Optimization in continuous domains by learning and simulation of Gaussian networks

Pedro Larrañaga, Ramon Etxeberria, José A. Lozano and José M. Peña
Intelligent Systems Group
Dept. of Computer Science and Artificial Intelligence
University of the Basque Country
E-20080 Donostia–San Sebastián, Spain
ccplamup@si.ehu.es, ramon@edunet.es, lozano@si.ehu.es, ccbpepaj@si.ehu.es

Abstract

This paper shows how the Gaussian network paradigm can be used to solve optimization problems in continuous domains. Some methods of structure learning from data and simulation of Gaussian networks are applied in the Estimation of Distribution Algorithm (EDA) as well as new methods based on information theory are proposed. Experimental results are also presented.

1 Estimation of Distribution Algorithms approaches in continuous domains

Figure 1 shows a schematic of the EDA approach for continuous domains. We will use $\boldsymbol{x} = (x_1, \ldots, x_n)$ to denote individuals, and D_l to denote the population of N individuals in the l-th generation. Similarly, D_l^{Se} will represent the population of the selected Se individuals from D_l. In the EDA [9] our interest will be to estimate $f(\boldsymbol{x} \mid D^{Se})$, that is, the joint probability density function over one individual $\boldsymbol{x}$ being among the selected individuals. We denote as $f_l(\boldsymbol{x}) = f_l(\boldsymbol{x} \mid D_{l-1}^{Se})$ the joint density of the l-th generation. See [6], [7], [8], [10] for revisions of EDA approaches.

In all the works where the EDA approach has been proposed for optimization in continuous domains –[12], [13], [14]– the joint density function is factorable as a product of unidimensional and independent normal densities.

$$f_{\boldsymbol{X}}(\boldsymbol{x}; \boldsymbol{\mu}, \sum) = \prod_{i=1}^{n} f_{\mathcal{N}}(x_i; \mu_i, \sigma_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{1}{2}(\frac{x_i - \mu_i}{\sigma_i})^2}. \quad (1)$$

EDA
  $D_0 \leftarrow$ Generate N individuals randomly
  Repeat for $l = 1, 2, \ldots$ until a stopping criterion is met
    $D_{l-1}^{Se} \leftarrow$ Select $Se \leq N$ individuals from D_{l-1}
    $f_l(\boldsymbol{x}) \leftarrow$ Estimate the density function
    $D_l \leftarrow$ Sample N individuals from $f_l(\boldsymbol{x})$

Figure 1: Pseudocode for the EDA approach.

2 Gaussian networks

Let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a set of random variables. We will use x_i to denote a value of X_i, the i-th component of $\boldsymbol{X}$, and $\boldsymbol{y}$ to denote a value of $\boldsymbol{Y} \subseteq \boldsymbol{X}$. A probabilistic graphical model for $\boldsymbol{X}$ is a graphical factorization of the joint probability density function. The representation consists of two components: a structure and a set of local probability densities.

The structure S for $\boldsymbol{X}$ is a directed acyclic graph (DAG) that represents the assertions that X_i and $\{X_1, \ldots, X_n\} \backslash \boldsymbol{Pa}_i^S$ [1] are independent given $\boldsymbol{Pa}_i^S$, $i = 2, \ldots, n$. In this paper, we assume that the local probability densities depend on a finite set of parameters $\boldsymbol{\theta}_S \in \boldsymbol{\Theta}_S$. The particular case to be considered is when each variable is continuous and each local density function is the linear-regression model:

$$f(x_i \mid \boldsymbol{pa}_i^S, \boldsymbol{\theta}_i) \equiv \mathcal{N}(x_i; m_i + \sum_{x_j \in \boldsymbol{pa}_i} b_{ji}(x_j - m_j), \frac{1}{v_i}). \quad (2)$$

Given this form, a missing arc from X_j to X_i implies that $b_{ji} = 0$ in the former linear-regression model. The

[1] $\boldsymbol{Pa}_i^S$ represents the set of parents –variables from which an arrow is coming out– of the variable X_i in the probabilistic graphical model with structure given by S.

local parameters are given by $\boldsymbol{\theta}_i = (m_i, \boldsymbol{b}_i, v_i)$, where $\boldsymbol{b}_i = (b_{1i}, \ldots, b_{i-1i})^t$ is a column vector. We call a probabilistic graphical model constructed with this local density functions a *Gaussian network* [15].

In [11] (p. 98–99) a method to sample a multivariate normal distribution is presented. The method generate instantiations of $\boldsymbol{X}$ by computing X_1, then X_2 conditioned on X_1, and so on. An adaptation of this method, on the basis of the Probabilistic Logic Sampling (PLS) algorithm ([5]) is used in this paper. For the simulation of an univariate normal distribution, a simple method based on the sum of 12 uniform variables is chosen.

3 New approaches to optimization based on learning and simulation of Gaussian networks

3.1 Univariate approach $UMDA_c^G$

In this case the factorization of the joint density function is as follows:

$$f_l(\boldsymbol{x}; \boldsymbol{\theta}^l) = \prod_{i=1}^{n} f_l(x_i, \boldsymbol{\theta}_i^l). \tag{3}$$

The estimation of the parameters is performed by their maximum likelihood estimates.

3.2 Bivariate approach $MIMIC_c^G$

The approach that we propose in this section constitutes an adaptation of the $MIMIC$ algorithm ([2]) applied to continuous domains where the underlying probability model for every pair of variables is assumed to be a bivariate Gaussian.

Given a permutation $\pi = (i_1, i_2, \ldots, i_n)$, we define the class of density functions, $\mathcal{F}_\pi(\boldsymbol{x})$,

$$\mathcal{F}_\pi(\boldsymbol{x}) = \{f_\pi(\boldsymbol{x}) = f(x_{i_1} \mid x_{i_2}) \cdot \ldots \cdot f(x_{i_{n-1}} \mid x_{i_n}) \cdot f(x_{i_n})\} \tag{4}$$

where $f(x_{i_n})$ and $f(x_{i_j} \mid x_{i_{j+1}})$, $j = 1, \ldots, n-1$, follow normal density functions. Our target is to choose the permutation π^* whose associated $f_{\pi^*}(\boldsymbol{x})$ minimizes the Kullback-Leibler information divergence between the true density function, $f(\boldsymbol{x})$, and the density functions, $f_\pi(\boldsymbol{x})$, of the class $\mathcal{F}_\pi(\boldsymbol{x})$.

Following an analogous development to the done in [2], we obtain that the previous target is equivalent to find π^* that minimizes

$$J_\pi(\boldsymbol{x}) = h(X_{i_1} \mid X_{i_2}) + \ldots + h(X_{i_{n-1}} \mid X_{i_n}) + h(X_{i_n}) \tag{5}$$

where $h(X \mid Y)$ denotes the mean uncertainty in X given Y, and $h(Y)$ denotes the Shanon entropy.

By searching over all $n!$ permutations it is possible to find π^*. However, due to efficiency reasons, we will adapt the algorithm proposed by [2] to the bivariate Gaussian distribution as appears below.

Theorem 1 ([17] p. 167) *Let $\boldsymbol{X}$ be an n dimensional normal density function, $\boldsymbol{X} \rightsquigarrow \boldsymbol{\mathcal{N}}(\boldsymbol{x}; \boldsymbol{\mu}, \sum)$, then the entropy of $\boldsymbol{X}$ is:*

$$h(\boldsymbol{X}) = \frac{1}{2} n(1 + \log 2\pi) + \frac{1}{2} \log \mid \sum \mid . \tag{6}$$

Applying this result to univariate and bivariate normal density functions in order to define the $MIMIC_c^G$, we obtain:

$$h(X) = \frac{1}{2}(1 + \log 2\pi) + \log \sigma_X \tag{7}$$

$$h(X \mid Y) = \frac{1}{2}\left[(1 + \log 2\pi) + \log(\frac{\sigma_X^2 \sigma_Y^2 - \sigma_{XY}^2}{\sigma_Y^2})\right] \tag{8}$$

where $\sigma_X^2 (\sigma_Y^2)$ denotes the variance of the univariate $X(Y)$ variable and σ_{XY} denotes the covariance between the variables X and Y.

The $MIMIC_c^G$ works as a straightforward greedy algorithm with two steps. In the first one, the variable with the smallest sample variance is chosen. In the second step the variable X, whose estimation of $\frac{\sigma_X^2 \sigma_Y^2 - \sigma_{XY}^2}{\sigma_Y^2}$ with respect to the previous variable, Y, is the smallest is picked up.

3.3 Approach based on edge exclusion tests $EGNA_{ee}$

Many graphical model selection procedures start by making the $\binom{n}{2}$ single edge exclusion tests –excluding the edge connecting X_1 and X_2 corresponds to accepting the null hypothesis $H_0 : w_{12} = 0$, being the alternative $H_A : w_{12}$ unspecified– evaluating the likelihood ratio statistic and comparing it to a χ^2 distribution. In this section we will use the likelihood ratio test, T_{lik} [16].

The likelihood ratio test statistic to exclude the edge between X_1 and X_2 from a graphical Gaussian model

```
Create any complete DAG representing the initial
Gaussian network structure
Calculate the rejection region boundary (t_0) by
the Newton-Raphson method
for i = 1, 2, ..., n
  for j = 1, 2, ..., n
    if X_i is parent of X_j then
      Calculate T_lik
      if T_lik ≤ t_0 then
        Remove the arc between X_i and X_j
```

Figure 2: Pseudocode for the learning of Gaussian network structures by edge exclusion tests.

is $T_{lik} = -n \log(1 - r^2_{12|rest})$ where $r_{12|rest}$ is the sample partial correlation of X_1 and X_2 adjusted for the remainder $X_3, \ldots, X_n$. The latter can be expressed ([17] p. 189) in terms of the maximum likelihood estimates of the elements of the inverse variance matrix as $r_{12|rest} = -\hat{w}_{12}(\hat{w}_{11}\hat{w}_{22})^{-\frac{1}{2}}$.

In [16] the distribution function of the previous statistic under the null hypothesis is obtained:

$$F_{lik}(x) = G_{\mathcal{X}}(x) - \frac{1}{2}(2n+1)x\frac{1}{\sqrt{(2\pi)}}x^{\frac{-1}{2}}e^{\frac{-1}{2}x}N^{-1} \tag{9}$$

where $G_{\mathcal{X}}(x)$ is the distribution function of a $\mathcal{X}^2_1$ random variable. Thus, for a 5 % test, the rejection region is given by the resolution of the following equation:

$$0.95 = G_{\mathcal{X}}(x) - \frac{1}{2}(2n+1)x\frac{1}{\sqrt{(2\pi)}}x^{\frac{-1}{2}}e^{\frac{-1}{2}x}N^{-1}. \tag{10}$$

By a simple manipulation, the resolution of the previously mentioned equation turns into finding the root of an equation. The Newton-Raphson method used in our experiments, is only an example of suitable methods for such a purpose. Figure 2 shows a schematic of our algorithm for the learning of Gaussian network structures by edge exclusion tests.

3.4 Approach based on the BGe metric $EGNA_{BGe}$

In [4], a continuous version of the BDe metric for Gaussian networks called BGe is obtained. The metric is based upon the fact that the normal–Wishart distribution is conjugate with respect to the multivariate normal. This fact allows us to obtain a closed formula for the computation of the marginal likelihood of the data given the structure. The scoring metric is obtained by means of a theorem that provides the result from which it is possible to prove that the marginal likelihood for a general Gaussian network can be calculated using the following formula:

$$f(D \mid S) = \prod_{i=1}^{n} \frac{f(D^{X_i \boldsymbol{Pa}_i} \mid S_c)}{f(D^{\boldsymbol{Pa}_i} \mid S_c)} \tag{11}$$

where each term is of the form given in Equation (13), and where $D^{X_i \boldsymbol{Pa}_i}$ is the database D restricted to the variables $X_i \cup \boldsymbol{Pa}_i$.

Combining the results provided by the theorems given in [3] (p. 178; p. 180), in [4] the authors obtain:

$$f(D \mid S_c) = (2\pi)^{-\frac{nN}{2}}(\frac{\nu}{\nu+N})^{\frac{n}{2}}\frac{c(n,\alpha)}{c(n,\alpha+N)}|T_0|^{\frac{\alpha}{2}}|T_N|^{-\frac{\alpha+N}{2}} \tag{12}$$

where the $c(n,\alpha)$ is defined as follows:

$$c(n,\alpha) = \left[2^{\frac{\alpha n}{2}}\pi^{\frac{n(n-1)}{4}}\prod_{i=1}^{n}\Gamma(\frac{\alpha+1-i}{2})\right]^{-1}. \tag{13}$$

This result yields a metric for scoring the marginal likelihood of any Gaussian network.

See [4] for a discussion about the three components of the user's prior knowledge that are relevant to learn Gaussian networks: (1) the prior probabilities $p(S)$, (2) the parameters α and ν, and (3) the parameters $\boldsymbol{\mu}_0$ and T_0.

In our implementation, we will use the least informative prior knowledge. The prior precision matrix T_0 will be the unit matrix. The prior means vector $\boldsymbol{\mu_0}$ will consist of the middle values of the variables, i.e, if $a \leq X_i \leq b$ then $\mu^i_0 = \frac{a+b}{2}$. Finally, the parameters α and ν will be equal to $n+2$, the smallest possible equivalent sample sizes. Thus, the data will always overcome the weaknesses of the prior knowledge. Local search is used as the search procedure.

4 Experimental results

We have selected three functions to compare the proposed algorithms with a version of Evolutionary Strategies (ES) [1].

$$F_1(\boldsymbol{x}) = 1/(10^{-5} + \sum_{i=1}^{n}|y_i|)$$
$$-0.16 \leq x_i \leq 0.16;\ \ y_1 = x_1;\ \ y_i = x_i + y_{i-1}\ \ i = 2, \ldots, n \tag{14}$$

$$F_2(\boldsymbol{x}) = \sum_{i=1}^{n} \left[(x_1 - x_i^2)^2 + (x_i - 1)^2 \right] \quad -10 \leq x_i \leq 10 \tag{15}$$

$$F_3(\boldsymbol{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) \quad -600 \leq x_i \leq 600 \tag{16}$$

While the task in $F_1(\boldsymbol{x})$ is maximization, we seek for the minimization of the other two functions. In the three functions $n = 10$. The size of the population is the same in all the algorithms. These values are set to 2000, 2000 and 750 respectively. The algorithms are stopped when 300,000 evaluations are done. We use truncation selection as the selection method of choice. The number of selected individuals is set up to half of the population.

Table 1 summarizes the mean function value of the best solutions found over 100 runs. See [7] for a more detailed analysis of the results.

Table 1: Mean values reached by the algorithms.

	F_1	F_2	F_3
$UMDA_c^G$	53,460	0.13754	0.011076
$MIMIC_c^G$	58,775	0.13397	0.007794
$EGNA_{ee}$	100,000	0.09914	0.008175
$EGNA_{BGe}$	100,000	0.0250	0.012605
ES	5,910	0	0.034477

Acknowledgement

This work was partially supported by the University of the Basque Country under project UPV 140.226-EB131/99, and by Spanish *Ministerio de Educación y Cultura* under AP97 44673053 grant. We thank E. Bengoetxea and the anonymous reviewers for useful suggestions.

References

[1] T. Bäck (1996). *Evolutionary Algorithms in Theory and Practice*. Oxford University Press.

[2] J. S. De Bonet, C. L. Isbell and P. Viola (1997). MIMIC: Finding Optima by Estimating Probability Densities. *Advances in Neural Information Processing Systems, Vol. 9* .

[3] M. DeGroot (1970) *Optimal Statistical Decisions*. McGraw/Hill, New York.

[4] D. Geiger and D. Heckerman (1994). *Learning Gaussian networks*. Technical Report MST-TR-94-10. Microsoft Advanced Technology Division, Microsoft Corporation, Seattle, Washington.

[5] M. Henrion (1988). Propagating uncertainty in Bayesian networks by probabilistic logic sampling. *Uncertainty in Artificial Intelligence 2*. North-Holland, Amsterdam, 149–163.

[6] P. Larrañaga, R. Etxeberria, J. A. Lozano, B. Sierra, I. Inza and J. M. Peña (1999a). A review of the cooperation between evolutionary computation and probabilistic graphical models. *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*. La Habana, 314–324.

[7] P. Larrañaga, R. Etxeberria, J. A. Lozano and J. M. Peña (1999b). *Optimization by learning and simulation of Bayesian and Gaussian networks*. University of the Basque Country, Technical Report EHU-KZAA-IK-4-99.

[8] P. Larrañaga, R. Etxeberria, J. A. Lozano, J. M. Peña (2000). Combinatorial optimization by learning and simulation of Bayesian networks. *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, Stanford.

[9] H. Mühlenbein and G. Paaß(1996). From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV*, 178–187.

[10] M. Pelikan, D. E. Goldberg and F. Lobo (1999). *A Survey of Optimization by Building and Using Probabilistic Models*. University of Illinois at Urbana–Champaign. IlliGAL Report No. 99018.

[11] B. D. Ripley (1987). *Stochastic Simulation*. John Willey and Sons.

[12] S. Rudlof and M. Köppen (1996). Stochastic hill climbing by vectors of normal distributions. *Proceedings of the First Online Workshop on Soft Computing (WSC1)*. Nagoya, Japan.

[13] M. Sebag and A. Ducoulombier (1998). Extending Population-Based Incremental Learning to Continuous Search Spaces. *Parallel Problem Solving from Nature - PPSN V*. Berlin, Springer-Verlag, 418–427.

[14] I. Servet, L. Trave-Massuyes and D. Stern (1997). Telephone network traffic overloading diagnosis and evolutionary techniques. *Proceedings of the Third European Conference on Artificial Evolution, (AE'97)*, 137–144.

[15] R. Shachter and C. Kenley (1989). Gaussian influence diagrams. *Management Science*, **35**, 527–550.

[16] P. W. F. Smith and J. Whittaker (1998). Edge exclusion tests for graphical Gaussian models. *Learning in Graphical Models*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 555–574.

[17] J. Whittaker (1990). *Graphical models in applied multivariate statistics*. John Wiley and Sons.