Blauth, Pigeot:

Using Genetic Algorithms for Model Selection in Graphical Models

Sonderforschungsbereich 386, Paper 278 (2002)

Online unter: http://epub.ub.uni-muenchen.de/

Using Genetic Algorithms for Model Selection in Graphical Models

Angelika Blauth*[1], Iris Pigeot**[1]

*Department of Statistics, University of Munich, Ludwigstr. 33, D–80539 München
**Institute of Statistics, University of Bremen, Postfach 330440, D–28334 Bremen
Bremen Institute for Prevention Research and Social Medicine, Linzer Str. 8–10, D–28359 Bremen

Summary

Model selection in graphical models is still not fully investigated. The main difficulty lies in the search space of all possible models which grows more than exponentially with the number of variables involved. Here, genetic algorithms seem to be a reasonable strategy to find good fitting models for a given data set. In this paper, we adapt them to the problem of model search in graphical models and discuss their performance by conducting simulation studies.

Keywords: *Genetic algorithms; Graphical models; Model selection*

1 Introduction

Modern technologies make it nowadays quite easy to collect huge amounts of data in course of empirical studies which results in the challenging demand to extract the substantial information and find certain patterns inside the data. Such multivariate data sets usually imply a complex association structure which cannot be analyzed in a straightforward manner based on standard statistical software without running the risk of loosing important information. Graphical models are a sensible approach to cope with these problems.
The basic idea is to represent a multivariate model in a graph where each element is to be interpreted in a purely statistical sense. The vertices represent the variables and the (non–)existence of edges between them can be interpreted as marginal or conditional (in)dependence statements based on certain rules and under specific assumptions.
Although the theory of graphical models seems to be examined thoroughly, there still is the problem to find a well fitting model for a given data set. The main difficulty lies in the

[1]Financial support by the SFB 386, Deutsche Forschungsgemeinschaft is gratefully acknowledged.

search space of all possible models which grows more than exponentially with the number of variables involved. Model selection itself can be seen as an optimization problem where the global optimum corresponds to the best fitting model. It is then the task of a selection strategy to find this model and not to get stuck in other sub–optimal models. Currently, there exists a variety of strategies, based on both a frequentist and a Bayesian point of view. Some of them are restricted to certain types of graphical models whereas others are more generally applicable. All of these methods have their advantages and disadvantages but all have in common that they may not find the global optimum. Since genetic algorithms have been successfully applied to various optimization tasks over the years it is here to examine whether this approach can better cope with the above mentioned problem than more conventional selection techniques.

This paper is organized as follows. First, we briefly introduce graphical models and some well–known selection strategies in Section 2. We then sketch genetic algorithms in general in Section 3, whereas we present the adaptation of these algorithms to model search in graphical models in some more detail in Section 4. The behavior of the derived genetic algorithms is investigated by means of a simulation study. The obtained results are discussed in depth in Section 5. We conclude with a short discussion.

2 Graphical Models and Model Selection

The basic idea of graphical models is to represent a multivariate data set by a graph $\mathcal{G} = (V, E)$ where each element is to be interpreted in a purely statistical sense. The vertices of such a graph reflect the random variables of the model and the edges the association structure, where E is a subset of $V \times V$ of ordered pairs of distinct vertices. Depending on the scale or on the underlying association structure of the involved variables we can distinguish several types of graphical models. In this paper, we will restrict ourselves to undirected graphical models. Here, a missing edge between two vertices is interpreted as conditional independence between the corresponding variables given all remaining one if certain distributional assumptions are fulfilled. For instance, a model may contain simultaneously discrete and continuous variables. In this case, a Conditional Gaussian (CG) distribution (Lauritzen and Wermuth, 1989) turns out as appropriate. In case of only discrete or continuous variables a multinomial distribution or a multivariate normal distribution may be adequate, respectively. Two types of models are of particular interest: In the *saturated model* the underlying graph is complete and in the *main effects model* no edge is present in the underlying graph. If not otherwise stated we describe an undirected graphical model by denoting the different cliques of the underlying graph, separated by a comma. For example if $V = \{a, b, c\}$, the saturated model is written as abc and the main effects model as a, b, c. For a deeper insight into graphical models we refer to the books of Edwards (2000), Lauritzen (1996), or Whittaker (1990).

A key task when applying graphical models to a data set is to select the appropriate model. Different strategies have been proposed for this purpose. Some of them can be used for certain types of graphical models only, whereas others are more generally applicable. The short overview, given here, will focus on the more general selection techniques *stepwise selection* and the *Edwards–Havránek (EH) procedure* which will be compared with the genetic algorithms based on simulated data sets in Section 5.

Selecting a graphical model the stepwise procedures proceed as follows, where mainly the number of associations is of interest. *Backward selection* starts with the saturated model and thus there are no (conditional) independence restrictions present. Step by step, individual edges are then removed until the remaining ones are all significant with regard to a certain criterion. At each step the edge that has the worst (non–significant) value according to the significance criterion is removed. In case of *forward selection*, the process starts with the main effects model. Analogously to backward selection, individual edges are now added to the model according to their importance until the stopping criterion is again fulfilled.

The EH procedure (Edwards and Havránek 1985, 1987) is based on search algorithms which seek the most parsimonous models, i.e. those with the lowest number of parameters, consistent with the data based on an overall fitting strategy. The procedure is appropriate for all model families that form a lattice (Edwards and Havránek 1987). For graphical models the simplest models are those that have as many conditional independencies as possible and thus the lowest number of edges. The EH procedure is based on two rules: The first states that if a model is *accepted*, i.e. judged to be consistent with the data, then all supermodels that include the former one are also accepted. The second rule states that if a model is *rejected*, i.e. judged to be inconsistent with the data, then all of its submodels are also rejected. Two terms are introduced to reflect these concepts: A model is called *w–accepted* (weakly accepted) if it includes an accepted model, and it is called *w–rejected* (weakly rejected) if it is included by a rejected model. At any stage of the selection process the whole range of possible graphical models is divided into three disjoint sets. One set consists of all w–accepted models. Another set contains all w–rejected models and between these two sets those models can be found of which the consistency with the data has still to be verified. At each step either the minimal or the maximal models in the undetermined set are tested and the sets of w–accepted and w–rejected models are updated according to the results of the corresponding test. The procedure stops when all possible models are classified as w–accepted or w–rejected. Note that this procedure may result in more than one model because different models may be judged as being consistent with the data which are equally sparse.

For examplary applications of this strategy see e.g. Edwards (2000, p. 146ff) or Edwards and Havránek (1985).

3 Genetic Algorithms

Genetic algorithms (GA) are used to solve search and optimization problems. Their name is due to a close similarity to evolutionary processes. In evolution theory, a population of individuals evolves by natural selection and survival of the fittest. This was first investigated by Charles Darwin in 1859 (The Origin of Species, Darwin 1958). Genetic algorithms mimic the evolution of creatures in that they try to find more and more better solutions for the given problem by producing more and more better descendants.

Genetic algorithms mainly consist of five different parts. The application of a GA to a given problem requires to define a possible solution such that a GA can handle it. Such a process has to be performed for each parameter of the problem and the result of this process is called *gene*. The genes are then joined forming a string called *chromosome*. All possible chromosomes together build the *search space*. In a pure genetic algorithm, a binary alphabet (that means an alphabet only consisting of 0 and 1) is used for coding the chromosomes. Thus, each gene (and with that the whole chromosome) consists of several *bits*. Then, an *initial population* has to be generated, and a *fitness function* has to be assigned, which measures the fit of a given solution. For running the algorithm it is important to *select* the parents and to *recombine* them to generate offspring in a certain way. This part of the algorithm is called *reproduction*. Here, it has to be ensured that the space in which the solution can be found, called *problem space*, is searched in an efficient manner. The two mostly used operators in the reproduction step are *crossover*, which generates a new chromosome by combining two chromosomes, and *mutation*, which changes one or more bits of a chromosome. Both operators are applied with a different probability. The search is finished when prespecified *termination criteria* are satisfied. However, these criteria do not always imply convergence to the global optimum. It might occur that the algorithm gets stuck in a local optimum instead. It is, therefore, advisable to run the algorithm, regardless of the termination criteria, a situation dependent minimal number of iterations.

4 Genetic Algorithms Applied to Model Selection in Graphical Models

Let us now deal with the necessary steps to adapt genetic algorithms to model search in graphical models. Such an adaption implies fixing the various parameters involved in genetic

algorithms. Whereas general hints can be given for some of these parameters, others have to be learned by simulation studies.
Since a conditional independence graph corresponds to a particular graphical model with a given independence structure, both the graph and the model formula are an appropriate choice for the chromosomes of a genetic algorithm. In this paper, we utilize the graph because the reproduction operators can be easily adapted to its internal representation as an upper triangular matrix. To introduce this structure we first have to define an adjacency matrix.

Definition 4.1 (Adjacency matrix) *An adjacency matrix $\mathcal{A}(\mathcal{G})$ is a representation of an undirected graph $\mathcal{G} = (V, E)$, $|V| = k$, i.e. a $k \times k$ matrix with entries $e_{ij}, i, j = 1, \ldots, k$, such that $e_{ij} = 1$ if and only if an edge exists between i and j, $i \neq j$, $e_{ij} = 0$ if and only if there is no edge between the two vertices, and $e_{ij} = 1$ for $i = j$.*

Note that an adjacency matrix of an undirected graph is symmetric. Thus, it is sufficient to work only with the upper triangular matrix. Following Laplante (2000), we represent such a matrix as follows.

Definition 4.2 (Upper triangular matrix) *Let $\mathcal{A}(\mathcal{G})$ be a $k \times k$ adjacency matrix of an undirected graph $\mathcal{G} = (V, E)$, $|V| = k$, with entries $e_{ij}, i, j = 1, \ldots, k$. An upper triangular matrix $\vec{\mathcal{A}}(\mathcal{G})$ is a vector consisting of elements $e_l, l = 1, \ldots, \frac{(k+1)k}{2}$, which stores an entry $e_{ij}, i \leq j$, of the corresponding adjacency matrix $\mathcal{A}$ in location $l = k(i-1) + j - \frac{i(i-1)}{2}$.*

When applying GAs, the next step is to fix the size N of an initial population and to initialize its elements. In our case the population is randomly initialized such that each possible edge has a probability of 0.5 to be in the model. Choosing the initial population purely at random follows the general guidelines for running a GA, whereas the size N mainly depends on the structure of the function to be optimized. If it is well–behaved a small size is sufficient to work with. If the function is somehow difficult, e.g. if it is multimodal or if it is very flat in the surroundings of the minimum or maximum, a larger size is needed. Simulations are therefore helpful to decide on the appropriate N, reported in Section 5.

To calculate the fitness of the models at each step of the algorithm we need proper criteria which are able to rate given models. Usual goodness–of–fit criteria will typically rate more complicated models, i.e. those with more parameters, to be better than sparser ones. A selected model, however, should have only as many parameters as necessary to give an adequate representation of the association structure underlying the data. One criterion accounting for this trade–off between model complexity and goodness–of–fit is the *BIC* (Bayesian information criterion, Schwarz 1978). It penalizes the complexity of an estimated model by an additional term, depending on the number of parameters of the model and the sample size.

Definition 4.3 (Bayesian Information Criterium) *Let $\hat{\theta}$ be the unique maximum of the likelihood function $L(\theta)$ corresponding to a certain graphical model. Let m be the number of parameters of a given model and n be the sample size. Then, the BIC (Bayesian information criterion) is defined as*

$$BIC = -2\ln(\hat{\theta}) + m\log(n).$$

The BIC corrects for the dependence of the fit on the sample size: the larger a sample, the more "significant" an effect has to be in absolute terms. As a result, the BIC does not tend to overfit the data, especially if we have a larger number of observations. Since the model with the smallest BIC is regarded as the best model the task is to minimize f. The ML–estimator itself is calculated via the *iterative proportional scaling* (IPS) algorithm (Demming and Stephan 1940) when considering pure models and via the *modified iterative proportional scaling* (MIPS) algorithm (Frydenberg and Edwards 1989) in case of mixed models.

In the selection step, we weight the chromosomes of a generation with their BIC–value, so that chromosomes with a lower BIC are more likely to reproduce than chromosomes with a higher one. In our optimization problem we apply *tournament selection* which is based on the ranks of the chromosomes. It randomly draws a small number j of chromosomes in one iteration and selects the best one from this set of j elements to be a parent of the next generation. With N being the size of the population this process is repeated N times. A typical value adapted in many applications is $j = 2$ (Michalewicz 1996, p. 61). Besides this aspect we apply an elitist genetic algorithm, which means that the best model is maintained. This ensures, in theory, that the global optimum will be found (Rudolph 1994).
Next, we perform crossover on the selected chromosomes with a certain probability p_c for each chromosome. We have two different types of crossover operators, the *one–point crossover for graphs* and the *union–intersection crossover* which are introduced in Definition 4.4 and 4.6.

Definition 4.4 (One–point crossover for graphs) *Let $\vec{\mathcal{A}}_1(\mathcal{G}_1)$ and $\vec{\mathcal{A}}_2(\mathcal{G}_2)$ be upper triangular matrices for the graphs $\mathcal{G}_1 = (V, E_1)$ and $\mathcal{G}_2 = (V, E_2)$, $|V| = k$, respectively, with elements e_l and $f_l, l = 1, \dots, \frac{(k+1)k}{2}$. Further, let r be a random value drawn from the discrete uniform distribution $U[1, \frac{(k+1)k}{2}]$. Combining the elements of $\vec{\mathcal{A}}_1$ and $\vec{\mathcal{A}}_2$ to get two children $\vec{\mathcal{C}}_1$ and $\vec{\mathcal{C}}_2$ by:*

$$\begin{aligned}\vec{\mathcal{C}}_1 &= e_1, \dots e_r, f_{r+1}, \dots, f_{\frac{(k+1)k}{2}},\\ \vec{\mathcal{C}}_2 &= f_1, \dots f_r, e_{r+1}, \dots, e_{\frac{(k+1)k}{2}}\end{aligned}$$

is called a one–point crossover for graphs.

Before defining the union–intersection crossover it is necessary to introduce the union and intersection operators for upper triangular matrices.

Definition 4.5 (Union and intersection for upper triangular matrices) *Let $\vec{A}, \vec{B}$ be upper triangular matrices with $|\vec{A}| = |\vec{B}| = u$ and elements $e_l, f_l,\ l = 1, \ldots, u$. The intersection operator $e_l \wedge f_l$ and the union operator $e_l \vee f_l$ are defined by listing all possible outcomes in Table 1.*

e_l	f_l	$e_l \wedge f_l$	$e_l \vee f_l$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Table 1: The union and intersection operators for upper triangular matrices

The intersection operator $\vec{\mathcal{A}} \wedge \vec{\mathcal{B}}$ and the union operator $\vec{\mathcal{A}} \vee \vec{\mathcal{B}}$ of two upper triangular matrices are defined by elementwise union and intersection $e_l \wedge f_l,\ l = 1, \ldots, u$, and $e_l \vee f_l,\ l = 1, \ldots, u$, respectively.

We are now ready to define the second type of crossover.

Definition 4.6 (Union–intersection crossover) *Let $\vec{\mathcal{A}}_1(\mathcal{G}_1)$ and $\vec{\mathcal{A}}_2(\mathcal{G}_2)$ be upper triangular matrices for the graphs $\mathcal{G}_1 = (V, E_1)$ and $\mathcal{G}_2 = (V, E_2)$, $|V| = k$, respectively. Combining the elements of $\vec{\mathcal{A}}_1$ and $\vec{\mathcal{A}}_2$ to get two children $\vec{\mathcal{C}}_1$ and $\vec{\mathcal{C}}_2$ by:*

$$\begin{aligned} \vec{\mathcal{C}}_1 &= \mathcal{A}_1 \wedge \mathcal{A}_2 \ \textit{(intersection)}, \\ \vec{\mathcal{C}}_2 &= \mathcal{A}_1 \vee \mathcal{A}_2 \ \textit{(union)} \end{aligned}$$

is called a union–intersection crossover for graphs.

Since the 1–entries of an upper triangular matrix are edges of an undirected graph the operations defined in Definition 4.6 correspond to the usual intersection and union of graphs. The crossover operator to be applied is chosen randomly for each pair of parents so that a wide range of new graphs can be generated.
In the last step of the reproduction phase each chromosome of the new generation undergoes mutation with a certain low probability p_m. The mutation operator is defined as follows.

Definition 4.7 (Mutation on graphs) *Let $\vec{\mathcal{A}}(\mathcal{G})$ be an upper triangular matrix for the graph $\mathcal{G} = (V, E), |V| = k$, with elements $e_l,\ l = 1, \ldots, \frac{(k+1)k}{2}$, and r be a random value drawn from the discrete uniform distribution $U[1, \frac{(k+1)k}{2}]$. Building a new upper triangular matrix $\vec{\mathcal{B}}$ out of $\vec{\mathcal{A}}$ such that*

$$\vec{\mathcal{B}} = [e_1, \ldots, \neg e_r, \ldots, e_{\frac{(k+1)k}{2}}]$$

with $\neg e_r = 1$ if $e_r = 0$ and $\neg e_r = 0$ if $e_r = 1$ is called mutation on graphs.

Definition 4.7 corresponds to inverting the status of one edge in the graph, i.e. removing the edge if it is present and adding it if it is not present.

Considering termination, we perform simulation studies to find appropriate numbers of iterations for the algorithm (cf. Section 5).

5 Simulation

Although there exist some theoretical results explaining the circumstances under which GAs find the global optimum of a problem, applications do not always follow this theory. This is mainly due to the asymptotic nature of these results on the one hand and to the limited population size on the other hand. Nevertheless, based on static or dynamic criteria it is possible to check whether the chosen coding and genetic operators are appropriate at all to find the global maximum for a given problem. For the problem of model search in graphical models we performed a detailed analysis in Blauth (2002). As conclusion it seems as if the problem of model search is easy for a genetic algorithm to solve and that the selected coding and genetic operators are suitable.
Further, the application of a genetic algorithm implies the adjustment of several parameters like the probabilities for crossover and mutation, the size of the population, and the number of iterations. Different parameter constellations may be successful depending on the kind of application.
Therefore, in Section 5.1 we investigate which parameter constellations are promising for our problem. In addition, we compare the genetic algorithm with more standard selection procedures as forward and backward selection and the Edwards–Havránek procedure in Section 5.2.

5.1 Parameter Constellations

In the following simulation studies we consider two different data sets: one consisting of six discrete variables and the other of five variables where two are discrete and three continuous. Although it may be desirable to investigate more than these two data sets by means of simulation results they are sufficient to get a first impression of the behavior of the genetic algorithm in this context. In addition, we restrict ourselves to rather small models since then an exhaustive search is possible to identify the "best" model with which the one resulting from the genetic algorithm and from other strategies will be compared.
The chosen values for the probability of mutation p_m, the probability of crossover p_c, the size of population N, and the number of iterations T are summarized in Table 2.
The rationale behind these values is as follows. We investigate the behavior of the algorithm

parameter	**value**
Number of iterations T	[50,. . . , 200]
Size of population N	15, 30, 50
Probability of mutation p_m	0.01, 0.1, 0.5
Probability of crossover p_c	0.7, 0.8, 0.9

Table 2: Parameter values of the different simulations

depending on the number of iterations T in a range of 50 to 200. It is expected that a smaller number of iterations leads to rather unstable results and a higher number does not lead to any further improvement The values for the size of the population N as 15, 30, and 50 reflect a small, medium, and large number of different models in one population. The role of mutation is discussed in the literature on GAs: While some references state that a very small mutation rate p_m is sufficient to ensure that an algorithm does not get stuck in parts of the search space, others are convinced that a quite high mutation rate may be important for the success of a GA. We therefore vary the parameter from 0.01 over 0.1 to 0.5. Since a high probability of crossover p_c is required for an efficient search through the problem space, this parameter ranges from 0.7 to 0.9.

In data set (a) we simulate the distribution of six discrete variables $V = \{a, b, c, d, e, f\}$, each having two levels and 10000 observations. On six vertices we have a total of 32768 different graphical models. We generated the data by drawing samples from a multinomial distribution with the underlying association structure cf, be, bd, ac, ab. An exhaustive search resulted in cf, be, bd, ac being the best model with respect to the BIC which has a value of 73785.4344. It differs from the original imputed one in that edge ab misses. This difference may occur due to sampling errors and it can be seen as a confirmation of the well–known statement that a chosen measure of fit may emphasizes a different model than the original one. Table 3 shows summary statistics for the BICs of the models. It consists of the best model according to the BIC, the minimum, lower quartile, medium, upper quartile, and maximum values of the BIC.
Data set (b) consists of 1000 observations on five variables, $V = \{a, b, x, y, z\}$, where two of them, a, b, are discrete and three of them, x, y, z, are continuous. The discrete variables have three and two levels, and each of the continuous ones follows a normal distribution. Overall, we have a search space of 1024 different graphical models. Using exhaustive search we find the main effects model a, b, x, y, z as the one with the lowest BIC of 21341.96. It corresponds exactly to the model that was used to generate the data. Summary statistics for the BIC calculated for the different models are given in Table 4.
To investigate the influence of the different parameter values we proceed as follows. In a first

best model:	cf, be, bd, ac
BIC:	
minimum	73785.434
lower quartile	73906.856
median	74398.099
upper quartile	74703.379
maximum	74899.737

Table 3: Summary statistics for the BIC of data set (a)

best model:	a, b, x, y, z
BIC:	
minimum	21341.959
lower quartile	21393.390
median	21416.635
upper quartile	21447.030
maximum	21627.247

Table 4: Summary statistics for the BIC of data set (b)

step we look at the search dynamics of single runs: Here, we present the results of the simulations by tables, where important information about the best model found and the median BIC of the final population are listed. Besides, the trend of the average BIC curve for each iteration is analyzed. In a second step we perform 1000 runs on the parameter constellations which seem to be successful due to the result of the first step. Each run is based on a different seed for the random number generator to reflect the randomness of the algorithm. This step should give further knowledge about the reliability of the algorithm. We restrict ourselves to discussing the most important and interesting results without dwelling into details obtained from all variations of combining the involved parameters which is not expected to yield a deeper understanding of the GA.

Let us first consider data set (a): To get an idea of the performance of the algorithm, results for different parameter constellations are shown in Table 5 for $T = 200$ iterations.
As it can be seen, the algorithm finds the best model according to the BIC in all but one case, requiring only a small number of iterations. Only constellation iii) yields misleading results. This model differs from the best one in one extra edge only. Further, the median BIC of the final population is in any case smaller than the lower quartile when calculating the median BIC over all the possible models (cf. Table 3). This suggests that the final population may

no.	parameters			best model	it.	med. BIC
	N	p_c	p_m			
i)	15	0.8	0.1	be, bd, ac, cf (1)	23	73785.43
ii)	30	0.7	0.01	be, bd, ac, cf (1)	71	73785.43
iii)	30	0.8	0.01	bed, ac, cf (**43**)	62	73803.50
iv)	30	0.8	0.1	be, bd, ac, cf (1)	8	73794.28
v)	30	0.8	0.5	be, bd, ac, cf (1)	7	73801.06
vi)	30	0.9	0.01	be, bd, ac, cf (1)	50	73785.43
vii)	50	0.8	0.1	be, bd, ac, cf (1)	14	73792.73

Table 5: Results for data set (a) with $T = 200$ iterations. **Best model** lists the best model found by the algorithm together with its rank. The number of the iteration at which the algorithm first reached the best model is given by **it.** and the median BIC of the final population is given by **med. BIC**.

consist of rather well fitting models.
The impact of the mutation probability p_m can be clearly seen comparing the different curves of Figure 1. For each curve we use another mutation rate, namely $p_m = 0.01, p_m = 0.1$, and $p_m = 0.5$, whereas the other parameters remain unchanged. The lower the mutation probability the smoother the curve of the median, where $p_m = 0.5$ and $p_m = 0.1$ yield similar results. Since an unstable curve indicates that the mutation probability is too high we can preliminarily conclude that $p_m = 0.5$ and $p_m = 0.1$ may be too large.

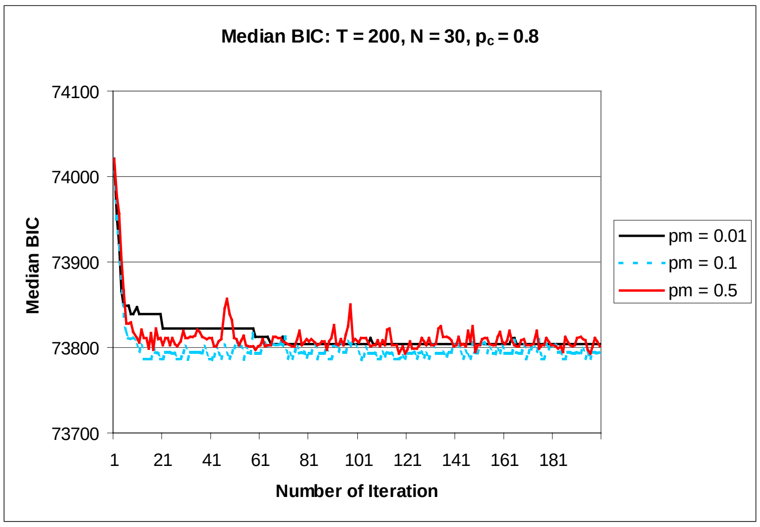

Figure 1: Median BIC of data set (a) for varying probability of mutation

Let us now investigate the effect of the crossover probability p_c. Contrasting the curves of

Figure 2, where we vary the parameter p_c from 0.7, 0.8, up to 0.9, we notice that these three plots hardly differ from each other, which means that varying the crossover probability in this range has no substantial impact on the performance of the algorithm.

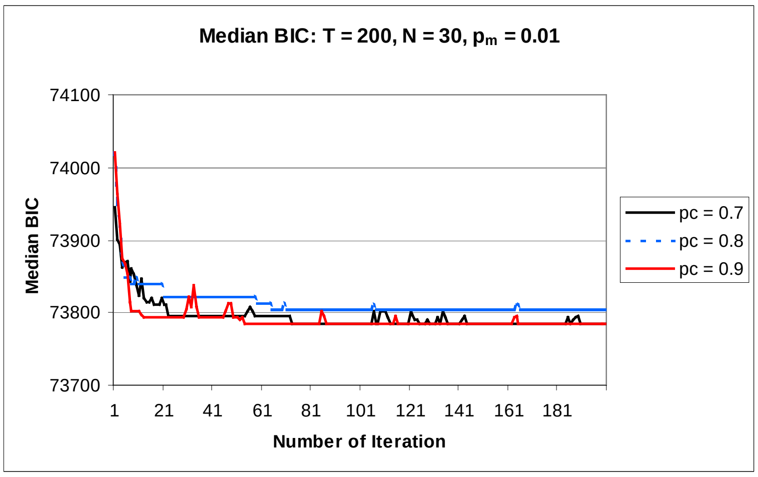

Figure 2: Median BIC of data set (a) for varying probability of crossover

The effect of different population sizes can be clearly seen from the curves of Figure 3. With a size of only $N = 15$ the curves are unstable and we have several very high peaks. In contrast, the curves for $N = 30$ and $N = 50$ are both rather flat and look quite similar. Due to these results and since a higher population size leads to a longer runtime of the algorithm it seems justified to consider a population size of 30 as sufficient to get stable results.

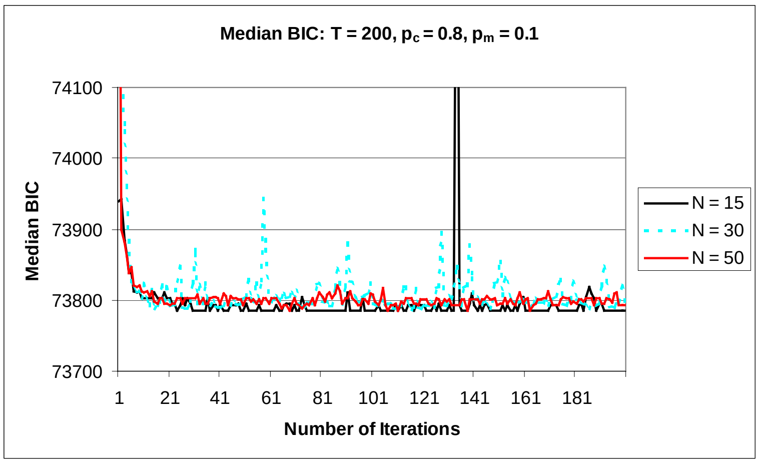

Figure 3: Median BIC of data set (a) for varying population sizes

We can easily analyze the behavior of the algorithm for different iteration numbers by looking at the curves of the different plots. Each figure shows that there is no essential progress in the average BICs from about 80 iterations on. Since the best model is always found after less than 100 iterations, an iteration size of 100 seems to be sufficient. No visible positive effect is obtained from increased iteration runs, and it should be emphasized that even with a very large number of iterations there is still some instability of the algorithm. The main reason for this is that mutation can result in poorly fitting models.
Summarizing the results from above we can state that in the given situation of six discrete variables, the constellations $N = 30$ or $N = 50$, $p_m = 0.01$ or $p_m = 0.1$, and $p_c = 0.8$ are most promising, where the choice of p_c as 0.8 is not mandatory. Besides, an iteration number $T = 100$ should be sufficient. Before analyzing the suggested parameter constellations further, we have to check whether the proper constellation may change if we have a different number of variables in our data set and another best fitting model. Therefore, we analyze now data set (b) consisting of five variables, where the main effects model is the best one.

As before, the results for different parameter constellations are shown in Table 6 for $T = 200$ iterations.

no.	**parameters**			**best model**	**it.**	**med. BIC**
	N	$\mathbf{p_c}$	$\mathbf{p_m}$			
i)	15	0.8	0.1	a, b, y, xz **(2)**	10	21348.34
ii)	30	0.7	0.01	a, b, x, y, z (1)	3	21341.96
iii)	30	0.8	0.01	a, b, x, y, z (1)	3	21341.96
iv)	30	0.8	0.1	a, b, x, y, z (1)	3	21363.40
v)	30	0.8	0.5	a, b, x, y, z (1)	5	21375.95
vi)	30	0.9	0.01	a, b, x, y, z (1)	3	21341.96
vii)	50	0.8	0.1	a, b, x, y, z (1)	2	21365.56

Table 6: Results for data set (b) with $T = 200$ iterations. **Best model** lists the best model found by the algorithm together with its rank. The number of the iteration at which the algorithm first reached the best model is given by **it.** and the median BIC of the final population is given by **med. BIC**. The last entry **av. no.** describes the average number of different models within the several populations during the run of the algorithm.

As in the previous case, the correct model is found most of the times, only in one case the second best model is found. Again, this model differs from the correct one in only one additional edge. If we look at the number of iterations needed to reach the best model we see that the best model is always found very quickly, even faster than in the case before where we had six variables, which may be due to the smaller search space. As in data set (a), the median BIC of the final population is smaller than the one of the whole population, which is again a hint that the models in the final population are quite well fitting.

Comparing the curves for the several parameter constellations yields rather analogous conclusions as in the discrete example, as for instance much smoother curves for $p_m = 0.01$ (cf. Figure 4). In addition, the Median BIC is also smaller for this particular mutation probability than for the other choices. The effect of different crossover probabilities (Figure 5) is the same as in data set (a), whereas a slight difference can be observed from the curves for different population sizes (Figure 6). Here, the curve for $N = 15$ is not such unstable as it was in case (a). Note that again a higher number of iterations does not lead to substantial improvements in the average BICs, where now even 20 iterations are sufficient.

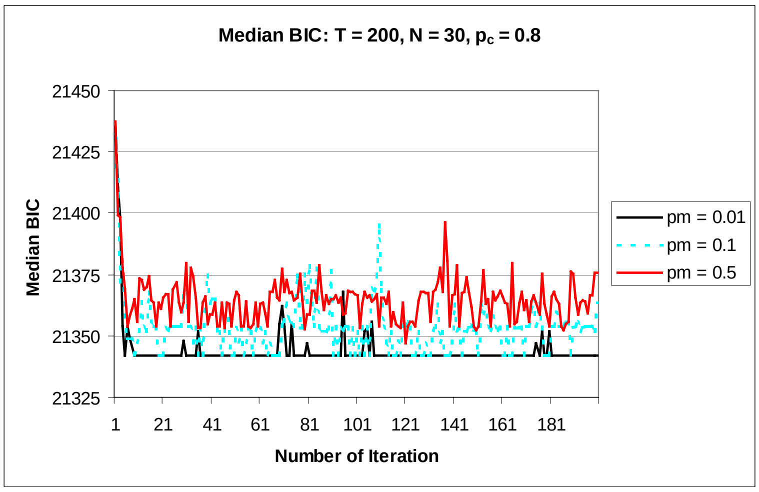

Figure 4: Median BIC of data set (b) for varying probability of mutation

Summarizing the results, we can conclude that the genetic algorithm applied here to five variables comes up with a similar behavior as in example (a) consisting of six variables. Again, combinations of $N = 30$ or $N = 50$, $p_m = 0.01$ or $p_m = 0.1$, and $p_c = 0.8$ seem to be successful.

Now we run 1000 simulations for some constellations on both data sets to gain further insights. The results are summarized in Table 7. The first number of each entry gives the percentage of times the best model according to the BIC is found out of 1000 simulations, and the value in brackets describes the percentage of times the next most frequent model results. It should be emphasized that the percentage of the latter is always much lower than that of the best model. Hence, we may say that the preference for the best model is rather high compared to other models in each of the parameter constellations. Comparing i) and ii) we see that in both data situations choosing $p_m = 0.01$ is much more promising than

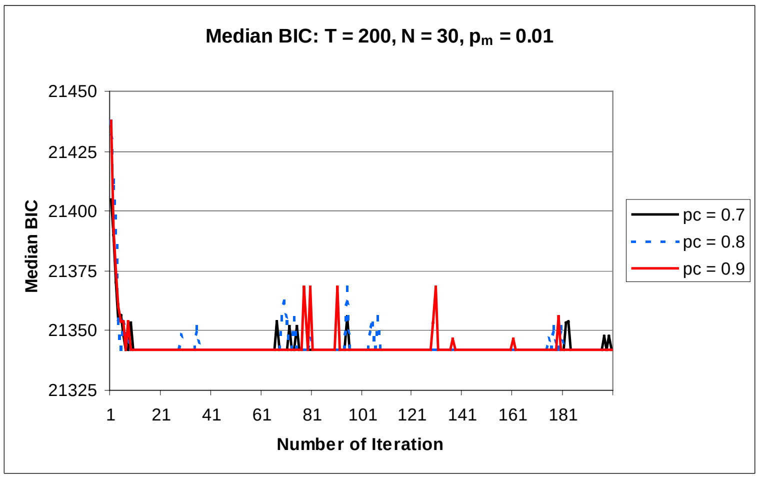

Figure 5: Median BIC of data set (b) for varying probability of crossover

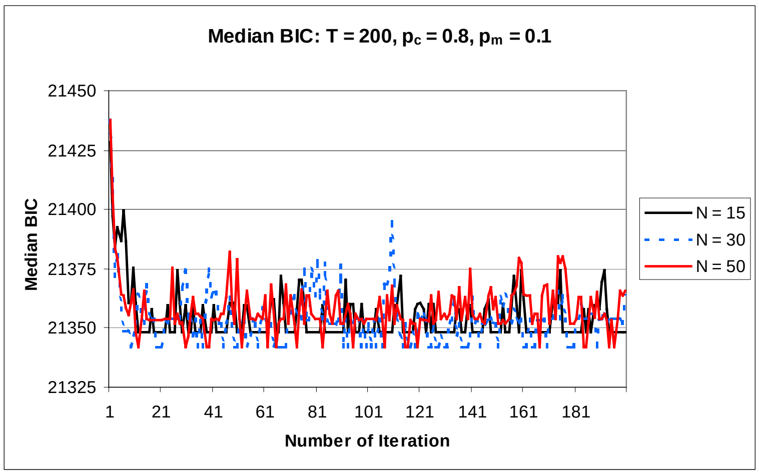

Figure 6: Median BIC of data set (b) for varying population sizes

$p_m = 0.1$: With $p_m = 0.1$ we obtain the best model in less than half of the simulation runs only for each data set, whereas with $p_m = 0.01$ we reach this model in 72.3% of the runs on data set (a) and even in 88.9% of the runs on data set (b). Based on these results we investigate whether using a greater population size leads to even more successful runs. The results for $N = 50$ are listed in entry iii). Here, the best model is found slightly more often in case (a) (74.8% instead of 72.3%) but in case (b) we obtain this model only in 80% of the runs instead of 88.9%. Since there are more variables in example (a) we may conclude that a larger population size for data sets consisting of a larger number of variables is indeed sensible. In contrast, choosing too large a population size for a problem may worsen the performance of the algorithm.

no.	**parameters**			**discrete data set**	**mixed data set**
	N	$\mathbf{p_c}$	$\mathbf{p_m}$	in %	in %
i)	30	0.8	0.01	72.3 (5.3)	**88.9 (1.9)**
ii)	30	0.8	0.1	35.9 (5.1)	49.7 (9.2)
iii)	50	0.8	0.01	**74.8 (3.3)**	80.0 (4.9)

Table 7: Results for 1000 runs on each data set

The following conclusions can be drawn from the simulations in this section: The parameter constellation turns out to be a crucial point. Our results suggest that the mutation rate should be $p_m = 0.01$, whereas the crossover rate and the iteration number of the algorithm seem to have only minor influence if chosen in a proper range (see above). To define a suitable population size N is difficult. It seems to depend on the number of variables in the data set. It would be necessary to perform more runs on simulated data sets, especially on ones with a larger number of variables to further study the behavior of the algorithm. In the next section, the performance of the genetic algorithm is compared to other search techniques.

5.2 Comparison with Other Strategies

Let us first report the comparison of the genetic algorithm with backward and forward selection. We perform the backward selection starting with the saturated model and the forward selection beginning with the main effects model. The criterion for deleting and adding edges, respectively, is the BIC instead of the commonly used deviance, so that we can easily compare the results with those of the genetic algorithm. The strategies lead to the models given in Table 8. The forward selection performed on data set (b) yields a wrong model including one extra edge az and being the sixth best model in terms of the BIC.

	data set (a)	**data set (b)**
backward sel.	be, bd, ac, cf (1)	a, b, x, y, z (1)
forward sel.	be, bd, ac, cf (1)	az, b, x, y **(6)**

Table 8: Results for the backward and forward selection

Though these results look rather promising, stepwise selection procedures have the drawback that a deletion or addition of an edge at some stage of the procedure implies that the corresponding variable will not be considered again at any step of the whole process. This may reduce the search space such that the part containing the true model will not be visited during the selection process. Thus, both the genetic algorithm and the stepwise procedures are quite convincing with respect to finding the true model in the simulation studies, but both can give wrong models depending on the choice of the available options of the algorithm. Regarding runtime, however, it is obvious that stepwise selection outperforms genetic algorithms, since in each iteration only one model has to be calculated as opposed to a whole set of models.

Let us now turn to the EH procedure, which leads to the minimal accepted models listed in Table 9.

data set (a)	**data set (b)**
$abcef, abcdf$	a, b, x, y, z
cf, bf, bde, ac	
$acdef, abcde$	
$acdef, abcef$	

Table 9: Minimal models accepted by the EH procedure

As it can be seen, four different models are obtained for data set (a), but none of them coincides with the model cf, be, bd, ac which is the best one when performing the genetic algorithm on the data set. This may be due to the fact that the EH procedure uses the likelihood ratio criterion instead of the BIC as a goodness of fit criterion. Another reason may be that the results of the EH procedure – like those of the stepwise selection – depend on the sequence of steps the algorithm has to perform (cf. Edwards and Havránek 1985). As a first conclusion we may say that the data itself do not emphasize a particular model and that the data set should be further analyzed in this respect. Case (b) is different: The only minimal accepted model found is the main effects model, which is also detected by the genetic algorithm. Thus, it seems as if here the data clearly favor this model.
Regarding runtime, the EH procedure has to be preferred to the genetic algorithm. In the two simulations, for example, the EH procedure needs 17 steps in the discrete case and only one step in the mixed case to terminate. In general, this model selection algorithm reduces the search space considerably to only a few models to be estimated (cf. Edwards and Havránek 1985, 1987) which saves a lot of time, although in the worst case all possible models may have to be analyzed.

The simulations show as a preliminary result that the genetic algorithm provides a reasonable alternative for model selection, but it suffers from its runtime which is much longer than that for the traditional algorithms: At each step of the algorithm a whole population of models has to be estimated. For a detailed analysis of the time complexity of the algorithm we refer to Blauth (2002), where also some enhancements to speed up the algorithm can be found.

6 Discussion

Besides the presented simulations it would be also interesting to investigate situations with more than six variables in the data set to get a deeper knowledge about the performance of the genetic algorithm. But in such situations, the best model cannot be found anymore by searching all models. Since the BIC does not always reflect the imputed model (cf. data set (a) of the simulation study) we cannot specify which model has to be found by the algorithm and thus we desisted from investigating such large models. To cope with this situation an overall measure for graphical models would be helpful.
One disadvantage of the GA seems to be that the best model according to a performance criterion is not always found. But there are some facts which defuse this problem: First, the final population consists at least of rather well fitting models, which is especially helpful when investigating larger data sets, because here it is not sensible anymore to speak of "the best" model. Second, the best model found by the GA mostly differs from the "real" best model in just a few edges. This may then be used as starting model of a backward or a forward selection to reach the best model. This would most of the time prevent the stepwise selection algorithm from running into local extrema.

Potential enhancements for the genetic algorithm can go in at least two directions. First, the algorithm should be also adapted to model search for DAGs and chain graphs. In both situations, the structure of the underlying graph has to be accounted for when choosing the proper operators like crossover or mutation and when thinking about keeping the runtime of the algorithm as low as possible. The second extension should aim at further accelerating the model search. First ideas to cope with this task are described in Blauth (2002), but more advanced strategies may be possible.
As conclusion, though genetic algorithms are not the panacea, they offer a promising alternative to other selection strategies when performing model search in graphical models.

References

Blauth, A. (2002). *Model Selection in Graphical Models with a special Focus on Genetic Algorithms.* Logos Verlag, Berlin.

Darwin, C. (1958). *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.* 6th ed., Penguins Books, New York.

Demming, W. & Stephan, F. (1940). On a least square adjustment of a sampled frequency table when the expected marginal totals are known. *The Annals of Mathematical Statistics* **11**, 427–444.

Edwards, D. (2000). *Introduction to Graphical Modelling.* 2nd ed., Springer–Verlag, New York.

Edwards, D. & Havránek, T. (1985). A fast procedure for model search in multidimensional contingency tables. *Biometrika* **72**, 339–351.

Edwards, D. & Havránek, T. (1987). A fast model selection procedure for large families of models. *Journal of the American Statistical Association* **82**, 205–213.

Frydenberg, M. & Edwards, D. (1989). A modified iterative scaling algorithm for estimation in regular exponential families. *Computational Statistics and Data Analysis* **8**, 142–153.

Laplante, P.A. (Ed.) (2000). *Dictionary of Computer Science, Engineering, and Technology.* CRC Press LLC, Boca Raton.

Lauritzen, S.L. (1996). *Graphical Models.* Clarendon Press, Oxford.

Lauritzen, S.L. & Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statistics* **17**, 31–57.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs* (3rd ed.). Springer, New York.

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks, Special Issue on Evolutionary Computation* **5**, 96–101.

Schwarz, G. (1978). Estimating the dimensions of a model. *The Annals of Statistics* **6**, 461–464.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics.* Wiley, Chichester.