Probability and Random Processes

Project Report on

"Expanding From Discrete To Continuous Estimation Of Distribution Algorithms"

GROUP - MEMBERS

1. Shriyanshi Srivastava	**9919103158**
2. Prakash Singh Rautela	**9919103160**
3. Devendra Singh Chauhan	**9919103161**
4. Sachin Kumar Chauhan	**9919103175**

SUBMITTED TO: ***DR. AMITA BHAGAT***

Acknowledgment

We would like to start this project by expressing our special thanks of gratitude to our mentor, *Dr. Amita Bhagat* for giving us the opportunity to study and present our findings on the topic "Discrete and Continuous Probability Distribution".

This project helped us to understand different areas of real life application of probability distribution. We would like to extend our sincere and heartfelt gratitude towards our mentor who has helped us in this endeavor with her active guidance, cooperation and encouragement.

At last, we would like to thank all our friends and family who directly or indirectly helped us in completing this project report.

Index

Introduction

Estimation of Distribution Algorithms (EDAs) is a new phase of Evolutionary Computation. Algorithms in evolutionary optimization guide their search through statistics based on a vector of samples, often called a population. By using this stochastic information, non–deterministic induction is performed in order to attempt to use the structure of the search space and thereby aid the search for the optimal solution. In order to perform induction, these samples are combined so as to generate new solutions that will hopefully be closer to the optimum. As this process is iterated, convergence is intended to lead the algorithm to a final solution.

In the genetic algorithm [11, 14] and many variants thereof, values for problem variables are often exchanged and subsequently individually adapted. Another way of combining the samples is to regard them as being representative of some probability distribution. Estimating this probability distribution and sampling more solutions from it, is a global statistical type of inductive iterated search. Such algorithms have been proposed for discrete spaces [2–4, 12, 13, 15, 17, 19, 21], as well as in a limited way for continuous spaces [5, 10, 15, 22, 23]. An overview of this field has been given by Pelikan, Goldberg and Lobo [20].

Objective

Our goal in this paper is to apply the search for good probability density models to continuous spaces. To this end, we formalize the notion of building and using probabilistic models in a framework named IDEA. We show how we can adjust existing techniques to be used in the continuous case. We thereby define evolutionary optimization algorithms. Using a set of test functions, we validate their performance.

The remainder of this paper is organized as follows:
In section 3, we present the IDEA framework. In section 4, we describe a few existing algorithms that build and use probabilistic models. In section 5, we state some derivations of probability density functions (pdfs). Our final conclusions are drawn in section 6.

The IDEA Framework

We write $a = (a0, a1,..., a|a|-1)$ for a vector a of length $|a|$. The ordering of the elements in a vector is relevant. We assume to have l random variables available, meaning that each sample point is an l dimensional vector. We introduce the notation $ac = (ac0 , ac1 ,...,ac|c|-1)$ Let $L = (0, 1,...,l - 1)$ be a vector of l numbers and let $Z = (Z0, Z1,...,Zl-1)$ be a vector of l random variables. We assume that we have an l dimensional cost function C(zL) which without loss of generality we seek

to minimize. Without any prior information on C(zL), we might as well assume a uniform distribution over Z. Now denote a probability distribution that is uniformly distributed over all zL with C(zL) ≤ θ and that has a probability of 0 otherwise, by Pθ(Z). In the discrete case we have:

$$P^{\theta}(\boldsymbol{\mathcal{Z}})(z\langle\boldsymbol{\mathcal{L}}\rangle) = \begin{cases} \frac{1}{|\{z'\langle\boldsymbol{\mathcal{L}}\rangle | C(z'\langle\boldsymbol{\mathcal{L}}\rangle) \leq \theta\}|} & \text{if } C(z\langle\boldsymbol{\mathcal{L}}\rangle) \leq \theta \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Note that if we find Pθ∗ (Z) where θ∗ = minzL{C(zL)}, a single sample drawn from Pθ∗ (Z) provides an optimal solution zL∗ . A probability distribution is made up of a probability density structure (pds) and a probability density function (pdf) for each element in the pds. In graphical models literature, a pds is also called a factorization. Let ab be the splicing of a and b such that the elements of b are placed behind the elements of a, giving |ab| = |a|+|b|. Using graphical modelling [9], we can denote any non–clustered pds with conditional probabilities P(Za|Zb) = P(Zab)/P(Zb). We let π(·) be a function that returns a vector π(i)=(π(i)0, π(i)1,...,π(i)|π(i)|−1) of indices denoting the variables that Zi is conditionally dependent on. We call the graph that results when taking the Zi as nodes and having an arc from node Zi to node Zj if and only if i ∈ π(j), the pds graph. The only required condition to be able to express any non–clustered pds using conditional probabilities, is that this graph needs to be acyclic. By using a permutation vector ω, the definition of a pds (π, ω) that models conditional factorizations, can be formalized as follows:

$$P_{(\pi,\boldsymbol{\omega})}(\boldsymbol{\mathcal{Z}}) = \prod_{i=0}^{l-1} P(Z_{\boldsymbol{\omega}_i} | Z\langle\pi(\boldsymbol{\omega}_i)\rangle) \quad (2)$$

$$\text{such that } \forall_{i\in\boldsymbol{\mathcal{L}}}\langle\boldsymbol{\omega}_i \in \boldsymbol{\mathcal{L}} \wedge \forall_{k\in\boldsymbol{\mathcal{L}}-(i)}\langle\boldsymbol{\omega}_i \neq \boldsymbol{\omega}_k\rangle\rangle$$

$$\forall_{i\in\boldsymbol{\mathcal{L}}}\langle\forall_{k\in\pi(\boldsymbol{\omega}_i)}\langle k \in \{\boldsymbol{\omega}_{i+1}, \boldsymbol{\omega}_{i+2}, \ldots, \boldsymbol{\omega}_{l-1}\}\rangle\rangle$$

Each P(Zωi |Z π(ωi)) from equation 2 is a special case multivariate conditional pdf. This means that any such conditional pdf along with a pds (π, ω) defines a probability distribution over Z. In general, we denote a pds by f. The pds is constrained to be of a certain form. For instance, in the case of equation 2, the constraints impose f to describe a directed acyclic graph. We denote the constrained space of all possible structures by C. A probability distribution over Z is then formally denoted by Pf(Z), f ∈ C. Denote the largest function value of a selection of samples at iteration t by θt. We find a pds and estimate each pdf to best approximate Pθt (Z). We can then sample from the resulting probability distribution to get more

samples. By formalizing this rationale in an iterative algorithm, we define the Iterated Density Estimation Evolutionary Algorithm (IDEA):

$\mathrm{IDEA}(n, \tau, m, sel(), rep(), ter(), sea(), est(), sam())$	
Initialize an empty vector of samples	$\boldsymbol{\mathcal{P}} \leftarrow ()$
Add and evaluate n random samples	***for*** $i \leftarrow 0$ ***to*** $n-1$ ***do***
	$\quad \boldsymbol{\mathcal{P}} \leftarrow \boldsymbol{\mathcal{P}} \sqcup \textsc{NewRandomVector}()$
	$\quad c[\boldsymbol{\mathcal{P}}_i] \leftarrow C(\boldsymbol{\mathcal{P}}_i)$
Initialize the iteration counter	$t \leftarrow 0$
Iterate until termination	***while*** $\neg ter()$ ***do***
Select $\lfloor \tau n \rfloor$ samples	$\quad (z^0\langle \boldsymbol{\mathcal{L}} \rangle, z^1\langle \boldsymbol{\mathcal{L}} \rangle, \ldots, z^{\lfloor \tau n \rfloor - 1}\langle \boldsymbol{\mathcal{L}} \rangle) \leftarrow sel()$
Set θ_t to the worst selected cost	$\quad \theta_t \leftarrow c[z^k\langle \boldsymbol{\mathcal{L}} \rangle]$ such that
	$\qquad \forall_{i \in \boldsymbol{\mathcal{N}}_\tau} \langle c[z^i\langle \boldsymbol{\mathcal{L}} \rangle] \leq c[z^k\langle \boldsymbol{\mathcal{L}} \rangle] \rangle$
Search for a pds $\mathfrak{f}$	$\quad \mathfrak{f} \leftarrow sea()$
Estimate each pdf in $\hat{P}_\mathfrak{f}(\boldsymbol{\mathcal{Z}})$	$\quad \{\hat{P}(\cdot) \vert \hat{P}(\cdot) \Leftarrow \hat{P}_\mathfrak{f}(\boldsymbol{\mathcal{Z}})\} \leftarrow est()$
Create an empty vector of new samples	$\quad \boldsymbol{\mathcal{O}} \leftarrow ()$
Sample m new samples from $\hat{P}_\mathfrak{f}(\boldsymbol{\mathcal{Z}})$	$\quad$ ***for*** $i \leftarrow 0$ ***to*** $m-1$ ***do***
	$\qquad \boldsymbol{\mathcal{O}} \leftarrow \boldsymbol{\mathcal{O}} \sqcup sam()$
Replace a part of $\boldsymbol{\mathcal{P}}$ with a part of $\boldsymbol{\mathcal{O}}$	$\quad rep()$
Evaluate the new samples in $\boldsymbol{\mathcal{P}}$	$\quad$ ***for each*** unevaluated $\boldsymbol{\mathcal{P}}_i$ ***do***
	$\qquad c[\boldsymbol{\mathcal{P}}_i] \leftarrow C(\boldsymbol{\mathcal{P}}_i)$
Update the generation counter	$\quad t \leftarrow t+1$
Denote the required iterations by t_{end}	$t_{\text{end}} \leftarrow t$

In the IDEA framework, we have that Nτ = (0, 1,..., τn − 1), τ ∈ [1 n , 1], sel() is the selection operator, rep() replaces a subset of P with a subset of O, ter() is the termination condition, sea() is a pds search algorithm, est() estimates each pdf and sam() generates a single sample from Pˆf(Z). The notation Pˆ(·) ⇐ Pˆf means that Pˆ(·) is one of the pdfs that is implied by the model f. The IDEA is a true evolutionary algorithm in the sense that a population of individuals is used from which individuals are selected to generate new offspring with. Using these offspring along with the parent individuals and the current population, a new population is constructed. By referring to the iterations in the IDEA as generations, the evolutionary correspondence is even more obvious. Note that in the IDEA, we have used the approximation notation Pˆθt f (Z) instead of the true distribution Pθt f (Z). An approximation is required because the determined distribution is based upon samples and the underlying density model is an assumption on the true model. This means that even though we might achieve Pˆθt f (Z) = Pθt f (Z), in general this is not the case. If we set m to (n − τn), sel() to selection by taking the best τn vectors and rep() to replacing the worst (n − τn) vectors by the new sampled vectors, we have that θk+1 = θk − ε with ε ≥ 0. This assures that the search for θ∗ is conveyed through a monotonically decreasing series θ0 ≥ θ1 ≥ ... ≥ θt end. We call an IDEA with m, sel() and rep() so chosen, a monotonic IDEA. If we set m in the IDEA to n and set rep() to replace P with O, we obtain the EDA by M¨uhlenbein, Mahnig and

Rodriguez [17]. In the EDA however, the threshold θt cannot be enforced. Note how EDA is thus an instance of IDEA.

Probability density structure search algorithms

In order to search for a pds, a metric is required that guides the search. In effect, this poses another optimization problem. The metric we use in this paper is a distance metric to the full joint pds (π+, ω+), $\forall i \in L \;\; \omega^+ i = i \wedge \pi^+(i) = (i + 1, i + 2, \ldots, l - 1)$. The distance metric is defined by the Kullback–Leibler (KL) divergence. We write Y instead of Z from now on to indicate the use of continuous random variables instead of either the discrete or continuous case. Using our definitions, the KL divergence can be written as [7]:

$$D(\hat{P}_{(\pi^+,\boldsymbol{\omega}^+)}(\boldsymbol{\mathcal{Y}})||\hat{P}_{(\pi,\boldsymbol{\omega})}(\boldsymbol{\mathcal{Y}})) = -h(\hat{P}_{(\pi^+,\boldsymbol{\omega}^+)}(\boldsymbol{\mathcal{Y}})) + \sum_{i=0}^{l-1} h(\hat{P}(Y_{\boldsymbol{\omega}_i}|Y\langle\pi(\boldsymbol{\omega}_i)\rangle)) \quad (3)$$

Let a L, b L where a L means that a contains only elements of L. In equation 3, h(Y a) is the multivariate differential entropy and h(Y a|Y b) is the conditional differential entropy. Let

$$dy\langle \boldsymbol{a} \rangle = \prod_{i=0}^{|\boldsymbol{a}|-1} dy_i$$

be shorthand notation for the multivariate derivative. We then have:

$$h(P(Y\langle \boldsymbol{a} \rangle)) = -\int P(Y\langle \boldsymbol{a} \rangle)(y\langle \boldsymbol{a} \rangle)\ln(P(Y\langle \boldsymbol{a} \rangle)(y\langle \boldsymbol{a} \rangle))dy\langle \boldsymbol{a} \rangle \quad (4)$$

$$h(P(Y\langle \boldsymbol{a} \rangle|Y\langle \boldsymbol{b} \rangle)) = h(P(Y\langle \boldsymbol{a} \sqcup \boldsymbol{b} \rangle)) - h(P(Y\langle \boldsymbol{b} \rangle)) \quad (5)$$

As the term

$$h(\hat{P}_{(\pi^+,\boldsymbol{\omega}^+)}(\boldsymbol{\mathcal{Y}}))$$

in equation 3 is constant, an algorithm that searches for a pds can use the KL divergence by minimizing the sum of the conditional entropies imposed by (π, ω). This will cause the pds search algorithm to search for a pds as close as possible to (π+, ω+) subject to additional constraints. The probabilistic models used in previously proposed algorithms range from lower order structures to structures of unbounded complexity. It has been empirically shown by Bosman and Thierens [6] that a higher order pds is required to solve higher order building block problems. We

shortly state three previously introduced pds search algorithms that we use in our experiments. In the univariate distribution, all variables are regarded independently of each other. The PBIL by Baluja and Caruana [2], the cGA by Harik, Lobo and Goldberg [13], the UMDA by M¨uhlenbein and Paaß [18], and all known approaches in the continuous case prior to the IDEA [10, 22, 23], use this pds. It can be modelled by $\forall i \in L\ \pi(i) = () \wedge \omega i = i$, giving:

$$\hat{P}_{(\pi,\boldsymbol{\omega})}(\boldsymbol{\mathcal{Z}}) = \prod_{i=0}^{l-1} \hat{P}(Z_i).$$

In the MIMIC algorithm by De Bonet, Isbell and Viola [4], the pds is a chain which is constrained to $\pi(\omega l-1) = () \wedge \forall i \in L-(l-1)\ \pi(\omega i)=(\omega i+1)$, giving

$$\hat{P}_{(\pi,\boldsymbol{\omega})}(\boldsymbol{\mathcal{Z}}) = (\prod_{i=0}^{l-2} \hat{P}(Z_{\boldsymbol{\omega}_i}|Z_{\boldsymbol{\omega}_{i+1}}))\hat{P}(Z_{\boldsymbol{\omega}_{l-1}}).$$

To find the chain, an O(l 2) greedy approximation algorithm is used in MIMIC to minimize the KL divergence. If the pds is constrained so that in addition to having an acyclic pds graph, each node may have at most κ parents, the pds is constrained to $\forall i \in L\ |\pi(i)| \leq \kappa$. This general approach is used in the BOA by Pelikan, Goldberg and Cant´u– Paz [19], as well as the LFDA by M¨uhlenbein and Mahnig [16] and the EBNA by Larra˜naga, Etxeberria, Lozano and Pe˜na. In the case of $\kappa = 1$, a polynomial time algorithm can be used to minimize the KL divergence [5]. In the case of $\kappa > 1$, a greedy algorithm is used that iteratively adds arcs to the pds graph. There are other special case algorithms, such as the optimal dependency trees approach by Baluja and Davies [3] and the ECGA by Harik [12]. Like the LFDA, the ECGA uses minimum description length as a search metric. This metric has the advantage that the resulting pds will not be overly complex. Using the KL divergence, this can only be influenced by adjusting κ because the KL divergence is merely a distance measure from a certain pds to (π+, ω+). We only regard the three described sea() algorithms in combination with the KL divergence metric. Note that using the KL metric and the (π+, ω+) pds is merely an instance of the IDEA framework. This is also the case for using a certain pdf. The framework is to be seen separately from the algorithms that can be modelled by it.

Probability density functions

Next to the pds search algorithms from section 3, we require to specify a pdf to use. It follows from sections 2 and 3 that we require to know the multivariate differential entropy as well as the conditional pdf. In this section, we specify two well known pdfs that we use in our experiments within the IDEA framework. A widely used parametric pdf is the normal pdf. Let

$$\boldsymbol{\mathcal{S}} = (\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{|\boldsymbol{\mathcal{S}}|-1})$$

be the set of selected samples. The sample average in dimension j is then

$$\overline{Y_j} = \frac{1}{|\boldsymbol{\mathcal{S}}|} \sum_{i=0}^{|\boldsymbol{\mathcal{S}}|-1} \boldsymbol{y^i}_j.$$

The sample covariance matrix over variables y <a> is

$$\frac{1}{|\boldsymbol{\mathcal{S}}|} \sum_{i=0}^{|\boldsymbol{\mathcal{S}}|-1} (y^i\langle \boldsymbol{a} \rangle - \overline{Y}\langle \boldsymbol{a} \rangle)(y^i\langle \boldsymbol{a} \rangle - \overline{Y}\langle \boldsymbol{a} \rangle)^T.$$

Let

$$s'_{ij} = \boldsymbol{S}^{-1}(i,j).$$

The conditional pdf and the entropy can be stated as follows [5]:

$$f_{\mathcal{N}}(y_{\boldsymbol{a}_0} | y\langle \boldsymbol{a} - \boldsymbol{a}_0 \rangle) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(y_{\boldsymbol{a}_0} - \mu)^2}{2\sigma^2}} \tag{6}$$

$$\text{where } \sigma = \frac{1}{\sqrt{s'_{00}}}, \quad \mu = \frac{\overline{Y_{\boldsymbol{a}_0}} s'_{00} - \sum_{i=1}^{|\boldsymbol{a}|-1} (y_{\boldsymbol{a}_i} - \overline{Y_{\boldsymbol{a}_i}}) s'_{i0}}{s'_{00}}$$

$$h(Y\langle \boldsymbol{a} \rangle) = \frac{1}{2}(|\boldsymbol{a}| + \ln((2\pi)^{|\boldsymbol{a}|}\det(\boldsymbol{S}))) \tag{7}$$

The non–parametric normal kernels pdf places a normal pdf over every available sample point. Let si be a fixed standard deviation in the i-th dimension. The conditional pdf and the entropy can then be stated as follows [8]:

$$f_{\mathcal{N}_K}(y_{\boldsymbol{a}_0} | y\langle \boldsymbol{a} - \boldsymbol{a}_0 \rangle) = \sum_{i=0}^{|\boldsymbol{\mathcal{S}}|-1} \boldsymbol{\nu}_i \frac{1}{\mathfrak{s}_{\boldsymbol{a}_0}\sqrt{2\pi}} e^{\frac{-(y_{\boldsymbol{a}_0} - \boldsymbol{y^i}_{\boldsymbol{a}_0})^2}{2s^2_{\boldsymbol{a}_0}}} \tag{8}$$

$$\text{where } \boldsymbol{\nu}_i = \frac{e^{-\sum_{j=1}^{|\boldsymbol{a}|-1} \frac{(y_{\boldsymbol{a}_j} - \boldsymbol{y^i}_{\boldsymbol{a}_j})^2}{2s^2_{\boldsymbol{a}_j}}}}{\sum_{k=0}^{|\boldsymbol{\mathcal{S}}|-1} e^{-\sum_{j=1}^{|\boldsymbol{a}|-1} \frac{(y_{\boldsymbol{a}_j} - \boldsymbol{y^k}_{\boldsymbol{a}_j})^2}{2s^2_{\boldsymbol{a}_j}}}}$$

$$h(Y\langle \boldsymbol{a} \rangle) = \tag{9}$$

$$\frac{1}{2}\ln\left(|\boldsymbol{\mathcal{S}}|^2(2\pi)^{|\boldsymbol{a}|}\prod_{j=0}^{|\boldsymbol{a}|-1}\mathfrak{s}^2_{\boldsymbol{a}_j}\right) - \int f_{\mathcal{N}_K}(y\langle \boldsymbol{a} \rangle)\ln\left(\sum_{i=0}^{|\boldsymbol{\mathcal{S}}|-1} e^{-\sum_{j=0}^{|\boldsymbol{a}|-1} \frac{(y_{\boldsymbol{a}_j} - \boldsymbol{y^i}_{\boldsymbol{a}_j})^2}{2s^2_{\boldsymbol{a}_j}}}\right) dy\langle \boldsymbol{a} \rangle$$

An alternative pdf to the two described above, is the histogram pdf. Using this pdf however does not scale up very well [7] and leads to an exponential iteration running time in the order of rκ where r is the amount of bins to use in each dimension. The normal pdf is very efficient but very cluster insensitive. The normal kernels pdf is very sensitive to clusters but may very quickly overfit the data. In addition, the running time each iteration for using the latter pdf tends to be a lot greater than for the normal pdf.

Conclusion

In this project, we have applied the search for good probability density models to continuous spaces. We have formalised the notion of building and using probabilistic models in a new framework named IDEA. We have shown how we can adjust existing techniques to be used in the continuous case.
We have utilised the algorithmic system IDEA for demonstrating iterated density estimation evolutionary algorithms. These calculations utilise density estimation procedures to probability distribution over the factors that code an issue to perform optimisation. To this end, a probability density structure should be found and therefore be utilised in density estimation. For a bunch of existing search algorithms, we have applied and tried them in the IDEA framework using two distinctive density estimation models.. The experiment demonstrates that building and utilising probabilistic models for continuous optimisation problems is promising. This, in blend with its demonstrating abilities, shows that the IDEA is general, appropriate and compelling.

References

1. https://link.springer.com/chapter/10.1007/3-540-45356-3_75

2. https://homepages.cwi.nl/~bosman/publications/2000_expandingfromdiscrete.pdf