A Unified Account of Population-Based Stochastic Optimization Methods

Mark Andrews
Department of Psychology
University College London

1 February, 2006

Abstract

In this paper, we aim to provide a theoretical framework through which to understand a variety of stochastic optimization methods. We describe a general procedure for stochastic optimization that is based on the EM algorithm. We show that this method turns out to be the general case of a large number of population-based stochastic optimization procedures including genetic algorithms, stochastic hill-climbing methods, the cross-entropy method, population-based incremental learning, and others. This unifying perspective may facilitate our understanding of the relative performance of these methods. For example, one of the key insights that this perspective provides is that different population-based procedures can be largely differentiated on the basis of the probability model that they (either explicitly or implicitly) assume underlies the problem domain. Procedures that assume probability models that are either inappropriate or inadequate for the problem domain can perform poorly on these problems.

1 Introduction

A typical optimization problem consists of a set $\mathcal{X}$ and a function $\phi\colon \mathcal{X} \mapsto \mathbb{R}$ that assigns a value or utility to each element of $\mathcal{X}$. The set $\mathcal{X}$ can be a continuous space, e.g. $\mathcal{X} \subseteq \mathbb{R}^d$, or a discrete and finite set, e.g. $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$. The objective of the optimization problem is to find

$$\hat{\mathbf{x}} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, \phi(\mathbf{x}), \tag{1}$$

or the element of $\mathcal{X}$ that has the highest value according to ϕ. The element $\hat{\mathbf{x}}$ is said to be the solution of the optimization problem.

Population-based methods for optimization represent candidate solutions as distribution of points, i.e. $\tilde{\mathbf{x}}_{1:M} \subseteq \mathcal{X}$. In these cases, the objective is to find

$$\hat{\mathbf{x}}_{1:M} = \underset{\{\tilde{\mathbf{x}}_{1:M}\}}{\operatorname{argmax}}\, \frac{1}{M}\sum_{m=1}^{M} \phi(\tilde{\mathbf{x}}_m), \tag{2}$$

or the distribution of points with the highest average value according to ϕ. The set of procedures commonly known as Genetic Algorithms (GAs), see e.g. [8], and Stochastic Hill-Climbing (SHC) methods, see e.g. [9], are two widely used and familiar examples of population-based optimization methods.

As any set of points, such as $\tilde{\mathbf{x}}_{1:M} \subseteq \mathcal{X}$, can be represented as a probability distributions over $\mathcal{X}$, as in $\mathrm{P}(\mathbf{x} \in \mathcal{X}) = \frac{1}{M}\sum_{m=1}^{M}\delta(\mathbf{x} - \tilde{\mathbf{x}})$, population-based methods for optimization can be generalized to include methods that explicitly describe a candidate solution as a probability distributions over the set $\mathcal{X}$. In these cases, if we parameterize a probability distribution over $\mathcal{X}$ according to $\mathrm{P}(\mathcal{X}\big|\boldsymbol{\theta})$, then an objective for this optimization problem could be

$$\hat{\theta} = \underset{\theta\in\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{i=1}^{N} \phi(\mathbf{x}_i)\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta}). \tag{3}$$

In other words, the objective is to find a probability distribution over the set $\mathcal{X}$ that will maximize $\mathcal{X}$'s expected value according to ϕ. Population-based methods that explicitly define probability distributions include population-based incremental learning (PBIL) [1, 2], the cross-entropy method (CE) [12], the MIMIC method of [4], and the Bayesian Optimization Algorithm of [11].

Equation (3) can be regarded as the general case of the objective function for population-based optimization methods. (We can see that (2) is a special case of (3), if we substitute $\frac{1}{M}\delta(\mathbf{x}_i - \tilde{\mathbf{x}})$ for $\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta})$ in (3) and regard the set of parameters as the set $\{\tilde{\mathbf{x}}_{1:M}\}$). In this paper, we describe an algorithm for optimizing

$$\mathcal{E}(\theta) = \log \sum_{i=1}^{N} \phi(\mathbf{x}_i)\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta}), \tag{4}$$

with respect to $\boldsymbol{\theta}$ (As log is a monotonic transformation, any optimum of $\mathcal{E}(\theta)$ will also be a solution of (3)). This method relies upon the iterative introduction and maximization of a free-energy lower bound on $\mathcal{E}(\theta)$, and is derived from similar principles to those of the EM algorithm of [6] and its generalizations, e.g. [10]. This method represents the general case of a large number of population-based stochastic optimization procedures. These include the already mentioned GAs, SHCs, PBIL, CE method, MIMIC, and BOA. These principles are also implemented in a reinforcement learning procedure known the relative payoff procedure (RPP), [7]. In fact, the RPP was shown by [5] to be a version of the EM algorithm. The analysis of [5] was the initial motivation for this work.

In what follows, we describe a general algorithm for population-based optimization that is based on principles similar to those of the EM algorithm. We illustrate how this algorithm works using a simple toy optimization problem. We then proceed to explain how the population-based search methods mentioned above can be seen as special cases of this general method. This leads to the insight that the central feature that differentiates between these population-based methods is the probability distribution that they assume. We conclude with a discussion of the role played by the probability model that is assumed in

population-based optimization methods. We demonstrate by way of a set of examples how probability distributions that are ill-suited to the problem domain can lead to poor performance.

2 Algorithm Optimizing $\mathcal{E}(\theta)$.

In this section, we apply the generalized EM algorithm, e.g. [6, 10], for the purposes of optimizing (4). As a reminder, EM for maximum-likelihood estimation works as follows. In a probabilistic model $\mathcal{M}$, with parameters $\mathcal{A}$ and latent-variables $\mathcal{B}$, we are interested to maximize the log-likelihood

$$\mathcal{L}(\mathcal{A}) = \log \int d\mathcal{B}\ \mathrm{P}\big(\mathcal{D}, \mathcal{B}\big|\mathcal{A}\big)\,. \tag{5}$$

To do so, we express (5) in an equivalent form

$$\mathcal{L}(\mathcal{A}) = \int d\mathcal{B}\ \mathrm{q}(\mathcal{B}) \log \frac{\mathrm{P}\big(\mathcal{D}, \mathcal{B}\big|\mathcal{A}\big)}{\mathrm{q}(\mathcal{B})}, \tag{6}$$

where $\mathrm{q}(\mathcal{B}) = \mathrm{P}\big(\mathcal{B}\big|\mathcal{D}, \mathcal{A}\big)$. Holding $\mathrm{q}(\mathcal{B})$ constant we iteratively maximize (6) with respect to $\mathcal{A}$ and then re-compute $\mathrm{q}(\mathcal{B})$. If these iterations are exact, they will monotonically increase $\mathcal{L}(\mathcal{A})$. The generalized EM allows for approximations in calculating $\mathrm{q}(\mathcal{B})$ and for increasing, rather than just maximizing, $(\mathcal{B}$.

To optimize $\mathcal{E}(\theta)$ in a manner similar to the generalized EM algorithm, e.g. [6, 10], we first introduce $\mathcal{F}(\mathrm{q}, \boldsymbol{\theta})$, a lower bound on $\mathcal{E}(\theta)$, as follows:

$$\mathcal{E}(\theta) = \log \sum_{i=1}^{N} \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i) \frac{\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta})}{\mathrm{q}(\mathbf{x}_i)}, \tag{7}$$

$$\geq \sum_{i=1}^{N} \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i) \log \frac{\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta})}{\mathrm{q}(\mathbf{x}_i)}, \tag{8}$$

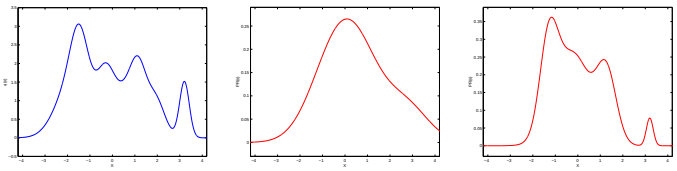
$$= \mathcal{F}(\mathrm{q}, \boldsymbol{\theta}). \tag{9}$$

The equality (7) holds trivially for any function $\mathrm{q}\colon \mathcal{X} \mapsto \mathbb{R}$. As log is a concave function, then (8) follows from Jensen's inequality [1], so long as $0 \leq \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i) \leq 1$ and $\sum_{i=1}^{N} \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i) = 1$.

We optimize $\mathcal{F}(\mathrm{q}, \boldsymbol{\theta})$ with respect to both q and $\boldsymbol{\theta}$. This proceeds iteratively, first by holding $\boldsymbol{\theta}$ constant and optimizing wrt to q, and then by holding q constant and optimizing wrt $\boldsymbol{\theta}$. In other words, having chosen an arbitrary

[1] Jensen's Inequality: If f is a concave function and $0 \leq \lambda_i \leq 1$ and $\sum_{i=1}^{N} \lambda_i$, then

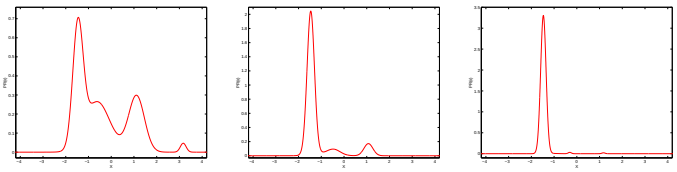
$$f(\sum_{i=1}^{N} \lambda_i x_i) \geq \sum_{i=1}^{N} \lambda_i f(x_i).$$



(a) Objective: $\phi(\mathbf{x})$

(b) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ at $t = 1$. $\mathcal{E}(\boldsymbol{\theta}^t) = .5132$

(c) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ at $t = 3$. $\mathcal{E}(\boldsymbol{\theta}^t) = .7100$



(d) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ at $t = 5$. $\mathcal{E}(\boldsymbol{\theta}^t) = .8248$

(e) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ at $t = 10$. $\mathcal{E}(\boldsymbol{\theta}^t) = 1.0183$

(f) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ at $t = 25$. $\mathcal{E}(\boldsymbol{\theta}^t) = 1.0873$

Figure 1: An illustration of the general procedure for stochastic optimization described in Section 2.

initial parameter setting $\boldsymbol{\theta}^0$, for $t \geq 0$, we recursively perform the following two steps:

$$\mathrm{q}^t(\mathbf{x}) = \underset{\mathrm{q}}{\operatorname{argmax}}\, \mathcal{F}(\mathrm{q}, \boldsymbol{\theta}^t), \quad \text{(E-step)} \tag{10}$$

$$\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta}), \quad \text{(M-step)}. \tag{11}$$

If for any $\boldsymbol{\theta}^t$, we can find a $\mathrm{q}^t(\mathbf{x})$ such that (8) is an equality then this iterative procedure will always monotonically increase $\mathcal{E}(\theta)$ with respect to $\boldsymbol{\theta}$. The iteration of (10) and (11) for $0 \leq t \leq T$ will result in a sequence $\{\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta}^t\}_{t=0}^{t=T}$. For any t, $\mathcal{E}(\boldsymbol{\theta}^{t+1}) \geq \mathcal{E}(\boldsymbol{\theta}^t)$ as

$$\overbrace{\mathcal{E}(\boldsymbol{\theta}^t) = \mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta}^t)}^{\text{E-step}} \overbrace{\leq \mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta}^{t+1})}^{\text{M-step}} \overbrace{\leq \mathcal{E}(\boldsymbol{\theta}^{t+1})}^{\text{Jensen's Inequality}} . \tag{12}$$

2.1 E-step

For parameter setting $\boldsymbol{\theta}^t$, we find $\mathrm{q}^t(\mathbf{x})$ as defined by (10) by maximizing $\mathcal{F}(\mathrm{q}, \boldsymbol{\theta}^t)$, subject to the constraint that

$$\sum_{i=1}^{N} \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i) = 1. \tag{13}$$

Enforcing this normalizing constraint with Lagrange multipliers, we find the point where the derivative of

$$\mathcal{F}(\mathrm{q}, \boldsymbol{\theta}^t) + \lambda \left(1 - \sum_{i=1}^{N} \mathrm{q}(\mathbf{x}_i)\phi(\mathbf{x}_i)\right), \tag{14}$$

vanishesand obtain the solution

$$\mathrm{q}^t(\mathbf{x}_i) = \frac{\mathrm{P}(\mathbf{x}_i|\boldsymbol{\theta}^t)}{\sum_{j=1}^{N} \mathrm{P}(\mathbf{x}_j|\boldsymbol{\theta}^t)\phi(\mathbf{x}_j)}. \tag{15}$$

Substituting (15) into (8), we see that $\mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta}^t) = \mathcal{E}(\theta)$.

2.2 M-step

From (15) and (8), we have $\mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta})$

$$= \frac{1}{Z} \sum_{i=1}^{N} \mathrm{P}(x_i|\boldsymbol{\theta}^t)\phi(\mathbf{x}_i) \log \mathrm{P}(\mathbf{x}_i|\boldsymbol{\theta}) + \mathcal{H}(\mathrm{q}^t(\mathbf{x})), \tag{16}$$

where

$$Z = \sum_{j=1}^{N} \mathrm{P}(x_j|\boldsymbol{\theta}^t)\phi(x_j), \tag{17}$$

and $\mathcal{H}(\mathrm{q}^t(\mathbf{x}))$ is the entropy of the distribution $\mathrm{q}^t(\mathbf{x})$. As neither Z and $\mathcal{H}(\mathrm{q}^t(\mathbf{x}))$ depend on $\boldsymbol{\theta}$, $\mathrm{argmax}_{\boldsymbol{\theta}}\, \mathcal{F}(\mathrm{q}^t(\mathbf{x}), \boldsymbol{\theta})$

$$= \underset{\boldsymbol{\theta}}{\mathrm{argmax}} \sum_{i=1}^{N} \mathrm{P}(x_i|\boldsymbol{\theta}^t)\phi(\mathbf{x}_i) \log \mathrm{P}(\mathbf{x}_i|\boldsymbol{\theta}). \tag{18}$$

It is useful to note that maximizing (16) is identical to minimizing the Kullback-Liebler divergence between $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta})$ and the probability distribution defined by $\frac{1}{Z}\mathrm{P}(x|\boldsymbol{\theta}^t)\phi(\mathbf{x}_i)$. In other words, the algorithm as a whole can be viewed as iteratively fitting the probability distribution at time $t+1$ to normalized product of the probability distribution at time t and the objective function $\phi(\mathrm{x})$. This results in a sequence of probability model inductions that will converge to a model that will maximize the expectation given in (4).

2.3 Sampling

Our objective function in the M-step is

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathrm{P}(\mathbf{x}_n \big| \boldsymbol{\theta}^t) \phi(\mathbf{x}_n) \log \mathrm{P}(\mathbf{x}_n \big| \boldsymbol{\theta}). \tag{19}$$

In anything other than small problems, we can not evaluate each of the N values that $\mathbf{x}_n$ can take, and it is necessary to approximate $\mathrm{P}(\mathcal{X} \big| \boldsymbol{\theta}^t)$ as

$$\mathrm{P}(\mathbf{x}_n \big| \boldsymbol{\theta}^t) \approx \frac{1}{M} \sum_{m=0}^{M} \delta\left(\mathbf{x}_n - \tilde{\mathbf{x}}_m\right) \tag{20}$$

where $\{\tilde{\mathbf{x}}_m\}_{m=0}^{M} \sim \mathrm{P}(\mathbf{x}_n \big| \boldsymbol{\theta}^t)$, and $M \ll N$. Substituting (20) into (19) and ignoring the constant $\frac{1}{M}$, we have our new objective for the M-step as

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{m=1}^{M} \phi(\tilde{\mathbf{x}}_m) \log \mathrm{P}(\tilde{\mathbf{x}}_m \big| \boldsymbol{\theta}). \tag{21}$$

The pseudo-code for this method is provided in Algorithmm 1.

Algorithm 1 Generalized EM for optimizing $\mathcal{E}(\theta)$.

Initialize: $t = 0$, $\boldsymbol{\theta}^t \leftarrow \boldsymbol{\theta}_{\mathsf{rand}}$, $\mathsf{converge} = 0$

while $\neg\mathsf{converge}$ **do**

Sample:

$$\{\tilde{\mathbf{x}}_m^t\}_{m=1}^{M} \sim \mathrm{P}\big(\tilde{\mathbf{x}} \big| \boldsymbol{\theta}^t\big)$$

Evaluate:

$$\{\phi(\tilde{\mathbf{x}}_m^t)\}_{m=1}^{M}$$

Maximize:

$$\boldsymbol{\theta}^{t+1} \leftarrow \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{m=1}^{M} \phi(\tilde{\mathbf{x}}_m^t) \log \mathrm{P}\big(\tilde{\mathbf{x}}_m^t \big| \boldsymbol{\theta}\big)$$

if $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t$ **then**

$\mathsf{converge} = 1$

else

$t = t + 1$

end if

end while

2.4 A Toy Problem

In Figure 1, we illustrate a simple one-dimensional optimization problem. Finding the maxima of functions of one-dimension spaces is not a particularly chal-

lenging problem, yet this toy problem is valuable for the clarity it provides. Subfigure (a) depicts an objective function $\phi(\mathbf{x})$ over $\mathbb{R}^1$. As is evident, there is a single global maximum, with three additional local maxima. According to our probabilistic interpretation of the problem of optimization, our objective to find the probability distribution that will maximize $\int d\mathbf{x}\phi(\mathbf{x})\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. To do so, following the specifications of Algorithm 1, at each time t we sample from a distribution $\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta}^t)$, evaluate each sample according to $\phi(\mathbf{x})$ and then effectively fit a new model $\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta}^t)$ to the distribution defined by $\frac{1}{Z}\mathrm{P}(\mathbf{x}_i\big|\boldsymbol{\theta}^t)\phi(\mathbf{x}_i)$.

The form of the probability model in this example is a Mixture of Gaussians model. In other words, $\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta}) \triangleq \sum_{k=1}^{K}\mathcal{N}(\mathbf{x}\big|\mu_k,\sigma_k^2)\pi_k$, where μ_k, σ_k^2 and π_k are the means, variances and mixing proportions of the kth component Gaussian. From (21), the objective is

$$\underset{\{\boldsymbol{\mu},\boldsymbol{\sigma},\boldsymbol{\pi}\}}{\operatorname{argmax}} \sum_{m=1}^{M}\phi(\tilde{\mathbf{x}}_m)\log\sum_{k=1}^{K}\mathcal{N}(\tilde{\mathbf{x}}\big|\mu_k,\sigma_k^2)\pi_k. \tag{22}$$

This can be solved by a routine application of the EM algorithm for (weighted) maximum-likelihood estimation. The E-step yeilds

$$\mathrm{P}(k\big|\tilde{\mathbf{x}}_m,\boldsymbol{\theta}) = \frac{\mathcal{N}(\tilde{\mathbf{x}}\big|\mu_k,\sigma_k^2)\pi_k}{\sum_{k'=1}^{K}\mathcal{N}(\tilde{\mathbf{x}}\big|\mu_{k'},\sigma_{k'}^2)\pi_{k'}}. \tag{23}$$

The M-step yields

$$\pi_k = \sum_{m=1}^{M}\frac{\phi(\tilde{\mathbf{x}}_m)}{\sum_{m'=1}^{M}\phi(\tilde{\mathbf{x}}_{m'})}\mathrm{P}(k\big|\tilde{\mathbf{x}}_m,\boldsymbol{\theta}), \tag{24}$$

$$\mu_k = \frac{\sum_{m=1}^{M}\mathrm{P}(k\big|\tilde{\mathbf{x}}_m,\boldsymbol{\theta})\phi(\tilde{\mathbf{x}}_m)\tilde{\mathbf{x}}_m}{\sum_{m'=1}^{M}\mathrm{P}(k\big|\tilde{\mathbf{x}}_{m'},\boldsymbol{\theta})\phi(\tilde{\mathbf{x}}_{m'})} \tag{25}$$

and

$$\sigma_k^2 = \frac{\sum_{m=1}^{M}\mathrm{P}(k\big|\tilde{\mathbf{x}}_m,\boldsymbol{\theta})\phi(\tilde{\mathbf{x}}_m)\|\tilde{\mathbf{x}}_m-\mu_k\|}{\sum_{m'=1}^{M}\mathrm{P}(k\big|\tilde{\mathbf{x}}_{m'},\boldsymbol{\theta})\phi(\tilde{\mathbf{x}}_{m'})}. \tag{26}$$

Subfigure (b) shows the initial probability distribution, or $\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta}^1)$, and Subfigures (c), (d), (e) and (f) show the probability models learned at times 3, 5, 10 and 25, respectively. The process starts out by fitting the probability distribution $\mathrm{P}(\mathbf{x}\big|\boldsymbol{\theta})$ to the general form of the objective $\phi(\mathbf{x})$. This is evident in Subfigure (c). It proceeds to place mass primarily in the vicinity of points corresponding to maxima of $\phi(\mathbf{x})$, e.g. Subfigure (d). Eventually, the mass in the vicinity of local maxima recede, Subfigure (e), and all mass is placed around the global maximum, Subfigure (f). In this example, the sequence of probability models converge to one that is sharply peaked around the value $\mathbf{x} = -1.4981$ (the global maximum of $\phi(\mathbf{x})$ occurs at $\mathbf{x} = -1.5052$).

3 Stochastic Search Methods

In this section, we describe how the method described above provides the general case of the a wide range of population-based procedures such GAs, SHC methods, PBIL, the CE method, MIMIC and BOA.

3.1 Genetic Algorithms

Genetic algorithms, as mentioned in Section 1, represent candidate solutions to optimization problems as distributions, or populations, of points, i.e. $\tilde{\mathbf{x}}_{1:M}$. The aim of the GA is to evolve this distribution using fixed evolution rules, so that the average value of the candidate solution according to $\phi\colon \mathcal{X} \mapsto \mathbb{R}$ increases. In general, the update rules take the following form: A randomly chosen, initial population is evaluated according to the objective function, and the highest scoring vectors are retained[2]. The members of this high-scoring set are then paired and *crossed-over* to form new *offspring* vectors. This progeny set is then taken to be the next generation, and the evaluation, selection and reproduction procedures are iterated. The manner of the so-called reproduction procedures can vary considerably. A common scenario, and one we will concentrate on here, is known as one-point crossover. In this scenario, two vectors $\mathbf{x}_A = x_{1:K}^{[A]}$ and $\mathbf{x}_B = x_{1:K}^{[B]}$ generate offspring vectors by randomly choosing a position $1 \leq k \leq K$ and crossing-over so as to form $\mathbf{x}_C = [x_{1:k}^{[A]}\ x_{k+1:K}^{[B]}]$. It is common that the resulting offspring are then perturbed by random mutation, such as by the addition of independent Gaussian noise with variance σ^2 to each element of the vector $\mathbf{x}_C$. Here, we show that this procedure is equivalent to representing a candidate solution to an optimization problem as a probability distribution $\mathrm{P}\big(x\big|\boldsymbol{\theta}\big)$. This probability distribution is then updated according to the Equation 21, with the result that the expected value of the probability distribution increases.

Given a population $\tilde{\mathbf{x}}_{1:M}$ whose members repeatedly pair and crossover in the one-point crossover manner, the set of offspring vectors can be as described as being a sample from the probability distribution

$$\mathrm{P}\big(\mathbf{x}\big|\tilde{\mathbf{x}}_{1:M}\big) = \sum_{k=1}^{K-1} \mathrm{P}(k) \sum_{m=1}^{M} \frac{\mathcal{N}(\tilde{x}_{1:k}^{[m]}, \Sigma^{[k]})}{M} \sum_{m' \neq m} \frac{\mathcal{N}(\tilde{x}_{k+1:K}^{[m']}, \Sigma^{[K-k]})}{M-1}. \tag{27}$$

Here, $\mathrm{P}(k)$ is the probability of choosing location k as the crossover point, and $\Sigma^{[k]} = \sigma^2 \mathbf{I}_k$ is a $k \times k$ Gaussian covariance matrix with σ^2 on the diagonal and zero elsewhere ($\mathbf{I}_k$ is a $k \times k$ identity matrix), with $\Sigma^{[K-k]}$ defined analogously. As can be seen, sampling from (27) involves choosing a location k according to $\mathrm{P}(k)$, choosing m uniformly at random and then sampling a k length vector from the Gaussian with covariance $\Sigma^{[k]} = \sigma^2 \mathbf{I}_k$ centered on $\tilde{x}_{1:k}^{[m]}$. This is followed by

[2]The proportion retained is variable.

choosing m' uniformly at random from the set $\{1, 2, \ldots m-1, m+1 \ldots m\}$, and then sampling a $K-k$ length vector from the Gaussian with covariance $\Sigma^{[K-k]} = \sigma^2 \mathbf{I}_{K-k}$ centered on $\tilde{x}^{[m']}_{k+1:K}$. The resulting K length sample vector is formed by concatenating the k and $K-k$ length vectors. This process is identical to drawing pairs uniformly at random from the population $\tilde{\mathbf{x}}_{1:M}$, choosing a crossover point k according to $\mathrm{P}(k)$, crossing-over the pair of vectors at this point, finally mutating each vector so formed by adding Gaussian noise, with fixed variance σ^2, independently to the each of its elements.

In a GA, an initial population of vectors is evaluated according to $\phi \colon \mathcal{X} \mapsto \mathbb{R}$, and the vectors $\{\tilde{\mathbf{x}}_m \colon \phi(\tilde{\mathbf{x}}_m) > \gamma_0\}$ are selected. Here, γ_0 is the $1-\epsilon$-quantile score of the set $\tilde{\mathbf{x}}^{[0]}_{1:M}$. If we take the free parameters of (27) to be the centres of the Gaussian distributions, finding

$$\theta^{t+1} = \operatorname*{argmax}_{\theta} \sum_{\{m \colon \phi(\tilde{\mathbf{x}}^m) > \gamma_0\}} \log \mathrm{P}\big(\tilde{\mathbf{x}}^m \big| \theta\big) \tag{28}$$

is simply a matter of setting the centres to be the set $\{\tilde{\mathbf{x}}_m \colon \phi(\tilde{\mathbf{x}}_m)\}$. The next generation then is a sample from this new distribution

$$\begin{aligned} \mathrm{P}\big(\mathbf{x}\big|\theta^{t+1}\big) = \sum_{k=1}^{K-1} \mathrm{P}(k) \sum_{\{m \colon \phi(\tilde{\mathbf{x}}_m) > \gamma_0\}} \frac{\mathcal{N}(\tilde{x}^{[m]}_{1:k}, \Sigma^{[k]})}{\sum_{m=1}^{M} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_0)} \\ \times \sum_{\{m' \colon \phi(\tilde{\mathbf{x}}^{m'}) > \gamma_0 \cap m' \neq m\}} \frac{\mathcal{N}(\tilde{x}^{[m']}_{k+1:K}, \Sigma^{[K-k]})}{\sum_{m' \neq m} \mathbb{I}(\phi(\tilde{\mathbf{x}}^{m'}) > \gamma_0)}. \end{aligned} \tag{29}$$

This process is a special case of Algorithm 1, with the exception that the ϕ function is thresholded and the function to be maximized with respect to θ is

$$\sum_{m}^{M} \mathbb{I}\left(\phi(\tilde{\mathbf{x}}^{[m]}) > \gamma\right) \log \mathrm{P}\Big(\tilde{\mathbf{x}}^{[m]}\big)\big|\theta\Big) \tag{30}$$

rather than

$$\sum_{m}^{M} \phi(\tilde{\mathbf{x}}^{[m]}) \log \mathrm{P}\Big(\tilde{\mathbf{x}}^{[m]}\big|\theta\Big). \tag{31}$$

3.2 Stochastic Hill-climbing

Stochastic hill-climbing (SHC) is perhaps the simplest stochastic optimization method. Applied to the problem described in Section 1, it works as follows. From the set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, we choose a uniformly random subset $\mathcal{Y}_0 = \{\mathbf{x}_{y_0(1)}, \mathbf{x}_{y_0(2)}, \ldots, \mathbf{x}_{y_0(m)}, \ldots, \mathbf{x}_{y_0(M)}\}$, where y_0 is a vector indicating M elements of $\mathcal{X}$, and usually $M \ll N$. We sample, with replacement, from the set $\mathcal{Y}_0$, and randomly perturb each sample to obtain the set $\tilde{\mathcal{Y}}_0 = \{\tilde{\mathbf{x}}_{y_0(1)}, \tilde{\mathbf{x}}_{y_0(2)}, \ldots, \tilde{\mathbf{x}}_{y_0(\tilde{m})} \ldots, \mathbf{x}_{y_0(\tilde{M})}\}$, where $\tilde{M} > M$. We evaluate the $\tilde{M}$ vectors according $\phi \colon \mathcal{X} \mapsto \mathbb{R}$, and choose the

M highest scoring vectors to obtain the new set $\mathcal{Y}_1 = \{\mathbf{x}_{y_1(1)}, \mathbf{x}_{y_1(2)}, \ldots, \mathbf{x}_{y_1(m)}, \ldots, \mathbf{x}_{y_1(M)}\}$. The hope is that at each iteration t,

$$\sum_{m=1}^{M} \phi\left(\mathbf{x}_{y_t(m)}\right) \geq \sum_{m=1}^{M} \phi\left(\mathbf{x}_{y_{t-1}(m)}\right). \tag{32}$$

Although there can be no guarantees of success, the SHC method has been shown to be surprisingly effective in combinatorial optimization test cases, [9].

In SHC, the procedure of drawing perturbed samples from some set $\mathcal{Y}_t$ can be likened to estimating the probability distribution from which $\mathcal{Y}_t$ is taken, and then drawing samples from this distribution. For example, if the vector $\mathbf{x}_n = [x_1^{[n]}, x_2^{[n]}, \ldots, x_i^{[n]}, \ldots, x_d^{[n]}]^\top$ is binary, it can be perturbed by randomly flipping each bit $x_d^{[n]}$ with a fixed probably ρ, in which case

$$\mathrm{P}\big(\mathbf{x}\big|\mathbf{x}_n\big) = \prod_{i}^{d} \rho^{(1-\delta_n^i)}(1-\rho)^{\delta_n^i}, \tag{33}$$

where the Kronecker delta $\delta_n^i = 1$ if $x_i = x_i^{[n]}$, and zero otherwise. Likewise, drawing perturbed samples from $\mathcal{Y}_t$ is equivalent to sampling from

$$\mathrm{P}\big(\mathbf{x}\big|\mathcal{Y}_t\big) = \frac{1}{M}\sum_{m=1}^{M}\prod_{i}^{d} \rho^{(1-\delta_m^i)}(1-\rho)^{\delta_m^i}. \tag{34}$$

We can regard (33) as a kernel function centered on $\mathbf{x}_n$ with a concentration parameter ρ, and (34) as a a normalized sum of M kernels centered on the M vectors of $\mathcal{Y}_t$. Estimating the probability distribution from which $\mathcal{Y}_t$ can be seen as equivalent to finding

$$\boldsymbol{\theta}_t = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{m=1}^{M} \log \mathrm{P}\big(\mathbf{x}_{y_t(m)}\big|\boldsymbol{\theta}\big) \tag{35}$$

where the parameters $\boldsymbol{\theta}$ specify the centers of M kernels. This is a simple matter of setting a center to each data points in $\mathcal{Y}$. Choosing $\boldsymbol{\theta}_t$ according to (35), and then drawing samples from $\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}_t\big)$ is equivalent to drawing perturbed samples from $\mathcal{Y}_t$.

The SHC method as a whole can be viewed as assuming that the distribution $\mathrm{P}\big(\mathcal{X}\big|\boldsymbol{\theta}\big)$ is parameterized by a set $\boldsymbol{\theta}$ that specify the centers of M kernels, as in (33). Beginning with a random initial parameter setting $\boldsymbol{\theta}_0$, samples $\{\tilde{x}_{\tilde{y}_0(\tilde{m})}\}_{\tilde{m}}^{\tilde{M}}$ are drawn from $\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}_0\big)$. These samples are evaluated according to $\phi\colon \mathcal{X} \mapsto \mathbb{R}$ to form the set $\{\phi(\tilde{\mathbf{x}}_{y_0(\tilde{m})})\colon \phi(\tilde{\mathbf{x}}_{y_0(\tilde{m})}) > \gamma_0\}$ where γ_0 is the median value of $\{\phi(\tilde{x}_{\tilde{y}_0(\tilde{m})})\}_{\tilde{m}}^{\tilde{M}}$. The probability distribution $\{\phi(\tilde{\mathbf{x}}_{y_0(\tilde{m})})\colon \phi(\tilde{\mathbf{x}}_{y_0(\tilde{m})}) > \gamma_0\}$ is then estimated, and the procedure is repeated. This is again a version of the general procedure described above, with the exception that a threshold of $\phi(\mathbf{x})$ is taken.

3.3 Population-Based Incremental Learning

The Population-Based Incremental Learning (PBIL) procedure, [1, 2], is intended as a generalization of the genetic algorithm. Instead of each iteration t of the algorithm being described by a population or subset of the solution space, it is summarized by a multi-variate Bernoulli probability vector $\boldsymbol{\theta}_t = [p_{t1}, p_{t2}, \ldots, p_{ti}, \ldots, p_{td}]$, with each $p_{ti} \in [0, 1]$. Samples based on this probability vector are drawn according to

$$\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}_t\big) = \prod_{i=1}^{d} p_{ti}^{x_i}(1 - p_{ti})^{1-x_i}, \tag{36}$$

to form $\{\tilde{x}_m\}_{m=1}^{M}$. These samples are evaluated according $\phi\colon \mathcal{X} \mapsto \mathbb{R}$. The new probability vector

$$\boldsymbol{\theta}_{t+1} = [p_{t+1,1}, p_{t+1,2}, \ldots, p_{t+1,i}, \ldots, p_{t+1,d}]$$

is updated by calculated by

$$p_{t+1,i} = \frac{\sum_{m=1}^{M} x_i^{[m]} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t)}{\sum_{m=1}^{M} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t)}, \tag{37}$$

where γ_t is a percentile threshold score. From this, the Bernoulli probability vector is updated as the proportion of times that $x_i^{[m]}$ takes the value 1 amongst the fit members of the population.

If we assume a probability model $\mathrm{P}\big(\mathbf{x}_m\big|\boldsymbol{\theta}\big)$ in (21) that is a multi-variate Bernoulli distribution, then it is easily verified that the solution to (21) is

$$p_{t+1,i} = \frac{\sum_{m=1}^{M} x_i^{[m]} \phi(\mathbf{x}_m)}{\sum_{m=1}^{M} \phi(\mathbf{x}_m)}, \tag{38}$$

It is clear that the PBIL update method in (37) is closely related to this, but with the additional feature that the evaluated samples are ranked and only those above a threshold are used for the probability model update.

3.4 Cross-Entropy Method

The cross-entropy (CE) method, [12], is a method for rare-event simulation, or calculating expectations $\int dx\, \psi(x)\mathrm{P}(x)$ when $\psi(x) \approx 0$ for most x. It is based on iteratively calculating optimal importance sampling distributions. Applied to an optimization problem as in Section 1, the CE method considers this problem analogous to finding an optimal distribution $\mathrm{P}\Big(\mathbf{x}|\hat{\boldsymbol{\theta}}\Big)$ that would sample from states in $\mathcal{X}$ that are close to optimal values according to $\phi\colon \mathcal{X} \mapsto \mathbb{R}$. Applying their method for deriving an optimal importance sampler to finding $\hat{\boldsymbol{\theta}}$ they obtain iterations

$$\boldsymbol{\theta}^{t+1} \leftarrow \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{m=1}^{M} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t) \log \mathrm{P}\big(\tilde{\mathbf{x}}_m\big|\boldsymbol{\theta}\big)\,, \tag{39}$$

where $\{\mathbf{x}_m\}_{m=1}^M$ are drawn from $\sim \mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}^{t-1}\big)$, and γ_t is again the $1-\epsilon$-quantile score of the set $\{\mathbf{x}_m\}_{m=1}^M$ according to $\phi\colon \mathcal{X} \mapsto \mathbb{R}$ ($\mathbb{I}$ is an indicator function, and equal to one if its argument is true). In examples, they consider the case, like with PBIL, where $\boldsymbol{\theta}$ is a multi-variate Bernoulli vector, i.e. $\boldsymbol{\theta}_t = [p_{t1}, p_{t2}, \ldots, p_{ti}, \ldots, p_{td}]$. In this case, (39) leads to the update procedure

$$p_{t+1,i} = \frac{\sum_{m=1}^M x_i^{[m]} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t)}{\sum_{m=1}^M \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t)}. \tag{40}$$

If the CE method is taken in its most general form, it is precisely a version of the general method we describe in Section 2. The only difference between the two methods is that in CE the score according to ϕ is thresholded. The performance of CE in any problem will depend largely upon the probability model in (39). In many examples, the probability models are chosen for their analytical simplicity. Such is the case when the probability model is taken to be a multi-variate Bernoulli vector. While this choice greatly facilitates the implementation of algorithms, it can weaken the method and will lead to poor performance in difficult problems, as will be explained in Section 4.

3.5 Non-independent stochastic search methods

In GAs, PBIL, and the example implementations of the CE that we described, the probability model on which they are based is a multivariate Bernoulli distribution. This model assumes that the individual elements of a candidate vector $\mathbf{x}_m$ contribute independently to the fitness of the fitness $\phi(\mathbf{x}_m)$. In order to introduce more general probability models, new methods have been proposed. For example, the works by [4], [3], and [11] have noted the inadequacy of the independence assumptions inherent in the multivariate Bernoulli probability distribution, and have proposed alternatives that model interactions between elements. While the multivariate Bernoulli model assumes that the distribution over the state-space $\mathcal{X}$ factors as the product of marginal distributions, i.e.

$$\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}\big) = \prod_{i=1}^d \mathrm{P}\big(x_i\big|\boldsymbol{\theta}\big)\,, \tag{41}$$

the model proposed in both [4] and [3] assumes

$$\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}\big) = \mathrm{P}\big(x_{\eta_1}\big|\boldsymbol{\theta}\big) \prod_{i=1}^{d-1} \mathrm{P}\big(x_{\eta_{i+1}}\big|x_{\eta_i}\boldsymbol{\theta}\big)\,, \tag{42}$$

for some permutation η of the integers 1 to d, and the model proposed by [11] assumes

$$\mathrm{P}\big(\mathbf{x}\big|\boldsymbol{\theta}\big) = \prod_{i=1}^{d-1} \mathrm{P}\big(x_i\big|\mathsf{PA}(x_i), \boldsymbol{\theta}\big)\,, \tag{43}$$

where $\mathsf{PA}(x_i)$ denote the graphical-model parents of variable x_i. In each case, these methods proceed by fitting their probability models to the fit members

of the population. Ideally this done by finding the optimal set of conditional dependencies. However, greedy heuristics are used to avoid the potential complexity of these searches.

Algorithm 2 General form of Genetic Algorithms, Stochastic Hill-climbing, PBIL, Cross-Entropy method and Non-independent search methods. The distinction between these methods lies in the assumption of the form of $\mathrm{P}\big(\tilde{\mathbf{x}}_m\big|\boldsymbol{\theta}\big)$ in the Maximization step (The γ_t term is variable threshold).

Initialize: $t = 0$, $\boldsymbol{\theta}^t \leftarrow \boldsymbol{\theta}_{\mathsf{rand}}$, $\mathsf{converge} = 0$

while $\neg\mathsf{converge}$ **do**

Sample:

$$\{\tilde{\mathbf{x}}_m^t\}_{m=1}^M \sim \mathrm{P}\big(\tilde{\mathbf{x}}\big|\boldsymbol{\theta}^t\big)$$

Evaluate:

$$\{\phi(\tilde{\mathbf{x}}_m^t)\}_{m=1}^M$$

Maximize:

$$\boldsymbol{\theta}^{t+1} \leftarrow \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{\tilde{m}=1}^{\tilde{M}} \mathbb{I}(\phi(\tilde{\mathbf{x}}_m) > \gamma_t) \log \mathrm{P}\big(\tilde{\mathbf{x}}_m\big|\boldsymbol{\theta}\big)$$

if $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t$ **then**

$\mathsf{converge} = 1$

else

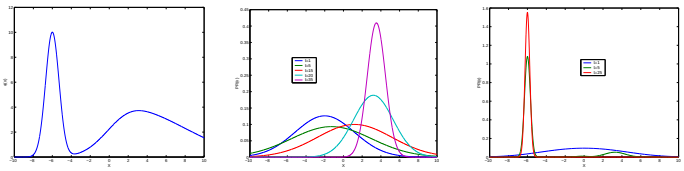
$t = t + 1$

end if

end whilemememe

4 The Role of the Probability Model

The adequacy of the general procedure described in Section 2, and of the special cases described in Section 3, depends to an important extent upon the probability models that are implicitly or explicitly assumed. For example, assuming that individual elements of a candidate vector $\mathbf{x}_m$ contribute independently to the fitness of the fitness $\phi(\mathbf{x}_m)$ is a poor general model, and will perform poorly in many problems. Consider the cases in the following table: A probability model that assumes that each element acts independently will lead to sub-optimal performance. The problem represented here is a classical example of the XOR problem: Either the first two elements of the vector are positive, or else the last two are positive. The second and third vectors have high fitness, but the average of these two vectors has a low fitness. The fitness landscape of the XOR problem is bi-modal, but the probability model is unimodal.



(a) Objective: $\phi(\mathbf{x})$

(b) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ in a non mixture model at $t = \{1, 5, 15, 20, 25\}$.

(c) $\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}^t)$ in mixture model at $t = \{1, 5, 25\}$.

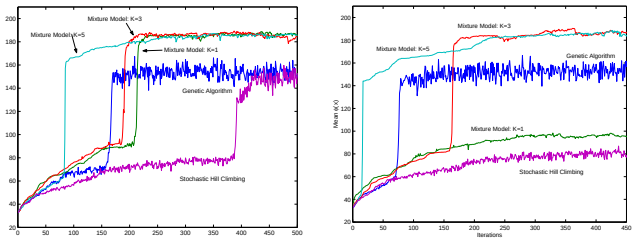
Figure 2: A comparison a mixture and non-mixture model on a bimodal optimization problem.

$\mathbf{x}$	$\phi(\mathbf{x})$
0000	Bad
0011	Good
1100	Good
1111	Bad
$\boldsymbol{\theta} = [.5.5.5.5]$	Bad

4.1 Comparison of Mixture and Nonmixture Probability Models

This point can be illustrated using the toy problem described previously in Section 2. Consider the objective function shown in Figure 2 (a). This problem has an obvious global maximum, and one considerably suboptimal local maximum. Subfigure (b) shows the performance of a single Gaussian model. The single Gaussian initially spreads its mass intermediate between both maxima. The sub-optimal maximum in this problem has a wide but shallow base. This eventually dominates the model, exerting more influence than the narrow base of the global maximum. The model eventually converges towards the suboptimal maximum. On the other hand, Subfigure (c) shows the performance of a mixture of $k = 3$ Gaussians. The flexibility afforded by the mixture model allows the model to place mass in the vicinity of both maxima. Eventually the global maximum dominates.

This point can be further illustrated by considering the Four-Peaks and Six-Peaks problems. The Four peaks problem is introduced in [2]. Binary vectors are assigned scores depending on the number of consecutive zeros beginning at position x_1, and consecutive ones ending a position x_d. If $\overrightarrow{\mathsf{zero}}$ is the number of



(a) Four-Peaks (b) Six-Peaks

Figure 3: A comparison of five models on the Four-Peaks and Six-Peaks problems. In both cases, the candidate binary-vectors were of length 100 and the $\nu = 10$.

zeros beginning at x_1, and $\overleftarrow{\mathsf{one}}$ is the of ones ending on x_d, the score is

$$\phi(\mathbf{x}) = \mathsf{max}(\overrightarrow{\mathsf{zero}}, \overleftarrow{\mathsf{one}}) + \begin{cases} d, & \text{if } \overrightarrow{\mathsf{zero}} > \nu \wedge \overleftarrow{\mathsf{ones}} > \nu \ , \\ 0, & \text{otherwise,} \end{cases} \tag{44}$$

where ν is a variable parameter. The Six Peaks is a generalization of the Four Peaks problem and introduced by [4]. It is based on binary vectors and $\overrightarrow{\mathsf{zero}}$ and $\overleftarrow{\mathsf{one}}$ are defined as above, while $\overrightarrow{\mathsf{one}}$ and $\overleftarrow{\mathsf{zero}}$ are the the number of consecutive ones beginning at position x_1, and consecutive zeros ending a position x_d. The objective function is

$$\begin{aligned} \phi(\mathbf{x}) =& \mathsf{max}(\overrightarrow{\mathsf{zero}}, \overleftarrow{\mathsf{one}}, \overrightarrow{\mathsf{one}}, \overleftarrow{\mathsf{zero}}) \\ &+ \begin{cases} d, & \text{if } \overrightarrow{\mathsf{zero}} > \nu \wedge \overleftarrow{\mathsf{ones}} > \nu \ , \\ d, & \text{if } \overrightarrow{\mathsf{ones}} > \nu \wedge \overleftarrow{\mathsf{zeros}} > \nu \ , \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{45}$$

The role of the probability model in these problems can be illustrated by comparing multivariate Bernoulli models with mixtures of multivariate Bernoulli models. A mixture of K multivariate Bernoulli distributions is

$$\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \prod_{i=1}^{d} p_{ki}^{x_i} \ (1-p_{ki})^{(1-x_i)} \ . \tag{46}$$

where $\{p_{k1}, p_{k2}, \ldots, p_{ki}, \ldots, p_{kd}\}$ is a set multivariate Bernoulli parameters, i.e. $p_{ki} \in [0, 1]$, and $\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_k, \ldots, \pi_K]$ is a multinomial distribution specifying the probability of each of the K distributions. Using this model the computation of (21) can be performed again using an EM algorithm for (weighted) maximum likelihood estimation.

In comparison to the Four-Peak problem, the Six-Peak problem has symmetrically opposite maxima, i.e. both $\overrightarrow{\mathsf{zero}} > \nu \wedge \overleftarrow{\mathsf{ones}} > \nu$ and $\overrightarrow{\mathsf{ones}} > \nu \wedge \overleftarrow{\mathsf{zeros}} > \nu$ have high value. A single multivariate Bernoulli model will be caught between these two maxima. By contrast, the Four-Peak problem does not have these diametrically opposite maxima. A single multivariate model will perform relatively better on this problem.

This is seen in Figure 3 (a) and (b), where we show the average performance of 5 models over 10 separate runs. The models in each case are a single multivariate Bernoulli model, a $k = 3$ mixture model, and a $k = 5$ mixture model. For purposes of comparison, we also include a stochastic hill-climber (mutation rate of .01) and a single crossover Genetic algorithm (also with a mutation rate of .01). In the Six-Peaks problem, the mixture models ($k = 3$ and $k = 5$) perform well, while the single multivariate model is on average unable to attain high rates of fitness. By contrast, in the Four-Peak problem the single multivariate model performs almost as well as the mixture models. These results seem to imply that problems which can be characterized as multi-modal are difficult for methods that assume unimodal probability distributions.

5 Conclusion

The purpose of this paper is threefold. First, we explicitly describe optimization in probabilistic terms as maximizing the functional

$$\mathcal{E}(\theta) = \log \sum_{i=1}^{N} \phi(\mathbf{x}_i) \mathrm{P}(\mathbf{x}_i \big| \boldsymbol{\theta}),$$

and derive a iterative method for this problem that is based upon the generalized EM algorithm, [10, 6]. Second, we show that many population based optimization procedures implicitly implement this procedure. These methods include GAs, SHCs, PBIL, CE, MIMIC and BOA. Third, we aim to use this perspective to provide insight into the behavior of population-based stochastic optimization methods. For example, we show that one of the important factors that differentiate between population-based search methods is the probability model that they (explicitly or implicitly) assume. We demonstrated this in Section 3. In the final section, we demonstrated that population-based method that use ill-suited probability models can perform poorly. For example, we showed how population-based methods that assume unimodal distributions can perform poorly on multi-modal optimization problems.

References

[1] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,. Technical Report CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, PA, 1994.

[2] S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In A. Prieditis and S. Russel, editors, *The Int. Conf. on Machine Learning 1995*, pages 38–46, San Mateo, CA, 1995. Morgan Kaufmann Publishers.

[3] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. In *International Conference on Machine Learning*, 1997.

[4] J. S. D. Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding optima by estimating probability densities. In *Advances in Neural Information Processing Systems*. MIT Press, 1997.

[5] P. Dayan and G. E. Hinton. Using EM for reinforcement learning. *neural computation*, 9:271–278, 1997.

[6] M. M. Dempster, N. M. Laird, and D. B. Jain. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, pages 1–38, 1977.

[7] G. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40:185–234, 1989.

[8] M. Mitchell. *An Introduction to Genetic Algorithms*. Bradford Books, 1998.

[9] M. Mitchell, J. H. Holland, and S. Forrest. When will a genetic algorithm outperform hill climbing. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 51–58. Morgan Kaufmann Publishers, Inc., 1994.

[10] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer Academic Publishers, 1998.

[11] M. Pelikan, D. E. Goldberg, and E. Cant-Paz. BOA the bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, pages 525–532, 1999.

[12] R. Rubinstein and D. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. Springer-Verlag, 2004.