figure above, it can be seen that at day $y=6$, the 2nd element of U is $5.245861e-05$ and 2nd element of the vector $(1-P)$ is 0.000109999 . So the state of the 2nd person should be “I” (Infected). When the state of the

2nd person is checked, it is seen that it was “S” (Susceptible) at it=5 (day=5) and changed to “I” (Infected) at it= 6 (day=6). Therefore, this part works correctly.

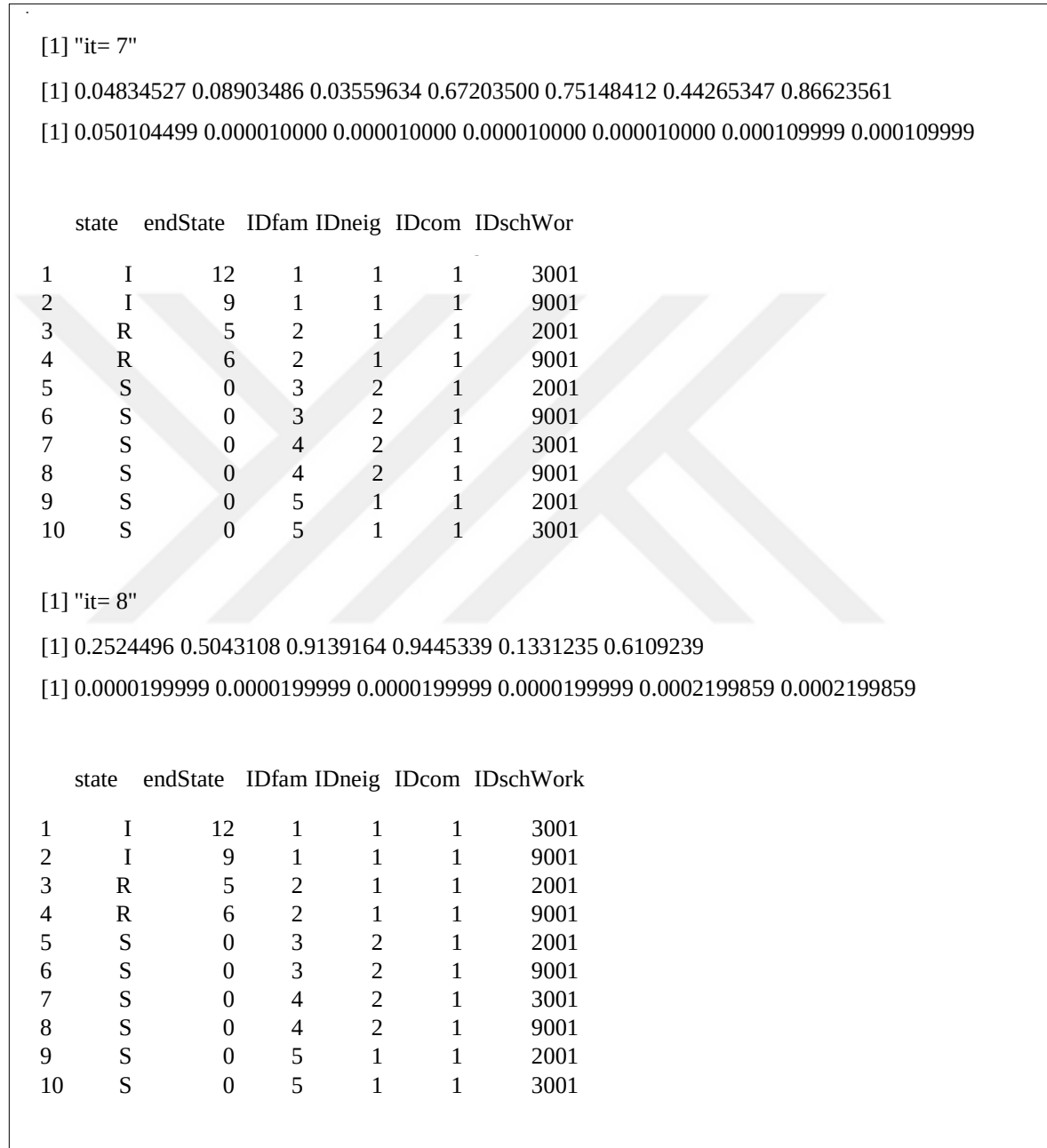


Figure 3.10. Outputs generated for controlling the correctness of the implementation.

It is expected that the 2nd person will end its infection at it=9 (day=9). When it=9 (day=9) is checked, the state of the 2nd person is changed to “R” (Recovered) as expected (see Figure 3.10 and Figure 3.11).

[1] "it= 9"						
[1] 0.3923188 0.9594665 0.5857290 0.4253387 0.6486275 0.3921224						
[1] 0.0000199999 0.0000199999 0.0000199999 0.0000199999 0.0002199859 0.0002199859						
	state	endState	IDfam	IDneig	IDcom	IDSchWork
1	I	12	1	1	1	3001
2	R	9	1	1	1	9001
3	R	5	2	1	1	2001
4	R	6	2	1	1	9001
5	S	0	3	2	1	2001
6	S	0	3	2	1	9001
7	S	0	4	2	1	3001
8	S	0	4	2	1	9001
9	S	0	5	1	1	2001
10	S	0	5	1	1	3001

Figure 3.11. Outputs generated for controlling the correctness of the implementation.

3.5. Code Performance and an Experiment with a Synthetic Population Structure

Synthetic primitive population matrices are created in this part to test the performance of the code and to make a first experiment. The neighborhoods of the created populations consist of 250 families of size 2 (one adult and one child). Populations of size 10000, 5000 and 2000 consist of 20, 10, 8 neighborhoods, respectively. All adults are assumed to go to work. In Figure 3.12, a screenshot from one of these created synthetic primitive population matrices is shown.

	IDfam	IDneig	IDcomm	IDSchWork
1	1	1	1	4003
2	1	1	1	9002
3	2	1	1	4001
4	2	1	1	9009
5	3	1	1	5001
6	3	1	1	9003
7	4	1	1	5001
8	4	1	1	9008
9	5	1	1	5001
10	5	1	1	9001
11	6	1	1	4003
12	6	1	1	9006
13	7	1	1	4002
14	7	1	1	9010
15	8	1	1	4002

Figure 3.12. An example part from one of the primitive synthetic population matrices.

The performance of the simulation code is tested by making experiments with 20, 50, 100, 150 days; 100, 200, 300 independent repetitions. With these experiments, the complexity of the code is tried to be determined. Runtime durations of the experiments are measured and given in Table 3.2. Inputs that are used while testing the code performance are the following:

- (i) Minimum disease duration = 3 days
- (ii) Vector holding the CDF of disease duration = (0.4,0.7,0.8,0.9)
- (iii) Number of initially infected people (I_0) = 12

The infection probabilities:

- (i) For a population of size 10000 with 8 schools and 8 workplaces: 0.003905 in family mixing groups (pinfFam), 0.0074976 in school mixing groups (pinfSchool), 0.0005467 in work mixing groups (pinfWork), 0.000000781 in neighborhood mixing groups (pinfNei) and 0.0000004686 in community mixing group (pinfComm).

- (i) For a population of size 5000 with 4 schools and 4 workplaces: 0.0039275 in family mixing groups (pinfFam), 0.0075408 in school mixing groups (pinfSchool), 0.00054985 in work mixing groups (pinfWork), 0.0000007855 in neighborhood mixing groups (pinfNei) and 0.0000004713 in community mixing group (pinfComm).
- (ii) For a population of size 2000 with 4 schools and 2 workplaces: 0.0004985 in family mixing groups (pinfFam), 0.0018943 in school mixing groups (pinfSchool), 0.0006979 (pinfNei) and 0.0000005982 in community mixing group (pinfComm).

Table 3.2. Runtime durations(s) of the experiments with an approximate R_0 value of 1.4.

Size refers to population size.

#rep, population size, days	duration(s)	#rep, population size, days	duration(s)	#rep, population size, days	duration(s)
100, 2000, 20	40	200, 2000, 20	79	300, 2000, 20	125
100, 5000, 20	88	200, 5000, 20	177	300, 5000, 20	266
100, 10000, 20	171	200, 10000, 20	343	300, 10000, 20	518
100, 2000, 50	93	200, 2000, 50	184	300, 2000, 50	302
100, 5000, 50	211	200, 5000, 50	418	300, 5000, 50	869
100, 10000, 50	410	200, 10000, 50	837	300, 10000, 50	1232
100, 2000, 100	144	200, 2000, 100	289	300, 2000, 100	475
100, 5000, 100	368	200, 5000, 100	745	300, 5000, 100	1532
100, 10000, 100	765	200, 10000, 100	1521	300, 10000, 100	2293
100, 2000, 150	137	200, 2000, 150	307	300, 2000, 150	458
100, 5000, 150	391	200, 5000, 150	773	300, 5000, 150	1223
100, 10000, 150	869	200, 10000, 150	1750	300, 10000, 150	2665

In Table 3.2, it is seen that runtime duration and number of days have a linear relation at the beginning, until around 50th day. Moreover, it can be seen that after day=100, runtime duration (in seconds) is not changing significantly with an increase in days of simulation. It is highly probable that the number of infected people is generally peaking early in the simulations and after having infecting everyone it can, the epidemic dies out. Since the loop is exited when no more infected remains in the population, the simulation finishes when there is no infected left and so its duration does not increase. In addition, it is understood

that, using the same number of independent repetitions and simulated days, the relation between the population size and the runtime duration is nearly linear. Lastly, it is seen that, using the same population size and number of simulated days, the relation between the number of independent repetitions and the runtime duration is nearly linear.



4. SELECTING MODEL PARAMETERS

In this part, the code that generates a population matrix that can be used as an input for the simulation is described and the choice of the model parameters is discussed.

To apply the simulation code and to make conclusions about possible disease spread in Turkey, it is necessary to prepare the input. In other words, to decide in detail on the model parameters that will be used in the simulation. For this purpose, I formed the data of a typical middle sized district in Turkey, for example in Marmara Region and I built a simulation model and observed its performance. The code that is built to form the data of an average middle- sized district in Turkey is also able to form different population matrices of different size and properties.

In Section 4.1, the useful information that can be obtained in Turkey is examined and then, the created population structure and the use of the population generation code are described. In Section 4.2, parameter calibration is discussed.

4.1. Creating the Population Matrix and a Synthetic Population Structure

In this section, the generated population data, its properties, its assumptions and the use of population generation code are discussed. The code to generate population matrix can be found in Appendix A. The created population structure is aimed to be realistic and to carry the main characteristics of a typical Turkish district so that the simulation model has a practical usage for Turkey. It is also flexible and can be used to form different population matrices (see Section 4.1.1 for details). Although the time for a master's thesis was not enough to gather all the information related to a population, since the main aim was in any case to build a model that is flexible, I did not gather detailed information. Instead, I went to an education ministry office to learn how many schools there are in a middle-sized Turkish district, and the average number of children in schools. I learned that in a district with

population size of approximately 30000 people, there are approximately 6 preschools, 6 primary schools, 6 middle schools and 6 high schools and the average size of schools is around 300 people. I found approximate working percentage of adults which is around 15% [20]. Moreover, I found that around 10% of the population consists of senior people who are aged above or equal to 65 [21]. These values are then used as input in the population generating function and their usage is expressed in detail in the later parts of this section.

The generated population matrix contains the information related to each individual of the population. It gives the “family ID number” of a person and indicates the size of the family that this person is in. It also gives neighborhood ID, community ID of a person. Logically, members of each family must have the same family, the same neighborhood and the same community ID (see Figure 4.1). It is assumed that members from each family and from each neighborhood may go to any school or any workplace in the community and that school ID or work ID do not relate to the family or neighborhood that a person is in. Since in Turkey, it is not unusual for children to go to schools outside of their neighborhoods and for the adults to go to work outside of their neighborhoods, assignment of children to schools of their age group and assignment of adults to workplaces are made randomly. All workplaces have approximately the same size and all schools have approximately the same size. Since the infection probabilities in differently sized workplaces and in differently sized schools are very difficult to calibrate, differently sized workplaces and differently sized schools have not been taken into consideration (see Section 4.2 for calibration details).

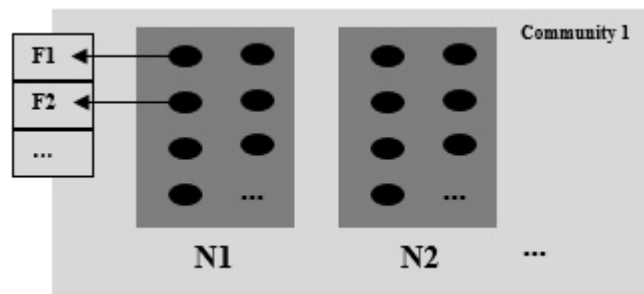


Figure 4.1. Family, neighborhood and community structure.

One of the main inputs of the population generation program is `fsizeTable` which gives the family structure of each neighborhood (see Figure 4.2).

	fsize	fquantity
1	1	100
2	2	50
3	3	34
4	4	37
5	5	5
6	6	3
7	7	1

Figure 4.2. An example of `fsizeTable`. The column `fsize` refers to family size and the column `fquantity` indicates how many families of each size there are, in one neighborhood.

The population size of one neighborhood can be found by summing the product of `fsize` (family size) and `fquantity` (number of families) of each row of `fsizeTable`. For example, by using the information shown in Figure 14, it can be calculated that there are:

$$1 * 100 + 2 * 50 + 3 * 34 + 4 * 37 + 5 * 5 + 6 * 3 + 7 * 1 = 500 \text{ people in one neighborhood.}$$

Other inputs of the program are `IDcomm`, the community ID number (which has been taken as 1 for this code), number of neighborhoods (`nnei`) in a community, `propSenior` which indicates the proportion of seniors to total “seniors+adults”, `ageDistrChild` which is a vector including the percentage of children in terms of age groups and `schoolTab` which is a matrix holding the school numbers in a community. For example, if `ageDistrChild` is `c(0.15,0.20,0.20,0.20,0.25)`, 15% of the population belongs to the age group called “baby”, 20% to “preschool”, 20% to “primary school”, 20% to “middle school”, and 25% to “high school”. In the `agegroup` column of the population structure matrix, code 1 refers to “baby”,

code 2 refers to “pre-school student”, code 3 refers to “primary school student”, code 4 refers to “middle school student”, code 5 refers to “high school student”, code 6 refers to “adult” and code 7 refers to “senior”. As for schoolTab input, a matrix is taken: For example, if the matrix is created with the `matrix(c(1,0,2,2,3,2,4,2,5,2,6,0,7,0),nrow=2)` command, this means that there is 0 school for the age group called “baby” in the community (it will always be 0 since “baby” is defined as children between 0-2 year olds and they don’t go to school at all.), that there are 2 pre-schools, 2 primary schools, 2 middle schools, 2 high schools, 0 school for adults and 0 school for seniors (Naturally adults and seniors don’t go to school, so these values will always be 0).

There is also workInfo input which includes the percentage of adults who go to work and the total number of workplaces. For instance, if the input is `workInfo=c(0.85,10)`, this means that 85% of adults are going to work in the community and there are in total 10 workplaces.

Assumptions of the code generating the population are as the following:

- (i) Equal family structure in each neighborhood.
- (ii) Baby is defined as 0-2 yrs.
- (iii) 1 size and 2 size families include adults or seniors.
- (iv) 3 and 3+ sized families include 2 adults and children (No seniors).

A list including the inputs to form the population matrix and their explanations can be found in Table 4.1.

Table 4.1. Inputs to form the population matrix and their explanations.

Name	Explanation
fsizeTab:	Table giving the family structure of each neighborhood, has two columns: “fsize” referring to family size and fquantity indicating how many families of each size there are in one neighborhood.
IDcomm:	Community ID number, always 1 for this code.
nnei:	Number of neighborhoods in a community.
propSenior:	Proportion of seniors to “seniors+adults”.
ageDistrChild:	Vector including the percentage of children in each age group.
schoolTab:	Matrix holding the total school numbers in a community.
workInfo:	Percentage of adults who go to work and the total number of

Some other information about the code is as the following:

- (i) If number of adults is not enough (so the number of seniors is too high), a correction message will appear: "According to fsizeTab, number of seniors is too high, so the percentage of seniors is set to "..."
- (ii) Infected individuals are assumed to be immune to re-infection during the simulation.

While forming the population structure matrix, community ID is used as 1 since the population that is desired to be analyzed in the simulation is considered as a district in Marmara Region, in Turkey and its size is approximately 10000 people. However, to create bigger population, for example a population of 100000 individuals, it is enough to increase the number of neighborhoods, which is an input to the program. Runtime to create the synthetic population is 0.86 seconds for 2000 people, 0.87 seconds for 10000 people, and 5.14 seconds for 100000 people. The code creates a matrix with 6 columns that are “IDfam”,

“size”, “IDneig”, “IDcomm”, “agegroup”, “IDSchWork”. As an output, it gives 4 of these 6 columns, “IDfam”, “IDneig”, “IDcomm”, “IDSchWork”. The reason that “size” and “agegroup” columns are created and not given as an output is that they are used to create IDSchWork column for now, so they are not used directly in the simulation program for now, but they might be useful in later research. An image of the created matrix can be seen below as Figure 4.3. The population matrix that is given as the function output is shown in Figure 4.4.

	IDfam	size	IDneig	IDcomm	agegroup	IDSchWork
8912	4122	4	18	1	6	9010
8913	4122	4	18	1	2	2001
8914	4122	4	18	1	5	5001
8915	4123	4	18	1	6	9001
8916	4123	4	18	1	6	9007
8917	4123	4	18	1	5	5002
8918	4123	4	18	1	3	3002
8919	4124	4	18	1	6	9006
8920	4124	4	18	1	6	9010
8921	4124	4	18	1	4	4001
8922	4124	4	18	1	3	3002
8923	4125	4	18	1	6	9010
8924	4125	4	18	1	6	9010
8925	4125	4	18	1	1	NA
8926	4125	4	18	1	1	NA
8927	4126	4	18	1	6	NA
8928	4126	4	18	1	6	9010

Figure 4.3. An example from matrix that is used to get the output population matrix.

	IDfam	IDneig	IDcomm	IDSchWork
792	411	2	1	9003
793	411	2	1	2002
794	412	2	1	9006
795	412	2	1	9007
796	412	2	1	3001
797	413	2	1	9006
798	413	2	1	NA
799	413	2	1	2002
800	414	2	1	9003
801	414	2	1	9006
802	414	2	1	2001
803	415	2	1	9003
804	415	2	1	9006
805	415	2	1	4001
806	415	2	1	NA
807	416	2	1	9008
808	416	2	1	9009
809	416	2	1	4002

Figure 4.4. An example from the population matrix formed to be used in the simulation model.

In IDSchWork column, codes between 2000 and 3000 are for preschools, between 3000 and 4000 are for primary schools, between 4000 and 5000 are for middle schools, between 5000 and 6000 are for high schools, between 9000 and 9999 are for adult workplaces. For example, if the code 3001 indicates a school for to age group 3, which is a primary school and that it is the first school of that age group; if the code 9010 is seen, this indicates the workplace with number 10.

The resulted population matrix is of size 10000, has 20 neighborhoods of equal sizes. Family sizes are between 1 and 7. The percentage of seniors among all above 18 years is 15%, working percentage of adults is 85%, total workplace number is 10, each workplace has equal size and the size of 1 workplace is 600 people. Total school number is 8 with 2 preschools, 2 primary schools, 2 middle schools and 2 high schools and population size of one school is 300.

4.1.1. Use of Population Structure Code to Generate Different Population Matrices

In this section, the use of the population generation code to generate different population matrices is explained. The code that is used to generate a synthetic Turkish population can also be used to generate population matrices of different sizes. In Section 4.1, the inputs of population generating function, `popdat()`, and the output forms are shown in detail. Here, additional details are given in case someone wants to change its parameters and generate a different population matrix. For example, a population of 100000 people can easily be generated by choosing the number of neighborhoods equal to 200 and the size of one neighborhood equal to 500.

The population generation code first creates a neighborhood matrix with the information coming from `fsizeTable` (see Figure 4.2 for `fsizeTable`). It then generates multiple neighborhoods according to `nnei` input of population generating (`popdat`) function. For example, if the size of one neighborhood is calculated to be 500 and if a population of size 2500 is desired, then, by choosing `nnei=5`, this population can be generated. The population generating function uses `addAge` subfunction to add randomly age group info

according to the code assumptions explained in Section 4.1 and schoolWorkF function to assign people to schools according to their age group information. Finally, the main function popdat, generates the resulting data frame to be used as population data. The detailed population generation code can be found in Appendix A. An example of the popdat function usage to form a population matrix of size 10000 is shown in the figure below:

```
popdat(fsizeTab=fsizeTable, nnei=20,
propSenior=0.15,ageDistrChild=c(0.20,0.20,0.20,0.20,0.20),
schoolTab=matrix(c(1,0,2,2,3,2,4,2,5,2,6,0,7,0), nrow=2), workInfo=c(0.85,10),
IDcomm=1)
```

Figure 4.5. Example of population generating function (popdat) with all its inputs.

4.2. The Information that the Simulation Model is Built on and Infection Probability Calibration

The simulation program uses as input the population matrix, infection probabilities within mixing groups (family, school, workplace, neighborhood and community), disease duration information and the initial number of infected people to initialize the state matrix.

To calibrate the infection probabilities, I built an extra function that calculates the approximate R_0 value, the basic reproduction number for the inputted population matrix. The R_0 value, as mentioned in Part 2, is a central quantity in epidemiology measuring the transmission potential of infectious diseases [22]. It is defined as the expected number of secondary cases produced by a single infected in a completely susceptible population [23]. In the function, I calculated the approximate R_0 -contribution (to the total approximate R_0 of the population) coming from each mixing level. I assumed the approximate R_0 -contribution of each mixing group is the expected number of secondary cases produced by a single infected in each mixing group. The general form of the used formula to compute the approximate R_0 -contribution of each mixing group is: $(n-1) * p_{inf} * E(\text{Disease Duration})$ where n refers to the mixing group size, p_{inf} is the transmission probability within mixing

group and $E(\text{Disease Duration})$ is the expected number of days of disease calculated from the pdf of the disease duration. In other words, the approximate R_0 -contribution of the community mixing level to the approximate total R_0 is calculated by multiplying “the size of community-1” (here, since the infected person cannot infect himself again, 1 is subtracted), “infection transmission probability within community” and “expected disease duration” with each other. Then, the same formula is computed for neighborhood mixing level. For school/work mixing level, firstly the proportion of people who work and the proportion of people who go to school are found. Then, R_0 -contribution coming from the mixing in school and R_0 -contribution coming from the mixing in workplace are calculated by the following formulae, respectively:

- (i) $(\text{average school size}-1) \cdot \text{pinfSchool} \cdot \text{ExpDur}$
- (ii) $(\text{average workplace size}-1) \cdot \text{pinfWork} \cdot \text{ExpDur}$.

By using the proportions of people who go to school, who go to work and using the above formulae, a weighted average for school/work mixing level is computed: This value is recorded as the R_0 -contribution coming from the school/work mixing level. Lastly, for each household size, an approximate contribution value to R_0 is computed. From the population matrix, the proportion of people of each household size to total number of community is found. Then, approximate R_0 -contribution of each family size is calculated by the formula $(i-1) \cdot \text{pinfFam} \cdot \text{ExpDur}$ where i denotes the household sizes in the community, for instance, from 1 to 7. The approximate R_0 -contribution from each household size is stored in an array. Likewise, the proportion of each household size in the whole community is computed and stored in another array. Afterwards, the sum of the products of approximate R_0 -contribution of each household size by the calculated proportions of each household size is computed. The simplified formula for this operation is as the following: $\sum_{h=1}^H C_h * P_h$ where, h is the household size ranging from 1 to H , C_h is the approximate calculated R_0 -contribution of household of size h and P_h is the calculated proportion of household size h to the whole community. As a result, this value is recorded as the R_0 -contribution coming from family mixing level. At the end, to find the approximate R_0 value, the sum of all

contributions coming from all mixing levels is calculated. Inputs of this function are given in Table 4.2.

Table 4.2. Inputs of the function.

Input Name	Description
popDat	Population matrix
pinfFam	Infection transmission probability within family
pinfSchool	Infection transmission probabilities within school
pinfWork	Infection transmission probabilities within work
pinfNei	Infection transmission probabilities within neighborhood
pinfComm	Infection transmission probabilities within community
minDayDisease	Minimum disease duration
P	Vector holding the pdf of disease duration. First entry belongs to

An example output of the function, which gives R_0 -contribution coming from each mixing level and the total approximate R_0 for the inputted population, can be seen in the following figure:

Approximate R_0 calculated for this population is: 1.602291
Contribution of community mixing group to the approximate R_0 of this population is: 0.2674561
Contribution of neighborhood mixing groups to the approximate R_0 of this population is: 0.2669479
Contribution of school/work mixing groups to the approximate R_0 of this population is: 0.8002926
Contribution of mixing in schools to the approximate R_0 of this population is: 0.5346134
Contribution of mixing in workplaces to the approximate R_0 of this population is: 0.2656792
Contribution of family mixing groups to the approximate R_0 of this population is: 0.2675943
Contribution of family mixing groups by size are: 0 0.1365277 0.2730554 0.4095832 0.5461109 0.6826386 0.8191663

Figure 4.6. Output of the function.

The function is shown with its detailed comments in the following figure:

```
R0_Calc<-function(popDat=popData,pinfFam=0.098,pinfSchool=0.006,pinfWork=0.00051,
  pinfNei=3.84e-4,pinfComm=1.92e-5,minDayDisease=3,p=c(0.4,0.3,0.1,0.1,0.1)){
#popDat...population matrix; pinfFam...infection probability in family mixing groups; pinfSchool...
#infection probability in school; mixing groups; pinfWork...infection probability in work mixing groups;
#pinfNei...infection probability in neighborhood mixing groups; pinfComm...infection probability in
#community mixing group; minDayDisease...minimum disease duration (in days)
#p...vector holding pdf values for disease duration, first entry is for minDayDisease
#Function for approximate R0 Calculation for a given population structure
nnei<-length(table(popDat$IDneig)) #number of neighborhoods
#expected disease duration calculation
ExpDur<-numeric(length(p))
for(i in 1:length(p)) ExpDur[i]<-(minDayDisease+i-1)*p[i]
ExpDur<-sum(ExpDur)
#R0 contribution coming from community mixing group
commContr<-(length(popDat[,1])-1)*pinfComm*ExpDur
#R0 contribution coming from neighborhood mixing group
neiContr<-((length(popDat[,1])/nnei)-1)*pinfNei*ExpDur
#R0 contribution coming from sch/Work mixing group
numberOfSchools<-length(which(as.numeric(levels(as.factor(popDat$IDSchWork)))<9000))
numberOfWorkplaces<-
length(which(as.numeric(levels(as.factor(popDat$IDSchWork)))<9999&as.numeric(levels(as.factor
(popDat$IDSchWork)))>9000))
aveSchSize<-length(which(popDat$IDSchWork>2000&popDat$IDSchWork<9000))/numberOfSchools
aveWorkSize<-
length(which(popDat$IDSchWork>9000&popDat$IDSchWork<9999))/numberOfWorkplaces
```

Figure 4.7. Function to calculate approximate R_0 value.

```

schContr<-(aveSchSize-1)*pinfSchool*ExpDur
workContr<-(aveWorkSize-1)*pinfWork*ExpDur

schWeight<-length(which(popDat$IDSchWork>2000&popDat$IDSchWork<9000))/length(popDat[,1])
workWeight<-length(which(popDat$IDSchWork>9000&popDat$IDSchWork<9999))/length(popDat[,1])
weightedSWContr<-schContr*schWeight+workContr*workWeight

householdSizes<-levels(as.factor(popDat$size))

lengthHS<-length(householdSizes)

#famContr: vector holding approximate R0-contribution value coming from each household size
famContr<-numeric(lengthHS)

for(i in 1:lengthHS) famContr[i]<-(i-1)*pinfFam*ExpDur
famProportion<-numeric(lengthHS) #proportion of each household size
for(i in 1:lengthHS) famProportion[i]<-table(popDat$size)[i]/sum(table(popDat$size))

#weighted sum of all contributions from each household size
wghtedTotalFamContr<-sum(famContr*famProportion)

#Calculated approximate R0 value is the sum of all contributions coming from all mixing groups
approximateR0<-wghtedTotalFamContr+weightedSWContr+neiContr+commContr

cat("Approximate R0 calculated for this population is: ",approximateR0,"\n",

    "Contribution of community mixing group to the approximate R0 of this population is:
",commContr,"\n",

    "Contribution of neighborhood mixing groups to the approximate R0 of this population is:
",neiContr,"\n",

    "Contribution of school/work mixing groups to the approximate R0 of this population is:
",weightedSWContr,"\n",

    "Contribution of mixing in schools to the approximate R0 of this population is: ",schContr,"\n",

    "Contribution of mixing in workplaces to the approximate R0 of this population is: ",workContr,"\n",

    "Contribution of family mixing groups to the approximate R0 of this population is:
",wghtedTotalFamContr,"\n",

    "Contribution of family mixing groups by size are: ",famContr)}

```

Figure 4.7. Function to calculate approximate R_0 value. (cont.)

For the population that is explained in Section 4.1, assumptions that I made about R_0 -contributions coming from each mixing level are as the following:

- (i) I assumed that R_0 -contribution (to total R_0) coming from mixing in schools is 2 times of R_0 contribution (to total R_0) coming from the mixing in workplaces, since children are more vulnerable to infection than the adults.
- (ii) I assumed that R_0 -contribution (to total R_0) coming from mixing in family, in community and in neighborhood are nearly the same, in total the half of the total approximate R_0 calculated for the given population.

Herein, it should be noted that, since the intention of this thesis was to be realistic I did not use different infection transmission probabilities within each school group (pre-school, primary school, middle school, high school) or child to child and child to adult infection transmission parameters for each mixing level, since the values related to these parameters are very hard to predict in real life and calibration gets more difficult when there are too many parameters. As a result, I used one single infection transmission probability for school, one for work, one for family, one for neighborhood and one for community in the simulation model.

Finally, to get a desired approximate R_0 value for the inputted population (described in Section 4.1) and to provide the abovementioned assumptions, I made the following analyses and calculations:

To apply the assumption i above, in other words to get R_0 -contribution (to total R_0) coming from mixing in schools being 2 times of R_0 -contribution (to total R_0) coming from the mixing in workplaces, the following formula should hold:

$$(\text{aveSchoolSize}-1)*\text{pinfSchool}*\text{schWeight}*\text{ExpDur} = 2*(\text{aveWorkSize}-1)*\text{pinfWork}*\text{workWeight}*\text{ExpDur}$$

For the inputted population, the values to be used in the above equation are as the following:

- (i) aveSchoolSize, average school size = 283 people;
- (ii) aveWorkSize, average workplace size = 612 people;
- (iii) schWeight, weight of people who go to school calculated from IDSchWork column of the population matrix = 0.2264;
- (iv) workWeight, weight of people who go to work calculated from IDSchWork column of the population matrix = 0.612.

It is obvious that people with NA in IDSchWork column of the population matrix does not contribute to the approximate R_0 -calculation coming from school/work mixing level. When the values are put into the equation, the following simplified formula is found: $11,71378 * \text{pinfWork} = \text{pinfSchool}$.

With similar calculations I found out that at the end, to get the desired R_0 -contributions (to total R_0) from each mixing level, the relation between the probabilities should be as follows:

- $\text{pinfNei} = 20.04 * \text{pinfComm}$
- $\text{pinfFam} = 5102 * \text{pinfComm}$
- $\text{pinfWork} = 2.674 * \text{pinfComm}$
- $11,71378 * \text{pinfWork} = \text{pinfSchool}$

I planned to take $\text{pinfSchool} = 0.006$, so in that case, pinfWork should be 0.00051, pinfFam should be 0.098, pinfNei should be $3.84e-4$ and pinfComm should be $1.92e-5$. At the last step, after ensuring that the desired R_0 -contributions (to total R_0) from each mixing level is applied, to get the total approximate R_0 , I multiplied all infection probabilities with a factor. I made several trials, some of them with their outputs can be seen in the following figure:

```

fac<.1
R0_Calc(popDat=popData,pinfFam=fac*0.098,pinfSchool=fac*0.006,pinfWork=fac*0.00051,pinfNei=fac*
3.84e-4,pinfComm=fac*1.92e-5,minDayDisease=3,p=c(0.4,0.3,0.1,0.1,0.1))

Approximate R0 calculated for this population is: 4.830543

Contribution of community mixing group to the approximate R0 of this population is: 0.8063194

Contribution of neighborhood mixing groups to the approximate R0 of this population is: 0.8047872

Contribution of school/work mixing groups to the approximate R0 of this population is: 2.4127

Contribution of mixing in schools to the approximate R0 of this population is: 1.611738

Contribution of mixing in workplaces to the approximate R0 of this population is: 0.8009623

Contribution of family mixing groups to the approximate R0 of this population is: 0.806736

Contribution of family mixing groups by size are: 0 0.4116 0.8232 1.2348 1.6464 2.058 2.4696

```

Figure 4.8. Trial with factor value=1 and the function output.

In Figure 4.8, the output shows that the approximate calculated R_0 for the inputted population is 4.830543. My aim was to get an approximate R_0 value of 1.6. So, I needed to decrease the factor value in the `R0_Calc` function call.

```

fac<-0.4
R0_Calc(popDat=popData,pinfFam=fac*0.098,pinfSchool=fac*0.006,pinfWork=fac*0.00051,pinfNei=fac*3.8
4e-4,pinfComm=fac*1.92e-5,minDayDisease=3,p=c(0.4,0.3,0.1,0.1,0.1))

Approximate R0 calculated for this population is: 1.932217

Contribution of community mixing group to the approximate R0 of this population is: 0.3225277

Contribution of neighborhood mixing groups to the approximate R0 of this population is: 0.3219149

Contribution of school/work mixing groups to the approximate R0 of this population is: 0.9650801

Contribution of mixing in schools to the approximate R0 of this population is: 0.6446951

Contribution of mixing in workplaces to the approximate R0 of this population is: 0.3203849

Contribution of family mixing groups to the approximate R0 of this population is: 0.3226944

Contribution of family mixing groups by size are: 0 0.16464 0.32928 0.49392 0.65856 0.8232 0.98784

```

Figure 4.9. Trial with factor value=0.4 and the function output.

In Figure 4.9, it can be seen that the approximate calculated R_0 value is 1.932217. So, I decreased the factor value more to get an approximate R_0 value of 1.6. By making similar trials, trying different “fac” values and checking the function outputs, I found out that the “fac” value to get an approximate R_0 of 1.6, for the inputted population is 0.3317 when $\text{pinfFam}=0.098$, $\text{pinfSchool}=0.006$, $\text{pinfWork}=0.00051$, $\text{pinfNei}=3.84\text{e-}4$, $\text{pinfComm}=1.92\text{e-}5$. The final trial and the function output that indicates that the calculated approximate R_0 values is 1.6 can be seen in Figure 4.10:

```

fac<-0.3317

R0_Calc(popDat=popData,pinfFam=fac*0.098,pinfSchool=fac*0.006,pinfWork=fac*0.00051,pinfNei=fac*
3.84e-4,pinfComm=fac*1.92e-5,minDayDisease=3,p=c(0.4,0.3,0.1,0.1,0.1))

Approximate R0 calculated for this population is: 1.602291

Contribution of community mixing group to the approximate R0 of this population is: 0.2674561

Contribution of neighborhood mixing groups to the approximate R0 of this population is: 0.2669479

Contribution of school/work mixing groups to the approximate R0 of this population is: 0.8002926

Contribution of mixing in schools to the approximate R0 of this population is: 0.5346134

Contribution of mixing in workplaces to the approximate R0 of this population is: 0.2656792

Contribution of family mixing groups to the approximate R0 of this population is: 0.2675943

Contribution of family mixing groups by size are: 0 0.1365277 0.2730554 0.4095832 0.5461109 0.6826386
0.8191663

```

Figure 4.10. Trial with factor value=0.3317 and the function output.

The calculated approximate R_0 value gives an idea about how the disease characteristics may be. For instance, if the calculated R_0 value for an experiment is less than “1”, one can think, that no outbreak would occur. But for non-homogeneous populations, the experiments that I have done and their results show that if in one mixing group there is (a) highly infective person (s), even if the total approximate R_0 value is less than 1, outbreak may be seen. The outbreaks that are seen in those cases may have some of their characteristics different than the ones in homogeneous populations with the same R_0 value (see Part 6, results of the experiments, for details).

5. SCENARIOS

After baseline model building phase, by adding details, a variety of scenarios can be simulated and in this part, different scenarios are discussed. By simulating these different scenarios, the aim is to see the impact of intervention techniques used to slow down the disease spread. For example, vaccines are generally utilized in public health to prevent infections by giving the person immunity to the disease. Or, as Longini *et al.* [3] did, as another intervention technique, usage of antiviral agents against disease spread can be considered. Another detail to be added to the baseline model may be the fact that some people stay at home when they understand that they got the disease. With these ideas in mind, I tried 2 scenarios: Firstly, I thought about the vaccination case. I searched for an answer to the question “If a part of the population gets immune to the disease, how different would the simulation results be?” Secondly, I thought about the impact of people who stay home after they got infected. The results of simulations are given in Part 6.

5.1. Scenario 1: Vaccination

In this scenario, it is assumed that a part of the population is vaccinated and the vaccinated ones get immune with a probability `probImm`. Therefore, a new input is added to the simulation code: `probImm`, which refers to the probability that a person is vaccinated and gets immune.

To apply this idea, a new random vector of the length of susceptible people in the population is generated. If the corresponding random number of a person is less than `probImm`, then the person gets immune and passes directly to “R (Recovered)” state. So, there will be recovered, at the start, just after the initialization of the population. The following code part in the figure shows how entities pass directly and randomly from S to R, in case of immunity due to vaccination.

```
UImm<-runif(length(which(pdat$state=="S")))
pdat$state[which(pdat$state=="S")][UImm<probImm] <-"R"
```

Figure 5.1. Code part that shows passing directly from S to R.

It is clear that the number of people that will not be affected from the infection will increase in this scenario and the attack rate calculation will change. After the simulation, the attack rate for this scenario will be calculated by the following formula:

$\text{percInfected} = (\text{npeople} - \text{nImm} - \text{I0} - \text{nS}) / (\text{npeople} - \text{nImm} - \text{I0})$ where, percInfected is the attack rate, the percentage of the whole population that is affected by the infection, npeople is the size of whole population, nImm is the number of people who got immune to the disease by vaccination, I0 is the initial number of infected people seeded to the population and nS is the number of people who are still susceptible.

5.2. Scenario 2: Stay Home

In this scenario, the idea of people staying home when infected is studied. Usually, during the first day of the infection, people do not know that they are infected or even if they do, they are usually spreading the virus to people that they contacted in the first day of their infection before they return home. So the first day, people stay in contact with others in each mixing group and spread the virus and from the 2nd day of infection till the end of infection, they stay home. According to my code, during the 1st day of the infection, people are spreading the disease in each mixing group and from the 2nd day, they only spread the infection in their home. In that case, I added a new input to the code: probHome , which refers to the probability that a person stays home once he is infected. I created a random vector (UHome) with a length equal to the number of newly infected people of that day. Indexes of the newly infected people are hold in the vector inewI . If among the newly infected people of that day, in the main for loop, the corresponding random number for a person is greater than or equal to probHome , then his state will change from “S (Susceptible)” to “I (Infected)”. If the corresponding random number for a person is less than probHome , then he

will pass from state “S” (Susceptible) to state “IH” (Infected and will stay home the second day) which means that he will stay home from the second day of his infection by passing to state “H” (Home). The code showing this process is shown in Figure 5.2:

```
if(!is.null(inewI))
{
  UHome <- runif(length(inewI)) inewIH <- inewI[UHome<probHome]
  inewI <- inewI[UHome>=probHome]}
```

Figure 5.2. A section from the code.

I implemented passing from “IH” state to “H” state at the 2nd day of infection by letting all people with current state “IH” pass to state “H” at the following day, by writing in the beginning of the for loop the following command:

```
pdat$state[pdat$state=="IH"]<-"H" #let all people with state “IH” pass to state “H”
```

Figure 5.3. A section from the code that lets all people with state "IH" pass to state "H".

With the above line in Figure 5.3, each morning, if there is someone in state “IH”, he passes directly to “H” state.

The code is designed so that all people in state “IH” are at their first day of infection and can spread the virus in each of their mixing groups that they are in. Therefore, I made a change in the update function that uses the table command to count the number of infected people. I used “substr” function to take only the first letter of a state into consideration while counting: substr(pdat\$state,1,1).

Therefore, during the simulation, if a person has the state “IH” at the first day of infection, he will be counted as “I”. This change is important for calculating the correct transmission probabilities. Thanks to this change, when a person has state “IH” in the first day of infection, he will be counted as “I” (Infected) and be able to spread the disease in all mixing groups.

After having passed from state “IH” to state “H”, a person will be able to spread the infection only in the family. It is clear that when someone stays home, he is able to infect other members of his family. The formula for the probability not to be infected is therefore changed to the following: $P = [(1-p_f)^{I_f+H_f} (1-p_{sch})^{I_{sch}} (1-p_n)^{I_n} (1-p_c)^{I_c}]$ where p_f , p_{sch} , p_n , p_c are the transmission probabilities within family, within school, within neighborhood and within community, respectively. I_f , I_{sch} , I_n and I_c are the number of infected people in family mixing group, school mixing group, neighborhood mixing group and community mixing group and H_f denotes the number of people that got infected and are staying home.

As mentioned in Section 3.3, for the people that are in state “I”, end times were generated. In this scenario, with the people in state “I”, the code randomly generates end times also for the people at state “H”, in the main for loop. And as for the people that are in state “I”, for the people that are in state “H”, if the current time of the loop is equal to the end time generated for the person, the person passes to state “R” (Recovered). The code in the following figure controls that a person, finishing his staying home period, passes from state “H” to state “R”.

```
if(!is.null(inewI)){
  #inewI: vector holding the indexes of newly infected people
  ivHend <- (1:npeople)[pdat$state=="H" & pdat$endState==it]
  pdat$state[ivHend] <- "R"}
```

Figure 5.4. A section from the code.

In this scenario, similar to the idea explained in Section 3.2, if there are no people in state “I”, “IH” or “H”, the for loop stops, using the break command shown in the following figure:

```
if(length(which(pdat$state=="I")) ==0 & length(which(pdat$state=="H")) ==0 &
length(which(pdat$state=="IH")) ==0) break
```

Figure 5.5. Code showing exiting the loop with break command.

5.2.1. Debugging of Scenario 2

In this section, debugging and controlling the correctness of the implementation of this scenario is discussed. I controlled every step of the program by using a trace object. I built a small, simple population of 10 people, for simplicity. A screenshot of this population matrix can be seen in the following figure:

	IDfam	IDneig	IDcomm	IDSchWork
1	1	1	1	3001
2	1	1	1	9001
3	2	1	1	2001
4	2	1	1	9001
5	3	2	1	2001
6	3	2	1	9001
7	4	2	1	3001
8	4	2	1	9001
9	5	1	1	2001
10	5	1	1	3001

Figure 5.6. Screenshot of the primitive population matrix formed for debugging.

The first line of trace prints the current time (it). The second line prints the random vector of uniform numbers used to determine whether or not a newly infected person will stay at home the following day (UHome). The third line prints `inewIH`, the indices of the randomly selected people among the newly infected ones that will pass to state “IH” and then to state “H” the following day. Fourth and fifth lines shows state and endstate columns of the matrix, respectively. Finally the sixth line shows the total number of infected and susceptible people in the whole community.

```
[1] "it= 1"
logical(0) numeric(0) logical(0)
[1] "I" "S" "S" "S" "S" "S" "S" "S" "I" "S"
[1] 4 0 0 0 0 0 0 0 7 0
  I R
1 2 0
[1] "it= 2"
integer(0) [1] 0.277068
[1] 10
  [1] "I" "S" "S" "S" "S" "S" "S" "S" "I" "IH"
  [1] 4 0 0 0 0 0 0 0 7 9
  I R
1 3 0
```

Figure 5.7. Tracing output for for days 1 and 2.

At day 2, it can be seen that `UHome=0.277068` is generated. `probHome` for this run was set to 0.5. So, it is expected that the person with with ID=10 passes to state “IH” if `UHome < probHome`, to state “I” if `UHome > probHome`. 0.277068 is less than 0.5, so the 10th person should pass to state “IH” at day 2, and to state “H” at the following day, day 3. Another check would be the following: At day 2, there are 3 people counted as infected in the community table. When the 10th person with state “IH” passes to state “H” at day 3, the number of infected decreases to 2. It should also be noted that end time for the person with state “IH” is generated and equal to 9.

```

[1] "it= 3"
logical(0) numeric(0) logical(0)
[1] "I" "S" "S" "S" "S" "S" "S" "S" "I" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 2 0
[1] "it= 4"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "I" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 1 1

```

Figure 5.8. Tracing output for days 3 and 4.

At the 4th day, it is seen that the person with ID=1 passed from state I to state R. The table is also updated. There is 1 infected person and 1 recovered person. Checking of this part of the code was made in Section 3.3.

```

[1] "it= 5"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "I" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 1 1
[1] "it= 6"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "I" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 1 1

```

Figure 5.9. Tracing output for days 5-9.

```

[1] "it= 7"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "R" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 0 2
[1] "it= 8"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "R" "H"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 0 2
[1] "it= 9"
logical(0) numeric(0) logical(0)
[1] "R" "S" "S" "S" "S" "S" "S" "S" "R" "R"
[1] 4 0 0 0 0 0 0 0 7 9
  I R
1 0 3

```

Figure 5.9. Tracing output for days 5-9 (cont.).

It is seen that on day 9, the 10th person passed to state “R” from state “H” and the number of recovered in the table is updated to 3. It should also be noted that the end time generated for the 10th person is 9 and at day 9, the person correctly finishes the infection.

6. EXPERIMENTS AND RESULTS

In this part, results of simulation runs with a population of size 10000, with 1000 independent repetitions and during 150 days are shown (see Part 4 for input population details). As other parameters, the number of initially infected people is 12, the minimum disease duration is 3 days, the vector holding the CDF of disease duration is (0.4, 0.7, 0.8, 0.9). The infection probability values are 0.0325 for pinfFam , 0.002 for pinfSchool , $1.7\text{e-}4$ for pinfWork , $1.3\text{e-}4$ for pinfNei , $6.37\text{e-}6$ for pinfComm (see Part 4 for calibration of the infection probabilities).

6.1. The Baseline Model Results

In this section, results of the baseline simulation runs are presented. The approximate calculated R_0 value is 1.6 (see Section 4.2 for details of approximate R_0 calculation and calibration of the infection probabilities). As the approximate calculated R_0 is 1.6, a high attack rate is expected.

A table showing the simulation outputs, a plot showing the average number of infected people, a histogram of attack rates, a histogram of the disease peaking day number, and a histogram of maximum number of infected on a single day of each repetition can be seen with details and the interpretations in the following table and figures. The total simulation time on a moderate notebook computer is 6010 seconds and the execution time for 1 simulated day is 0.105 seconds.

Table 6.1. Simulation outputs of the baseline model.

Average of maximum number of infected at a single day	1622
The average time between the start and peak point	19.7 days
The average time till the epidemic dies out	57.2 days
The average attack rate	0.551

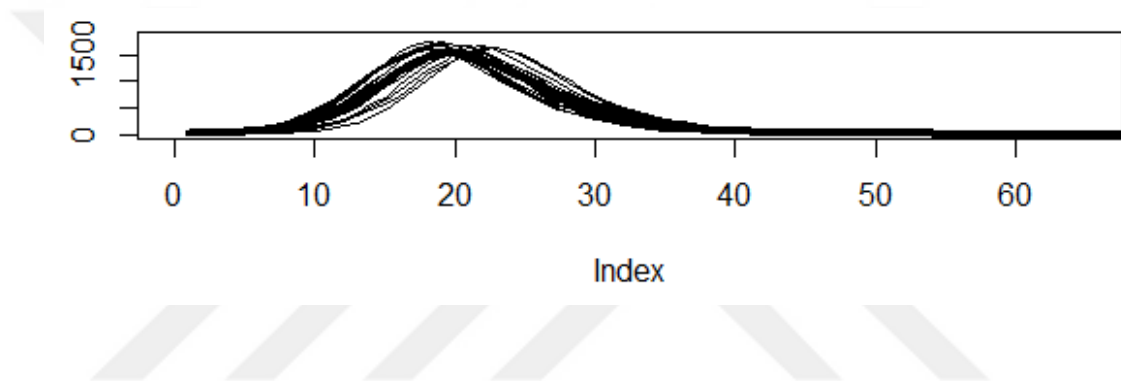


Figure 6.1. A plot showing the average number of infected people per day and number of infected people per day for random repetitions.

In Figure 6.1, the thick black line shows the average number of infected people during each day of the baseline simulation model. It is seen that the disease is peaking on average around at day 20. However, from the thin lines representing the paths of different independent repetitions, it is seen that peaking on days 17 or 24 is also possible. This is an expected result and occurs because of the randomness included in the model. This randomness is the reason that each thin line on the figure above has different characteristics. Also, It can be concluded from the figure that the epidemic dies out totally around the day 55, when the number of infected people reaches zero.

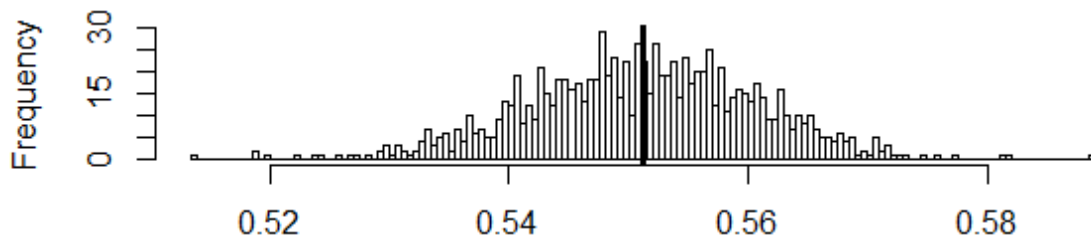


Figure 6.2. Histogram of attack rates of the baseline model, mean value is 0.551.

Figure 6.2 shows the percentage of people that are infected during the epidemic, in other words the attack rate. The mean attack rate is 0.551 (shown in figure with dark black line) has 95% probability that it is between 0.5505515 and 0.5517484 assuming that the original random variable is normally distributed (Since the number of repetitions is 1000, using the central limit theorem, normal distribution assumption can be used.). It can also be concluded that the attack rates are mostly between 0.515 and 0.585. It is important to notice that, since the number of initially infected people is 12, outbreak tendency is expected in independent repetitions. Assuming that no outbreak occurs when the percentage of infected people in the population is less than or equal to 10%, an outbreak rate is calculated. It is 100% for this scenario, which means there is outbreak in all repetitions.

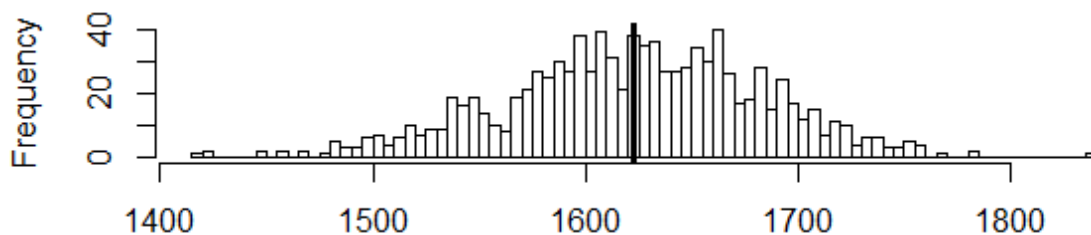


Figure 6.3. Histogram of maximum number of infected on a single day of each repetition.

Figure 6.3 shows that maximum number of infected on a single day is mostly between 1400 and 1800. The mean value is 1622 (shown with black line). The 95% confidence interval for the mean is [1618.438, 1625.94].

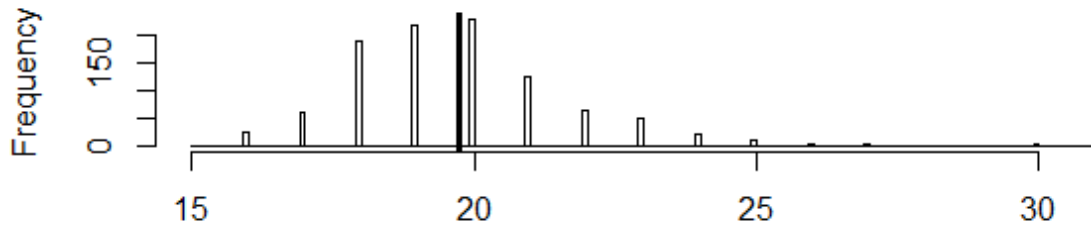


Figure 6.4. Histogram of disease peaking day of each independent repetition.

In Figure 6.4, it is seen that the disease peaking day is on average 19.7 (shown with thick black line) and peaking occurs generally between 14th and 26th days. The 95% confidence interval for the mean is [19.61514, 19.86086].

The epidemic dies out in this scenario on average in 57.2 days. From the histogram shown in Figure 6.5, It is seen that the epidemic generally dies out between days 45 and 80, but there are a few which last more than 80 days.

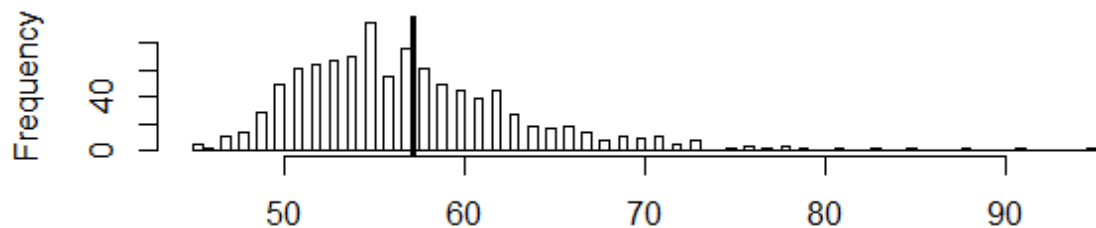


Figure 6.5. Histogram of time when the epidemic dies out.

6.2. Results of the Scenarios

This section presents the results of several combinations of the scenarios that are explained in Part 5. The approximate R_0 calculation for each section is shown in details in its beginning. The simulations start with 12 initially infected people. The other parameters are the same as the ones in Section 6.1. Additively, these experiments include probImm and probHome parameters. As mentioned in Part 5, probImm is the probability that a person gets vaccinated and immune and probHome is the probability of staying home when infected.

6.2.1. Results with probHome 0.2

For this scenario, probHome is 0.2, probImm=0 and the approximate R_0 value is 1.4. It is calculated by taking the days of mixing (in contact groups) into consideration (see the details in Section 4.2 for approximate R_0 calculation). It is clear that when a person stays home after the 1st day of his infection, he only mixes with his family and so, he will have only 1 day of mixing in all other levels.

While calculating the approximate R_0 value for this scenario, following steps (that are different from the ones in Section 4.2) have been taken:

- (i) For 20% of the population, the following formula, that is very similar to the one in Section 4.2, is used:

$$\text{pinfComm} * (\text{CommSize} - 1) * 1 + \text{pinfNei} * (\text{NeiSize} - 1) * 1 + \text{pinfSchool} * (\text{aveSchoolSize} - 1) * \text{schoolWeight} * 1 + \text{pinfWork} * (\text{aveWorkSize} - 1) * \text{workWeight} * 1 + \sum_{i=1}^I \text{pinfFam} * (i - 1) * \text{famProportion}[i] * 4.2, \text{ where } i \text{ represents family size from 1 to } I(\text{maximum family size}).$$

Figure 6.6. First step of calculating the approximate R_0 value.

- (ii) For the rest, that is 80%, the same formula as in Section 4.2 is used:

$$(\text{pinfComm} * (\text{CommSize} - 1) + \text{pinfNei} * (\text{NeiSize} - 1) + \text{pinfSchool} * (\text{aveSchoolSize} - 1) * \text{schoolWeight} + \text{pinfWork} * (\text{aveWorkSize} - 1) * \text{workWeight} + \sum_{i=1}^I \text{pinfFam} * (i - 1) * \text{famProportion}[i]) * 4.2.$$

Figure 6.7. Second step of calculating the approximate R_0 value.

The result $0.5851543 * 0.2 + 1.601 * 0.8 = 1.4$ is taken as the approximate calculated R_0 value for this case.

Since R_0 is approximately 1.4, the average attack rate is anticipated to be less than the previous case with baseline model that had R_0 value of 1.6: It is seen in Table 6.2 that the average attack rate for this case is 0.475 thus, as expected, less than the average baseline model attack rate.

Table 6.2. Simulation outputs of Scenario 2 with probHome=0.2.

Average of maximum number of infected at a single day	1096
The average time between the start and peak point	21.4 days
The average time till the epidemic dies out	58.4 days
The average attack rate	0.475

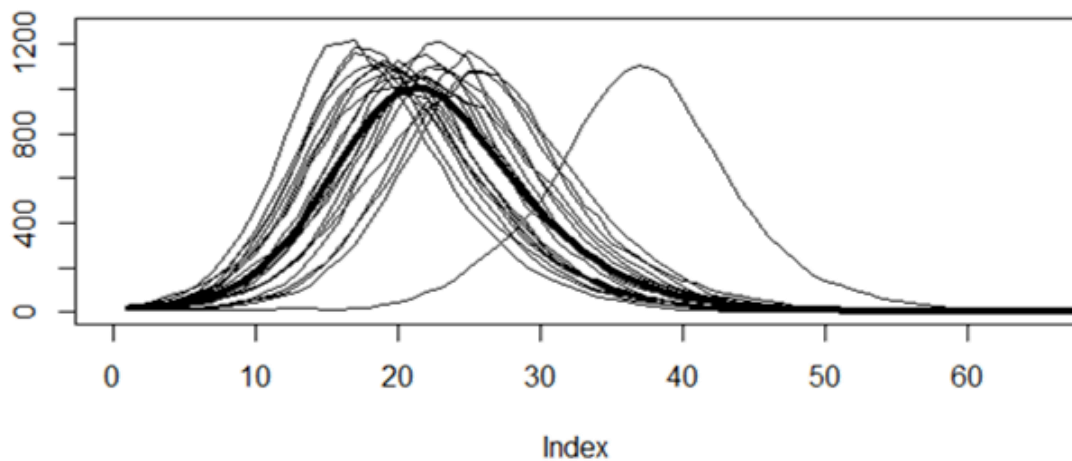


Figure 6.8. A plot showing the average number of infected people per day and the number of infected people per day for random repetitions with $\text{probHome}=0.2$ and $\text{probImm}=0$.

In the figure above it is seen from the thick black line that the disease peaks on average at around 21st day. Due to the randomness used in the model, it is seen from the lines representing independent repetitions that, peaking at around 18th, 27th and even on the 38th days is also possible. Figure also shows that, after day 60, no infection is seen among the population.

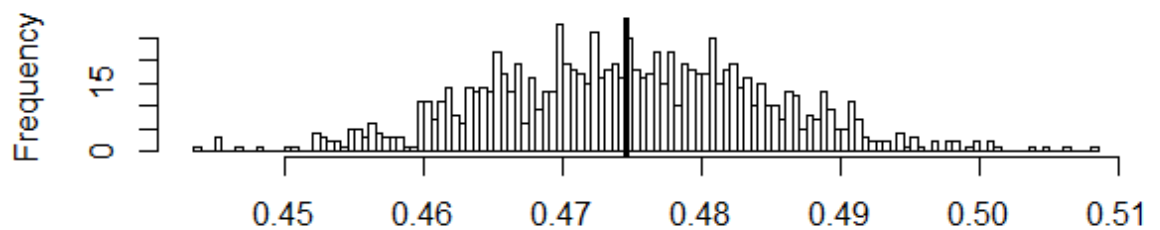


Figure 6.9. Histogram of attack rates with $\text{probHome}=0.2$ and $\text{probImm}=0$.

In Figure 6.9, it is seen that the average attack rate for this scenario is between 0.44 and 0.51. Its mean is 0.4745295 for this case with 95% confidence interval [0.473907, 0.4751521].

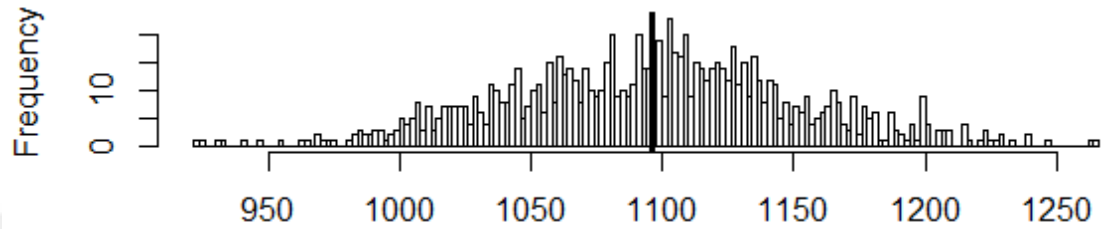


Figure 6.10. Histogram of maximum number of infected of each independent repetition with probHome=0.2 and probImm=0.

The figure above shows the histogram of maximum number of infected of each independent repetition. It is usually between 900 and 1300, its mean is 1096 and 95% confidence interval for the mean is [1092.791, 1099.657].

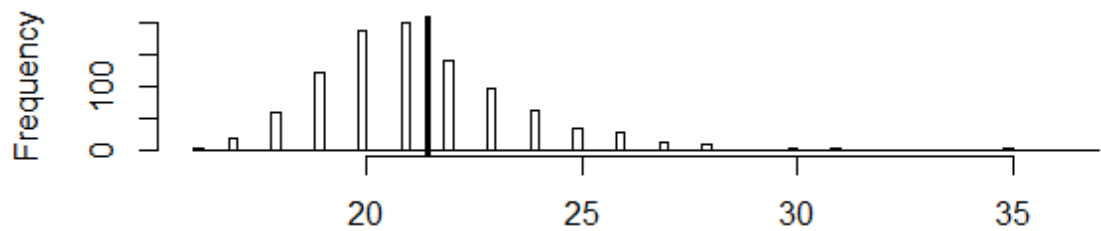


Figure 6.11. Histogram of disease peaking days of each repetition.

In Figure 6.11, it is seen that peaking occurs generally between 16th and 30th days with mean 21.4. 95% confidence interval for the mean is [21.25549, 21.57451]. The mean in baseline model case with R_0 approximately equal to 1.6 was around 19.7, it can be commented out that when the approximate R_0 value is higher, peaking occurs more quickly.

6.2.2. Results with probHome 0.5

Approximate calculated R_0 for this case is 1.1 (see Section 4.2 and 6.2.1 for details). Outputs of the simulation can be found in the table and figures below:

Table 6.3. Simulation outputs with probHome=0.5, probImm=0.

Average of maximum number of infected at a single day	439
The average time between the start and peak point	25.7 days
The average time till the epidemic dies out	63.2 days
The average attack rate	0.332

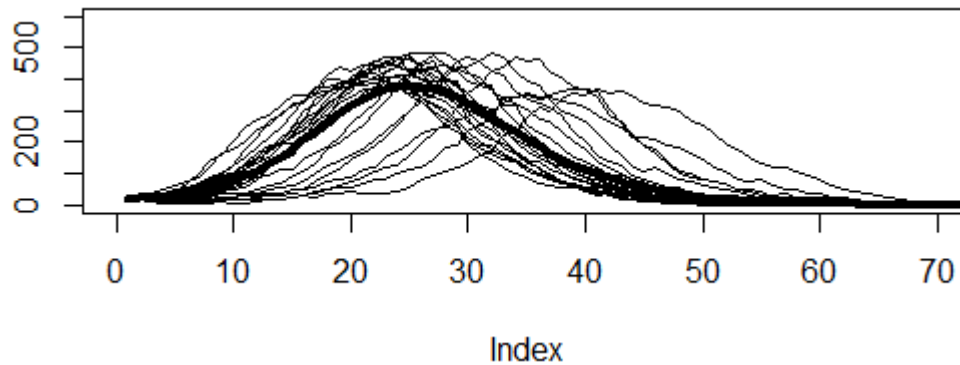


Figure 6.12. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.

In Figure 6.12, the average number of infected during each day of simulation is shown with a dark line. The other lines represent several independent repetitions. It is seen that after day 55, epidemic dies out in most repetitions, approximately after day 70, no infected is seen in any repetition. Peaking of disease occurs at approximately at day 25, with average number of infected around 400. The reason why this number is small is because the people that are

infected are staying home with a probability 0.5 and this prevents disease transmission after the first day in contact groups with the exception of family contact group.

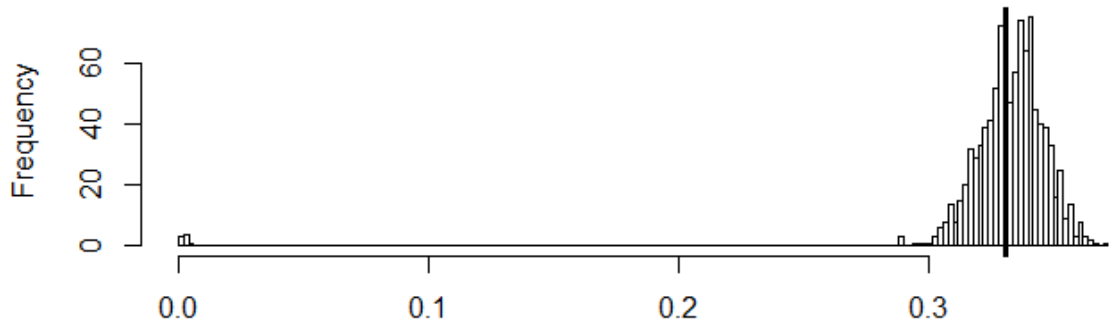


Figure 6.13. Histogram of attack rates with probHome=0.5.

Figure 6.13 shows the histogram of attack rates with staying home probability 0.5. It is seen that the percentage of people that are infected of each repetition is between 0 and 0.4 and mostly between 0.3 and 0.4. Its mean is 0.332 with 95% confidence interval of [0.330527, 0.334037].

The proportion of repetitions with attack rate less than or equal to 0.1 for this case is 0.006. In other words, with a probability of 0.994, there will be an outbreak.

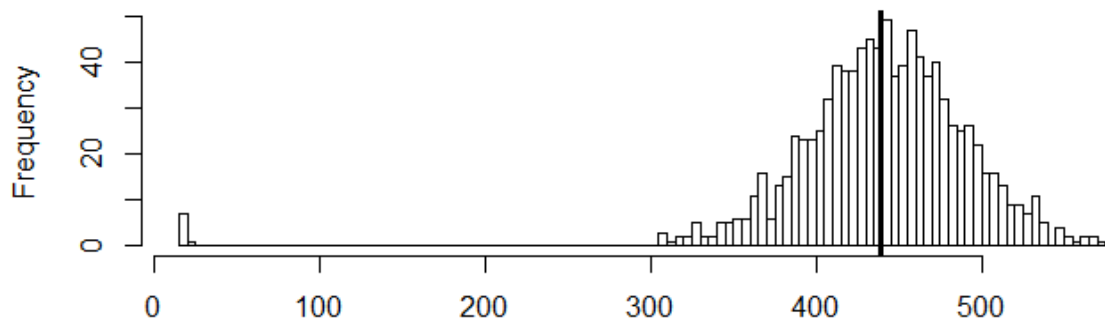


Figure 6.14. Histogram of maximum number of infected on a single day of each repetition.

Figure 6.14 shows the histogram of the maximum number of infected on a single day of each repetition. Its mean is 439; most of the observed values are between 300 and 600. As mentioned earlier in this section, with a probability of 0.994 there is outbreak.

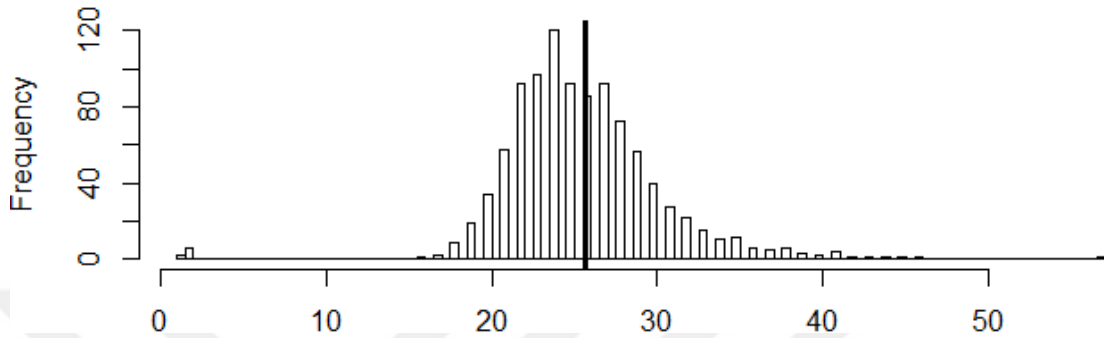


Figure 6.15. Histogram of disease peaking days of each repetition.

In Figure 6.15, it is seen that peaking occurs generally between 15th and 35th days with mean 25.7. 95% confidence interval for the mean is [25.36214, 25.96786]. In the case with probHome=0.2, probImm=0 and with R_0 approximately equal to 1.4, the mean was 21.2, it can be commented out again that, when the approximate R_0 value is higher, peaking occurs more quickly.

6.2.3. Results with probImm=0.15 and probHome=0.2

In this section, results of the simulations with probability of staying home when infected is equal to 0.2 and probability of being vaccinated and getting infected is 0.15 are analyzed. The approximate R_0 is calculated to be 1.19. The R_0 in school mixing group is 1.698.

The approximate R_0 value of the case where probHome=0.2 and probImm=0 was calculated to be 1.4 in Section 6.2.1. When probImm=0.15, the proportion of susceptible people in the population is 0.85. Thus, the resulting R_0 value 1.19 is found for this section

by multiplying the proportion of susceptible people in the population that is 0.85 with the previously calculated approximate R_0 value for the case with probHome=0.2 that is 1.4. With an approximate R_0 value of 1.19, the relative decrease (to the previous sections) of the “attack rate”, of the “average maximum number of infected on a single day” and of “disease duration interval” are expected. The results can be found in the table and figures below:

Table 6.4. Simulation outputs of combined scenarios with probImm=0.15 and probHome=0.2.

Average of maximum number of infected at a single day	624
The average time between the start and peak point	25 days
The average time till the epidemic dies out	62.7 days
Execution time for 1 simulated day	0.294
The average attack rate	0.38

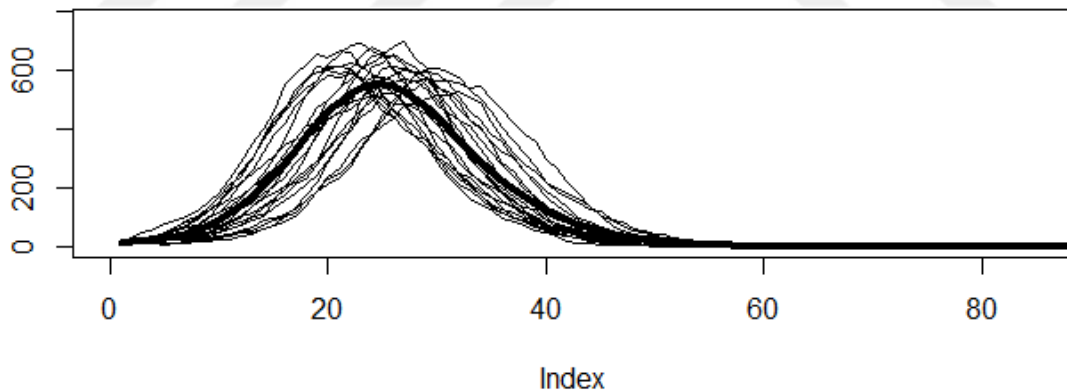


Figure 6.16. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.2 and probImm=0.15.

With an approximate R_0 value of 1.19, outbreak is expected. It can be seen from the figure above that outbreak occurs. The number of infected people is peaking around at day 25 and the mean of maximum number of infected people on a single day is around 600. It

was around 1100 without vaccination (when $\text{probImm}=0$) thus, it can be interpreted that vaccination decreases the average number of infected people on a single day.

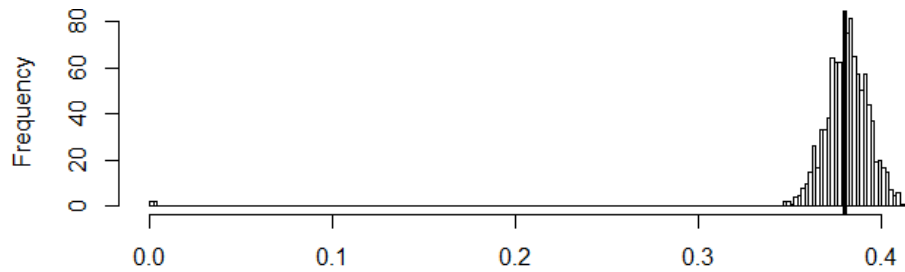


Figure 6.17. Histogram of attack rates of combined scenarios with $\text{probImm}=0.15$ and $\text{probHome}=0.2$.

The figure above shows the histogram of attack rates. Mean for attack rates for this case is 0.38 and 95% confidence interval for the mean is $[0.3782287, 0.3815176]$. Proportion of repetitions that have an attack rate less than or equal to 0.1 is 0.004. So, it can be interpreted that with a probability 0.996, an outbreak occurs.

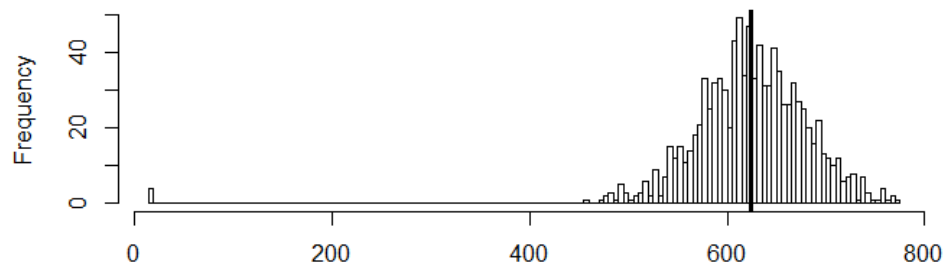


Figure 6.18. Histogram of maximum number of infected on a single day of each repetition.

Figure 6.18 represents the histogram of maximum number of infected people on a single day of each repetition. The mean for maximum number of infected on a single day of each repetition is 624 with 95 confidence interval $[619.6508, 627.7172]$. Almost all values are between 450 and 850.

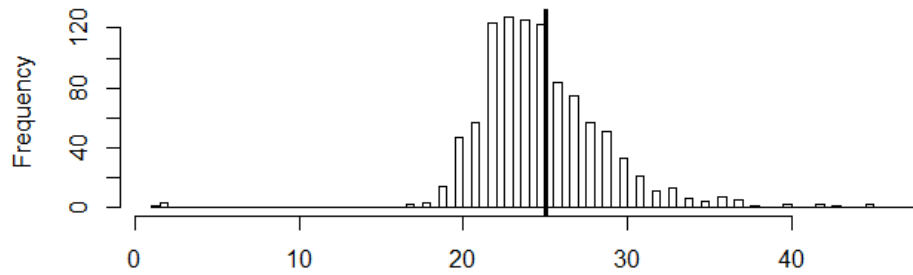


Figure 6.19. Histogram of disease peaking days.

In this case, as can be seen from the figure above, peaking occurs generally between 15th and 35th days with mean 25; 95% confidence interval for the mean is [24.79132, 25.29868]. The mean in baseline model case with R_0 approximately equal to 1.6 was around 19.9, it can be concluded that when R_0 is greater, peaking occurs more quickly.

6.2.4. Results of Simulation with probImm=0.4 and probHome=0.2

Approximate calculated R_0 value is 0.84 (see Section 4.2 and 6.2.1 for details.). In the following table and figures, simulation outputs are presented:

Table 6.5. Simulation outputs of combined scenarios with probImm=0.4 and probHome=0.2.

Average of maximum number of infected at a single day	104
The average time between the start and peak point	37.1 days
The average time till the epidemic dies out	84.3 days
Execution time for 1 simulated day	0.0798 seconds
The average attack rate	0.164

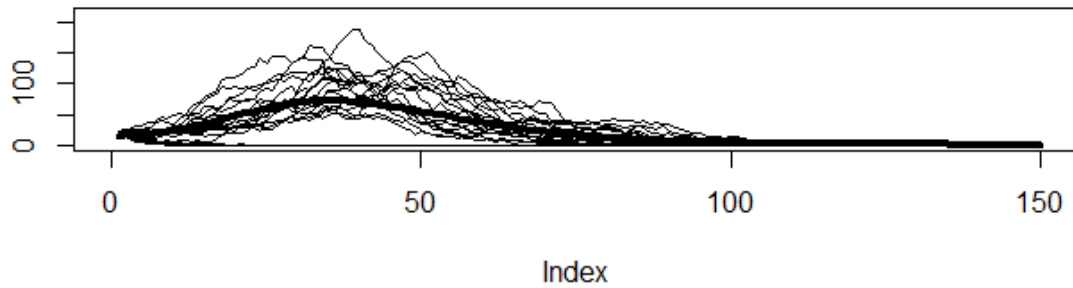


Figure 6.20. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with $\text{probHome}=0.2$ and $\text{probImm}=0.4$.

Figure 6.20 and Table 6.6 show that there is an outbreak in this case. This is an interesting and unexpected result with such a low R_0 value. The reason of seeing such a result is expressed as the following: In the beginning, among 12 initially infected people, the mean number of initially infected children is 2.8 and the R_0 -contribution in schools is approximately 1.199 for this case. Thus, even though 0.84 is a low R_0 value and no outbreak is expected with such an R_0 value less than 1 in a homogeneous population, the simulation results show that, when the R_0 -contribution in one or more mixing groups is above 1, even though the approximate R_0 value for the whole population is low, disease outbreak may occur. In other words, the infection spread in one mixing group (the schools, in this research) affects the overall disease spread characteristics and these characteristics are highly related to the parameterization that is explained in detail in Section 4.2.

The calculated approximate R_0 value gives an idea about how the disease characteristics may be. If the calculated R_0 value for an experiment is less than “1”, one can think, that no outbreak would occur. But for non-homogeneous populations, the experiments that I have done and their results show that if in one mixing group there are highly infective people, even if the approximate R_0 value is less than 1, an outbreak may occur. The outbreaks that are seen in those cases may have different characteristics different than an outbreak in a homogeneous population with the same R_0 value.

As mentioned earlier in Section 4.2, the R_0 -contribution coming from mixing in schools are forming a big part of the total calculated R_0 value. So, even though the

approximate total R_0 value of the whole population is less than 1, outbreak may occur if the infection spread in schools is high. Table 6.6 shows the average attack rates in each school in this experiment when the average attack rate calculated for the whole population is 0.164. It can be seen in the table below that even though the average attack rate is low for the whole population, the attack rates in schools are high. Thus, even though the outbreak expectation is low regarding the average total attack rate and the approximate calculated R_0 value, the high R_0 -contribution in schools mixing level may result in a disease outbreak. The characteristics of this outbreak differ from those in perfectly homogeneous populations with the same R_0 . Consequently, it would be appropriate to say that, approximate R_0 value calculation for non-homogeneous populations should not be evaluated alone and the parameterization in simulations and the mixing in each level of the population should be carefully examined.

Table 6.6. Average attack rates of 8 schools when the approximate calculated R_0 is 0.84 and when the average total attack rate is 0.164.

1	0.4998355
2	0.4775839
3	0.4202564
4	0.4284982
5	0.4117153
6	0.4459441
7	0.4344086
8	0.4414695

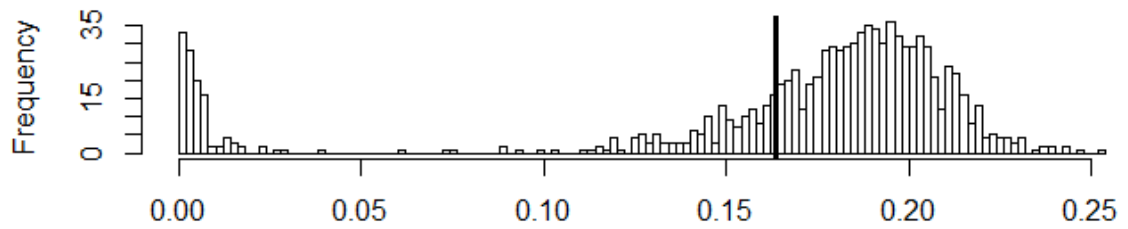


Figure 6.21. Histogram showing the attack rates of simulations with $\text{probHome}=0.2$ and $\text{probImm}=0.4$.

Mean for attack rate for this scenario is 0.164 with 95% confidence interval [0.1597689, 0.1674196]. The proportion of independent repetitions where attack rate is less than or equal to 0.1 for this case is 0.122. Thus, assuming that when the attack rate in a repetition is less than 0.1 there is no outbreak, the outbreak probability for this case is 0.878.

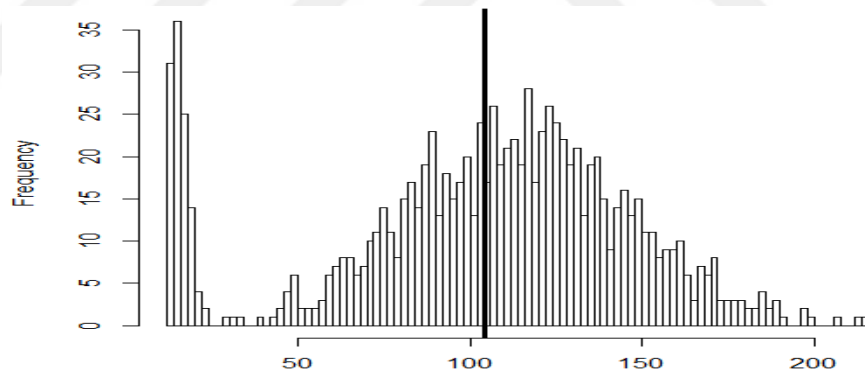


Figure 6.22. Histogram of maximum number of infected on a single day of each repetition.

The average maximum number of infected on a single day of each repetition, 16.7 with 95% confidence interval [16.52265, 16.88335], is less than the previous case in Section 6.5, which was 29.8 on average. As can be seen from the histogram on Figure 6.22, in most of the cases there are more than 75 infected people in maximum on a single day, thus, as mentioned before in this section, outbreak probability is high for this case.

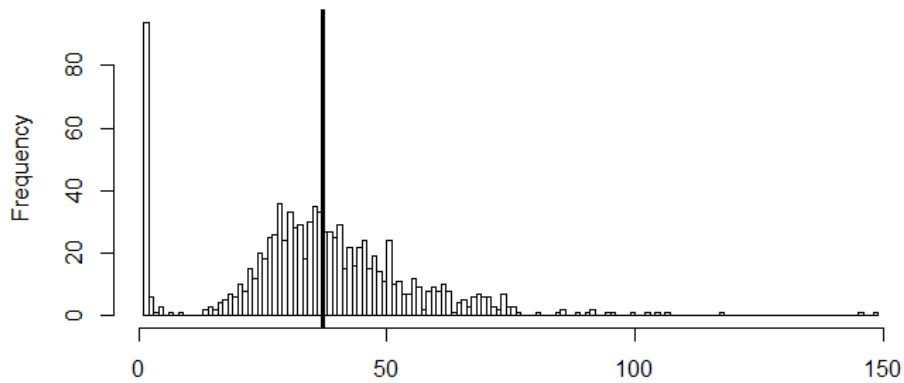


Figure 6.23. Histogram of disease peaking days.

The figure above shows the histogram of disease peaking days. It can be seen on the left-hand side of the histogram that the disease dies out in a few repetitions (out of 1000 repetitions) since the maximum number of infected (the peak point) is seen in the beginning of these repetitions. The mean of peaking day is 37.1.

6.2.5. Results of Simulation with probImm=0.15 and probHome=0.5

The approximate calculated R_0 for this case is 0.935. Simulation outputs can be seen in the following table and figures:

Table 6.7. Simulation outputs with probHome=0.5 and probImm=0.15.

Average of maximum number of infected at a single day	190
The average time between the start and peak point	32.3 days
The average time till the epidemic dies out	75.7 days
Execution time for 1 simulated day	0.0918 seconds
The average attack rate	0.229

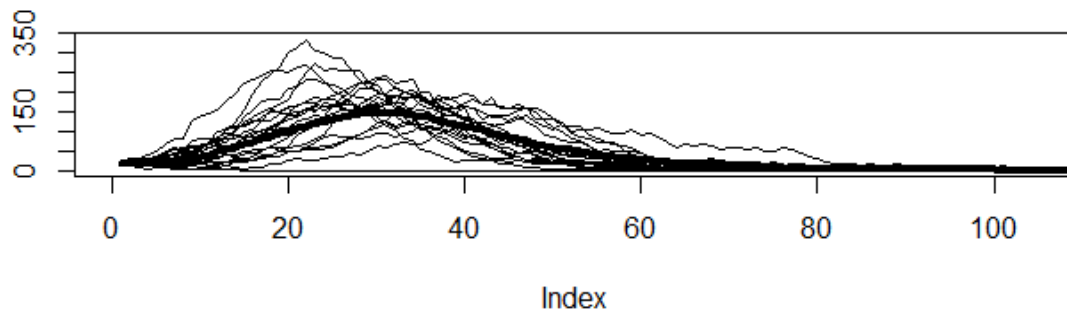


Figure 6.24. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.15.

The plot in Figure 6.24 shows that the disease in this case is peaking around the day 32 and is dying out around at day 75. It can be seen that there are some repetitions where the disease dies out immediately after the start. This is because of the stochastic nature of the simulation model and an expected result. The outbreak is seen in this experiment even though the approximate calculated R_0 is 0.935. The average number of the infected children at the beginning is 2.6 for this case and the R_0 -contribution coming from mixing in schools is 1.24. This high R_0 -contribution above 1 is the reason of seeing a disease outbreak (see Section 6.2.4 for details.).

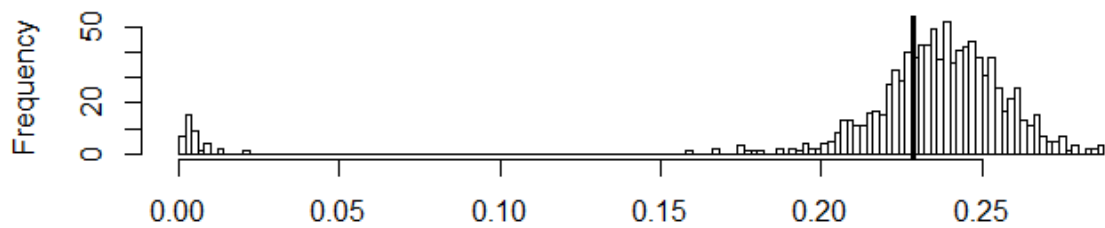


Figure 6.25. Histogram of attack rates with probHome=0.5 and probImm=0.15.

The mean attack rate for this scenario is 0.229 with 95% confidence interval [0.2256 304, 0.2316415]. The proportion of repetitions that have an attack rate less than or equal to 0.1 is 0.039. They can be seen in the left-hand side of the second histogram in the above figure.

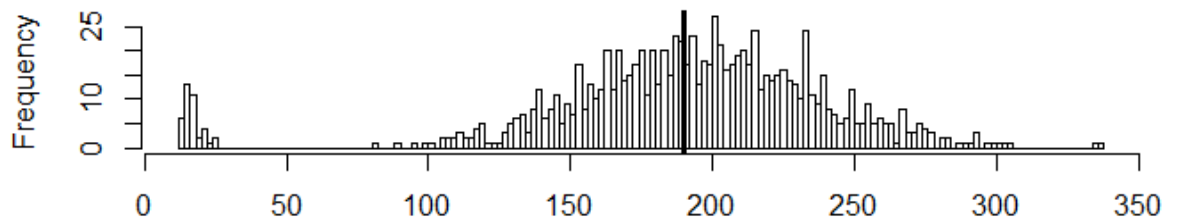


Figure 6.26. Histogram of maximum number of infected on a single day of each repetition.

The figure above shows the histogram of maximum number of infected on a single day for each repetition. Maximum number of infected on a single day is on average 190 with 95% confidence interval [190.09, 193.3111].

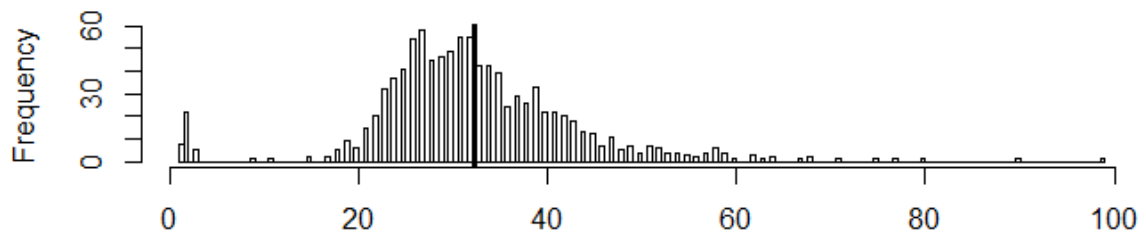


Figure 6.27. Histogram of peaking days.

Figure 6.27 represents the histogram of the day numbers (of each repetition) where the amount of infected people is the highest. The mean these numbers is 32.3, with 95% confidence interval of [31.58576, 32.96224].

6.2.6. Results of Simulation with probImm=0.4 and probHome=0.5

Approximate calculated R_0 value is 0.66 for this case (see Section 4.2 and 6.2.1 for details.). In the following table and figures, simulation outputs are presented:

Table 6.8. Simulation outputs of combined scenarios with probImm=0.4 and probHome=0.5.

Average of maximum number of infected at a single day	21.6
The average time between the start and peak point	9.73 days
The average time till the epidemic dies out	37.3 days
Execution time for 1 simulated day	0.177 seconds
The average attack rate	0.0226

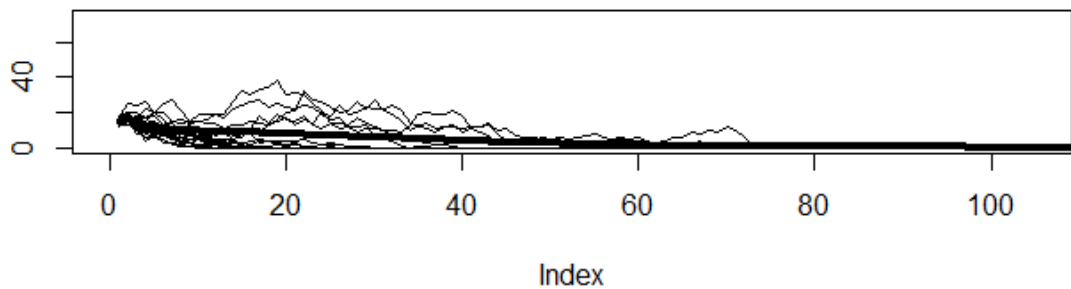


Figure 6.28. A plot showing the average number of infected people per day and number of infected people per day for random repetitions with probHome=0.5 and probImm=0.4.

In Figure 6.28, the thick black line shows the average number of infected people per day. It is understood from this plot that the disease has a tendency of dying out on average. In some repetitions, there are some fluctuating lines showing that there may occur small outbreaks in some repetitions. This is because of the stochastic nature of the simulation model and an expected result. In this case, R_0 -contribution coming from mixing in schools

is approximately 0.876 so, no outbreak is expected in schools and in the whole population (see Section 6.2.4 for details).

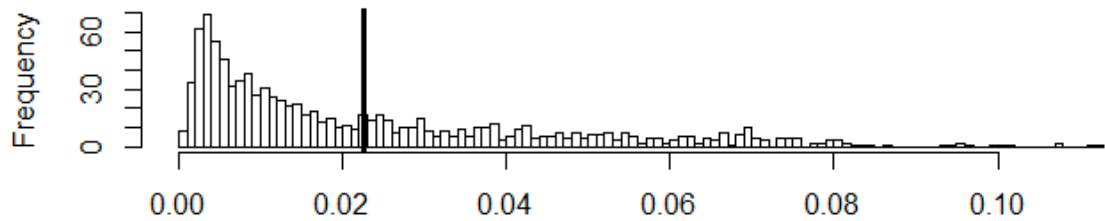


Figure 6.29. Histogram showing the attack rates of simulations with probHome=0.5 and probImm=0.4.

Mean for attack rate for this scenario is 0.0226 with 95% confidence interval [0.02125919, 0.02402738]. The proportion of independent repetitions where attack rate is less than or equal to 0.1 for this case is 0.994, so the disease has a high tendency in this case.

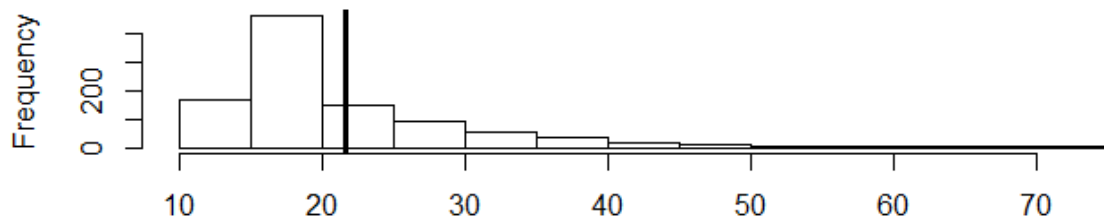


Figure 6.30. Histogram of maximum number of infected on a single day of each repetition.

As can be seen from the histogram above, the average maximum number of infected on a single day of each repetition is 21.6 with 95% confidence interval [21.03976, 22.07224].

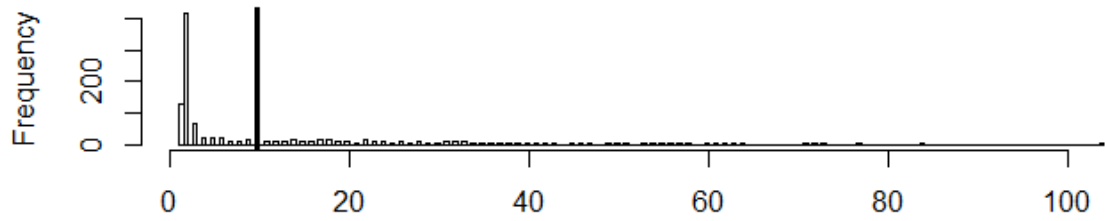


Figure 6.31. Histogram of disease peaking days.

Figure 6.31 represents the histogram of the day numbers (of each repetition) where the amount of infected people is the highest. The mean of these numbers is 9.73, with 95% confidence interval of [8.867213, 10.59879]. A large majority of all repetitions suggests that the maximum number of infected is seen in their beginning which means that the disease is inclined to die in most of the repetitions. As mentioned earlier in this section, the proportion of repetitions which have an attack rate less than or equal to 0.1 is 0.994 for this case and the above histogram verifies this statement.

7. CONCLUSION

The main aim of this thesis was to implement a flexible agent based (individual based) simulation of disease spread models in R mathematical programming language. This implementation is made to have a practical usage everywhere, especially in Turkey. A synthetic input population matrix carrying the characteristics of Turkish population structure is designed. The simulation program can work with any arbitrary population matrix, therefore it is flexible. The implementation is successful, tested and flows without any problems. It can be run in any personal notebook, its runtime is around 100 minutes for 1000 repetitions.

After the baseline model building, two scenarios are designed and simulated to predict the effectiveness of vaccination and staying home in case of infection. The results and their interpretations are shown in Part 6. It can be concluded from the results that both vaccination and staying home when infected decrease the attack rate and the average maximum number of infected on a single day significantly compared to the baseline model results.

One of the other achievements of this thesis is the function that is built to suggest a calibration procedure for infection probabilities using the R_0 -contribution coming from different mixing groups.

While analyzing the results of the experiments, an unexpected outbreak has been observed for a setting with an approximate R_0 value of 0.84. The implementation was checked again and it is sure that the outbreak, that was observed in the experiment, with an R_0 value less than 1 does not come from a mistake in the implementation. Then, the approximate R_0 value for different mixing levels of the population was calculated. It turned out that the R_0 -contribution of schools was greater than 1, even though the approximate calculated R_0 for the whole population was below 1. This shows that, interpreting the average R_0 value for non-homogeneous populations is difficult. Just finding the average approximate

R_0 value for non-homogeneous populations may not be enough. In other words, the R_0 value defined in the literature of agent based disease spread models with mixing groups does not lead to the expected behavior, meaning that the R_0 values are misleading and that calculating the average R_0 value does not fully describe the disease spread behavior.



REFERENCES

1. Brauer, F., C. Castillo-Chavez, and C. Castillo-Chavez, *Mathematical models in population biology and epidemiology*, Springer, 2001.
2. Trottier, H. and P. Philippe, "Deterministic modeling of infectious diseases: theory and methods", *The Internet Journal of Infectious Diseases*, Vol. 1, No. 2, pp. 3, 2001.
3. Longini, I.M., M.E. Halloran, A. Nizam and Y. Yang, "Containing pandemic influenza with antiviral agents", *American journal of epidemiology*, Vol. 159, No. 7, pp. 623-633, 2004.
4. Arifin, S.N., G.R. Madey, and F.H. Collins, *Spatial Agent-Based Simulation Modeling in Public Health: Design, Implementation, and Applications for Malaria Epidemiology*, John Wiley & Sons, 2016.
5. Chao, D.L., M.E. Halloran and I.M. Longini, "School Opening Dates Predict Pandemic Influenza A (H1N1) Outbreaks in the United States", *Journal of Infectious Diseases*, Vol. 202, No. 6, pp. 877-880, 2010.
6. Longini Jr, I.M. and J.S. Koopman, "Household and community transmission parameters from final distributions of infections in households", *Biometrics*, Vol.38, No.1, pp. 115-126, 1982.
7. Longini, I.M., A. Nizam, S. Xu, K. Ungchusak, W. Hanshaoworakul, D.A. Cummings and M.E. Halloran, "Containing pandemic influenza at the source", *Science*, Vol. 309, No. 5737, pp. 1083-1087, 2005.

8. Ajelli, M., B. Gonçalves, D. Balcan, V. Colizza, H. Hu, J.J. Ramasco, S. Merler and A. Vespignani, "Comparing large-scale computational approaches to epidemic modeling: agent-based versus structured metapopulation models", *BMC infectious diseases*, Vol. 10, No. 1, pp. 1, 2010.
9. Khalil, K.M., et al., *An agent-based modeling for pandemic influenza in Egypt*, in *Handbook on Decision Making*. 2012, Springer. p. 205-218.
10. Perez, L. and S. Dragicevic, "An agent-based approach for modeling dynamics of contagious disease spread", *International journal of health geographics*, Vol. 8, No. 1, pp. 1, 2009.
11. Milne, G.J., J.K. Kelso, H.A. Kelly, S.T. Huband and J. McVernon, "A small community model for the transmission of infectious diseases: comparison of school closure as an intervention in individual-based models of an influenza pandemic", *PloS one*, Vol. 3, No. 12, pp. e4005, 2008.
12. Mic, P., *Predicting case numbers in the epidemics with basic reproduction number and analysis of hospital resources sufficiency.*, MSc, Cukurova University, 2014.
13. Aydin, E., *The level of knowledge and activities of provincial health directorate managers in Turkey about control of the communicable diseases and the outbreaks data related to outbreaks.*, MSc, Ankara University, 2006.
14. Carhan, A., A.B. Altas, N. Albayrak , Y. Uyar, "2007-2008 Kış Sezonunda Türkiye'nin Dokuz İlinde Influenza Sürveyans Sonuçları", *Mikrobiyoloji Bulteni*, Vol. 43, No.2, pp. 235-241, 2009.
15. Çıkman, A., M. Berktas, M. Parlak, Y. Bayram and G. Yaman, "Evaluation of Influenza A (H1N1) Cases in Van Region ", *Nobel Med*, Vol.10, No.3, pp. 36-40, 2014.

16. Küçük, O., S. Bicer, T. Giray, D. Cöl, G. Ciler Erdağ, M. Uğras, Y. Gürol, A. Vitrinel, "Evaluation of the Patients with Influenza Viruses", *Çocuk Enfeksiyon Dergisi*, Vol. 7, pp. 87-91, 2013.
17. Sakarya, S.Ö., *The ratio of registered tuberculosis patients at Tuberculosis Dispensaries Notification System in İzmir and the factors affect on this ratio*, PhD, Ege University, 2008.
18. Ertem, M., F. Sevenscan, V. Dorman, N. Ozcullu, V. Ormanlı, N.K. Kubat, N. Albayrak and B. Altas, "Pandemic (H1N1) influenza in Diyarbakir, 2009", *Turkish Journal of Public Health*, Vol. 9, No. 2, 2011.
19. Kebabci, N., H. Akalın, G. Bölük, A. Oğuz-Ayarcı, E. Kazak, U. Topal, H. Yorulmaz, S. Akköse, T. Özvatan-Şener, E. Aslan, D. Köprücüoğlu, Y. Heper, E. Yılmaz, F. Kahveci, R. Mıstık, S. Helvacı, "Pandemik İnfluenza A (H1N1) 2009 Deneyimi", *Klimik Dergisi*, Vol. 25, No.3, pp. 117-121, 2012.
20. Central Dissemination System, Labour Force Statistics. <https://biruni.tuik.gov.tr/medas/?kn=72&locale=en>, accessed at May 2017.
21. Turkish Statistical Institute, Elderly Statistics, 2015. <http://www.turkstat.gov.tr/PreHaberBultenleri.do?id=21520>, accessed at May 2017.
22. Chowell, G. and F. Brauer, *The Basic Reproduction Number of Infectious Diseases: Computation and Estimation Using Compartmental Epidemic Models*, in *Mathematical and Statistical Estimation Approaches in Epidemiology*, Springer Netherlands: Dordrecht. p. 1-30, 2009.
23. Weber, D.J. and J.C. Thomas, *Epidemiologic Methods for the Study of Infectious Diseases*. Oxford University Press Inc, USA, 2001.

APPENDIX A: POPULATION GENERATING CODE

```

fsizeTable<- t(matrix(c(1,100,2,50,3,34,4,37,5,5,6,3,7,1),nrow=2))
colnames(fsizeTable)<-c("fsize","fquantity")
fsizeTable<-data.frame(fsizeTable)

makepdagneig <- function(fst=fsizeTable,IDneig=1,IDcomm=1){
  fst <- data.frame(fst)
  np <- sum(fst$fsize*fst$fquantity)
  m1 <- matrix(ncol=4,nrow=np)
  colnames(m1)<- c("IDfam","fsize","IDneig","IDcomm")
  m1 <- data.frame(m1)
  m1[,3]<- IDneig
  m1[,4]<- IDcomm
  nf1 <-fst[1,2]
  m1[1:nf1,2]<- 1 #all have size 1
  m1[1:nf1,1]<- 1:nf1
  nf2<-fst[2,2]
  m1[nf1+(1:(nf2*2)),2]<- 2
  m1[nf1+(1:(nf2*2)),1]<- rep(nf1+(1:nf2),each=2)
  nf<-numeric(7)

```

Figure A.1. Population generating code.

```

nf[1:2]<-c(nf1,nf2)
for(j in 3:7){
  j0 <- sum(fst$fquantity[1:(j-1)]*(1:(j-1)))
  nf[j]<- fst$fquantity[j]
  m1[j0+(1:(nf[j]*j)),2]<- j;
  m1[j0+(1:(nf[j]*j)),1]<- rep(cumsum(nf)[j-1]+(1:nf[j]),each=j)}
data.frame(m1)}

pdatcomm <- function(fst=fsizeTable,nnei=4,IDcomm=1){
IDneig=1:nnei
npop1neig <- sum(fst[,1]*fst[,2])
m2 <- matrix(nrow= nnei*npop1neig,ncol=4)
colnames(m2)<-c("IDfam","size","IDneig","IDcomm")
m2<--data.frame(m2)
for(i in IDneig){
  new <- makepdatneig(fst=fst,IDneig=IDneig[i],IDcomm=IDcomm)
  if(i>1){
    new$IDfam <- new$IDfam + max(m2$IDfam,na.rm=T)
  }
  m2[(1*npop1neig)+(i-1)*npop1neig,] <- new
}
m2
}

```

Figure A.1. Population generating code (cont.).

```

addAge<-
function(fst=fsizeTable,mat=res,nnei=4,ageDistrChild2=c(0.15,0.20,0.20,0.20,0.25),percSenior2=0.05)

{ fst      <- data.frame(fst)

  popnei    <- sum(fst$fsize*fst$fquantity)

  popcomm   <- popnei*nnei

  maxSen     <- (fst[1,2]*1+fst[2,2]*2)*nnei

  totAdSen   <- (fst[1,2]*1+(fst[2,2]+fst[3,2]+fst[4,2]+fst[5,2]+fst[6,2]+fst[7,2])*2)*nnei

  #same thing with maxAd.

  maxpercSen <- maxSen/totAdSen

  popChild=popcomm-totAdSen

  if(percSenior2>maxpercSen){

    warningMes <- function() warning("Percentage of seniors was too large so, it is set to ",
round(maxpercSen,digits=2))

    warningMes()

    percSenior2=maxpercSen

  }

  ageDistr = matrix(c(popChild*ageDistrChild2[1],
                      popChild*ageDistrChild2[2],
                      popChild*ageDistrChild2[3],
                      popChild*ageDistrChild2[4],
                      popChild*ageDistrChild2[5],
                      totAdSen*(1-percSenior2),
                      totAdSen*percSenior2))

```

Figure A.1. Population generating code (cont.).

```

adm <- t(data.frame(ageDistr))

#All families include at least 2 adults, one size families include only adults.

colnames(adm)<-c("baby","preschool","primary","middle","high","adult","senior")

mat$agegroup<-6 #set all to ADULT code


nf1 <-fst[1,2]
nf2 <-fst[2,2]

nf<-numeric(7)
nf[1:2]<-c(nf1,nf2)

#randomly sample from ages of WHOLE COMMUNITY
agec2<- rep(c(1,2,3,4,5),adm[1,c(1,2,3,4,5)])[sample(sum(adm[1,c(1,2,3,4,5)]))]

for(i in 1:nnei)
{
  for(j in 3:7)
  {
    nf[j]<- fst$fquantity[j]

    j0 <- sum(fst$fquantity[1:(j-1)]*(1:(j-1)))

    cInd <- popnei*(i-1)+j0+rep((1:nf[j])*j,j-2)-rep(0:(j-3),each=nf[j])

    mat[cInd,5]<- sample(agec2, length(cInd))

  }
}

mat[which(mat$size==1||mat$size==2),5]<-rep(c(6,7),c(totAdSen*(1-
percSenior2),totAdSen*percSenior2))[sample(length(which(mat$size==1||mat$size==2)))]

```

Figure A.1. Population generating code (cont.).

```

data.frame(mat)
}

schoolWorkF<-
function(fst=fsizeTable,nnei=4,mat=resAge,schoolTab2=matrix(c(1,0,2,18,3,45,4,37,5,31,
6,0,7,0),nrow=2),workInfo2=c(0.85,10)) {

  nAdult<-length(which(mat$Agegroup==6))

  workInfo2[1]<-nAdult*workInfo2[1]

  schoolTab2=data.frame(schoolTab2)

  mat$IDSchWork[mat$Agegroup==1]<- 9999 #set all babies(0-2 yrs) to No WORK NO
SCHOOL

  mat$IDSchWork[mat$Agegroup==7]<- 9999 #set all seniors to No WORK NO SCHOOL

  mat$IDSchWork[mat$Agegroup==6]<- 9999 #first set all adults to No WORK NO
SCHOOL

  mat$IDSchWork[which(mat$Agegroup==2)]<-
sample(c(2000+(1:schoolTab2[2,2])),length(which(mat$Agegroup==2)),replace=TRUE)

  mat$IDSchWork[which(mat$Agegroup==3)]<-
sample(c(3000+(1:schoolTab2[2,3])),length(which(mat$Agegroup==3)),replace=TRUE)

  mat$IDSchWork[which(mat$Agegroup==4)]<-
sample(c(4000+(1:schoolTab2[2,4])),length(which(mat$Agegroup==4)),replace=TRUE)

  mat$IDSchWork[which(mat$Agegroup==5)]<-
sample(c(5000+(1:schoolTab2[2,5])),length(which(mat$Agegroup==5)),replace=TRUE)

  mat$IDSchWork[sample(which(mat$Agegroup==6),workInfo2[1])]<-
sample(c(9000+(1:workInfo2[2])),workInfo2[1],replace=TRUE)

  data.frame(mat) }

```

Figure A.1. Population generating code (cont.).

```

popdat<-
function(fsizeTab=fsizeTable,nnei=20,percSenior=0.25,ageDistrChild=c(0.20,0.20,0.20,0.
20,0.20),schoolTab=matrix(c(1,0,2,18,3,45,4,37,5,31,6,0,7,0),nrow=2),workInfo=c(0.85,1
0),IDcomm=1){

  makepdtn eig()

  res1<-pdatcomm(fst=fsizeTab,nnei=nnei,IDcomm=IDcomm)

  res2<-
addAge(fst=fsizeTab,mat=res1,nnei=nnei,ageDistrChild2=ageDistrChild,percSenior2=perc
Senior)

  res3<-schoolWorkF(mat=res2,nnei=nnei,schoolTab2=schoolTab,workInfo2=workInfo)

  data.frame(res3)}

```

Figure A.1. Population generating code (cont.).