Erasmus University Rotterdam
Master Thesis Econometrics and Management Science
Operations Research and Quantitative Logistics

Calibrating parameters in the MISCAN model using a genetic algorithm

Author:
Lucie de Jonge
411585

Supervisor:
C.D. van Oosterom MSc
Second assessor:
Prof. dr. S. I. Birbil
External supervisors:
S.K. Naber PhD
I. Lansdorp-Vogelaar PhD

May 30, 2019

Abstract

Microsimulation disease models can be used to perform economic evaluations to inform policy makers on the long term costs and health outcomes of cancer screening and surveillance strategies. The Erasmus University Medical Center of Rotterdam, department of Public Health, developed the MISCAN model. Some parameters in the model should be estimated since they are not (directly) observable. These parameters are estimated through calibration, which is a method to obtain parameter estimates such that model outcomes fit the observed data. Currently, the Nelder-Mead simplex method is used, but that method does not perform well. Therefore, we proposed a new parameter search strategy called genetic algorithm. Using survival of the fittest and randomized changes, genetic algorithms try to find new solutions. Recombining two solutions (parent solutions) from a population and tweaking the parent solutions with certain probability, the aim is to create new and even better solutions (child solutions) in terms of fitness. We used the MISCAN model for esophageal cancer. Using different ways to initialize the algorithm and performing multiple calibration runs, we concluded that the proposed parameter search strategy seems to perform better in terms of running time and root mean squared prediction error for calibrating three parameters and therefore, it is a good alternative for the Nelder-Mead simplex method.

Contents

1 Introduction

The Erasmus University Medical Center of Rotterdam (EMC), department of Public Health, performs economic evaluations to inform policy makers on the long term costs and health outcomes of different interventions. These economic evaluations are performed by the EMC using mathematical disease models. EMC developed microsimulation models (MSM) called MIcrosimulation SCreening ANalysis (MISCAN) to evaluate cancer screening and surveillance strategies. MISCAN models are used for different projects across the world. In the United States of America (US), the Cancer Intervention and Surveillance Modeling Network (CISNET) uses MISCAN models to investigate different policies in cancer intervention and surveillance. Actually, the department of Public Health is part of CISNET which has six different groups (Colorectum, Esophagus, Breast, Lung, Prostate and Cervix), each consisting of at least three different research/modelling teams.

1.1 Microsimulation models

In MSM, the key element is that these models simulate a population on a individual-based level. Microsimulation can be used in highway traffic flowing, financial transactions or the development of a disease through a population, for example. The MISCAN model is a model to evaluate the screening and surveillance policy for different types of cancer. The model gives insights into the natural history of the disease. Also, it can give insights in the increase in incidence for specific cancer types. Lastly, it can be used to calculate the effectiveness and cost-effectiveness of a screening, surveillance and treatment policy.

The use of computer simulation models is increasing. There exist more models like the MISCAN model from EMC. For example, the Fred Hutchinson Cancer Research Center in Seattle, the Massachusetts General Hospital in Boston and the University of Minnesota developed such models. Each of these models differ in its modeling methodology (Kong et al., 2014). Currently, the MISCAN model has been developed not only for the evaluation of widely implemented screening programs for cervical, breast and colorectal cancer, but also for lung and esophageal cancer. MISCAN models are used to provide policy makers with a recommendation for a policy or to evaluate their current policy. In the Netherlands, projects are done for the National Institute for Public Health and the Environment for example.

MISCAN is a semi-Markov microsimulation model. A Markov model is a stochastic model where the states are discrete and the duration in these states are exponentially distributed. In particular, the Markov property holds, which means that the future state only depends on the current state and not on the states in the past. This means that the transitions between different states have a constant probability. MISCAN also uses discrete-event states and exponentially

distributed durations in these states. However, other types of durations and even dependence between durations are possible. The model simulates a population at individual levels. These individual levels are based on the fact that a person can evolve through different disease states, for example healthy (no abnormalities), preclinical cancer, and clinically (with symptoms) diagnosed cancer (Kong et al., 2014). The events in MISCAN, as defined by a Markov model, could be the birth and death of a person and transitions from one disease state to another. These events are simulated by the Monte Carlo method. Monte Carlo methods are computational algorithms that perform random sampling repeatedly.

1.2 Esophageal cancer

Throughout this thesis, the focus will be on early detection of esophageal cancer. Esophageal cancer is ranked eighth on the list of most common cancers in the world (Vizcaino et al., 2002). Looking at the mortality, it is ranked sixth (Kroep, 2015). The fatality rate, which can be defined as the rate of deaths within the group of people with esophageal cancer, is high. The 5-year survival rate is approximately 15%, which makes esophageal cancer a very lethal disease. Most of the esophageal cancers are squamous cell carcinoma (SCC) or esophageal adenocarcinoma (EAC). The difference between SCC and EAC is that EAC develops from the intestinal epithelium, also known as Barrett's esophagus (BE), and SCC from the squamous cells of the esophagus. The MISCAN model esophageal cancer focuses on EAC. BE is a state where the normal cells in the esophagus are replaced by intestinal metaplasia. It is assumed that BE can develop due to gastro-esophageal reflux disease (GERD). Also, heartburn and/or acid regurgitation, which are the symptoms of GERD, are linked to the development of BE and EAC. Before BE becomes EAC, BE develops from non-dysplastic BE to low-grade dysplasia, then to high-grade dysplasia and after that to EAC.

Another difference between SCC and EAC is that EAC occurs most in western countries like the United States of America and Europe, while SCC occurs most in countries in Asia and Africa (Kroep, 2015). In the United States, the incidence rate for EAC increases. For men, the incidence increased from 0.9 to 6.5 per 100,000 population from 1975 to 2009. For women, the increase was 0.2 to 0.9 per 100,000 population in the same period (Surveillance, Epidemiology, and End Results (SEER) program). Screening can be done since EAC can be detected early through endoscopic surveillance.

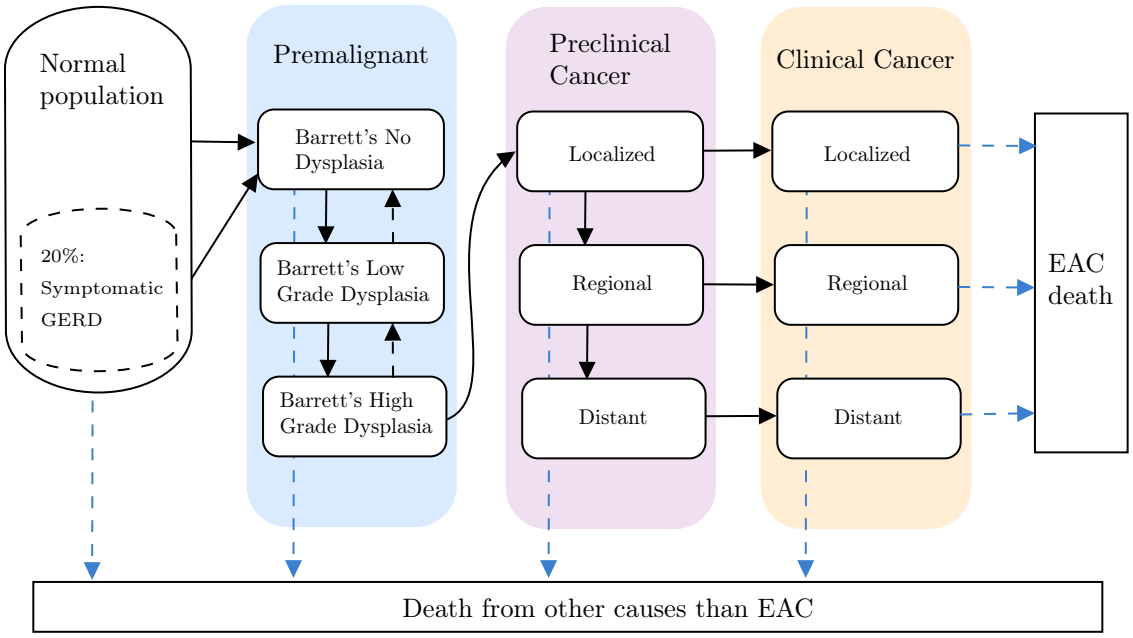

Figure 1: Graphical representation of Erasmus/UW-EAC model (Kroep et al., 2015)
GERD = gastroesophageal reflux disease, EAC = esophageal adenocarcinoma.

1.3 MISCAN EAC model

In cooperation with the University of Washington, the MISCAN model for esophageal cancer, also known as the Erasmus/UW-EAC model, was developed in 2011. Figure 1 shows a graphical representation of this model. In this figure, the different disease states are shown. The arrows denote which transitions are possible.

Like any MISCAN model, the Erasmus/UW-EAC model is structured in three components. First, we have the demography part. This part simulates the individual life histories in the absence of the disease. An individual life history are births and deaths of individuals not caused by EAC. Second, the model consists of a natural history component. The natural history component simulates the development of a cancer. It is assumed that twenty percent of the normal population has symptomatic (GERD). GERD can be the cause of BE, but BE could also occur without GERD. As can be seen in Figure 1, the development of EAC begins in BE without dysplasia, i.e. abnormal cell development. After that, it can develop to low or high grade dysplasia. From the high grade dysplasia, it can develop to preclinical localized EAC and following the arrows in the figure we can end up at clinical cancer and even death from EAC. Note that the difference between preclinical and clinical is that the symptoms are absent in the preclinical phase. Moreover, in the clinical phase, the cancer has been diagnosed and the patient can be treated. Last, we have the screening component. Here, the development of the specific cancer can be interrupted by screening. Screening can detect BE (blue part) and preclinical cancer. Figure 2 shows a graphical representation of the screening component for one hypothetical individual. The upper part describes the situation where

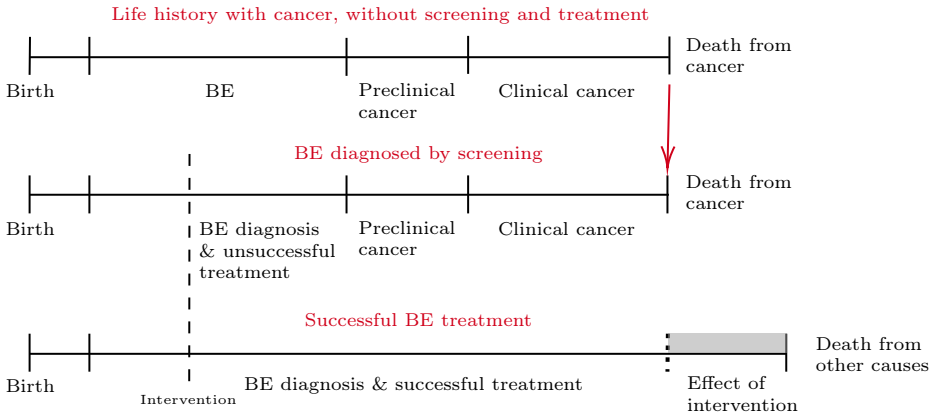

Figure 2: Graphical representation of screening and interventions (Kroep et al., 2015)
BE = Barrett's esophagus.

the individual did not get screening and unfortunately, this individual dies from EAC. The middle part describes the situation where BE was diagnosed but BE treatment was not successful, while in the bottom part BE treatment was successful and the person dies from other causes. BE treatment involves removing the BE to avoid development to cancer. An effect of screening is the difference in life-years between the simulated moment of death from cancer and moment of the death from other causes as can be seen in the grey part of 2. Moreover, an individual treated successfully for BE has benefit from the screening already before the simulated moment of death from cancer. The quality of life is already better after a successful treatment because this individual does not develop EAC

1.4 Calibration

For the model to be realistic, we need plausible parameter values. However, some parameters are neither directly observable nor can be calculated from existing data. An example of such a parameter is the mean duration of preclinical cancer since preclinical cancer does not present any symptoms as explained before. Therefore, performing a good parameter estimation is difficult (Van der Steen et al., 2016).

Such parameter estimation can be done by calibration. Calibration is a method to obtain parameter estimates such that the outcomes of a model using the parameter estimations fit the observed data. In this thesis, the observed data consists of the EAC incidence obtained from the Surveillance, Epidemiology, and End Results database (SEER) and the BE prevalence for the United States of America (US) determined by expert opinion.

Currently, MISCAN uses the Nelder-Mead simplex method to calibrate its parameters. Since calibration is a time-consuming task, it is not preferable to perform the calibration multiple times. If we want to calibrate a lot of parameters, more than 15, the current method does not perform

very well: it is not able to find the best solution. The method generates a solution, but we are able to improve on the solution manually because of local minima in which the method can get stuck. So, the aim of this thesis is to provide a new calibration method for MISCAN models, since in the existing literature, not much is written about specific calibration algorithms for microsimulation in disease models. Existing literature only provides an overview of a general calibration method as in Vanni et al. (2011) and a comparison of different goodness-of-fit criteria and parameter search methods is provided in Karnon and Vanni (2011). Another lack of the Nelder-Mead simplex method is that it provides just one solution instead of a set of solutions. This thesis proposes an algorithm which is able to calibrate multiple parameters and provides a set of solution. Genetic algorithms have these properties and therefore, the focus will be on genetic algorithms. We compare their performance with the performance of the Nelder-Mead simplex method to address this lack of literature. Moreover, after finding an alternative method for calibrating the parameters in MISCAN models, we can apply this to all the MISCAN models and, in the end, it may influence the current health policies.

The outline of the rest of this thesis is as follows. Section 2 provides more information on the data that is used. Section 3 gives give a detailed description of the problem. In Section 4, we show what kind of literature is already available regarding the problem. Then, in Section 5, we provide more detailed information on two different calibration algorithms already mentioned; the Nelder-Mead simplex method in Section 5.1 and genetic algorithms in Section 5.2. After that, we report the results in Section 6. Finally, we come up with a conclusion in Section 7

2 Data

As explained in the previous section, calibration is a method to match model outcomes with the observed data. This section gives an overview of the different types of data (observed and simulated) used in this thesis obtained from the SEER database and the MISCAN model. Also, some insights of the data are given in order to determine the number of individuals to be simulated in the calibration. First, we will explain the input and output of the MISCAN model and after that, we will explain hypothetical data sets.

2.1 Calibration targets and parameters

Table 1 gives an overview of the parameters that are calibrated in this thesis for the Erasmus/UW-EAC model with their interpretation. The Erasmus/UW-EAC model has more parameters which should be calibrated, but the parameters described in Table 1 describe a simplified version of the model and therefore, we chose these to be calibrated first in this thesis.

Parameter	Notation	Definition	Range
incB30	θ_{30}	Probability to develop BE at age 30	0-0.1
incB40	θ_{40}	Probability to develop BE at age 40	0-0.1
incB50	θ_{50}	Probability to develop BE at age 50	0-0.1
incB60	θ_{60}	Probability to develop BE at age 60	0-0.1
incB70	θ_{70}	Probability to develop BE at age 70	0-0.1
decrease7080	θ_{d80}	Probability to develop BE at age 80, as a percentage of the probability to develop BE at age 70	0-1
decrease8085	θ_{d85}	Probability to develop BE at age 85, as a percentage of the probability to develop BE at age	0-1
BARR	θ_{BARR}	Mean dwelling time of BE without dysplasia	1-40

Table 1: Overview of parameters calibrated for the Erasmus/UW-EAC model

The first seven parameters are used to simulate the onset of BE at different ages. The last two parameters of these are used to ensure a decrease in the probability to develop BE after age 70 which will be explained in Section 3. The last parameter in the table, 'BARR', is used for the mean dwelling time or duration for being in the premalignant state of Barrett's esophagus (BE) without dysplasia.

The EAC incidence by age from ages 0 to 100 is used as a calibration target and influenced by all the parameters in Table 1 while the BE prevalence is mostly related to the onset of BE at different ages. An upper bound for the BE prevalence is chosen by expert opinion. The BE prevalence obtained in the model should be below this upper bound. Note that the difference between incidence and prevalence is that incidence is defined over a period of time and the prevalence at a single moment in time. The EAC incidences defined by age have been updated according to the EAC incidence rates of the SEER database for men and women aged 20-84 in the US from 1975-2010 (Kroep, 2015). A new calibration is necessary in case the desired model outcomes are not corresponding to the data that was used for the previous model calibration. For example, a specific analysis considers a different period of time than the data that was used to calibrate the model parameters. The parameter estimates could therefore be inaccurate and should be re-estimated using the appropriate data.

For the probability to develop BE, it is required that the rate is increasing up to age 70 as has been found by Masclee et al. (2014). At age 70, there is a turning point. After this point, the probability should decrease (Kroep, 2015). We meet this requirement by multiplying the probability at age 70 by the multiplier for the age of 80. The probability to develop BE at age 85 is computed by multiplying the probability to develop BE at age 80 by the multiplier for age 85. In order to ensure a decrease, these multipliers are both numbers smaller than 1.

	Mean EAC incidence per age group (coefficient of variation)							
Sample size	0-19	20-24	25-29	30-34	35-39	40-44	45-49	50-54
10,000	0 (0)	0 (0)	0 (0)	1 (1)	1 (1)	2 (1)	4 (0.5)	7 (0.29)
100,000	0 (0)	0 (0)	1 (1)	6 (0.33)	13 (0.31)	22 (0.23)	40 (0.18)	65 (0.12)
1,000,000	0 (0)	1 (1)	13 (0.31)	57 (0.12)	126 (0.09)	221 (0.06)	401 (0.05)	649 (0.04)
10,000,000	0 (0)	6 (0.33)	134 (0.09)	567 (0.04)	1264 (0.03)	2211 (0.02)	4001 (0.02)	6481 (0.01)
100,000,000	0 (0)	64 (0.13)	1341 (0.03)	5655 (0.01)	12629 (0.009)	22120 (0.007)	40047 (0.005)	64819 (0.004)

Sample size	55-59	60-64	65-69	70-74	75-79	80-85	85-99	Total
10,000	7 (0.29)	15 (0.27)	23 (0.22)	27 (0.19)	25 (0.20)	18 (0.22)	16 (0.25)	146
100,000	64 (0.13)	139 (0.09)	214 (0.07)	251 (0.07)	238 (0.05)	175 (0.07)	156 (0.08)	1384
1,000,000	644 (0.04)	1393 (0.03)	2151 (0.02)	2539 (0.02)	2344 (0.02)	1790 (0.02)	1516 (0.02)	13855
10,000,000	6443 (0.01)	13930 (0.01)	21509 (0.006)	25108 (0.007)	23815 (0.006)	17468 (0.007)	15572 (0.008)	138510
100,000,000	64431 (0.004)	139258 (0.003)	215047 (0.002)	251058 (0.002)	238144 (0.002)	174761 (0.002)	155756 (0.002)	1385130

Table 2: 2000 simulation runs with different sample sizes

2.2 Hypothetical data set

In general, during the calibration of a model, the outcomes of the model are compared with the observed data in order to determine correctness of the parameter values. In order to evaluate our genetic algorithm, we should have data about the parameter values. We are not able to compare the estimated parameters with the underlying parameter values, because these underlying parameter values are unknown. Therefore, we simulate a hypothetical data set of observations with known, underlying parameters, such that we are able to compare them with the parameter estimates. The measurement to compare the results of two different parameter search strategies will be explained in Section 5.3. In order to determine the best number of individuals to be simulated, sample size, we give some insights in the calibration targets/model outcomes.

First, we give some insights into the EAC incidence per age group which is one calibration targets. Since MISCAN is a stochastic model, we are interested when the mean over multiple simulation runs would be stable for different sample sizes. So, we reported the mean cancer incidence at different ages for different sample sizes in Table 2. The larger the sample size the smaller the coefficient of variation for each age group as shown in the table.

We performed 2000 runs for five different sample sizes; 1e4, 1e5, 1e6, 1e7 and 1e8. Running the different sample sizes took approximately 48 seconds, 50 seconds, 1 minute and 3 seconds, 2 minutes and 54 seconds, and 21 minutes and 9 seconds, respectively.

Second, we have the BE prevalence as a calibration target for the parameters. We do not have evidence for the BE prevalence in the total population, since many people do not have any symptoms of BE and in order to diagnose BE, an endoscopy should be performed. The BE prevalence is estimated to be between 1.6% and 6.8% (Gilbert et al., 2011). The calibration target for the BE prevalence in the MISCAN model is defined as an upper bound. Since there is not much evidence for the BE prevalence, the upper bound for the BE prevalence was set to 100 percent and

Parameter	Notation	Value
BARR	θ_{BARR}	5.95439
incB30	θ_{30}	2.669 19e−5
incB40	θ_{40}	2.592 96e−5
incB50	θ_{50}	0.002 326 54
incB60	θ_{60}	0.000 480 822
incB70	θ_{70}	3.542 72e−5
decrease7080	θ_{d80}	0.239148
decrease8085	θ_{d85}	0.688884

Table 3: Overview of parameter values for the hypothetical data set

it was concluded that the simulated BE prevalence was not unreasonable using this upper bound. This means that the deviance for this calibration target is always zero. So, the deviance would not depend on this calibration target and therefore, we do not take the behaviour of this calibration target into account for deciding on the best sample size for the calibration.

Choosing the best sample size, there exists a trade-off between the running time and the accuracy of the model outcomes. The smaller the sample size the shorter the running time, but the smaller the number of people getting EAC and the variation is relatively large. Based on this, we decided to create a hypothetical data set by simulating a cohort of people using a sample size of 1,000,000 individuals using the vector of parameter values denoted in Table 3. After that, we create calibration targets based on the model outcomes obtained by the simulation for the hypothetical data set. Moreover, during the calibration, we used the same sample size in order to obtain the outcomes of the model.

3 Problem Description

This section provides a more detailed description of the problem. First, we will provide more information on calibration methods in general. We will give an overview of the structure of a calibration algorithm and how this is currently used in MISCAN models.

3.1 Structure of a calibration procedure

MSMs have several parameters as input. Some parameters can be obtained directly from the observed data, but some cannot. For a disease model such as MISCAN, the preclinical duration of the disease is not observable. For example, only data about the incidence of different cancers is available. Thus, a calibration algorithm tries to find parameter values such that the outcomes of the model fit the observed data. This means that we have a vector of observed data $\boldsymbol{\mu}$ and

that there is a model $M(\cdot)$ with a vector of K parameters $\boldsymbol{\theta}$ as an input, where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)$. Here, our model $M(\cdot)$ is the MISCAN model. So, the calibration algorithm tries to minimize the distance between the (random) model outcomes $M(\boldsymbol{\theta})$ and the observed data $\boldsymbol{\mu}$.

Below, we explain the structure of a calibration procedure according to some of the components discussed by Karnon and Vanni (2011):

1. Calibration targets
2. Goodness of fit criterion
3. Parameter search strategy
4. Convergence criteria
5. Stopping rule

Firstly, the calibration targets are the observed data. Examples for these calibration targets are a mean or a curve, but we use the observed data mentioned in Section 2.2.

Secondly, to be able to determine the distance between the calibration targets and the model outcomes, a calibration algorithm uses a Goodness-of-Fit (GOF) criterion. There are two types of GOF criteria, qualitative and quantitative. An example of a qualitative GOF is a visual representation of the model outcomes compared to the observed data. Through a figure or graph, we decide whether the model outcomes fit the observed data well. However, this is a very subjective GOF. Therefore, a quantitative GOF is used most of the time. Currently, the calibration for MISCAN models uses a likelihood-based GOF criterion with Poisson deviance in case of Poisson distributed data and with binomial deviance in case of binomial distributed data, since the calibration targets used for calibrating MISCAN models are either Poisson or binomial distributed (Van der Steen et al., 2016). An example of Poisson distributed data could be the number of interval cancers. Interval cancers are Poisson distributed since they can occur at every moment during the simulation. The number of screen-detected cancers is an example of binomial distributed data. A binomial distribution is the probability distribution of the number of successes. The success in the number of screen-detected cancers can be easily defined as when a cancer was detected by screening. So, there are two possible outcomes; either a cancer is detected by screening or it is not. The deviance can be interpreted as a distance measure for the observed outcome and the corresponding model outcome. The deviance $D_i(\cdot)$ for outcome i can be computed as follows (Van der Steen et al.,

2016).

$$\text{Poisson deviance}^{1,2}: \quad D_i(\boldsymbol{\theta}) = 2\left[\mu_i\left(ln\left(\frac{\mu_i}{M_i(\boldsymbol{\theta})}\right)\right) - (\mu_i - M_i(\boldsymbol{\theta}))\right], \tag{1}$$

$$\begin{aligned}\text{Binomial deviance}^{1,2}: \quad D_i(\boldsymbol{\theta}) = {} & 2\left[\mu_i\left(ln\left(\frac{\mu_i}{n_1}\right) - ln\left(\frac{M_i(\boldsymbol{\theta})}{n_2}\right)\right)\right] + \\ & 2\left[(n_1 - \mu_i)\left(ln\left(\frac{n_1 - \mu_i}{n_1}\right) - ln\left(\frac{n_2 - M_i(\boldsymbol{\theta})}{n_2}\right)\right)\right],\end{aligned} \tag{2}$$

where $M_i(\boldsymbol{\theta})$ equals the MISCAN model outcome i with the vector of estimated parameters $\boldsymbol{\theta}$ corresponding to the observed outcome μ_i, n_1 the number of persons used in the observed outcomes and n_2 the number of persons used in the model. Note that the Poisson deviance does not depend on the size of the data set or model outcomes. So, the observed data and the model outcomes should have the same size or should be scaled to the same size when using the Poisson deviance.

Other examples of GOF criteria are least squares or chi-squared (χ^2). Currently, the total GOF $D(\cdot)$ is computed as an unweighted sum of the GOF for each observed outcome $D_i(\cdot)$:

$$D(\boldsymbol{\theta}) = \sum_i D_i(\boldsymbol{\theta}). \tag{3}$$

A weighted sum could be used in case of different preferences in reaching the calibration targets. So, it could be that the model outcomes should fit a specific calibration target with higher importance than another calibration target.

Thirdly, we have the parameter search method or strategy. This part searches for the best parameter values such that the model outcomes fit the observed data. There are a lot of different methods in order to perform a calibration. The perfect parameter search strategy does not exist. However, the aim is to find the most appropriate method for the problem and compare these with other methods (Vanni et al., 2011). The current parameter search method will be discussed in Section 5.1 in more detail.

A fourth element of a calibration algorithm is the convergence criteria. These criteria determine whether a parameter set is acceptable. The convergence criteria could be defined as that the GOF criterion, the deviance in our case, for a parameter set should be below a certain value. In case of a set of parameters during the calibration algorithm, for example in our parameter search method, a convergence criterion could be that a parameter set is only accepted when the GOF is better or equal to the worst parameter set in the set of parameter sets.

In order to obtain results from the calibration, we should use a rule to stop the calibration, the stopping rule. Again, this stopping rule could be multiple things. We could use the convergence

[1] In case $M_i(\boldsymbol{\theta}) = 0$, we assumed $M_i(\boldsymbol{\theta}) = 0.5$ (Van der Steen et al., 2016)
[2] In case $\mu_i = 0$, then $\lim_{x \to 0} x \ln(x) = 0$ (Van der Steen et al., 2016)

criteria, but another example is a specified number of iterations. Another stopping rule could be that we want to have at least a predefined number, n, of solutions that are below a certain value as explained before in the part about the convergence criteria. At the end, we are able to find n solutions to the problem which satisfy being below that certain value.

3.2 Problem in MISCAN models

In MISCAN, local minima occur since there may exist multiple solutions for simulated individuals. For EAC, we only obtain the number of diagnosed cancers. We explain this more using a simplified example. Consider two individuals, a and b. Figure 3 shows two simulated individuals with a birth (left) and a moment of being diagnosed of EAC (right). The vertical lines between birth and the moment of EAC diagnosis represent the simulated onset of BE.

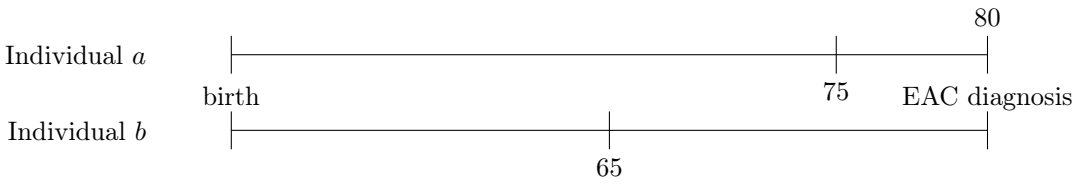

Figure 3: Examples of time line of cancer for individual a and b

In practice, the observed data only consists of the EAC diagnosis at age 80 in Figure 3. We do not know the exact probability by age of developing BE and the dwelling time of BE. As in Figure 3, individual a has a late BE onset and a short dwelling time and individual b has an early BE onset and a long dwelling time, which result in both cases to an EAC diagnosis at age 80. So, multiple combinations of the values for the parameters denoted in Table 1 exist such that the model outcomes fit the observed data.

As explained before, the problem with the Nelder-Mead simplex method is that the method can get stuck in a local optimum. Neddermeijer et al. (2000) mention that the Nelder-Mead simplex method is a local search method which means that it is not guaranteed that it is able to find global optima. In order to avoid getting stuck in local optima, the calibration should be done using different starting values. However, this is not a workable solution, since calibration is a time-consuming task.

Moreover, some parameters have to be calibrated first in the Nelder-Mead simplex method and after that, other parameters are calibrated but the consequence is that the model outcomes related to the first calibrated parameter could have been changed. So, some estimators for different parameters are correlated. The new calibration method should also take correlation into account. As explained before, the probability of developing BE should increase by age. So, we should make

sure that

$$\gamma_i = \frac{\theta_i}{\theta_{i-1}}, \tag{4}$$

where θ_i denotes the probability to develop BE at age $i \in \{30, 40, 50, 60, 70\}$ and is larger than 1. After the age of 70, the probability to develop BE should decrease. We estimate the multiplier to ensure this decrease and the probabilities can be computed by

$$\gamma_{80} = \theta_{\text{d80}} \times \theta_{70}, \qquad 0 < \theta_{\text{d80}} < 1 \tag{5}$$

$$\gamma_{85} = \theta_{\text{d85}} \times \theta_{80}, \qquad 0 < \theta_{\text{d85}} < 1 \tag{6}$$

where θ_{d80} and θ_{d85} denote the parameters to ensure that the probability to develop BE decreases after the age of 70. Therefore, both parameters are restricted to be smaller than 1.

In summary, the problem is that the current parameter search strategy, the Nelder-Mead simplex method, does not perform very well. It is a local search method and that means that it can get stuck in local optima. Moreover, the Nelder-Mead simplex method just gives one solution and not a set of solutions. Therefore, EMC, the department of Public Health, is interested in implementing a new parameter search strategy for the calibration of MISCAN models.

4 Literature Review

Up until today, not much literature has been written about calibration of parameters for microsimulation disease models. Van der Steen et al. (2016) compare different goodness-of-fit criteria such as sum of squared errors (SSE), Pearson chi-square and a likelihood-based approach. They compared the performance of each criterion based on the root mean squared prediction error (RMSPE), the computation time and the impact on the estimated cost-effectiveness ratios. They conclude that the likelihood-based approach results in the best estimation for the parameters in a microsimulation disease model. However, they also conclude that there does not exist a lot of literature about descriptions of calibration algorithms, while calibration is such an important part of modeling. Stout et al. (2009) obtained articles in a MEDLINE search about cancer-screening models. MEDLINE is a database consisting of journal citations and abstracts for biomedical literature around the world. They show that cancer simulation models are used more often than before, but, again, there is not much literature about the description of calibration methods.

An exception on this is written by Kong et al. (2009). First, they explain how to handle multiple calibration targets into the GOF, which resulted in a weighted sum of each GOF for each calibration

target. Weights were assigned in order to show the differences in the importance of the targets. Second, they compare simulated annealing (SA) with a genetic algorithm (GA) in the calibration of the Lung Cancer Policy Model (LCPM) from the Massachusetts General Hospital in Boston. They conclude that both parameter search strategies perform better than a grid search. The computation time of a grid search increases as the number of parameters is increasing. Therefore, both SA and GA are good alternatives. SA seems to perform faster than GA but not significantly. They performed two different calibration scenarios and GA did not perform significantly worse than SA.

As already mentioned in Section 3, Vanni et al. (2011) provide an overview of different goodness-of-fit criteria and parameter search strategies. According to seven steps, they explain calibration algorithms. They conclude that the lack of standards in calibrating disease models reduces the credibility of calibration methods and their aim is to increase this credibility. Besides the meta-heuristics GA and SA, other methods are used to calibrate MSMs. Rutter et al. (2009) introduce a Bayesian method. This method uses Markov chain Monte Carlo. It has several advantages over other methods for the calibration of MSMs. First, multiple parameters can be calibrated simultaneously over multiple data sources. Second, this method provides a confidence interval around the estimated parameter values at the end of the calibration. Last, they are able to include some information in the calibration about the parameters through the prior distribution. They applied the method on their MSM for colorectal cancer.

Besides the medical field of calibrating parameters, MSMs can be used for highway traffic flowing and financial transactions as mentioned in Section 1. Kim et al. (2005) introduce a new method for calibrating traffic MSMs, since it was said that previous methods were not modeling correctly the traffic conditions. They included a genetic algorithm into the calibration and used several ways in order to evaluate a parameter set, such as the Moses' distribution free rank-like, the Wilcoxon rank-sum test and the Kolmogorov-Smirnov test and the mean absolute error ratio (MAER), which defines a parameter set to be the 'best' when it has the lowest MAER. They conclude that simple measurements, such as the MAER, for correctness of the simulation results could result in wrong conclusions.

Cobos et al. (2016) introduced two methods based on nondominated sorting genetic algorithm II (NSGA-II) and SA because their aim was to develop a multi-objective algorithm for the calibration of microsimulation traffic flow models, as the existing algorithms are not capable of handling such problems. They argue that methods such as tabu search and SA are good methods, but those do not guarantee that a global optimum is found, which is also the case for the Nelder-Mead simplex method. Hollander and Liu (2008) argue that there is a lot of literature about the methodologies

for calibrating MSMs, but they do not outline general calibration principles. They present a review of different methods for calibrating traffic MSMs, and, in the end, they come up with a recommendation on how to perform a calibration. They suggest that the most commonly used tools should be used instead of trying to use different methods. Moreover, they suggest that these traffic MSMs are excellent to investigate the variation in MSMs. Because of the stochasticity in traffic MSMs, it is important to do more simulation runs and they think that most calibration algorithms do not use enough iterations. The output of one traffic MSM run would result in a too small sample from a distribution we do not know. However, we know that calibration is a time-consuming task and therefore, multiple calibration runs would take too much time and is not very efficient.

Concluding, SA and GA are mostly used and suggested for calibrating MSMs in different fields. Moreover, algorithms based on them are suggested, as in Cobos et al. (2016). SA and GA perform better than grid searches, but SA does not guarantee to find a global optimum, which is also the case for the Nelder-Mead simplex method, but GA actually guarantees to find a global optimum. Also, the Bayesian method introduced by Rutter et al. (2009) seems to be very suitable as well for MSMs.

5 Parameter Search Strategies

This section describes two different parameter search strategies. First, we will discuss the parameter search strategy that is currently used for calibrating MISCAN models and after that, we will introduce another parameter search strategy which will be the main focus of this thesis. Last, we will provide a method to compare different parameter search strategies.

5.1 Nelder-Mead simplex method

5.1.1 Algorithm

As mentioned in Section 1, the current parameter search strategy, the Nelder-Mead simplex method, does not perform very well. In this section, we elaborate more on this parameter search strategy. We consider a function $D(\cdot)$ describing the total likelihood-based GOF, where $D_i(\cdot)$ is either Poisson or binomial distributed as in Equation (1) and (2). The Nelder-Mead simplex method is a derivative-free optimization (DFO) method. It was originally designed for the optimization of deterministic multidimensional functions, but it has been applied to stochastic functions many times (Neddermeijer et al., 2000).

The Nelder-Mead simplex works as follows for a stochastic function in $\mathbb{R}^K$. In each iteration,

it considers $K+1$ points with a simplex constructed by their convex hull. The aim is to remove the vertex with the worst function value and to replace it with a better vertex in terms of function value. If a better point cannot be found, the simplex shrinks. At the end, the best vertex is left and a solution is found (Wright and Nocedal, 1999). Algorithm 1 gives a mathematical description of the Nelder-Mead simplex method provided by Neddermeijer et al. (2000).

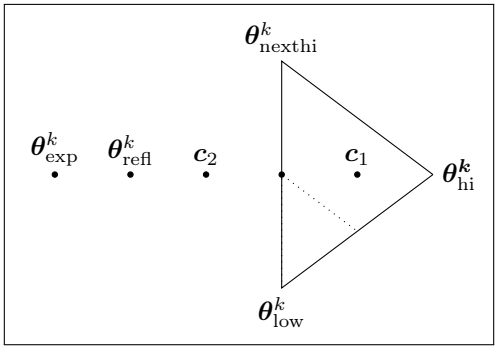

Figure 4: An iteration of the Nelder-Mead simplex method in $\mathbb{R}^2$

Figure 4 shows an example of the kth iteration for the Nelder-Mead simplex method in a two-dimensional parameter space ($\mathbb{R}^2$) based on a figure in Wright and Nocedal (1999). Consider three points in the kth iteration as solutions for the calibration, where we denote a point by $\boldsymbol{\theta}^k_{\text{low}}$ if it is the best solution among these three points in terms of GOF, by $\boldsymbol{\theta}^k_{\text{hi}}$ if it is the worst solution and by $\boldsymbol{\theta}^k_{\text{nexthi}}$ if it is the next worst solution after the worst solution. If it is not possible to obtain a better point than $\boldsymbol{\theta}^k_{\text{hi}}$, the simplex shrinks, as shown by the dotted line in Figure 4. If $\boldsymbol{\theta}^k_{\text{low}}$ is a local minimum, the algorithm is not capable of escaping from this local minimum anymore. This is done in the algorithm in lines 8 and 14 since the contracted points $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ were not better than or at least as good as $\boldsymbol{\theta}^k_{\text{hi}}$ and $\boldsymbol{\theta}^k_{\text{refl}}$. By the time we reach a local minimum, we would only be shrinking the simplex since the other points nearby are not better in terms of function values compared to the best solution so far.

Besides this description of the standard Nelder-Mead simplex method, Neddermeijer et al. (2000) provide us with several adaptive extensions in order to improve the method for stochastic simulation models like our MISCAN model. The main idea of the adaptive extension is that an additional criterion is implemented after each iteration. They propose four possible criteria. They introduce a criterion for the simplex size, a criterion whether the change in function value is sufficiently large, a criterion to check whether function values are sufficiently different and a criterion to retain the best solution so far and to re-evaluate the best vertex.

In the initialization of the algorithm, the algorithm asks for a starting point. This could be either a random starting point for example generated by taking a random number from the uniform distribution between the lower and upper bound provided by the range of the parameters or some

Algorithm 1 kth iteration of Nelder-Mead simplex method (Neddermeijer et al., 2000)

procedure NELDERMEAD($\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{K+1}\}, D(\cdot)$)
  $k \leftarrow 1$
  while stopping rule has not been met **do**
    Evaluate $\{D(\boldsymbol{\theta}_1^k), \ldots, D(\boldsymbol{\theta}_{n+1}^k)\}$
    $\boldsymbol{\theta}_{\text{low}}^k \leftarrow \min(\{D(\boldsymbol{\theta}_1^k), \ldots, D(\boldsymbol{\theta}_{n+1}^k)\})$
    $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \max(\{D(\boldsymbol{\theta}_1^k), \ldots, D(\boldsymbol{\theta}_{n+1}^k)\})$
    $\boldsymbol{\theta}_{\text{nexthi}}^k \leftarrow \max(\{D(\boldsymbol{\theta}_1^k), \ldots, D(\boldsymbol{\theta}_{n+1}^k)\} \backslash \{D(\boldsymbol{\theta}_{\text{hi}}^k)\})$
    Obtain $\boldsymbol{\theta}_{\text{cent}}^k$ as the centroid
    $\boldsymbol{\theta}_{\text{refl}}^k \leftarrow (1+\alpha)\boldsymbol{\theta}_{\text{cent}}^k - \alpha\boldsymbol{\theta}_{\text{hi}}^k$ ▷ Provided that $\alpha > 0$
    if $D(\boldsymbol{\theta}_{\text{refl}}^k) \geq D(\boldsymbol{\theta}_{\text{hi}}^k)$ **then**
      $\boldsymbol{\theta}_{c1}^k \leftarrow \beta\boldsymbol{\theta}_{\text{hi}}^k + (1-\beta)\boldsymbol{\theta}_{\text{cent}}^k$ ▷ Contracted vertex provided that $0 < \beta < 1$
      if $D(\boldsymbol{\theta}_{c1}^k) < D(\boldsymbol{\theta}_{\text{hi}}^k)$ **then**
        $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{c1}^k$
      else
        for $i = 1 : K+1$ **do**
          $\boldsymbol{\theta}_i^k \leftarrow \delta\boldsymbol{\theta}_i^k + (1-\delta)\boldsymbol{\theta}_{\text{low}}^k$ ▷ Provided that $\boldsymbol{\theta}_i^k \neq \boldsymbol{\theta}_{\text{low}}^k, 0 < \delta < 1$
        end for
      end if
    else if $D(\boldsymbol{\theta}_{\text{nexthi}}^k) < D(\boldsymbol{\theta}_{\text{refl}}^k) < D(\boldsymbol{\theta}_{\text{hi}}^k)$ **then**
      $\boldsymbol{\theta}_{c2}^k \leftarrow \beta\boldsymbol{\theta}_{\text{refl}}^k + (1-\beta)\boldsymbol{\theta}_{\text{cent}}^k$ ▷ Contracted vertex provided that $0 < \beta < 1$
      if $D(\boldsymbol{\theta}_{c2}^k) < D(\boldsymbol{\theta}_{\text{refl}}^k)$ **then**
        $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{c2}^k$
      else
        $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{\text{refl}}^k$
        for $i = 1 : K+1$ **do**
          $\boldsymbol{\theta}_i^k \leftarrow \delta\boldsymbol{\theta}_i^k + (1-\delta)\boldsymbol{\theta}_{\text{low}}^k$ ▷ Provided that $\boldsymbol{\theta}_i^k \neq \boldsymbol{\theta}_{\text{low}}^k, 0 < \delta < 1$
        end for
      end if
    else if $D(\boldsymbol{\theta}_{\text{low}}^k) < D(\boldsymbol{\theta}_{\text{refl}}^k) < D(\boldsymbol{\theta}_{\text{nexthi}}^k)$ **then**
      $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{\text{refl}}^k$
    else if $D(\boldsymbol{\theta}_{\text{refl}}^k) < D(\boldsymbol{\theta}_{\text{low}}^k)$ **then**
      $\boldsymbol{\theta}_{\text{exp}}^k \leftarrow \gamma\boldsymbol{\theta}_{\text{refl}}^k + (1-\gamma)\boldsymbol{\theta}_{\text{cent}}^k$ ▷ Expanded vertex provided that $\gamma > 1$
      if $D(\boldsymbol{\theta}_{\text{exp}}^k) < D(\boldsymbol{\theta}_{\text{low}}^k)$ **then**
        $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{\text{exp}}^k$
      else
        $\boldsymbol{\theta}_{\text{hi}}^k \leftarrow \boldsymbol{\theta}_{\text{refl}}^k$
      end if
    end if
    $k \leftarrow k+1$
  end while
end procedure

prior knowledge on the solution. Currently, prior knowledge is used in calibrating MISCAN models and this involves the solution of the previous calibration of MISCAN.

5.1.2 Stopping rule

As explained in Section 3, the stopping rule of a calibration algorithm can be several things. It can be a specific number of iterations, but it can also be based on the GOF criteria. Currently, for the calibration of MISCAN models, a stopping rule based on the progression within the simplex is used. To determine whether the progression is enough, Kendall's tau is used. After each iteration, it is determined whether the total deviance of the simplex, the progression, has made enough progression compared to the previous iteration. In other words, progression occurs when the total deviance in the simplex has decreased enough compared to the total deviance in the simplex of the previous iteration.

Kendall's tau is also known as Kendall rank correlation coefficient. This measure can be used to determine the ordinal association between two measured quantities. Using a tau-test, we can determine the statistical dependence of these two measured quantities.

Using an example of Kendall (1938), we explain Kendall's tau. Consider a set

$$4 \quad 7 \quad 2 \quad 10 \quad 3 \quad 6 \quad 8 \quad 1 \quad 5 \quad 9$$

where they should be ordered from 1 up to 10. When a pair is increasingly ordered, it will get a score of +1 and -1 otherwise. So, pair, 4 7, gets a score of +1 while a pair of 4, 2 gets a score of -1. We repeat this for every number. So, first, we create nine pairs with 4, after that with eight pairs with 7, seven pairs with 2 and so on. At the end, we obtain the following nine scores,

$$+3, \quad -2, \quad +5, \quad -6, \quad +3, \quad 0, \quad -1, \quad +2, \quad +1.$$

The sum of the nine scores is +5. The maximum score equals 45. In general, the maximum score can be calculated by $\frac{n(n-1)}{2}$ for n numbers increasingly ordered. So, for $n = 10$, $\frac{10(10-1)}{2} = \frac{10\times 9}{2} = \frac{90}{2} = 45$. Kendall's tau ($\tau$) can be calculated according to the following formula.

$$\tau = \frac{\text{actual score}}{\text{maximum possible score}} \tag{7}$$

Under the null hypothesis, τ has a mean of zero, and variance equal to $\frac{2(2n+5)}{9n(n-1)}$, which means that

there is no significant difference. With the mean and variance, we can compute the z-statistic.

$$z = \frac{\tau}{\sqrt{\frac{2(2n+5)}{9n(n-1)}}}$$

Using $\alpha = 0.05$, we reject the null hypothesis when $|z| > 1.96$.

The null hypothesis for the Nelder-Mead simplex method is defined as that there does not exist enough progression between two iterations. In general, the progression can also be evaluated every k iterations, where $k \geq 1$. So, the algorithm continues until the null hypothesis cannot be rejected anymore.

5.2 Genetic algorithm

In this thesis, the main focus will be on genetic algorithms. A genetic algorithm is a heuristic and especially, a meta-heuristic. While most heuristics get stuck at local optima and provide a limited number of solutions, meta-heuristics were introduced in order to be able to escape from local optima and provide a population of solutions at the end of the algorithm, which will be explained later in this section. A benefit compared to other meta-heuristics like SA or tabu search is that genetic algorithms look at an entire population of solutions instead of just considering one single solution. Genetic algorithms search through the whole neighbourhood in the population.

Genetic algorithms are based on the natural selection mechanism in biology. They use survival of the fittest and randomized changes to the solutions to obtain new solutions. The algorithm consists of five key elements:

- Population
- Fitness function
- Selection procedure
- Reproduction mechanism
- Mutation operation

The population consists of solutions that have been evaluated through the fitness function which measures their quality. In order to create new solutions, we should select two parent solutions. With these two parent solutions, we apply the reproduction mechanism which is called a cross-over operation. The reproduction mechanism provides two new solutions which are called the child solutions or offsprings. After that, with small probability, small changes could be applied to the child solutions using a mutation operation. In general, there are five steps to be taken during an iteration of a genetic algorithm:

- **Step 0:** Initialization of the population

  Form the initial population using the ranges of the parameters
- **Step 1:** Selection of parents

  Select a subset of parents according to a selection method.
- **Step 2:** Reproduction

  Form offsprings by applying cross-over.

  Apply mutation to the offsprings.
- **Step 3:** Generate population

  Add the acceptable offsprings to the population
- **Step 4:** Stopping criterion

  If a stopping criterion is met, STOP. Otherwise, go to Step 1.

One requirement for this algorithm is that the solution should be represented by a chromosome. This means that the solution should be a string of values. Binary encoding is the most commonly used way of representing a solution. However, our parameters are real-valued and cannot be represented by binary encoding. So, we represent the solution by its real values. According to Davis (1991), representing the solution by binary encoding is unnatural and unnecessary.

In order to explain the rest of the methodology, we should introduce some notation. A parent solution t consisting of K parameters to be calibrated can be represented by

$$\boldsymbol{\theta}_t = [\theta_{t1}, \ldots, \theta_{tK}], \tag{8}$$

where each parameter θ_{tj} is real-valued.

5.2.1 Initialization

First, we should initialize the algorithm. Initialization involves creating the first population of solutions. The initialization of the population can be done in two ways; either with or without prior knowledge on the parameter values. The population size, N, should be chosen carefully. Intuitively, the larger the complexity of the problem, which means the number of parameters considered, the bigger the population size should be. However, Ahn and Ramakrishna (2002) state that recent literature shows that it is not required to take the complexity of the problem into account in order to determine the population size. Increasing the population size has as downsides that it uses more memory and it takes more computation time. So, we try different population sizes and analyze the possible differences in the results. To create the initial population, we opt

for a range of each parameter and with these ranges, we generate the N sets of parameter values using a uniform distribution.

5.2.2 Fitness function

After initializing the population, we define our fitness function. The fitness function measures the quality of a solution and the higher the fitness the better the solution. The objective of a genetic algorithm is maximizing the fitness function value while the objective of a calibration is minimizing the GOF, the deviance. So, the fitness function value should increase as the deviance is decreasing. In other words, the lower the deviance, the higher the fitness function value. Therefore, the fitness function $F(\cdot)$ for a parent solution t is defined by

$$F(\boldsymbol{\theta_t}) = \frac{1}{D(\boldsymbol{\theta_t})} \tag{9}$$

where $D(\cdot)$ is the total GOF as defined in Equation (3) in Section 3.

5.2.3 Selection procedure

Next, we introduce selection procedure. A selection procedure selects a subset of the total population. Here, a subset consists of n parent solutions, where n is even and $n < N$, and these n parent solutions are used in the next step, the reproduction procedure. There are several ways to choose our parent solutions for reproduction. We consider the three most commonly known methods. We implement all three methods and we compare them at the end. The three different methods are

- Elitism
- Roulette Wheel
- Tournament Selection

First of all, there is a method called 'Elitism'. Here, the n fittest parent solutions are selected. In other words, we select the n parent solutions with the highest value for the fitness function.

Second, we have 'Roulette wheel' based on Lipowski and Lipowska (2012). Here, we select n parent solutions randomly with each parent solution s having a selection probability defined as the fitness of parent solution s proportional to the total fitness in the population. So, each time a parent solution has been selected, the parent solution is removed from the population of parent solutions considered through the selection procedure. This means that the total fitness in the population reduces during the selection procedure and the selection probability of each parent solution remaining in the population increases.

The higher the fitness of a parent solution, the better the fit and the higher the selection probability.

The last method is 'Tournament Selection'. Here, we randomly select z parent solutions, $z \leq N$, not restricted by n, without replacement and among these z individuals, we select one 'winner' using a tournament. In order to obtain n parent solutions, we should perform n tournaments. The winner of the tournament is the parent solution with the highest fitness function (Miller and Goldberg, 1995). The winner is inserted in a pool of all tournament winners. After n tournaments, we have our set of parent solutions.

5.2.4 Reproduction mechanism

In the selection procedure, we obtain our subset of n parent solutions. We divide the n selected parents into $\frac{n}{2}$ pairs of parents. These $\frac{n}{2}$ pairs of parents can each create two child solutions which will result in n child solutions in total. Most genetic algorithms use one-point crossover. This means that a pair of parent solutions is split and recombined at a crossover point. Let us explain this with an example.

Consider a pair of parent solutions $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ with four parameters ($K = 4$) to be calibrated as denoted in (10) and (11)

$$\boldsymbol{\theta}_1 = [\theta_{11}, \theta_{12}, \theta_{13}, \theta_{14}] \tag{10}$$

$$\boldsymbol{\theta}_2 = [\theta_{21}, \theta_{22}, \theta_{23}, \theta_{24}] \tag{11}$$

Parent solution $\boldsymbol{\theta}_t$ consists of four parameters, $\theta_{t1}, \theta_{t2}, \theta_{t3}$ and θ_{t4} for $t = 1, 2$. Assume that the one-point crossover occurs at point 3. So, after the third parameter, the parameter values are exchanged. This means that the child solutions $\boldsymbol{\theta}_{c_1}$ and $\boldsymbol{\theta}_{c_2}$ can be constructed as

$$\boldsymbol{\theta}_{c_1} = [\theta_{11}, \theta_{12}, \theta_{13}, \theta_{24}], \tag{12}$$

$$\boldsymbol{\theta}_{c_2} = [\theta_{21}, \theta_{22}, \theta_{23}, \theta_{14}]. \tag{13}$$

So, the result is that the last parameters, θ_{14} and θ_{24}, are switched. Now that we have the child solutions, we should determine whether the child solutions get affected by a mutation.

5.2.5 Mutation operation

The mutation operation is based on the method described by Wright (1991). First of all, we should determine whether a child solution is selected for mutation. In biology, it is not always the case

that a mutation happens. So, let p_M be the the probability that the given child solution is selected to be mutated, also known as the mutation rate. We should choose the value of the mutation rate carefully. If we choose the mutation rate to be too low, we obtain a population with too little variation, while choosing the mutation rate to be too high, we obtain a population with too much variation.

After knowing that the given parameter should be mutated, we should determine in which direction we should mutate. Each direction, below or above the parameter value, has a probability of $\frac{1}{2}$ of being chosen.

We should limit the size a parameter θ_{tj} for parent solution t could mutate. Let us say that M is the maximum mutation size. This means that we are allowed to change the parameter value to at most $(1 - M) \times \theta_{tj}$ if it is determined that we are going to decrease the value, and to at most $(1 + M) \times \theta_{tj}$ if it is determined that we are going to increase the value. In Table 1, the ranges for each parameter are given. We should be careful that the parameter should not get values outside these ranges. In order to do so, we add at most M% of the difference between the parameter value and its bound.

5.2.6 Update population

At the end of an iteration, the population should be updated. The child solutions are evaluated using the fitness function. If a child solution has a higher fitness than the parent solution with the lowest fitness currently in the population, it is added to the population and the other parent solution is removed. If the child solution is not better than any parent solution in terms of fitness, we reject the child solution.

5.2.7 Complete calibration algorithm

Algorithm 2 gives a detailed description of the calibration using the genetic algorithm as a parameter search strategy. The three selection methods mentioned in Section 5.2.3 are denoted as SelecProc() in Algorithm 2 in line 4. Since 'Elitism' is straightforward, only the pseudocode of 'Roulette wheel' and 'Tournament Selection' can be found in Algorithms 3, 4. Algorithm 2 requires an input vector, $\boldsymbol{R}$, of ranges of the parameters to be calibrated, the population size N, the mutation parameter p_M, the maximum mutation size $M\%$ and the fitness function $F(\cdot)$ as defined by Equation (9). As explained before, the parameter values cannot exceed their bounds as defined in $\boldsymbol{R}$ in the mutation operation. Therefore, the parameter value can mutate at most $M\%$ of the difference between the parameter value and its bounds as can be seen in lines 12 and 14. The calibration ends when the stopping rule is satisfied stated in line 3. At the end of the calibration algorithm, we obtain a population of parameter sets with a 'reasonable' fit. One single parameter

Algorithm 2 Calibration algorithm with a genetic algorithm

```
1: procedure GENETIC(R, N, p_M, M, F(·))
2:     P ← InitializePop(R, N)                          ▷ Generate initial population with size N
3:     while Stopping rule has not been met do
4:         S ← SelectProc(P, n)                         ▷ Returns n parent solutions for crossover
5:         C ← CrossOver(S)                             ▷ Returns n child solutions
6:         for t = 1 : n do
7:             p_t ← U(0, 1)
8:             for j = 1 : K do
9:                 if p_t ≤ p_M then                    ▷ Mutation operation
10:                    p_j ← U(−M, M)
11:                    if p_j ≥ 0 then
12:                        C(t, j) ← C(t, j) + (R(j, 'UB') − C(t, j)) · p_j      ▷ Increasing the value
13:                    else
14:                        C(t, j) ← C(i, j) + (C(t, j) − R(j, 'LB')) · p_j      ▷ Decreasing the value
15:                    end if
16:                end if
17:            end for
18:        end for
19:        for each child solution c do                 ▷ Update population
20:            Sort P in increasing order in the fitness
21:            if F(C(c)) > F(P(1)) then
22:                P ← P\{P(1)}
23:                P ← P ∪ {C(c)}
24:            end if
25:        end for
26:    end while
27: end procedure
```

set can be used for further analysis in MISCAN, for example the 'best' parameter set, and the population of parameter sets can be used for a (probabilistic) sensitivity analysis.

Algorithm 3 show the pseudocode for the selection method 'Roulette wheel'. This selection method selects n parent solutions with a certain selection probability as explained in Section 5.2.3. The implementation is as follows. First, we compute the total fitness, totalFIT, in the population as in line 5. Next, we draw n random numbers, r, between 0 and the total fitness shown in line 8. The for-loop in line 7 makes sure that there will be n numbers. Iterating through the population of parent solutions, we keep adding up the fitness, partFIT, until we exceed the random number r. If we do, we add the last considered parent in the subset of parent solutions, R, that are selected in line 13. We continue until n parent solutions are selected for the reproduction procedure.

Algorithm 3 shows the pseudocode for the selection method 'Tournament Selection'. As explained before, this selection method selects n parent solutions for the reproduction procedure by doing so-called 'tournaments'. In order to obtain n parent solutions, we should perform n tournaments as is stated in line 4 and in every tournament, we randomly select a subset B from the

Algorithm 3 Algorithm for the selection method 'Roulette wheel'

```
1: procedure RouletteWheel(P, n)
2:     totalFIT ← 0
3:     R ← ∅
4:     for each parent solution t do
5:         totalFIT ← totalFIT + F(t)          ▷ Compute the total fitness in population
6:     end for
7:     for t = 1 : n do
8:         r ← U(0, 1) × totalFIT
9:         partFIT ← 0
10:        for each parent solution t do
11:            partFIT ← partFIT + F(t)
12:            if partFIT ≥ r then
13:                R ← R ∪ {t}
14:                P ← P\{t}
15:            end if
16:        end for
17:    end for
18:    return R
19: end procedure
```

population consisting of z parent solutions as can be seen in line 5. We evaluate these parent solutions in their fitness in line 7 and the parent solutions b with the highest fitness in subset B is selected for the reproduction procedure. We continue until n parent solutions are selected.

Algorithm 4 Algorithm for the selection method 'Tournament Selection'

```
1: procedure Tournament(P, n)
2:     T ← ∅
3:     B ← ∅
4:     for t = 1 : n do
5:         B ⊆ P                               ▷ Randomly select a subset B from P
6:         P ← P\B
7:         f_B ← F(B)
8:         b is parent solution with highest fitness in f_B
9:         T ← T ∪ b
10:    end for
11:    return T
12: end procedure
```

Algorithm 5 shows the pseudocode for the crossover procedure described in Section 5.2.4. In the crossover procedure, two parent solutions are recombined at the crossover point in order to create child solutions. The crossover point is computed by taking a random number between 1 and the number of parameters - 1, l, als can be seen in lines 2 and 3. Lines 5 till 12 show the pseudocode for creating the child solutions. It is assumed that the set S is constructed such that two consecutive parent solutions form a pair and create a child solution.

Algorithm 5 Algorithm for the reproduction procedure 'One-point crossover'

procedure CROSSOVER(S)
  $l \leftarrow K - 1$
  CrossPoint $\leftarrow$ Uniform$(1, l)$
  for each parent solution t in S **do**
    for $j = 1 :$ CrossPoint **do**
      $C(i, j) \leftarrow S(i, j)$
      $C(i + 1, j) \leftarrow S(i + 1, j)$
    end for
    for $j =$ CrossPoint $+ 1 : l$ **do**
      $C(i, j) \leftarrow S(i + 1, j)$
      $C(i + 1, j) \leftarrow S(i, j)$
    end for
  end for
  return C
end procedure

5.3 Performance of parameter search strategies

As explained in Section 1.4, we want to compare our results with the current parameter search method, the Nelder-Mead simplex method, in order to determine whether a genetic algorithm preferred over the Nelder-Mead simplex method as the parameter search strategy in calibration MISCAN models. We compare our results in terms of running time and through the relative root mean squared prediction error (RMSPE) (Van der Steen et al., 2016). We use the relative RMSPE rather than the absolute RMSPE since expressing the deviation in percentages makes sure that the different deviations between the parameter estimate and the underlying parameter value have a comparable impact on the RSMPE. The relative RMSPE can be defined as follows.

$$\text{RMSPE} = \sqrt{\sum_{w=1}^{m} \frac{\sqrt{\sum_j \left(\frac{\hat{\theta}_j^w - \theta_j}{\theta_j}\right)^2}}{m}}, \tag{14}$$

where $\hat{\theta}_j^w$ denotes parameter estimate j in the wth calibration and θ denotes the known, underlying parameter value corresponding to the parameter estimate j. For each parameter search strategy, we should perform m calibration runs to compute the RMSPE. For the Nelder-Mead simplex method, the solution with the lowest deviance is left at the end and is used and for the genetic algorithm, we use the parameter set with the lowest deviance in the population of each calibration.

In order to determine whether the results of the calibration runs between both the different parameter search strategies as the different genetic algorithms are stochastically different, we use a Wilcoxon signed-rank test at a 5% significance level. In generally, this nonparameteric test can be used to determine whether there exists a significant stochastic different between two samples.

Applied to our results, we compare two sets of m prediction errors, $\sqrt{\sum_j \left(\frac{\hat{\theta}_j^w - \theta_j}{\theta_j}\right)^2}$, where each set is obtained by a different parameter search strategy.

6 Results

This section describes the results of implementing the methodology described in the previous section using R. We show the results in three steps. First, we applied the algorithm without any prior knowledge on the solution. Next, we introduced some prior knowledge into the population since currently the parameter search strategy contains some prior knowledge about the solution. Last, in order to evaluate the performance of the genetic algorithm and compare it with the Nelder-Mead simplex method, we performed multiple calibration runs using the same initial population in the genetic algorithm and the same starting point in the Nelder-Mead simplex method.

Since the running time for a single MISCAN simulation takes around 1 minute and 35 seconds with a sample size of 1,000,000, performing multiple times MISCAN simulations makes the running time of the calibration long compared to the time to perform the other operations, which is negligible. In particular, MISCAN is used in the fitness function. So, computing the fitness takes the most time. However, using genetic algorithms, the fitness function of multiple parent solutions can be computed in parallel.

The mutation rate p_M is set to 0.1 and the maximum mutation size M is set to 2.5%. Moreover, there was not much literature about specific numbers for the population and selection size. Therefore, we chose to try three different population and selection sizes. As explained in Section 5.2.1, increasing the population size results in more used memory space and more computation time. Therefore, we decided to use population sizes 30, 40 and 50, and selection sizes 8, 12, 20. For 'Tournament Selection', we select random parent solutions to do a 'tournament'. We select three parent solution ($z = 3$) for each 'tournament'.

6.1 Without starting values

First, we implemented the methodology of Section 5.2 and calibrated three parameters. Initially, the parameters that we calibrated were 'BARR', 'decrease7080' and 'decrease8085' as described in Table 1. Using the stopping rule as defined in Section 5.1.2, the Nelder-Mead simplex method is currently able to perform a calibration for 18 parameters resulting in a deviance of 74. In order to test our algorithm, we first defined the stopping rule such that all the parent solutions in the population should have a deviance of at most 100 or we limited the number of generations that can be performed by 1000.

Tables 7, 8 and 9 in Appendix A.1 show the result for just one single calibration run for three parameters. It is interesting that 'Roulette wheel', seems to work best in terms of running time. Moreover, 'Roulette wheel' is the only selection method that was able to reach a deviance of at most 100 in the population for all different combinations of population and selection size. The other two methods does not seem to perform different from each other. A remarkable thing is that the selection method 'Tournament Selection' was not able to reach a deviance below 100 at all in the population using a population size of 30 and a selection size of 8.

After that, we applied the algorithm for calibrating five and eight parameters. We add the parameters 'IncB30' and 'IncB40' for calibrating five parameters and 'IncB50', 'IncB60' and 'IncB70' for calibrating eight parameters. Unfortunately, after running the algorithm for a couple of days, the algorithm was not able to result in a solution.

6.2 With starting values

As explained in Section 5.1, a starting value for the parameters is used as an input for the Nelder-Mead simplex method. So, as a second step, we inserted starting values into the initial population. So, we added these starting values as parameter values into the population. As starting values, we used the starting values used for the current parameter search strategy for which the parameter values shown in Table 3 were obtained. The vector of parameters for the starting values is denoted in Table 4

Parameter	Notation	Value
BARR	θ_{BARR}	5.8388
incB30	θ_{30}	2.518e−5
incB40	θ_{40}	2.399 41e−5
incB50	θ_{50}	0.000 223 227
incB60	θ_{60}	0.000 471 659
incB70	θ_{70}	3.987 47e−5
decrease7080	θ_{d80}	0.450628
decrease8085	θ_{d85}	0.408561

Table 4: Overview of parameter values for the starting parameter set

We still used the stopping rule as explained in the previous subsection. It is not surprising that the results in terms of deviance were much lower and much faster than before compared to the results of Section 6.1, and we know that the algorithm is working.

6.2.1 Three parameters

First, our aim was to calibrate the models with three parameters to be calibrated, the same parameters as in Section 6.1. Again, we used the three selection methods, 'Elitism', 'Roulette wheel' and 'Tournament Selection'. Table 10, 11 and 12 in Appendix A show the results for three different population sizes and selection sizes. Within the three selection methods, we do not observe a clear difference in the results, but they all perform good. The lowest deviance, 11.10, is obtained using selection method 'Tournament Selection' for population size 40 and selection size 8.

6.2.2 Five parameters

Next, we tried to involve five parameters to calibrate. Besides 'BARR', 'decrease7080' and 'decrease8085', we included also 'incB30' and 'incB40'. As mentioned in Section 2.1, the probabilities to develop BE specified for several ages should increase by age. The results can be found in the Appendix in Tables 13, 14 and 15.

Within the three selection methods, we do not observe a clear difference in the results. As mentioned earlier, no results were obtained without prior knowledge, so this results show us that the algorithm works. The lowest deviance, 13.74, is obtained using selection method 'Roulette wheel' for population size 30 and selection size 12.

6.2.3 Eight parameters

Next, we tried to involve eight parameters to calibrate. Besides 'BARR', 'decrease7080' and 'decrease8085', 'incB30' and 'incB40', we included also 'incB50', 'incB60', 'incB70'. Tables 16, 17 and 18 in Appendix A.2.3 show the results for three different population and selection sizes.

Selection methods 'Roulette wheel' and 'Tournament Selection' seem to perform well without clear difference, while selection method 'Elitism' seems to take much more time for the calibration. However, 'Tournament Selection' has a longer running time compared to 'Elitism' for population size 50 and selection sizes 12 and 20. Again, using selection method 'Tournament Selection' resulted in the lowest deviance, 13.78, for population size 50 and selection size 12.

6.3 Multiple calibration runs

In order to compare the results from the genetic algorithm with the results from the Nelder-Mead simplex method, we perform multiple calibration runs for both the Nelder-Mead simplex method as for different genetic algorithms. Concluding from Section 6.2.3, we excluded 'Elitism' from this part of the results.

6.3.1 Nelder-Mead simplex method

In order to make a fair comparison with the different genetic algorithms for calibrating three parameters, we performed 20 calibration runs for three parameters using the Nelder-Mead simplex method using the same random starting point for each calibration run. The random starting point was $[29.0239; 0.3065; 0.0977]$ for the parameters [BARR; decrease7080; decrease8085]. As in Algorithm 1, the algorithm should have as input the initial simplex denoted by $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{K+1}\}$. The package 'neldermead' in R perturbs around this starting point in order to initialize the simplex. Other ways to initialize the simplex using this package are also possible.

The Nelder-Mead simplex method is able to find parameters such that the deviance is on average 15.72 which seems to be good. The RMSPE of the 20 calibration runs equals 1.5071. On average, it took the Nelder-Mead simplex method 52 iterations in 3 hours and 29 minutes to run one single calibration run.

6.3.2 Genetic algorithm

Here, we show the results of performing 20 calibration runs for each combination of selection method, population and selection size. We denoted the mean number of iterations, running time, minimum and maximum deviance including their bounds in the 20 populations. Next, we reported the root mean squared prediction error (RMSPE) as described in Section 5.3. Last, we report the p-value of the Wilcoxon signed-rank test for stochastic significant difference with Nelder-Mead simplex method.

While in the previous analysis a simple stopping rule was used, namely that the maximum deviance in the population should be at most 100 or at most 1000 iterations, we used a stopping rule based on the stopping rule of the current parameter search strategy as explained in Section 5.1.2. We evaluated the progression between two consecutive iterations.

Table 5: Results for the selection method 'Roulette wheel' with starting values using three parameters for twenty calibration runs

Population size	Selection size	Iterations	Running time	Min deviance	Max deviance	RMSPE	P-value
30	8	9	22 min	31.54 (15.81 – 55.89)	1335.74 (54.12 – 20621.91)	1.1431	**0.0033**
30	12	9	34 min	32.98 (17.45 – 54.52)	118.39 (45.45 – 244.22)	1.1259	**0.0004**
30	20	9	48 min	27.73 (12.76 – 45.95)	74.44 (40.81 – 172.10)	1.1147	**0.0008**
40	8	9	24 min	30.99 (13.49 – 53.04)	910.82 (71.40 – 4403.29)	1.1816	**0.0036**
40	12	9	34 min	29.11 (11.12 – 47.37)	151.63 (64.13 – 470.65)	1.1988	**0.0178**
40	20	9	49 min	27.71 (9.71 – 44.54)	78.30 (49.16 – 111.23)	1.0590	**0.0004**
50	8	9	26 min	30.04 (12.49 - 47.46)	1985.11 (194.51 - 16093.00)	1.1873	**0.0076**
50	12	9	37 min	27.75 (12.15 - 45.16)	255.50 (78.53 - 583.77)	1.2175	**0.0113**
50	20	9	51 min	25.33 (7.06 - 43.80)	104.85 (32.59 - 241.31)	1.4032	0.3935

Table 6: Results for the selection method 'Tournament Selection' with starting values using three parameters for twenty calibration runs

Population size	Selection size	Iterations	Running time	Min deviance	Max deviance	RMSPE	P-value
30	8	9	21 min	30.17 (11.79 - 48.67)	577.29 (73.09 - 8032.84)	1.1905	**0.0019**
30	12	9	32 min	31.71 (14.80- 43.22)	105.38 (59.72 - 190.65)	1.1808	**0.0065**
30	20	9	46 min	23.72 (12.29 - 44.07)	62.63 (30.73 - 94.04)	1.1818	**0.0046**
40	8	9	24 min	31.44 (20.17 - 45.73)	658.46 (68.27 - 3642.40)	1.1420	**0.0019**
40	12	9	34 min	27.23 (15.70 - 43.34)	116.63 (51.75 - 250.58)	1.2569	0.0760
40	20	9	49 min	24.01 (8.63 - 44.23)	76.87 (46.13 - 172.10)	1.1998	0.0598
50	8	9	26 min	31.98 (14.96 - 49.39)	2276.90 (152.33 - 11027.15)	1.2871	**0.0495**
50	12	9	37 min	26.09 (14.09 - 44.80)	227.24 (67.84 - 467.01)	1.3691	0.2280
50	20	9	52 min	27.12 (9.69 - 41.94)	88.45 (55.89 - 172.10)	1.2501	**0.0113**

Tables 5 and 6 show the results for 20 calibration runs for the selection method 'Roulette wheel' and 'Tournament Selection', respectively. The lowest RMSPE is obtained using a population size of 40 and a selection size of 20 using the selection method 'Roulette Wheel' and concluding from the p-value using a 5% significance level, this result is stochastically significantly different from the result of the Nelder-Mead simplex method and since the RMSPE is lower, this results is significantly better.

In Table 19 in Appendix A.3, the p-values are denoted for each combination of selection method, population and selection size in the genetic algorithm. Concluding from this table, 17 combinations performed stochastically different using a 5% significant level. These can be found by the bold numbers in the table. Using selection method 'Roulette wheel' and a population size of 50 and a selection size 20 resulted in a stochastic difference for almost all other combinations for the same selection method, only two were not stochastically different. Looking at Table 5, this is not a surprising result since the RMSPE of this combination is the highest of all. Also compared to the 'Tournament Selection', this combination is stochastically different for four out of nine combinations for selection method 'Tournament Selection'.

Also, we tested for significant stochastic difference between genetic algorithms and the Nelder-Mead simplex method. Looking at the p-values in Tables 5 and 6, we see that the results of only four combinations of population and selection sizes for both the selection methods are not significanlty stochastically different from the results of the Nelder-Mead simplex method. The RMSPE of Tables 5 and 6 are all smaller than the RMSPE in Section 6.3.1 and most of them are significantly stochastically different and therefore, they can be concluded to be significantly better.

Note that we can adjust the stopping rule in order to make it more strict and to achieve better results in terms of minimum deviance in the population and RMSPE. Looking deeper into the RMSPE, we noticed that the RMSPE is mostly affected by a bad estimation for the parameter

'decrease7080'. This can be due to an inaccurate calibration target for this parameter.

We also performed 20 calibration runs using a stopping rule based on Section 5.1.2 for five and eight parameters. However, it appeared that the progression between two consecutive iterations was not enough and therefore, the calibration terminated quickly for both five and eight parameters. Also, we used the stopping rule based on the progression between five consecutive iterations, but, again, the algorithm terminated quickly.

7 Conclusion

The EMC, Department of Public Health, uses a MIcrosimulation SCreening ANalysis (MISCAN) model to perform economic evaluations to inform policy makers on the long term costs and health outcomes of different interventions. MISCAN models have several parameters as input. However, some of the parameters are not (directly) observable and have to be found using calibration. Calibration involves finding parameter values such that model outcomes fit the observed data. Currently, Nelder-Mead simplex method is used for the calibration of this model. However, this method does not perform very well. It gets stuck in local optima and provides just one solution. Therefore, this thesis tried to implement a new algorithm for calibrating MISCAN models, a genetic algorithm. Especially, we used the MISCAN model for esophageal cancer.

Genetic algorithm are based on the natural selection mechanism in biology. This meta-heuristic is able to escape from local optima and provide a population of solutions at the end of the algorithm. Using survival of the fittest and randomized changes, a genetic algorithm tries to find new solutions. Recombining two solutions (parent solutions) from a population of solutions gives us new solutions (child solutions) and using the new solutions, we aim to find the 'best' solution in terms of the fitness value of the solution.

First, we implemented the algorithm without any prior knowledge. The results showed that calibrating three parameters worked good and selection method 'Roulette wheel' was the only selection method that was able to reach a deviance of at most 100 in the population. Moreover, 'Roulette wheel' also performs best in terms of running time compared to the other two methods. Unfortunately, calibrating five and eight parameters did not lead to convenient results.

Next, we inserted prior knowledge into the population and the running time was faster compared to the results without prior knowledge. Moreover, we were able to obtain results for five and eight parameters which means that the algorithm is working.

Last, as can be found in Section 6.3.1, we performed 20 calibration runs using three parameters for selection methods 'Roulette wheel' and 'Tournament Selection'. We also performed the 20 calibration runs using the Nelder-Mead simplex method. The RMSPEs for the genetic algorithm

using different population and selection sizes were lower compared to the RMSPE of the Nelder-Mead simplex method. Moreover, the running time for Nelder-Mead was longer than for the genetic algorithm. For a genetic algorithm, the results between selection method 'Roulette wheel' and 'Tournament Selection' are not all significantly stochastically different. The lowest RMSPE is obtained for 'Roulette wheel' using a population size of 40 and a selection size of 20 and this was even lower than the RMSPE of the Nelder-Mead simplex method. Moreover, this genetic algorithm is stochastically different in the prediction errors compared to the Nelder-Mead simplex method using a 5% significance level and therefore, it can be concluded that this genetic algorithm is significantly better.

One difference between 'Roulette wheel' and 'Tournament Selection' is that the selection method 'Roulette wheel' was the only selection method that was able to provide a population with deviances of at most 100 for all combinations of population and selection sizes without using any prior knowledge. Based on the results for calibrating three parameter, genetic algorithms using selection method 'Roulette wheel' with population size 40 and selection size 20 can be good alternatives for calibrating MISCAN models since they obtain (significantly) better results in terms of running time and RMSPE.

Further research can be done on optimizing the parameter settings for this algorithm and therefore improve the calibration for five and eight parameters, since the current strategy does not obtain desirable results. We know that the method works since we get results inserting prior knowledge into our genetic algorithm, but without this prior knowledge the algorithm stops quickly due to insufficient progression in the total deviance in the population. So, it would be interesting what kind of stopping rule should be used in order to get results for five and eight parameters. It could be that the stopping rule of Section 5.1.2 is not suitable for genetic algorithms. We adjusted this stopping rule such that the progression after five iterations was considered, but that did not lead to different results.

Also, the results of the Nelder-Mead simplex method could be improved in order to make a better comparison. Since the Nelder-Mead simplex method is a local search method and therefore, it is not guaranteed that it finds the global optimum, multiple calibration runs using both multiple random starting points as one random starting point should be performed. However, in this research, we chose to use only one random starting point to reduce the uncertainty of MISCAN and since the genetic algorithms uses the same random starting every calibration run.

Other research can be done involving the prior knowledge. Currently, a prior knowledge is inserted through the starting values of the simplex. In order to give some prior knowledge, but not as much as inserting a previous solution, we assigned distributions, other than the uniform

distribution, to the parameters based on the starting values currently used in the Nelder-Mead simplex method shown in Table 4. For five parameters, running the calibration without a stopping rule, we obtained after a couple of days a deviance of 12.96 using the selection method 'Roulette wheel', since this was concluded to be the most appropriate selection method, a population size of 30 and a selection size of 20. For the calibration of MISCAN models, we can use this kind of prior knowledge in the calibration. Moreover, further research could be done on the stopping rule in combination with prior knowledge.

References

Ahn, C. W. and Ramakrishna, R. S. (2002). A genetic algorithm for shortest path routing problem and the sizing of populations. *IEEE transactions on evolutionary computation*, 6(6):566–579.

Cobos, C., Erazo, C., Luna, J., Mendoza, M., Gaviria, C., Arteaga, C., and Paz, A. (2016). Multi-objective memetic algorithm based on NSGA-II and simulated annealing for calibrating CORSIM micro-simulation models of vehicular traffic flow. In *Conference of the Spanish Association for Artificial Intelligence*, pages 468–476. Springer.

Davis, L. (1991). Hybridization and numerical representation. *The Handbook of Genetic Algorithms*, pages 61–71.

Gilbert, E. W., Luna, R. A., Harrison, V. L., and Hunter, J. G. (2011). Barrett's esophagus: a review of the literature. *Journal of Gastrointestinal Surgery*, 15(5):708–718.

Hollander, Y. and Liu, R. (2008). The principles of calibrating traffic microsimulation models. *Transportation*, 35(3):347–362.

Karnon, J. and Vanni, T. (2011). Calibrating models in economic evaluation. *Pharmacoeconomics*, 29(1):51–62.

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Kim, S.-J., Kim, W., and Rilett, L. (2005). Calibration of microsimulation models using nonparametric statistical techniques. *Transportation Research Record: Journal of the Transportation Research Board*, (1935):111–119.

Kong, C. Y., Kroep, S., Curtius, K., Hazelton, W. D., Jeon, J., Meza, R., Heberle, C., Miller, M., Choi, S. E., Lansdorp-Vogelaar, I., van Ballegooijen, M., Feuer, E. J., Inadomi, J. M., Hur, C., and Luebeck, E. G. (2014). Exploring the recent trend in esophageal adenocarcinoma incidence and mortality using comparative simulation modeling. *Cancer Epidemiology and Prevention Biomarkers*, pages cebp–1233.

Kong, C. Y., McMahon, P. M., and Gazelle, G. S. (2009). Calibration of disease simulation model using an engineering approach. *Value in health*, 12(4):521–529.

Kroep, S. (2015). *Exploring the Natural History of Esophageal Adenocarcinoma and Possibilities for Early Detection and Intervention.* PhD thesis, Erasmus MC: University Medical Center Rotterdam.

Kroep, S., Lansdorp-Vogelaar, I., Van Der Steen, A., Inadomi, J. M., and Van Ballegooijen, M. (2015). The impact of uncertainty in barrett's esophagus progression rates on hypothetical screening and treatment decisions. *Medical decision making*, 35(6):726–733.

Lipowski, A. and Lipowska, D. (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6):2193–2196.

Masclee, G., Coloma, P., de Wilde, M., Kuipers, E., and Sturkenboom, M. (2014). The incidence of barrett's oesophagus and oesophageal adenocarcinoma in the united kingdom and the netherlands is levelling off. *Alimentary pharmacology & therapeutics*, 39(11):1321–1330.

Miller, B. L. and Goldberg, D. E. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212.

Neddermeijer, H. G., van Oortmarssen, G., Piersma, N., Dekker, R., and Habbema, D. (2000). Adaptive extensions of the nelder and mead simplex method for optimization of stochastic simulation models. Technical report.

Rutter, C. M., Miglioretti, D. L., and Savarino, J. E. (2009). Bayesian calibration of microsimulation models. *Journal of the American Statistical Association*, 104(488):1338–1350.

Stout, N. K., Knudsen, A. B., Kong, C. Y., McMahon, P. M., and Gazelle, G. S. (2009). Calibration methods used in cancer simulation models and suggested reporting guidelines. *Pharmacoeconomics*, 27(7):533–545.

Van der Steen, A., van Rosmalen, J., Kroep, S., van Hees, F., Steyerberg, E. W., de Koning, H. J., van Ballegooijen, M., and Lansdorp-Vogelaar, I. (2016). Calibrating parameters for microsimulation disease models: a review and comparison of different goodness-of-fit criteria. *Medical Decision Making*, 36(5):652–665.

Vanni, T., Karnon, J., Madan, J., White, R. G., Edmunds, W. J., Foss, A. M., and Legood, R. (2011). Calibrating models in economic evaluation. *Pharmacoeconomics*, 29(1):35–49.

Vizcaino, A. P., Moreno, V., Lambert, R., and Parkin, D. M. (2002). Time trends incidence of both major histologic types of esophageal carcinomas in selected countries, 1973–1995. *International journal of cancer*, 99(6):860–868.

Wright, A. H. (1991). Genetic algorithms for real parameter optimization. In *Foundations of genetic algorithms*, volume 1, pages 205–218. Elsevier.

Wright, S. and Nocedal, J. (1999). Numerical optimization. *Springer Science*, 35(67-68):7.

List of Symbols

General calibration algorithm

$\boldsymbol{\mu}$	vector of observed data
$\boldsymbol{\theta}$	vector of K parameters to be calibrated
μ_i	observed outcome i
$D(\boldsymbol{\theta})$	total deviance using vector of parameters $\boldsymbol{\theta}$
$D(\cdot)$	total deviance
$D_i(\boldsymbol{\theta})$	deviance for outcome i using vector of parameters $\boldsymbol{\theta}$
$D_i(\cdot)$	deviance function for outcome i
K	number of parameters to be calibrated
$M(\boldsymbol{\theta})$	(random) model outcomes for vector of parameters $\boldsymbol{\theta}$
$M(\cdot)$	the model
n_1	the number of persons used in the observed data
n_2	the number of persons used in the model

Genetic algorithm

$\boldsymbol{\theta}_b$	parent solution b with the highest fitness value in the tournament
$\boldsymbol{\theta}_t$	parent solution t
$\boldsymbol{\theta}_c$	child solution c
$\boldsymbol{\theta}_{tj}$	parameter j of parent solution t, where $j \in \{1, \ldots, K\}$
$\boldsymbol{R}$	matrix of ranges of the parameters
B	set of parent solutions in a tournament
C	set of child solutions
$F(\boldsymbol{\theta}_t)$	fitness function for parent solution t
$F(\cdot)$	fitness function
l	maximum of possible cross-over point
M	mutation size
N	population size
n	selection size, where $n < N$
P	set of parent solutions
p_M	mutation rate
p_t	selection probability of parent solution t
R	set of parent solutions selected by 'Roulette wheel'
r, p_t, p_j	random number between 0 and 1
S	set of selected parent solution

T	set of parent solutions selected by 'Tournament Selection'
z	tournament size, where $z \leq N$
CrossPoint	cross-over point
LB	lower bound
partFIT	subtotal of fitness in the population
totalFIT	the total fitness in the population
UB	upper bound

Comparing parameter search strategies

$\hat{\theta}_j^w$	estimated parameter value j in calibration run w
θ_j	known, underlying parameter value j
m	number of calibration runs to be performed

Nelder-Mead simplex algorithm

$\alpha, \beta, \gamma, \delta$	parameters for the Nelder-Mead simplex method between 0 and 1
$\boldsymbol{\theta}^k_{\text{cent}}$	the vertex in the centroid
$\boldsymbol{\theta}^k_{\text{exp}}$	expanded vertex
$\boldsymbol{\theta}^k_{\text{hi}}$	vertex with the highest value
$\boldsymbol{\theta}^k_{\text{low}}$	vertex with the lowest value
$\boldsymbol{\theta}^k_{\text{nexthi}}$	vertex with the highest value
$\boldsymbol{\theta}^k_{\text{refl}}$	θ^k_{hi} is reflected through centroid of the remaining vertices
$\boldsymbol{\theta}^k_{c_1}$	contracted vertex between $\boldsymbol{\theta}^k_{\text{hi}}$ and $\boldsymbol{\theta}^k_{\text{cent}}$
$\boldsymbol{\theta}^k_{c_2}$	contracted vertex between $\boldsymbol{\theta}^k_{\text{refl}}$ and $\boldsymbol{\theta}^k_{\text{cent}}$
τ	Kendall's tau, the rank correlation coefficient

Appendix A Results

A.1 Without starting values

Table 7: Results for the selection method 'Elitism' without starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	323	± 5h 50 min	66.50	99.50
30	12	161	± 4h 30 min	65.51	99.71
30	20	57	± 2h 35 min	66.18	99.15
40	8	474	± 11h 49 min	65.31	99.47
40	12	1000	± 26h 23 min	74.88	109.00
40	20	909	± 41h 25 min	78.90	99.82
50	8	1000	± 17h 46 min	99.60	249.92
50	12	184	± 5h 12 min	56.89	99.97
50	20	86	± 3h 58 min	51.34	99.78

Table 8: sults for the selection method 'Roulette wheel' without starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	79	± 1h 30 min	65.74	99.97
30	12	182	± 11h 25 min	68.13	99.25
30	20	40	± 1h 52 min	66.54	98.22
40	8	124	± 2h 19 min	70.39	98.88
40	12	41	± 1h 45 min	66.16	98.71
40	20	46	± 2h 0 min	72.10	99.75
50	8	162	± 3h 1 min	62.11	99.87
50	12	148	± 4h 8 min	44.51	99.65
50	20	90	± 6h 4 min	54.09	99.90

Table 9: Results for the selection method 'Tournament' without starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	461	± 11h 13 min	71.78	99.84
30	12	1000	± 27h 50 min	281.37	438.27
30	20	186	± 8h 47 min	78.53	99.68
40	8	1000	± 18h 10 min	75.42	102.71
40	12	688	± 19h 30 min	80.42	99.87
40	20	253	± 11h 40 min	67.33	99.81
50	8	869	± 15h 40 min	69.62	99.97
50	12	990	± 27h 36 min	74.69	99.68
50	20	279	± 12h 56 min	73.05	99.74

A.2 With starting values

A.2.1 Three parameters

Table 10: Results for the selection method 'Elitism' with starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	20	± 27 min	17.11	81.79
30	12	19	± 26 min	13.28	50.18
30	20	18	± 39 min	12.25	61.04
40	8	8	± 27 min	16.36	63.83
40	12	8	± 43 min	15.55	86.07
40	20	6	± 1h 17 min	11.66	72.70
50	8	19	± 39 min	16.11	97.44
50	12	13	± 37 min	16.75	64.50
50	20	15	± 1h 12 min	12.30	81.08

Table 11: Results for the selection method 'Roulette wheel' with starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	11	± 33 min	12.92	64.44
30	12	8	± 26 min	16.12	47.69
30	20	17	± 1h 21 min	16.24	42.93
40	8	23	± 47 min	17.78	57.94
40	12	8	± 28 min	17.86	44.01
40	20	15	± 1h 12 min	16.82	73.92
50	8	18	± 18 min	13.52	74.71
50	12	13	± 45 min	12.52	83.34
50	20	9	± 48 min	16.30	89.84

Table 12: Results for the selection method 'Tournament' with starting values using three parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	13	± 27 min	18.83	50.22
30	12	11	± 36 min	18.50	57.02
30	20	8	± 39 min	12.27	37.80
40	8	12	± 27 min	11.10	58.00
40	12	13	± 43 min	13.76	72.51
40	20	16	± 1h 17 min	14.27	43.20
50	8	17	± 39 min	17.82	84.65
50	12	10	± 37 min	12.77	59.57
50	20	16	± 1h 18 min	17.36	81.23

A.2.2 Five parameters

Table 13: Results for the selection method 'Elitism' with starting values using five parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	19	± 52 min	25.07	61.22
30	12	31	± 2h 7 min	19.51	73.39
30	20	13	± 1h 36 min	20.26	53.90
40	8	60	± 2h 10 min	15.73	56.31
40	12	32	± 2h 28 min	20.54	51.26
40	20	18	± 2h 13 min	19.88	45.18
50	8	131	± 4h 35 min	20.78	87.63
50	12	42	± 2h 39 min	17.96	64.87
50	20	15	± 1h 54 min	23.97	74.38

Table 14: Results for the selection method 'Roulette wheel' with starting values using five parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	33	± 1h 25 min	21.39	70.85
30	12	23	± 1h 47 min	13.74	62.91
30	20	16	± 1h 55 min	24.88	56.47
40	8	36	± 1h 40 min	17.64	88.76
40	12	12	± 59 min	23.96	70.88
40	20	26	± 2h 53 min	16.85	51.98
50	8	17	± 58 min	19.13	77.78
50	12	40	± 2h 41 min	17.07	57.69
50	20	17	± 2h 10 min	16.33	57.06

Table 15: Results for the selection method 'Tournament' with starting values using five parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	30	± 1h 22 min	17.41	55.40
30	12	17	± 1h 8 min	23.37	51.04
30	20	16	± 1h 57 min	20.46	56.20
40	8	31	± 1h 33 min	17.54	86.11
40	12	29	± 2h 11 min	22.50	73.37
40	20	15	± 1h 43 min	47.95	21.30
50	8	37	± 1h 47 min	15.79	70.70
50	12	28	± 2h 11 min	19.24	61.40
50	20	25	± 2h 49 min	19.07	55.92

A.2.3 Eight parameters

Table 16: Results for the selection method 'Elitism' with starting values using eight parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	47	± 5h 17 min	21.54	71.11
30	12	46	± 4h 46 min	20.61	77.18
30	20	29	± 6h 22 min	19.88	49.31
40	8	156	± 5h 18 min	15.82	61.55
40	12	271	± 15h 5 min	20.26	94.52
40	20	23	± 2h 18 min	18.89	53.10
50	8	491	± 15h 46 min	21.94	69.05
50	12	89	± 5h 40 min	13.78	81.50
50	20	47	± 4h 53 min	18.94	54.20

Table 17: Results for the selection method 'Roulette wheel' with starting values using eight parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	28	± 1h 1 min	22.78	64.77
30	12	74	± 4h 3 min	21.89	59.22
30	20	36	± 4h 2 min	23.37	64.96
40	8	24	± 1h 0 min	26.28	58.95
40	12	23	± 1h 44 min	20.89	57.60
40	20	32	± 3h 38 min	22.22	60.40
50	8	57	± 2h 7 min	20.81	63.19
50	12	41	± 2h 36 min	15.91	57.69
50	20	22	± 2h 29 min	19.69	62.51

Table 18: Results for the selection method 'Tournament' with starting values using eight parameters for one calibration run

Population size	Selection size	Number of generations	Running time	Min deviance	Max deviance
30	8	84	± 3h 36 min	19.83	59.17
30	12	22	± 1h 31 min	28.24	87.90
30	20	15	± 1h 31 min	22.80	48.58
40	8	55	± 2h 28 min	22.52	78.38
40	12	26	± 1h 52 min	21.34	63.39
40	20	44	± 4h 35 min	15.94	64.62
50	8	47	± 2h 36 min	13.32	83.29
50	12	93	± 6h 4 min	15.37	76.01
50	20	33	± 3h 24 min	21.05	67.75

A.3 Performance parameter search strategies

Table 19: P-values for Wilcoxon-rank test for different genetic algorithms

					'Roulette wheel'						'Tournament Selection'					
	N/n	30/8	30/12	30/20	40/8	40/12	40/20	50/8	50/12	50/20	30/8	30/12	30/20	40/8	40/12	40/20	50/8	50/12	50/20
'Roulette wheel'	30/8		0.818099159	0.490313914	0.776339442	0.560778164	0.18055693	0.91382871	0.71491593	**0.013833499**	0.797169499	0.588471377	0.725099894	0.714915935	0.198813426	0.516207731	0.198792361	0.0547763	0.579200556
	30/12			0.626247112	0.704870144	0.285078667	0.208325482	0.490212127	0.316761508	0.004503923	0.881686715	0.285169111	0.625849486	1	0.076389545	0.09349109	0.144040034	**0.011869103**	0.607252052
	30/20				0.37202063	0.228650692	0.41707706	0.456929508	0.176193715	**0.006830256**	0.498840244	0.255913151	0.498880519	0.665144288	0.098931599	0.223507588	0.098915771	**0.011877055**	0.386707993
	40/8					0.645560857	0.113517527	0.881719777	0.665085268	**0.021484198**	0.967631573	0.946071347	0.935319423	0.607201466	0.364751663	0.60726891	0.343674045	0.09349109	0.956847261
	40/12						**0.046770996**	0.588453874	0.87103921	0.051451137	0.714928728	0.655331195	0.766003268	0.432666654	0.52494803	0.892408848	0.524928587	0.16768397	0.828667499
	40/20							0.285282212	0.061928625	**0.001551574**	0.198813426	0.107494865	0.090892229	0.279160727	**0.014912205**	0.093521849	**0.017288246**	**0.003192189**	0.110498784
	50/8								0.745436908	**0.034866372**	0.978417759	0.892413864	0.95685535	0.745436908	0.25591315	0.473459878	0.37202063	0.088337091	0.849818021
	50/12									0.063867318	0.607218330	0.903107451	0.725087522	0.464923419	0.51618805	0.84981106	0.465070102	0.133228552	0.989207892
	50/20										**0.023073982**	**0.04248308**	**0.041114098**	**0.013820144**	0.2036602674	0.069931212	0.255823938	0.735244248	0.107494865
'Tournament Selection'	30/8											0.655346318	0.913836762	0.725050396	0.285282212	0.401720329	0.364796663	0.0547763	0.745483018
	30/12												0.714967096	0.490293562	0.267362155	0.516207731	0.409311912	0.098868285	0.935316395
	30/20													0.645607321	0.386707993	0.645622805	0.35770542	0.065842804	0.828675411
	40/8														0.159525084	0.432753012	0.155453585	**0.031493194**	0.424840028
	40/12															0.35772797	0.881719777	0.498860383	0.516207731
	40/20																0.551756661	0.223485871	0.786774932
	50/8																	0.542658101	0.51618805
	50/12																		0.194064776