Bayesian Inference in Estimation of Distribution Algorithms

(Corrected Version of CEC'07 paper: September 2008)

Marcus Gallagher, Ian Wood, Jonathan Keith and George Sofronov

***Abstract*— Metaheuristics such as Estimation of Distribution Algorithms and the Cross-Entropy method use probabilistic modelling and inference to generate candidate solutions in optimization problems. The model fitting task in this class of algorithms has largely been carried out to date based on maximum likelihood. An alternative approach that is prevalent in statistics and machine learning is to use Bayesian inference. In this paper, we provide a framework for the application of Bayesian inference techniques in probabilistic model-based optimization. Based on this framework, a simple continuous Bayesian Estimation of Distribution Algorithm is described. We evaluate and compare this algorithm experimentally with its maximum likelihood equivalent, UMDA_c^G.**

I. Introduction

Estimation of Distribution Algorithms (EDAs) [1], [2], [3] construct a probability distribution $p(\mathbf{x})$ over the search space $\mathcal{X}$ of an optimization problem and adaptively learn this model to drive the search process. Generally speaking, a probability density function is used to generate a sample of candidate solutions at each iteration of the algorithm. This sample is evaluated with respect to the objective function of the problem. A subset of the best points in the sample are selected and used to modify the search distribution so that (with higher probability) improved solutions are produced by sampling from the distribution in next generation. In the Cross-entropy method [4], selected points are used to modify the search distribution so as to decrease the Kullback-Leibler divergence from a degenerate distribution over the (estimated) optimal solution to the search distribution. EDAs can also be viewed in terms of a stochastic minimization process on the K-L divergence [5]. EDAs and the Cross-Entropy method are closely related and several instances of EDAs can be seen as equivalent to the Cross-Entropy method, depending on the choice of density function and implementation details of the algorithms.

The model fitting task within each iteration of an EDA is typically carried out by a maximum likelihood estimation procedure. An alternative statistical framework is provided by Bayesian inference. In recent years, Bayesian techniques have become increasingly widely used in the fields of machine learning and statistics. Surprisingly however, Bayesian inference has so far received very little attention in the EDA literature.

Marcus Gallagher is with the School of Information Technology and Electrical Engineering, University of Queensland 4072, Australia (email: marcusg@itee.uq.edu.au). Ian Wood is with the School of Mathematical Sciences, Queensland University of Technology, GPO Box 2434 Brisbane QLD 4001 Australia (email: wood@itee.uq.edu.au). Jonathan Keith is with the School of Mathematical Sciences, Queensland University of Technology, GPO Box 2434 Brisbane QLD 4001 Australia (email: j.keith@qut.edu.au). George Sofronov is with the Department of Mathematics, University of Queensland 4072, Australia (email:georges@maths.uq.edu.au).

Often, the family and structure of the model used in an EDA is fixed (e.g. a set of Bernoulli distributions to generate the bitstrings in the PBIL algorithm [6], or a factorized Gaussian distribution over a continuous search space in the UMDA_c algorithm [2]). For EDAs that use probabilistic graphical models, search is often performed to determine the model configuration. For example, the "Bayesian Optimization Algorithm" (BOA) [7] is an EDA that uses a Bayesian network as its density model. The structure of the network model is typically found using a greedy search over a suitable metric. While some of these metrics are derived from Bayesian modelling assumptions (e.g the BDe and BGe metrics [2]), their use in EDAs is quite different from performing Bayesian inference. Bayesian network parameters in EDAs are typically estimated from the data (the selected best points from the sample) using a maximum likelihood approach. Hence (despite the implications of its name), BOA does *not* involve Bayesian inference in its modelling process.

In this paper, we develop a framework for the application of Bayesian inference techniques for model fitting in EDAs (BayEDAs). Based on this framework, a simple continuous Bayesian Estimation of Distribution Algorithm is described. We evaluate and compare this algorithm experimentally with its maximum likelihood equivalent, UMDA_c^G.

The general idea of applying Bayesian inference in the context of EDAs has to some extent been considered (see [8] and the references therein). Zhang describes the notion of Bayesian inference in a canonical algorithm, and gives an example of using this idea by considering a prior based on the Boltzmann distribution for model parameters. Zhang discusses possibilities for the implementation of various modelling steps in this canonical algorithm but no specific algorithms are implemented or experimentally evaluated. An example is subsequently presented using a Helmholtz machine as the density model.

In contrast to this previous work, we isolate the idea of using Bayesian inference in model-based optimization. It is then possible to derive specific Bayesian algorithms based on probability density functions commonly used in the EDA literature which can be directly applied in various optimization settings. BayEDAs therefore have a direct connection with maximum-likelihood based EDAs and the cross-entropy method.

A brief outline of the paper is as follows. In Section II we develop a framework for the application of Bayesian inference in EDAs. Section III presents an instantiation of this framework with a continuous EDA based on a univariate Gaussian density model. Experimental results are reported in Section IV to provide insight into the behaviour of the

TABLE I

GENERAL PSEUDOCODE FRAMEWORK FOR AN EDA.

```
Given: population size $M$, selection parameter $\tau$
BEGIN (set $t = 0$) Generate $M$ individuals at random
REPEAT for $t = 1, 2, \ldots$ until stopping criterion is met
    Select $M_{sel} < M$ individuals via truncation selection
    Maximise the likelihood $p(D|\theta)$ to obtain a point estimate $\hat{\theta}$
    Sample $M$ individuals from $p_t(\mathbf{x}|\hat{\theta})$
    $t = t + 1$
ENDREPEAT
```

TABLE II

GENERAL PSEUDOCODE FRAMEWORK FOR A BAYESIAN EDA.

```
Given: population size $M$, selection parameter $\tau$
BEGIN (set $t = 0$) Generate $M$ individuals at random
REPEAT for $t = 1, 2, \ldots$ until stopping criterion is met
    Select $M_{sel} < M$ individuals via truncation selection
    Calculate the model posterior $p_t(\theta|D)$
    Sample $M$ individuals from $p_t(\mathbf{x}|D)$
    $t = t + 1$
ENDREPEAT
```

algorithm, demonstrate it's feasibility and compare it with UMDA$_c^G$. Some related work concerning Bayesian inference and optimization is discussed in Section V. Finally, Section VI presents some conclusions and outlines directions of future work.

II. MAXIMUM LIKELIHOOD AND BAYESIAN MODEL FITTING IN EDAS

A. Framework

Consider the optimization problem

$$\min(f(\mathbf{x})), \mathbf{x} \in \mathcal{X}$$

where $f(\mathbf{x})$ is the fitness or objective function, $\mathbf{x}$ is an individual solution point and $\mathcal{X}$ is the feasible search space. EDAs build a probabilistic model $p_t()$ over $\mathcal{X}$ at each generation t of the algorithm based on selected individuals. Pseudocode for an EDA is shown in Table I.

The probabilistic model in Table I is specified by a vector of parameters θ. The parameters in an EDA model are typically estimated as the maximum likelihood values

$$\hat{\theta} = \arg\max_{\theta} p(\{\mathbf{x}^1, ..., \mathbf{x}^{M_{sel}}\}|\theta)$$

An alternative to model fitting is provided by Bayesian Statistics. In the Bayesian framework, a prior distribution $p(\theta)$ is specified, reflecting our belief about the model parameters before seeing any data. Once a set of data $D = \{\mathbf{x}^1, \ldots, \mathbf{x}^{M_{Sel}}\}$ is observed, we update our belief using Bayes rule

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} \tag{1}$$

where $p(\theta|D)$ is the posterior distribution over the model parameters. The posterior predictive distribution for a future data point $\mathbf{x}$ is then obtained by integrating over the model parameters:

$$\begin{aligned} p(\mathbf{x}|D) &= \int p(\mathbf{x}, \theta|D)d\theta \\ &= \int p(\mathbf{x}|\theta, D)p(\theta|D)d\theta \\ &= \int p(\mathbf{x}|\theta)p(\theta|D)d\theta \\ &\propto \int p(\mathbf{x}|\theta)p(D|\theta)p(\theta)d\theta \end{aligned}$$

As mentioned above, in EDAs we need to be able to sample from $p_t(\mathbf{x}|D)$ to generate the population at generation $t+1$. A standard technique in Bayesian data analysis to do this is to sample from the joint distribution $p(\mathbf{x}, \theta|D)$, giving predicted observations $(\mathbf{x}^1, \theta^1), \ldots, (\mathbf{x}^M, \theta^M)$. Discarding the sample parameter vectors leaves us with a sample $x^1, \ldots, x^M$ drawn from the marginal distribution $p(\mathbf{x}|D)$. The joint distribution can sometimes be sampled from directly and otherwise can be sampled via Markov Chain Monte Carlo sampling of the unnormalised distribution $p(\mathbf{x}|\theta)p(D|\theta)p(\theta)$. General pseudocode for a Bayesian EDA is shown in Table II.

It is evident that an implementation of this Bayesian EDA framework will involve making choices for the type of model used (which specifies $p(\mathbf{x}|\theta)$ and the prior distribution $p(\theta)$). These choices are dependent on the type of problem to be solved (e.g continuous versus discrete solution variables). For some cases the calculation of the model posterior and posterior predictive distribution can be carried out simply, while other choices may require more sophisticated Markov Chain Monte Carlo sampling techniques.

III. APPLICATION TO FACTORIZED-MODEL CONTINUOUS EDAS

The simplest and most widely developed model in EDAs is a factorized product of univariate marginal distributions

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i) \tag{2}$$

A number of continuous EDAs ($\mathbf{x} \in \mathbb{R}^n$) have been developed using the factorized probability model given in (2). For the remainder of this paper we focus on continuous EDAs that utilize a univariate Gaussian distribution

$$p(x_i|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}(\frac{x_i - \mu_i}{\sigma_i})^2} \tag{3}$$

The learning/model estimation problem in this case requires a method for calculating the mean μ_i and variance σ_i parameters of $p(x_i)$. The standard model is to use a different σ_i parameter for each search dimension, leading to elliptical contours of equal probability with the constraint that the principal axes of these ellipses must be parallel to one of the coordinate axes of the space.

A. Continuous UMDA

The extension of the Univariate Marginal Distribution Algorithm (UMDA) [9] to continuous search spaces: UMDA_c^G [10], follows the general EDA framework of Table I and employs the model from (3). In UMDA_c^G, the mean parameters for the next generation are set as the sample mean of the selected population

$$\mu_{i,t} = \overline{x_{i,t}} = \frac{1}{M_{sel}} \sum_{j=1}^{M_{sel}} x_i^j \tag{4}$$

and the standard deviation (variance) parameters are set as the sample standard deviation (variance) of the selected population

$$\sigma_{i,t} = s_{i,t} = \sqrt{\frac{1}{M_{sel}-1} \sum_{j=1}^{M_{sel}} (x_i^j - \overline{x_{i,t}})^2} \tag{5}$$

At any given generation, $\mu_{i,t}$ and $\sigma_{i,t}$ represent the maximum likelihood estimates for the mean and standard deviation of each marginal distribution $p(x_i|\mu_i, \sigma_i^2)$.

The initial population ($t = 0$) is generated from a uniform distribution across the feasible search space. In the first generation, $\mu_{i,0}$ and $\sigma_{i,0}$ are estimated based on selected points from this random population. UMDA_c^G uses truncation selection: a fraction τ of the population with the best objective function values are retained for building/adapting the search model[1]. Therefore, only two algorithm parameters must be specified for an implementation of UMDA_c^G: the population size M and the selection parameter τ.

B. $\mathrm{BayEDA_{cG}}$: *A Continuous Bayesian EDA based on a univariate Gaussian model*

Using the factorized model from (2) above, a Bayesian EDA can be specified (here for simplicity we use x to refer to any one of the solution components x_i in a multidimensional problem). For a univariate Gaussian (Normal) model distribution, Bayesian inference can readily be carried out: the resulting expressions given here are drawn from Gelman et al. [11]. We consider the simplest case of a noninformative (flat) prior for the model parameters, expressing no preference for any particular values for the model parameters before observing any data. In this case, inference depends only on the data (selected individuals). The standard noninformative prior is uniform on $(\mu, \log \sigma^2)$ or

$$p(\mu, \sigma^2) \propto (\sigma^2)^{-1}$$

The joint posterior can be factorised as

$$p(\mu, \sigma^2|D) = p(\mu|\sigma^2, D)p(\sigma^2|D)$$

In this case, the marginal density for σ is

$$\sigma^2|D \sim \mathrm{Inv} - \chi^2(M_{sel} - 1, s^2) \tag{6}$$

[1]Rounding if $N \cdot \tau$ is not an integer.

TABLE III
ALGORITHM: $\mathrm{BayEDA_{cG}}$.

```
Given: population size $M$, selection parameter $\tau$
BEGIN (set $t = 0$) Generate $M$ individuals uniformly in $S$
REPEAT for $t = 1, 2, \ldots$ until stopping criterion is met
    Select $M_{sel} = \mathrm{Round}(M \cdot \tau)$ individuals via truncation selection
    Calculate sample mean $\overline{x}$ and variance $s^2$ of D to update model
    Sample $M$ individuals from $p_t(\mathbf{x}|D, \theta)$:
    FOR i=1:$M$
        Draw sample variance $\tilde{\sigma}^2 \sim \mathrm{Inv} - \chi^2(M_{sel} - 1, s^2)$
        Draw sample mean $\tilde{\mu} \sim N(\overline{x}, \tilde{\sigma}^2/(M_{sel}))$
        Draw new individual $\mathbf{x}_i \sim N(\tilde{\mu}, \tilde{\sigma}^2)$
    ENDFOR
ENDREPEAT
END
```

where s^2 is the sample variance of the data. The conditional density for μ is

$$\mu|D, \sigma^2 \sim N(\overline{x}, \sigma^2/M_{sel}) \tag{7}$$

where $\overline{x}$ is the sample mean of the data D.

The predictive distribution for $\tilde{x}$ given the data, μ and σ is

$$\tilde{x}|D, \mu, \sigma^2 \sim N(\mu, \sigma^2) \tag{8}$$

In the $\mathrm{BayEDA_{cG}}$ algorithm, sampling from the posterior predictive distribution $p(\tilde{x}|D)$ can be easily carried out in a three-step process. Firstly, a sample $\tilde{\sigma}^2$ is drawn from (6), then this sample is used to draw a sample $\tilde{\mu}$ from (7) and finally both samples are used to draw a sample $\tilde{x}$ from (8). The process is repeated M times to produce the population for use in the next generation.

The algorithm is summarized in Table III. Note that for implementation purposes, a random draw y from an inverse-χ^2 distribution can be obtained by firstly drawing a sample z from the χ^2 distribution and applying $y = s^2/z$. The χ^2 distribution is also a special case of the gamma distribution (see [11] for details).

IV. Experimental Results

In this Section we present simulation results for the $\mathrm{BayEDA_{cG}}$ algorithm, using several standard test functions, and for comparison results for UMDA_c^G on the same functions. Since $\mathrm{BayEDA_{cG}}$ and UMDA_c^G use the same (factorised Gaussian) model and data (while differing in the way inference is performed on the model parameters), we would expect the two algorithms to produce results that are similar in a number of respects. Our main aim is to gain some insight into the behaviour of the $\mathrm{BayEDA_{cG}}$ algorithm rather than attempting to claim state-of-the-art performance on these problems. Therefore, we make no attempt to tune the parameters of the algorithms to these problems. The test functions used are listed in Table IV. While these functions have some limitations, they are commonly used and sufficient here for illustrative purposes.

Firstly we consider a simple unimodal problem (the 1-D Sphere function) to show the experimental convergence of the algorithm compared to UMDA_c^G. For each algorithm,

TABLE IV

TEST FUNCTIONS USED IN THE EXPERIMENTS

Name	Function
Sphere	$f_{Sph}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$ $-5.12 \le x_i \le 5.12, f_{Sph}(\mathbf{x}^*) = 0$
Rastrigin	$f_{Ras}(\mathbf{x}) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$ $-5 \le x_i \le 5, f_{Ras}(\mathbf{x}^*) = 0$
Rosenbrock	$f_{Ros}(\mathbf{x}) = \sum_{i=1}^{n-1}(x_i^2 - x_{i+1})^2 + (x_i - 1)^2$ $-2 \le x_i \le 2, f_{Ros}(\mathbf{x}^*) = 0$
Griewangk	$f_{Gri}(\mathbf{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ $-600 \le x_i \le 600, f_{Gri}(\mathbf{x}^*) = 0$
Ackleys	$f_{Ack}(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e$ $-15 \le x_i \le 30, f_{Ack}(\mathbf{x}^*) = 0$

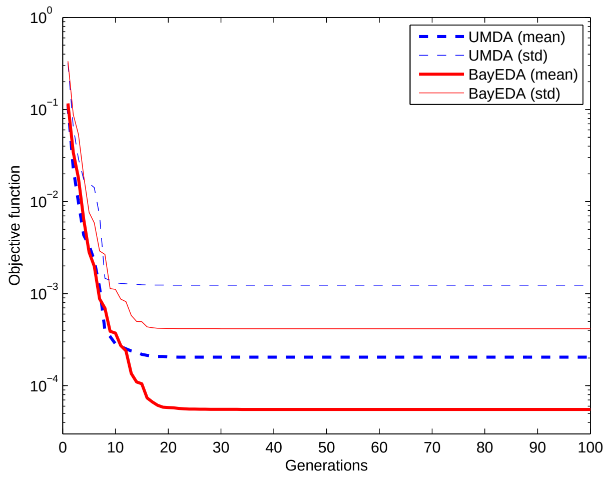

Fig. 1. Best-so-far performance curves for $\text{BayEDA}_{\text{cG}}$ and UMDA_c^G on the 1-D Sphere function. Shown are average performance and (average + standard deviation) curves.

$M = 40$ and $\tau = 0.9$. 100 trials were conducted over 100 generations, with results shown as mean and standard deviations over these trials. Figure 1 shows the performance of each algorithm as a function of generations (in terms of best solution value found so far). While the performance is fairly similar, $\text{BayEDA}_{\text{cG}}$ attains a lower value on average. Progress for UMDA_c^G appears to converge after about 15 generations, compared to around 20 generations for $\text{BayEDA}_{\text{cG}}$. The standard deviations on the curves for each algorithm are almost identical.

Figures 2 and 3 show the evolution of the model parameters for each algorithm. Note that for $\text{BayEDA}_{\text{cG}}$ these are first averaged over the M model parameter samples generated and used when sampling from the posterior predictive distribution of each population. Apart from a fluctuation in standard deviation around generation 5-15, the curves for the μ parameters are very similar (Figure 2). In Figure 3 however, we see a slower convergence for the $\text{BayEDA}_{\text{cG}}$ σ^2 estimate compared to that of UMDA_c^G. This explains the improved performance of $\text{BayEDA}_{\text{cG}}$ in Figure 1. The model sampling

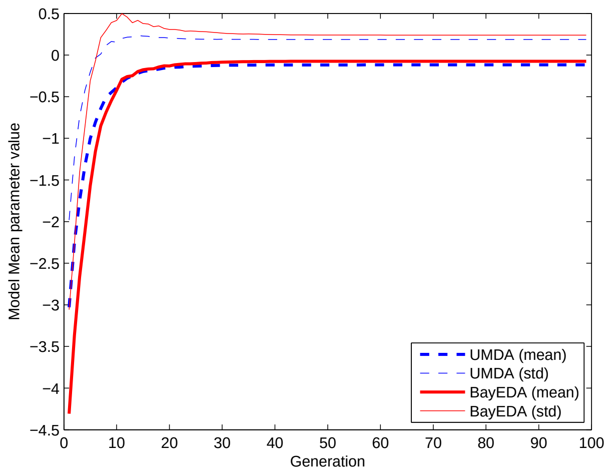

Fig. 2. Evolution of model mean parameter values for $\text{BayEDA}_{\text{cG}}$ and UMDA_c^G on the 1-D sphere function. Shown are average performance and (average + standard deviation) curves. For $\text{BayEDA}_{\text{cG}}$ the curves are averages of posterior samples and over runs.

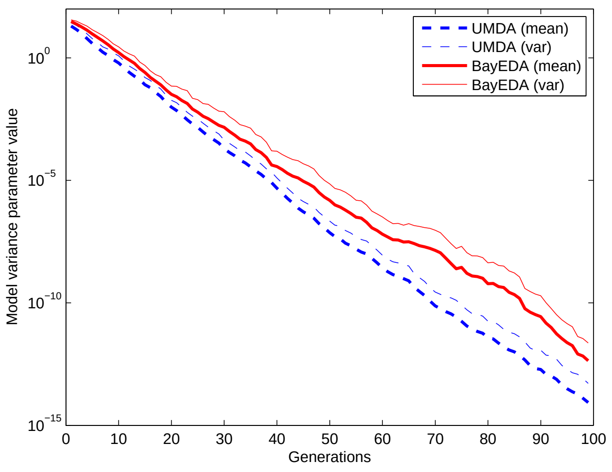

Fig. 3. Evolution of model variance parameter values for $\text{BayEDA}_{\text{cG}}$ and UMDA_c^G on the 1-D sphere function. Shown are average performance and (average + standard deviation) curves. For $\text{BayEDA}_{\text{cG}}$ the curves are averages of posterior samples and standard deviations over runs.

in $\text{BayEDA}_{\text{cG}}$ introduces a source of variability not present in UMDA_c^G, which increases the search diversity. For this problem the effect leads to an improvement in performance.

A further illustration of the dynamics of the $\text{BayEDA}_{\text{cG}}$ model is shown in Figure 4. For this experiment, $M = 40$ and $\tau = 0.95$ over 100 generations. The graph shows the evolution of a single trial run of the BayEDA model on the 1-D Rastrigin function, in terms of the sampled model parameter values over generations. The model samples follow a trajectory with mean values tending towards the location of the global optimum ($x = 0$) and variance values converging towards zero. However, clustering is evident in the points

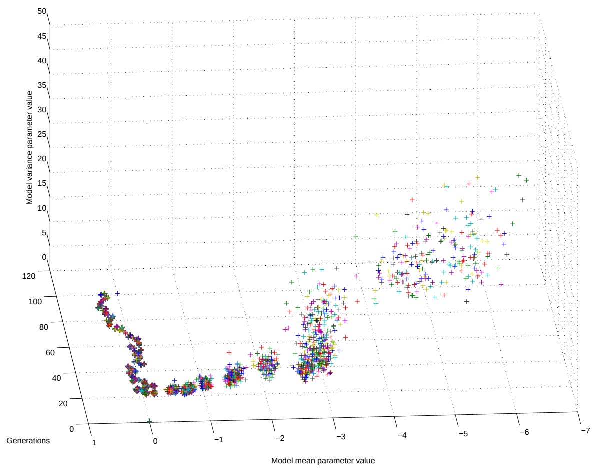

Fig. 4. A plot of the model posterior samples over a single run of $\mathrm{BayEDA_{cG}}$ on the 1-D Rastrigin function. Clustering is evident along the mean parameter axis and can be related to the structure of the problem.

on this trajectory, showing that the algorithm spends more time in certain areas of the search space. For reference, the 1-D Rastrigin function is shown in Figure 5 over the range explored by the algorithm. Comparing Figures 4 and 5 it can be seen that the clustering in model samples corresponds to the local minima of the function.

In the Bayesian framework, model parameter values are drawn from the model according to their posterior distribution. This fact can offer an explanation for the result shown in Figure 4. Selection will lead to a representation of the regions on the 1-D Rastrigin function close to each local optimum. This will make more likely the relative probability of Gaussian models with a mean peaked over a local optimum and a variance in proportion to the size of a local basin of attraction. As a consequence, models of such shape and location will tend to appear more frequently in samples from the posterior distribution. The influence of this effect will however depend on the test problem and the algorithm parameters. For this experiment, a very soft selection pressure was used and the initial population was uniform in the range $[-15, 5]$. This causes the model to evolve more slowly over the objective function surface and causes it to encounter more locally optimal basins of attraction before locating the region around the global optimum.

Experiments were also conducted on 10-D versions of the test functions given in Table IV. For each problem, 30 trials were conducted. Following the experiments in Chapter 8 of [2], we used $M = 2000$ and $\tau = 0.5$. For f_{Sph}, runs were for 100 generations and for all other functions runs were for 200 generations The results are summarized in Table V.

Overall the results show highly similar performance for the two algorithms. UMDA_c^G showed slightly better mean performance in most cases, although the standard deviation of the results for $\mathrm{BayEDA_{cG}}$ was almost always larger that of UMDA_c^G (possibly a result of the model variability

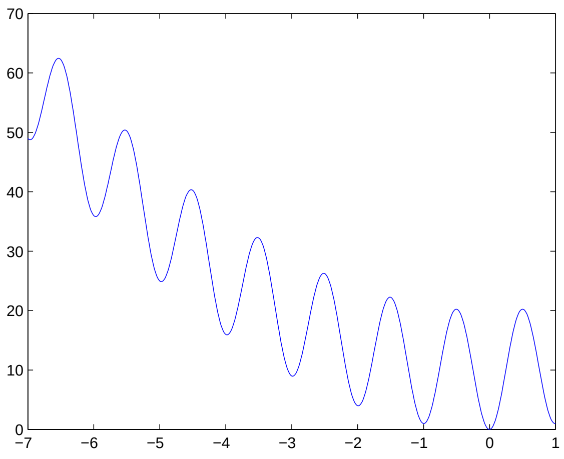

Fig. 5. The 1-D Rastrigin function in over the range covered by the results in Figure 4.

TABLE V
PERFORMANCE RESULTS ON 10D VERSIONS OF THE TEST FUNCTIONS.

Function	UMDA Mean (Std)	BayEDA Mean (Std)
Sphere	9.63E-09 (2.36E-09)	1.18E-08 (2.63E-09)
Rastrigin	7.12E-06 (8.15E-06)	1.56E-05 (2.16E-05)
Rosenbrock	8.21E+01 (2.04E-02)	8.21E+01 (2.41E-02)
Griewangk	7.54E-14 (2.45E-14)	1.08E-13 (2.86E-14)
Ackleys	1.96E-08 (2.75E-09)	2.11E-08 (3.42E-09)

discussed in the 1-D examples above). As mentioned above, no attempt was made to optimize the values of M and τ used in $\mathrm{BayEDA_{cG}}$ - a study of the sensitivity of the performance of the algorithms to the parameter values is outside the scope of this paper. The global optimum for each problem was found with high precision, apart from the Rosenbrock function for which convergence is known to be difficult.

V. Related Work

The literature concerned with the development of optimization techniques is both large and diverse. Optimization algorithms that construct some kind of statistical model and use this model to influence the search process can be found in areas such as Evolutionary Computation, Metaheuristics, Machine Learning and Engineering Design, as well as in the fields of stochastic and global optimization.

A. Objective Function Models

A different approach to model-based optimization is to construct a model using not only selected points visited during the search, but also the corresponding objective functions values for those points searched. Newton's method is a simple and well-known example of this class of techniques, fitting a local quadratic model at each iteration of the algorithm and directing the search using the optimum of the model (Sequential quadratic programming techniques generalise this idea) [12]. Response surface methodology [13] also utilizes optimization procedures that fit a low-order polynomial regression model to the $(\mathbf{x}_{ti}, S(\mathbf{x}_{ti}))$ data and use simple calculus to estimate the optima from the polynomial. Experimental design techniques are an important part of response surface approaches.

Stochastic process models of the objective function have also been widely considered as response surfaces and in model-based optimization, dating back to an algorithm introduced by Kushner in 1964 based on a 1-D Weiner process [14]. Subsequent work includes that of Stuckman [15], the Bayesian approach to global optimization of Mockus [16] and the P-algorithm of Zilinskas [17]. In engineering design, more sophisticated stochastic process models have been employed as response surfaces for model-based optimization. In particular, kriging models have received attention [18]. These techniques produce a surrogate model of the objective function, together with a value for the confidence of the model at any point $\mathbf{x}$. Together with information about the current best solution found, this confidence information is used to define criteria that indicate the expected utility of searching future points in the search space [19].

Previous work on model-based optimization can also be found in artificial intelligence and machine learning. Moore and Schneider use locally weighted regression to build a model of the objective function using all points evaluated during the search [20]. Boyan and Moore propose the STAGE algorithm [21], which learns an "evaluation function" which aims to predict the outcome of a local search algorithm. This evaluation function model is then used to guide future search.

VI. Conclusions

We have presented a new approach to model fitting in EDAs based on Bayesian inference, a method which utilizes prior distributions on model parameters and generates each successive population from the posterior predictive distribution. A simple implementation of this framework, $\mathrm{BayEDA_{cG}}$ using a Gaussian model with a non-informative prior was detailed and experimentally compared against the frequentist alternative, UMDA_c^G. The $\mathrm{BayEDA_{cG}}$ method outperformed the UMDA_c^G method on an example 1-dimensional optimization problem. Its performance on five 10-dimensional example problems was overall very similar to that of UMDA_c^G. The differences between the two results are primarily due to the slightly larger variance of the $\mathrm{BayEDA_{cG}}$ model.

We believe that there is considerable scope for future work in developing Bayesian techniques in EDAs. While the $\mathrm{BayEDA_{cG}}$ implementation described here is a continuous univariate model, multivariate models and/or models for discrete variables exist in the Bayesian literature and should be readily applicable to EDAs. In principle, Bayesian EDAs could be developed using many of the other probabilistic models commonly used in discrete and continuous EDAs (though some implementations will be more complex than others). Furthermore, the Bayesian framework opens up the possibility of utilizing prior distributions in the context of optimization. Although this paper only considers a simple noninformative prior over univariate Gaussian model parameters, informative priors may be useful to incorporate knowledge about optimization problems (e.g constraints) in a consistent way, something that is not possible with EDAs employing maximum-likelihood parameter estimation. In addition, conjugate priors can be utilized to produce efficient implementations of Bayesian inference for many commonly used distributions.

Acknowledgment

The authors would like to thank Dirk Kroese for contributing to the development of this work and related discussion. Ian Wood would like to acknowledge the support of the ARC Center for Complex Dynamic Systems and Control. Jonathan Keith and George Sofronov would like to acknowledge the support of an Australian Research Council (ARC) Discovery Grant (DP0556631) and additionally for JK a National Medical and Health Research Council (NHMRC) project grant (389892).

References

[1] S. Kern, S. D. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos, "Learning probability distributions in continuous evolutionary algorithms - a comparative review," *Natural Computing*, vol. 3, no. 1, pp. 77–112, 2004.

[2] P. Larrañaga and J. A. Lozano, Eds., *Estimation of Distribution Algorithms : A New Tool for Evolutionary Computation*. Kluwer, 2002.

[3] M. Pelikan, D. E. Goldberg, and F. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5–20, 2002.

[4] R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*, ser. Information Science and Statistics. Springer, 2004.

[5] M. Gallagher and M. Frean, "Population-based continuous optimization, probabilistic modelling and mean shift," *Evolutionary Computation*, vol. 13, no. 1, pp. 29–42, 2005.

[6] S. Baluja, "Population-Based Incremental Learning: A method for integrating genetic search based function optimization and competitive learning," School of Computer Science, Carnegie Mellon University, Tech. Rep. CMU-CS-94-163, 1994.

[7] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "BOA: The Bayesian optimization algorithm," in *Proc. Genetic and Evolutionary Computation Conference (GECCO'99)*, W. Banzhaf and et al., Eds. San Francisco, CA: Morgan Kaufmann, 1999, pp. 525–532.

[8] B.-T. Zhang, "A unified Bayesian framework for evolutionary learning and optimization," in *Advances in Evolutionary Computing: Theory and Applications*, ser. Natural Computing Series, A. Ghosh and S. Tsutsui, Eds. Springer, 2003, pp. 393–412.

[9] H. Mühlenbein, "The equation for response to selection and its use for prediction," *Evolutionary Computation*, vol. 5, pp. 303–346, 1998.

[10] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña, "Optimization by learning and simulation of Bayesian and Gaussian networks," University of the Basque Country, Spain, Tech. Rep. KZZA-IK-4-99, 1999.

[11] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, 2nd ed. Chapman and Hall/CRC, 2004.

[12] R. Fletcher, *Practical methods of optimization*, 2nd ed. Chichester, New York: Wiley, 1987.

[13] G. E. P. Box and N. R. Draper, *Empirical model-building and response surfaces*. Wiley, 1987.

[14] H. J. Kushner, "A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise," *Journal of Basic Engineering*, vol. 86, pp. 97–106, 1964.

[15] B. E. Stuckman, "A global search method for optimizing nonlinear systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 6, pp. 965–977, 1988.

[16] J. Mockus, “Application of Bayesian approach to numerical methods of global and stochastic optimization,” *Journal of Global Optimization*, vol. 4, pp. 347–365, 1994.

[17] A. Žilinskas, “A review of statistical models for global optimization,” *Journal of Global Optimization*, vol. 2, pp. 145–153, 1992.

[18] D. R. Jones, “A taxonomy of global optimization methods based on response surfaces,” *Journal of Global Optimization*, vol. 21, pp. 345–383, 2001.

[19] M. J. Sasena, P. Papalambros, and P. Goovaerts, “Exploration of metamodelling sampling criteria for contrained global optimization,” *Engineering Optimization*, vol. 34, no. 3, pp. 263–278, 2002.

[20] A. W. Moore and J. Schneider, “Memory-based stochastic optimization,” in *Advances in Neural Information Processing Systems*, vol. 8, 1996, pp. 1066–1072.

[21] J. Boyan and A. Moore, “Learning evaluation functions to improve optimization by local search,” *Journal of Machine Learning Research*, vol. 1, pp. 77–112, 2000.