Proceedings of the ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE 2017
August 6-9, 2017, Cleveland, Ohio, USA

DETC2017-67643

BAYESIAN NETWORK STRUCTURE OPTIMIZATION FOR IMPROVED DESIGN SPACE MAPPING FOR DESIGN EXPLORATION WITH MATERIALS DESIGN APPLICATIONS

Conner Sharpe
c_sharpe@utexas.edu
Mechanical Engineering Department
The University of Texas at Austin
Austin, Texas, USA

Clinton Morris
clint_morris@utexas.edu
Mechanical Engineering Department
The University of Texas at Austin
Austin, Texas, USA

Benjamin Goldsberry
goldsbe@arlut.utexas.edu
Mechanical Engineering Department
The University of Texas at Austin
Austin, Texas, USA

Carolyn Conner Seepersad
ccseepersad@mail.utexas.edu
Mechanical Engineering Department
The University of Texas at Austin
Austin, Texas, USA

Michael R. Haberman
haberman@arlut.utexas.edu
Mechanical Engineering Department
The University of Texas at Austin
Austin, Texas, USA

ABSTRACT

Modern design problems present both opportunities and challenges, including multifunctionality, high dimensionality, highly nonlinear multimodal responses, and multiple levels or scales. These factors are particularly important in materials design problems and make it difficult for traditional optimization algorithms to search the space effectively, and designer intuition is often insufficient in problems of this complexity. Efficient machine learning algorithms can map complex design spaces to help designers quickly identify promising regions of the design space. In particular, Bayesian network classifiers (BNCs) have been demonstrated as effective tools for top-down design of complex multilevel problems. The most common instantiations of BNCs assume that all design variables are independent. This assumption reduces computational cost, but can limit accuracy especially in engineering problems with interacting factors. The ability to learn representative network structures from data could provide accurate maps of the design space with limited computational expense. Population-based stochastic optimization techniques such as genetic algorithms (GAs) are ideal for optimizing networks because they accommodate discrete, combinatorial, and multimodal problems. Our approach utilizes GAs to identify optimal networks based on limited training sets so that future test points can be classified as accurately and efficiently as possible. This method is first tested on a common machine learning data set, and then demonstrated on a sample design problem of a composite material subjected to a planar sound wave.

INTRODUCTION

Advances in manufacturing processes and simulation capabilities have made it possible to design more complex materials and structures. These complex engineering system design problems are commonly broken down into subproblems, at which point the high-level challenge becomes managing the dependencies between the subproblems and identifying high-performance system-wide solutions. This is often a cumbersome process of iteratively adjusting candidate designs in pursuit of optimal performance. The expense of this optimization process grows with the problem's dimensionality, the degree of coupling between variables and subsystems, the nonlinearity of the underlying relationships, and the computational expense of the underlying simulation models.

As a means of addressing these complex systems design problems, design exploration techniques aim to explore the design space to identify sets of promising designs. Set-based design exploration methods, in particular, focus on identifying sets of satisfactory performance for each subproblem in a complex systems design problem instead of identifying unique optimal point-wise solutions. These sets of solutions can be intersected across subproblems to identify system-wide solutions with fewer total iterations compared to traditional point-wise optimization [1]. Figure 1 presents a simplified illustration of the set-based design paradigm.

A key challenge in the set-based approach is *mapping* the sets of promising solutions, especially for problems with complex, highly nonlinear design spaces. Interval-based

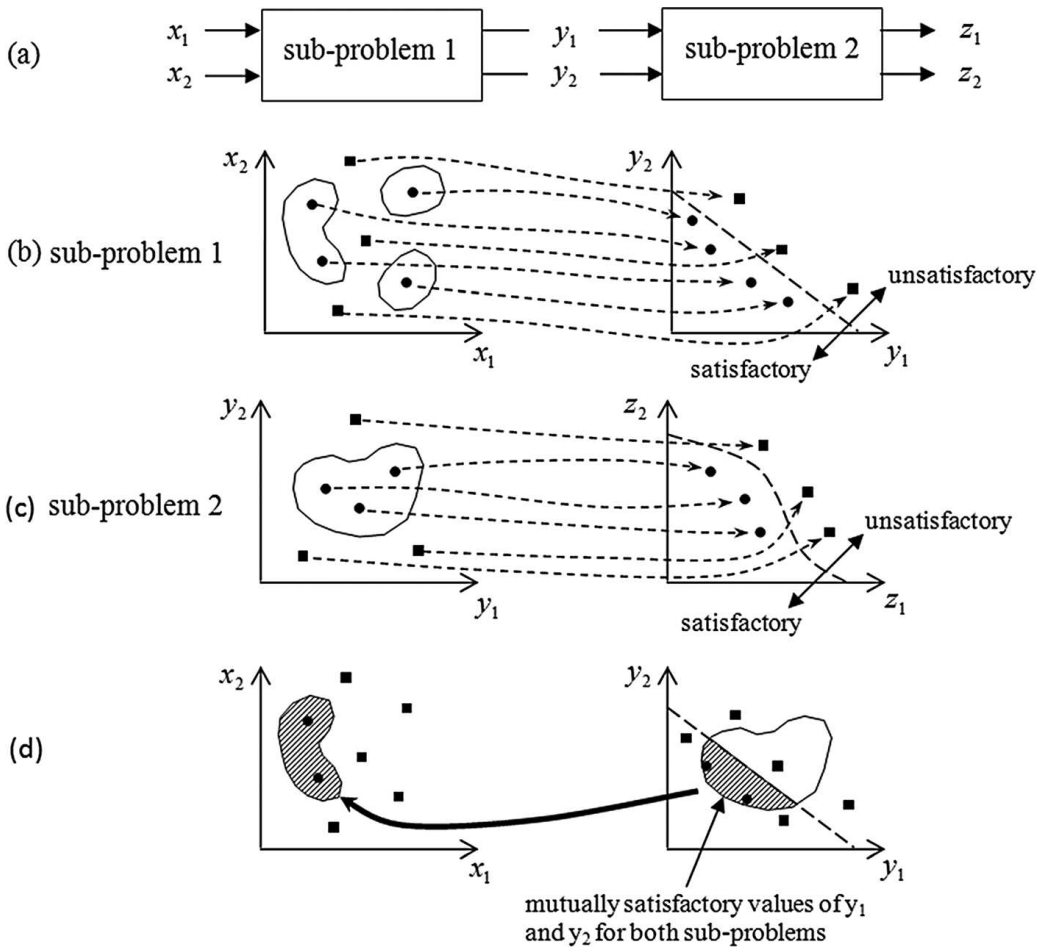
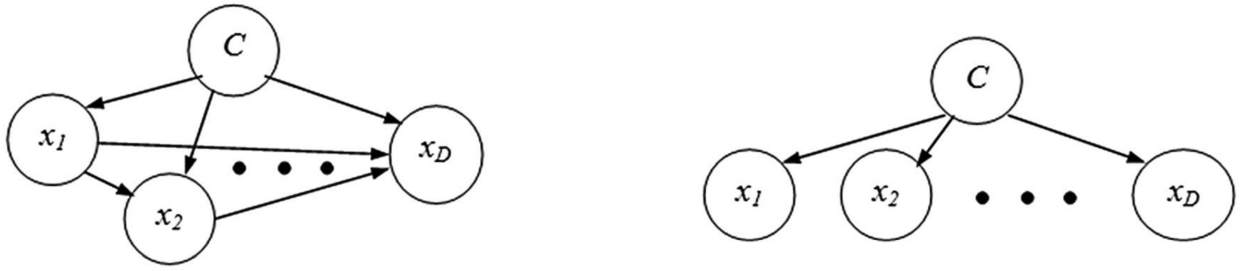


Figure 1. Mapping sets of mutually satisfactory designs for coupled subproblems.

techniques, which assign lower and upper bounds for each design variable [2], are simple to implement but limited to hyper-rectangular representations that can perform poorly for highly nonlinear problems. This challenge motivates the use of more flexible techniques that can map more complex design spaces and identify arbitrary sets of solutions. Kernel-based Bayesian network classifiers (BNCs) have been demonstrated to map regions of satisfactory performance more accurately [3,4], and they have been used to aid the search for solutions for complex multilevel materials design problems such as multilevel design of negative stiffness metamaterials and process-aware design of additively manufactured parts [5,6].

BNCs offer several advantages for design exploration. Bayesian statistics make it straightforward to incorporate new data along with prior knowledge for more efficient exploration. Additionally, Bayesian techniques are easily adapted to accommodate both continuous and discrete design variables, whereas many other machine learning techniques require homogeneous design spaces. Classifiers themselves are appropriate for non-unique mappings, which are common when mapping the inverse relationship between performance targets and design variables, whereas most surrogate models are more appropriate for unique forward mappings. Also, unlike discriminative classifiers, such as support vector machines, which have also been utilized in materials design applications [7] and output simply the "best" class, Bayesian classifiers are generative and utilize probabilistic algorithms to output the probability of class membership. Accordingly, they can provide confidence values associated with a classification and they can be straightforwardly integrated with optimization and adaptive search techniques to explore design spaces efficiently [8].

Identifying an appropriate network structure is a challenge for implementing BNCs, however. Typically, BNCs are based on naïve networks that encode no dependencies between design variables. This assumption is surprisingly robust, as naïve networks can often perform as well as, or even better than, more complex networks even in problems where there are variable dependencies. However, it has been shown that including appropriate dependence relationships between variables can increase classification accuracy (in our case accuracy of the design space maps) substantially in many cases [9,10]. Modeling too many dependencies, however, can needlessly increase computational expense because more complex network structures require more training points to provide accurate classification. It is therefore worthwhile to pursue methods to identify intermediate network structures that improve



$$p(\vec{x} \mid c_l) = p(x_D \mid x_{D-1}..., x_1, c_l)...p(x_2 \mid x_1, c_l)p(x_1 \mid c_l) \qquad p(\vec{x} \mid c_l) = p(x_D \mid c_l)...p(x_2 \mid c_l)p(x_1 \mid c_l)$$

Figure 2. Fully dependent (left) and fully independent (right) Bayesian networks.

classification performance and provide more accurate sets of promising solutions to designers even when the availability of training points is limited. In this paper, genetic algorithms are utilized to identify suitable network structures for BNCs, and the impact on classification accuracy is demonstrated for a sample machine learning problem and a materials design problem. The next section provides a more in-depth description of BNCs, followed by a description of the network optimization approach and then an application of the combined approach to test problems.

BAYESIAN NETWORK CLASSIFIERS

Bayesian networks are probabilistic directed acyclic graphical models (DAGs) used to represent a set of conditional dependence relationships that combine to form a joint probability distribution. The nodes of the graph represent random variables, and the directed arcs that connect the nodes in the graph represent dependencies between variables. The head of the arrow points to the child variable, while the tail of the arrow connects to the parent variable. In Bayesian analysis, a variable is assumed to be conditionally independent of all of its non-descendant variables given its parent variables. These graphs provide a convenient way of encoding the relationships between design variables for statistical analysis. BNCs utilize a class variable as a root node which serves as a parent variable to every analysis variable. Examples of fully-dependent and fully-independent (naïve) classifier networks can be seen in Figure 2.

BNCs utilize Bayes theorem to estimate the probability that a given data point of interest will fall in a particular class. In this context this probability is referred to as the posterior class probability, and is denoted as $p(c_l|\vec{x})$. In order to make this calculation one must first know the class-conditional probability of that data point $p(\vec{x}|c_l)$ as well as the prior probability of that class $p(c_l)$. These quantities are multiplied to form the joint probability that both the design point and the class will occur $p(c_l, \vec{x})$. The joint probability is then normalized by the probability of the data point of interest occurring at all, which is calculated as the sum of the joint probability distribution across all classes. This calculation can be seen in Equation 1.

$$\boldsymbol{p(c_l|\vec{x}) = \frac{p(\vec{x}|c_l)p(c_l)}{p(\vec{x})} = \frac{p(\vec{x}|c_l)p(c_l)}{\sum_{k=1}^{2} p(\vec{x}|c_k)p(c_k)}} \qquad (1)$$

In most settings it is impossible to know the necessary probability distributions *a priori*. In design problems the distributions are estimated by evaluating the performance of a set of candidate designs with known properties using forward models such as simulations or physical experiments. These training points are classified as high or low performance based on a performance threshold specified by the designer. These training points can then be used to estimate the posterior probability that any point of interest in the design space will fall into a given class without evaluating that point using the forward model. The prior probabilities of each class can be easily estimated by simply assigning probabilities according to the proportion of training points that fall into each class as a ratio of the total number of training points. The crux of this method lies in estimating the class-conditional probability of the relevant design point. In discrete design spaces the multinomial distribution can be estimated through use of a histogram. However, most design problems involve continuous variables. In these design spaces it is necessary to create distribution models that can extend the influence of discrete training points into the surrounding continuous space, which always incurs computational cost. One common method to construct these distributions is a technique known as kernel density estimation (KDE). In this method, a Gaussian kernel with tunable bandwidth is centered on each training point such that the influence of that point on the surrounding space is higher the closer you come to that point. The influence of all training points is then summed to create the overall map of the space.

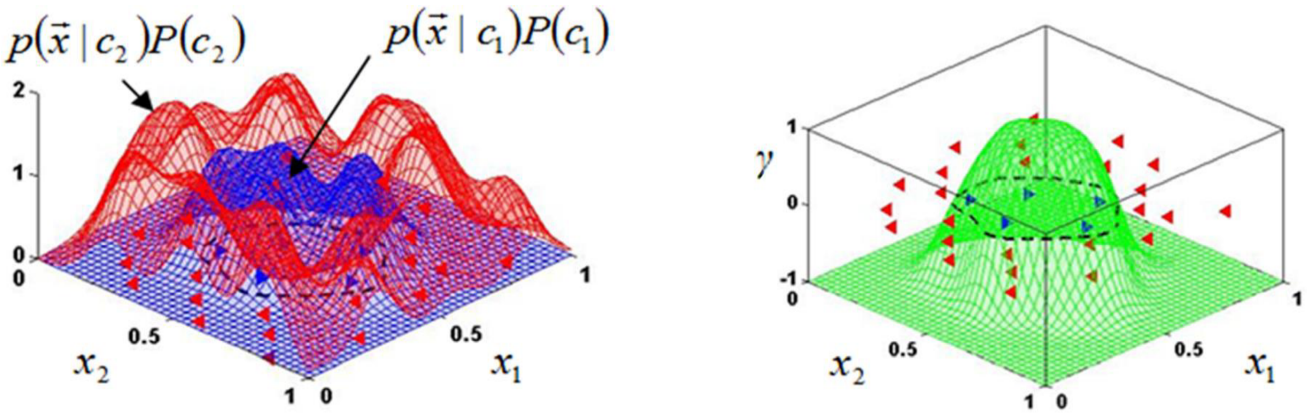


Figure 3. A set of points (designated as the blue and red triangles in the figure) are evaluated using a simulation model and classified using a performance threshold. The sets of high and low performance points are used as training points to construct separate kernel density estimates according to Equation 2 and scaled by the prior probabilities of the respective classes (left). The joint probability distributions are transformed using Equation 1 to obtain the posterior probabilities of class membership, which are compared according to Equation 5 to form a classification decision surface (right). The boundary of the satisfactory design region is encircled by the black dotted line.

The equation for determining the class conditional probability of a particular candidate design from the KDE is shown in Equation 2,

$$\boldsymbol{p(\vec{x}|c_l)} = \prod_{i=1}^{D} \boldsymbol{p(x_i|pa_i, c_l)}$$
$$= \prod_{i=1}^{D} \frac{\mathbf{1}}{\boldsymbol{N_l}} \sum_{j=1}^{N_l} \frac{\prod_{k=1}^{K} \boldsymbol{N_k^j(pa_{i,k}, \hat{x}_k^j, h_{k,l})}}{\sum_{r=1}^{N_l} \prod_{s=1}^{K} \boldsymbol{N_s^r(pa_i, \hat{x}_s^r, h_{s,l})}} \frac{\mathbf{1}}{\boldsymbol{h_{i,l}\sqrt{2\pi}}} \boldsymbol{e}^{\frac{(x_i - \hat{x}_i^j)^2}{2h_{i,l}^2}} \quad (2)$$

where $\vec{x}$ is the candidate design of interest, $\hat{x}^j$ are the training points, $h_{i,l}$ is the bandwidth of the Gaussian distribution for the i[th] dimension, pa_i is the set of parent variables for the i[th] dimension, N_l is the number of training points for class l, and D is the number of design variables. The bandwidth of the Gaussian kernels can have a significant impact on the classification accuracy [3,4,9]. Shahan et. al. established heuristics for setting bandwidth [3,11] in 2D spaces in the following form with $\sigma_{i,l}$ as the standard deviation of the normalized data for the i[th] dimension and α as a tunable parameter.

$$\boldsymbol{h_{i,l}} = \frac{\boldsymbol{\sigma_{i,l}\alpha}}{\sqrt{\boldsymbol{N_l}}} \quad (3)$$

This heuristic yielded sharp kernels which improved effectiveness in low-dimensional design spaces. However, increasing the dimensionality of the design space causes a much sparser covering of the space with the same number of training points. Smoother kernels with wider bandwidths spread the influence of each training point farther to accommodate for this effect. With this in mind the bandwidth heuristic is adapted to higher-dimensional spaces in the following form.

$$\boldsymbol{h_{i,l}} = \frac{\boldsymbol{\sigma_{i,l}\alpha}}{\sqrt[D]{\boldsymbol{N_l}}} \quad (4)$$

This heuristic increases the smoothing for each additional dimension, which combats the sparsity of larger design spaces. In this work the tuning parameter is set to $\alpha = 1.6$ by manual tuning of classification accuracies. However, methods for adaptively setting kernel bandwidth in higher-dimensional problems would be a valuable avenue for further research.

In design space classification tasks the KDE class-conditional likelihood calculation is completed for two different sets of training data, which have previously been classified as either satisfactory or unsatisfactory, depending on whether their predicted performance exceeds a threshold set by the designer. The posterior class probability can then be calculated for each class independently and compared for classification purposes according to Equation 5. The class with the higher probability is assigned to that test point, with points for which Equation 5 equals zero determining the decision boundary. A visual representation of a 2D design space map and classification decision surface can be seen in Figure 3.

$$\boldsymbol{Decide\ c_1\ if}: \quad (5)$$
$$\frac{\boldsymbol{p(\vec{x}|c_1)p(c_1)}}{\sum_{k=1}^{2} \boldsymbol{p(\vec{x}|c_k)p(c_k)}} - \frac{\boldsymbol{p(\vec{x}|c_2)p(c_2)}}{\sum_{k=1}^{2} \boldsymbol{p(\vec{x}|c_k)p(c_k)}} > \mathbf{0}$$

The central challenge of Bayesian classification, particularly in continuous design spaces, is to reduce computational cost. The KDE calculation is the main cost of Bayesian classification in continuous spaces. The cost of this process scales linearly with the number of parent variables involved in the analysis as can be seen in Equation 2 and clarified in Table 1.

Table 1. Time complexity of classification where N is the number of training points, D is the number of design variables, and L is the length of the longest chain of parent variables in the network.

Time Complexity		
Naïve	Intermediate	Fully-Dependent
O(N)	O(NL), 1<L<D	O(ND)

The easiest way to reduce computational cost is to reduce the number of parent variables, which is why naïve Bayes networks (networks with no parent variables) are very popular in many practices. However, removing parent variables that represent dependencies that are actually present in the system can increase the asymptotic error of the resulting classification [9,10]. It is clear from Equation 2 that the parent variables of each design variable could have a significant effect on the class conditional probability. The extent to which the elimination of relevant variable relationships affects accuracy depends on the system being evaluated. However, we hypothesize that engineering design problems may be particularly susceptible to this issue for two reasons. First, engineering design variables are commonly coupled in terms of their effect on performance, which yields more opportunities for variable dependencies to affect classification. Second, complex engineering systems often have forward models that are expensive to evaluate, with simulations taking anywhere from hours to days or even weeks to execute. In this context it is important to build a classifier that is as accurate as possible with limited training data. The goal is then to be able to identify and maintain important dependencies between variables while removing unnecessary variable connections to save time and resources and speed convergence.

Learning design variable network structures from data has the potential to provide even more value to the design process beyond increasing accuracy of design space mappings and time savings. These structures could reveal relationships between variables of which designers were previously unaware, causing them to reexamine the problem from a new perspective and enhancing innovation. The revelation of unintuitive variable dependencies through the network itself as well as surprising areas of high performance from the design space maps could combine to form a powerful tool for seeding new design ideas. In order for the designer to realize all of these advantages, an appropriate optimization method must be utilized which can learn appropriate network structures and yield improved mappings of the design space.

NETWORK STRUCTURE OPTIMIZATION

The optimization of Bayesian networks for classification accuracy is a difficult problem. The dependencies between variables that form the network structure must first be learned from the data itself, and then this structure must be used to map the space of interest and make predictions about future data. The learning of Bayesian network structure is a combinatorial problem, with binary entries used to represent the conditional dependence or independence between variables. The discrete nature of the problem presents a challenge for many optimization algorithms that depend on gradient information. Additionally, network performance can be highly multimodal, with small changes in network structure often creating large deviations in classification accuracy. Perhaps the greatest challenge is that the search space grows rapidly with the dimensionality of the network according to Equation 6.

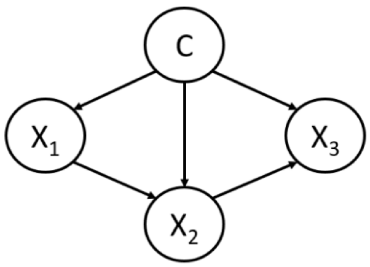
$$\boldsymbol{f(D)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{D}} (\boldsymbol{-1})^{\boldsymbol{i+1}} \binom{\boldsymbol{D}}{\boldsymbol{i}} \boldsymbol{2}^{\boldsymbol{i(D-i)}} \boldsymbol{f(D-i)} \qquad (6)$$

In light of these challenges for global optimization of network structures, local search methods or heuristic optimization techniques are commonly employed. Greedy hill climbing has been used to evaluate network structures, but this works best for sparse networks (networks with few parents) [12]. Evolutionary algorithms are well-suited to learn intermediate-density network structure from data [12,13]. There are many different styles of evolutionary algorithms, but they all share the same basic principles inspired by biological evolution. This work utilizes the well-established genetic algorithm (GA). The efficacy of GAs is attributed to the building block hypothesis, which states that strings of traits that lead to high performance are implicitly modelled and proliferated by the GA's probabilistic operators of selection, crossover and mutation [14].

Heuristic evolutionary techniques such as GAs have been used in the statistical community to optimize Bayesian network structure [12-13,15-16]. This work aims to bring these techniques into the design community for the first time by applying GAs to improve the accuracy of BNCs for design space mapping and exploration. The heuristic techniques fall under what is known as a score plus search method in which the GA is the search mechanism [13] and the score provides the fitness function for the GA. There are a number of established scoring metrics from statistics such as penalized maximum likelihood and minimum descriptive length, which both seek to find the maximum likelihood of the data with penalties for more complex models to avoid overfitting [10,13]. However, our approach is focused in particular on Bayesian network classifiers with the end goal of providing more accurate mappings of promising design spaces. In light of this goal, the fitness function is the classification accuracy yielded by each candidate network structure when tested on a large set of data reserved for validation.

The logical flow of the implementation of this method is illustrated in Figure 5. A critical component is forming suitable representations of the network structures for the operations required throughout the method. An example of how a candidate network structure is encoded in various forms can be seen in Figure 4. The parent matrix allows for convenient indexing in the process of structure generation to ensure variables remain in the appropriate ordering. The training structures hold the dimensional indices of the parent variables in the overall data set so that they can be accessed easily during the classification calculations. The GA population entries are reshaped into strings to allow for convenient crossover of selected network structures. Based on these representations, BNC calculations and GA operations can be performed efficiently. The crossover probability is set to 0.95 and the mutation probability is set to 0.05. These values encourage aggressive changes in the population for increased diversity in the search. The termination criteria is set as a maximum number of GA iterations based on time constraints and convergence rates which change with complexity of the network.

A topological ordering of the variables is enforced in this analysis to achieve some significant advantages. The design variables are ordered from one to nine, and parent variables are required to precede them in the order, e.g. variable 3 could be a parent of variable 4 but not vice versa. Topological ordering ensures there will be no cyclic networks that violate Bayesian criteria, and renders a much more tractable search space of possible networks. This assumption usually does not create vast differences in classification accuracy except in special cases [12], which may be explored in later work. The random mutations involved in GAs can often create cyclic networks that violate topological ordering. It is therefore necessary to implement a repair mechanism to ensure variables remain in their topological ordering. This mechanism is implemented by converting the population strings into the parent matrices and



$$Parent\ Matrix = \begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{matrix}$$

$$Training\ Structure = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}$$

$$Population\ Entry = 000100010$$

Figure 4. Representations of network structures used for different stages of GA calculations.

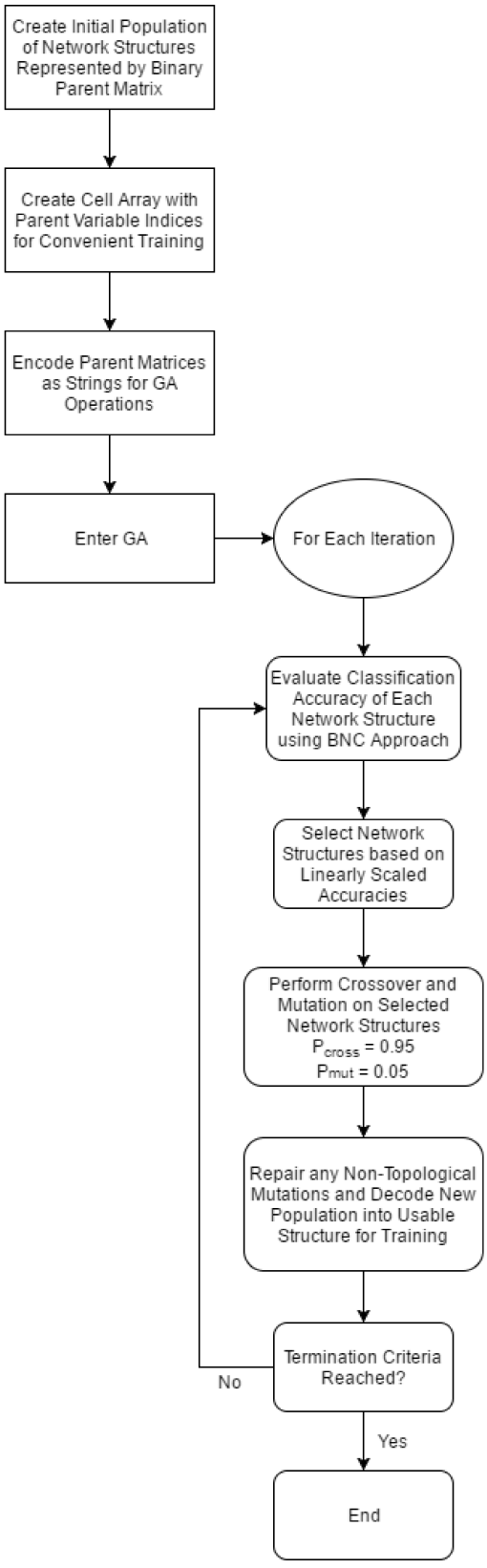


Figure 5. GA logic for optimizing classifier network structure.

enforcing all entries in the lower triangle of the parent matrices to be zeros.

PROBLEMS AND RESULTS

A simple instantiation of the genetic algorithm outlined in Figure 5 was used in this work. The method was first tested on a discrete data set in order to establish the efficacy of the method in this instance. The Monk's data set from the UC Irvine machine learning data repository [17] was chosen. The data set was six-dimensional and partitioned into 124 training points and 432 test points. A histogram was used to estimate the class-conditional likelihood of the data points. The initial network structures were generated randomly, with each possible parent variable having a 50% chance of being included in the network. No limitations on the maximum number of parent variables were set. A total of 20 candidate networks were included in the initial population, and the GA was terminated after 20 iterations. The GA was run five times to assess the consistency of classification accuracy improvement. Table 2 shows that the GA yielded significant improvements in accuracy. An example intermediate-density network identified by one of the runs is shown in Figure 6.

Table 2. Classification results for Monk's data set. The standard deviation across GA runs is displayed in parentheses.

Network Structure	Classification Accuracy % (σ)
Naïve	71.3
Fully-Dependent	77.3
Average GA Network	97.9 (0.49)

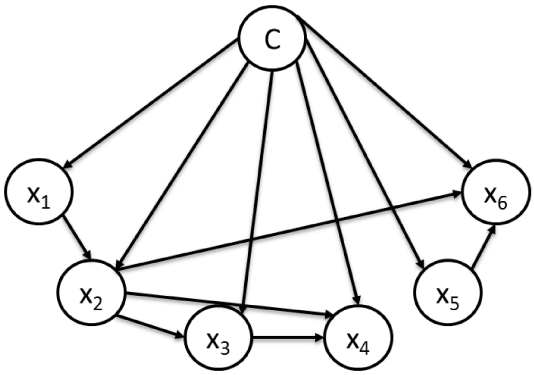


Figure 6. A sample network structure identified by the GA for the Monk's problem.

This problem established that a GA search of network structure with classification accuracy as the fitness function could drastically increase accuracy in some problem domains. The next step was to test this in a continuous design problem domain using the kernel-based BNC technique discussed previously in detail. An example problem of intermediate dimensionality was needed such that the search space of possible networks would be large enough to make traditional search impractical but modest enough to enable fine-tuning of the GA search algorithm as needed. In light of this goal, the problem chosen was the design of a 3-layer composite material submerged in water subjected to a planar sound wave. The goal of this materials design problem was to minimize the sound reflected back to the source. These types of acoustic damping materials could find use in sonar applications.

As shown in Figure 7 and Table 3, each layer of the composite material was defined by a distinct density, ρ, sound speed, c, and thickness, t. These design variables determined the performance of the composite. The performance objective was to obtain a reflection coefficient, R, of at least -6 dB. The reflection coefficient is defined as the ratio of the amplitude of the reflected wave to the amplitude of the source wave, which is equivalent to the difference between these amplitudes in the log space of decibels. The source of the planar sound wave was set to a driving frequency, f, of 1000 Hz. The water on either side of the composite was defined by its density, $\rho_0 = 1000$ kg/m^3, and sound speed, $c_0 = 1500$ m/s.

Table 3. Problem formulation for design of acoustic damping composite material.

Constants	$\rho_0 = 1000$ kg/m^3 $c_0 = 1500$ m/s $f = 1000$ Hz
Design Variables	ρ_1 , c_1 , t_1 ρ_2 , c_2 , t_2 ρ_3 , c_3 , t_3
Variable Bounds	800 kg/m^3 $< \rho <$ 2000 kg/m^3 100 m/s $< c <$ 2000 m/s 0.01 m $< t <$ 0.1 m
Performance Threshold	$R = P_R/P_S \leq$ -6 dB

As with the previous example, the GA method outlined in Figure 5 was used to identify the best network for the BNC. The difference in this instantiation, apart from using KDEs rather than histograms to estimate class-conditional probabilities, was in the initialization of the population. Due to the larger search space of possible networks, the population was initialized in a different way. Truly random initial structures in a problem of this dimensionality would have made it very difficult to find relatively sparse network structures. Therefore, varying levels of network complexity were included in the initial population. The initial population was generated by iteratively adding parent variables to a particular variable. The number of iterations was set equal to the maximum number of available parents. A new parent variable was added each iteration with a probability of 50%; if a new parent variable was added, that parent was chosen randomly from the available parents (any variable before the current variable in the ordering). This process was completed for 3 sets of 8 candidate network structures with the maximum number of parent variables limited to 2, 4, and 6 variables for each set, respectively. In addition, 14 instances of truly random structures were constructed through the same process as the discrete case. Finally, a naïve and a fully dependent network structure were included in the initial population to ensure that these networks, which could have been implemented a priori,

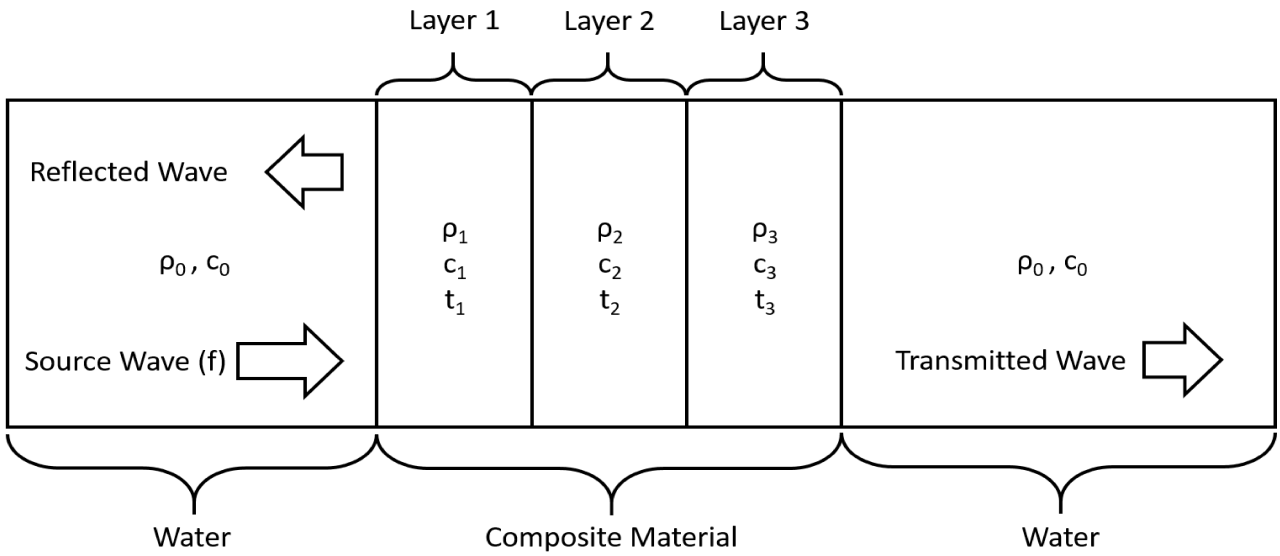


Figure 7. Problem schematic for design of acoustic damping composite material.

were not excluded from the search. A total of 40 candidate networks were included in the initial population.

The forward model of this problem evaluated candidate designs quickly, so classification accuracy in this instance was determined by using a set number of training points to train the KDEs and then validating the classifiers on a large set of test points. The Halton sequence was used to generate a large pool of design points. The training points were selected as the first 1000 points of each class in the sequence, yielding 2000 total training points. A sample of 2000 points from later in the sequence was used for validation.

Once the GA and BNC parameters were fully defined, the GA was executed 5 times to assess how consistently the GA improved classification accuracy. The average classification accuracy of the best network identified by each GA trial is compared against the naïve and fully-dependent networks in Table 4.

Table 4. Classification results for acoustic composite problem. Standard deviation across GA trials is displayed in parentheses.

Network Structure	Classification Accuracy % (σ)
Naïve	81.9
Fully-Dependent	83.1
Average GA Network	88.3 (0.18)

Table 5 documents a representative network structure found by the GA. This network demonstrated the best classification accuracy identified across all 5 GA trials.

Table 5. Best performing composite material network structure identified across all GA runs.

Variable	Parent Variables
ρ_1	
c_1	ρ_1
t_1	ρ_1 , c_1
ρ_2	ρ_1
c_2	c_1 , t_1 , ρ_2
t_2	c_1 , c_2
ρ_3	ρ_1 , t_1 , c_2
c_3	c_1 , c_2 , ρ_3
t_3	ρ_1 , ρ_2 , t_2 , ρ_3 , c_3

There were a number of commonalities between the networks identified by the different GA trials. Parent variables that appeared in at least 3 of the GA identified networks are shown in Table 6.

Table 6. Variable dependencies frequently identified in separate GA trials. The commonality implies that these dependency relationships are significant in the system.

Variable	Parent Variables
ρ_1	
c_1	ρ_1
t_1	ρ_1 , c_1
ρ_2	t_1
c_2	c_1 , ρ_2
t_2	c_1 , c_2
ρ_3	ρ_1 , c_2
c_3	c_1 , ρ_2 , c_2 , ρ_3
t_3	c_2 , c_3

The amount of training data to include is an important variable in this method because sparse training data can limit the effectiveness of especially complex network structures. Also, in higher-dimensional spaces more training points are needed to train accurate classifiers. To investigate the effect of the size of the training set, the accuracy of the GA-optimized networks was compared to the performance of a naïve (fully disconnected) and a fully connected network for a range of training points. The Halton sequence was again used to generate a large pool of design points. The training points were selected as the first X points of each class in the sequence with X ranging from 100 to 2000 in steps of 100. The same validation set of 2000 points from later in the sequence was used for validation. Figure 8 shows that for all training set sizes, the GA-optimized networks provide higher classification accuracy than either the naïve Bayes or the fully dependent network. The improvement is even more significant for training sets of more than 1500 points.

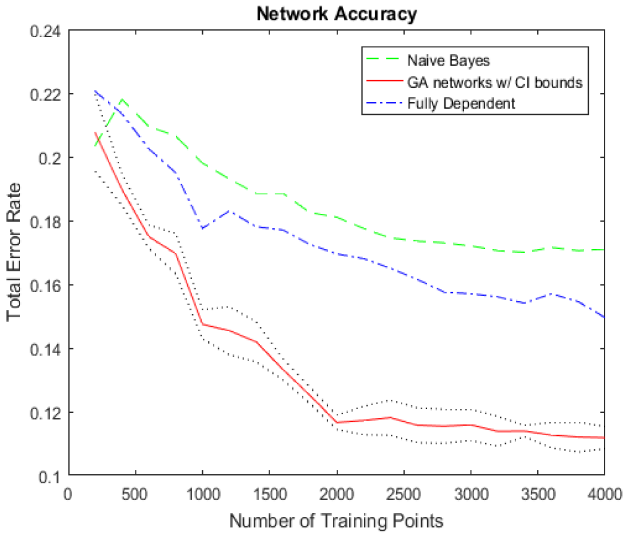


Figure 8. Convergence of classification accuracies as a function of the number of training points. The red line represents the average accuracy across GA networks, with the black dotted lines showing 95% confidence interval bounds.

Classification accuracies have largely converged for all three types of networks by approximately 2000 training points, with further increases in the size of the training set yielding minimal improvements in classification accuracy. This validates the choice of 2000 training points used in this work for network optimization. The rate of convergence is likely to change, however, for problems of different dimensionality or interrelationships among variables. Including more training data generally allows all networks, especially more complex networks, to classify additional points more accurately; however, computational expense and time constraints often limit the availability of training data. In this case, Figure 8 shows that if the simulation were more expensive and less data were available, utilizing GAs to find intermediate networks would still be the best strategy as the intermediate networks still outperform the a priori (naïve Bayes and fully-dependent) networks. Less training data would likely cause convergence to less dense networks as they converge more quickly in general.

DISCUSSION

The classifier network optimization was very successful in the case of the discrete Monk's data set. Improvements seen through network optimization in the continuous design problem were less dramatic but still significant. This may be due to the difference in how class conditional probabilities were calculated; the adjustment of Gaussian kernel products may not yield the same drastic effect as adding additional histogram bins to more accurately represent a multinomial distribution. Additionally, the differences may be due to the problems themselves. The dependencies in the Monk's problem may be more dependent on class than those of the acoustic composite problem, thus yielding a greater improvement in classification. This is supported by the fact that the naïve network has a stronger classification accuracy in the acoustic composite problem than the Monk's problem. As discussed previously, the effects of dependencies on classification accuracy can vary widely based on the type of dependency and the interactions between different groups of dependencies. Network optimization for classification accuracy does not necessarily identify the strongest dependencies, but those that affect classification most significantly. This is an important distinction, as certain types of dependencies have stronger influences on classification results. Variable dependencies that have a concentrated influence on one class of performance affect classification accuracy more significantly than dependencies that have a relatively consistent influence regardless of the class of performance. Additionally, particular network dependencies that favor one class over another can be cancelled out by the presence of other dependencies which favor the opposite class just as strongly [18]. Some of these mitigating factors are likely to be present in the composite material problem, as including the dependencies in the problem improves accuracy by a modest amount of about 7%. However, this jump in accuracy is still significant and would certainly improve the utility of these design mappings in a multilevel problem.

Inspection of the network structures revealed by the various iterations of the GA yielded significant commonality, increasing the likelihood that these dependencies are important to performance. The composite material appears to have strong interaction between the densities and sound speeds, with dependencies on thicknesses appearing far less often. This outcome matches physical intuition in this problem because the acoustic impedance of each layer is governed by the product of these properties. For the reasons cited above, it does not mean that these are the only significant dependencies in the system; however, the dependencies identified repeatedly by the GA do provide insight into interactions that effect design performance and may be exploited for improved designs. In this example, designers could utilize the interaction between densities and sound speeds to ensure the ratio of these properties for each layer yields a desirable sequence of impedances.

Overall, the GA consistently resulted in a classification accuracy superior to that of both the naïve and the fully-dependent networks in both problem paradigms. This method appears to be a promising avenue for improving the accuracy of design space mappings when limited training data is available.

FUTURE WORK

There are a number of promising avenues for future research in this area. For example, estimation of distribution algorithms (EDAs) are another branch of evolutionary algorithms that have been demonstrated to identify network structures more robustly than GAs in some cases. EDAs form explicit probabilistic models over the chosen representation of traits held by good solutions and sample this model for each subsequent generation [19]. This formal modelling of traits can capture interactions directly and lead to improved performance in some cases, but comes at a cost of increased computational expense [13,19]. It would be enlightening to explore these tradeoffs and compare the efficacy of EDAs to traditional GAs for design exploration and design space mapping applications.

A potential limitation of this work is that an ordering of the variables was assumed. In future work adapting the method to search various variable orderings in addition to variable dependencies could provide performance improvements. Furthermore, the nature of Bayesian networks allows prior information to be easily encoded before the optimization process begins. Utilizing expert knowledge to initialize the optimization with expected significant dependencies instead of random network structures could speed convergence and improve the likelihood that the GA finds optimal network structures.

Another challenge is to apply this approach to engineering systems design problems with much larger numbers of variables. As the number of variables increases, the computational complexity of the KDE method to estimate class-conditional probabilities can become prohibitive. One option to mitigate this difficulty could be to utilize Gaussian mixture models, which operate in a very similar manner to KDEs but use a much smaller set of Gaussian kernels, thus improving computational time. A second option would be to utilize proper orthogonal decomposition techniques to reduce the dimensionality of these problems by eliminating less significant design variables where appropriate. These approaches must be explored to scale this method to even more complex materials and systems design problems.

ACKNOWLEDGMENTS

The authors would like to acknowledge support from the National Science Foundation under Grant No. CMMI -1435548. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor. The authors would also like to acknowledge the support of The University of Texas at Austin Cockrell School of Engineering through the Engineering Doctoral Fellowship.

REFERENCES

[1] Sobek, D.K., Ward, A.C., Liker, J.K., 1999, “Toyota’s Principles of Set-Based Concurrent Engineering,” *Sloan Management Review*, **40**(2), pp. 67-84.

[2] Panchal, J.H., Fernandez, M.G., Christiaan, J.J., Paredis, J.K., Mistree, A., Mistree, F., 2007, “An Interval-Based Constraint Satisfaction (IBCS) Method for Decentralized, Collaborative Multifunctional Design,” *Concurrent Engineering*, **15**(3), pp. 309.

[3] Shahan, D.W., Seepersad, C.C., 2012, “Bayesian Network Classifiers for Set-Based Collaborative Design,” *Journal of Mechanical Design*, **134**(7), pp.1 14.

[4] John, G., Langley, P., 1995, “Estimating Continuous Distributions in Bayesian Classifiers,” *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers Inc,pp. 338-345.

[5] Matthews, J., Klatt, T., Seepersad, C.C., Haberman, M.R., Shahan, D.W., 2013, “Hierarchical Design of Composite Materials with Negative Stiffness Inclusions Using a Bayesian Network Classifier,” DETC2013-13128, *ASME IDETC Design Automation Conference*, Portland, Oregon.

[6] Rosen, D., 2015, “A Set-Based Design Method for Material-Geometry Structures by Design Space Mapping,” DETC 2015-4676, *ASME IDETC Design Automation Conference*, Boston, MA.

[7] Galvan, E., Malak, R.J., Gibbons, S., Arroyave, R., 2017, “A Constraint Satisfaction Algorithm for the Generalized Inverse Phase Stability Problem,” *Journal of Mechanical Design*, **139**(1), pp. 011401.

[8] Backlund, P., Shahan D.W., Seepersad C.C., 2015, “Classifier-guided Sampling for Discrete Variable, Discontinuous Design Space Exploration: Convergence and Computational Performance,” *Engineering Optimization*, **47**(5), pp. 579-600.

[9] Perez, A., Larranaga, P., Inza, I., 2009, “Bayesian Classifiers Based on Kernel Density Estimation: Flexible Classifiers,” *International Journal of Approximate Reasoning*, **50**(2), pp. 341-362.

[10] Friedman, N., Geiger, D., Goldszmidt, M., 1997, “Bayesian Network Classifiers,” *Machine Learning*, **29**(2-3), pp.131-163.

[11] Shahan, D., 2010, “Bayesian Network Classifiers for Set-based Collaborative Design”, PhD Dissertation, The University of Texas at Austin, Austin.

[12] Murphy, Kevin P., 2012, *Machine Learning A Probabilistic Perspective*, MIT Press, Cambridge, MA.

[13] Larranaga, P., Karshenas, H., Bielza, C., Santana, R., 2013, “A Review on Evolutionary Algorithms in Bayesian Network Learning and Inference Tasks,” *Information Sciences*, **233**, pp.109-125.

[14] Goldberg, D.E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

[15] Larranaga, P., Poza, M., Yurramendi, Y., Murga, R.H., Kuijpers, C.M.H., 1996, "Structure Learning of Bayesian Networks by Genetic Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(9), pp. 912-926.

[16] Heckerman, D., Geiger, D., Chickering, D., 1995, "Learning Bayesian Networks: the Combination of Knowledge and Statistical Data," *Machine Learning* **20**(3), pp. 197-243.

[17] Lichman, M., 2013, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, CA: University of California, School of Information and Computer Science.

[18] Zhang, H., 2004, "The Optimality of Naïve Bayes," *Proceeding of the Seventeenth Internatinal Florida Artificial Intelligence Research Society Conference*, Miami Beach, AIAA Press.

[19] Larranaga, P., Lozana, J., 2001, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers, Norwell, MA.