Abstract—This paper investigates the use of empirical and Archimedean copulas as probabilistic models of continuous estimation of distribution algorithms (EDAs). A method for learning and sampling empirical bivariate copulas to be used in the context of n-dimensional EDAs is first introduced. Then, by using Archimedean copulas instead of empirical makes possible to construct n-dimensional copulas with the same purpose. Both copula-based EDAs are compared to other known continuous EDAs on a set of 24 functions and different number of variables. Experimental results show that the proposed copula-based EDAs achieve a better behaviour than previous approaches in a 20% of the benchmark functions.

I. INTRODUCTION

In evolutionary optimization, the class of algorithms that employ probabilistic modeling are usually called estimation of distribution algorithms (EDAs) [10], [13], [17]. In EDAs probabilistic models are learnt from the selected individuals and used to generate new solutions. This is a significant difference with respect to other evolutionary algorithms based on crossover and mutation operators. EDAs have been successfully applied to solve problems with discrete and continuous representations. Although the main rationale behind the EDAs, i.e. learning and sampling from probabilistic models, is the same for discrete and continuous problems, there exist fundamental differences between the characteristics of these two domains. In this paper we analyze a class of EDAs for problems with continuous or real value representation [1], [11].

There are many different approaches to continuous optimization using EDAs. Most of this research applies Gaussian probabilistic models [8]. However, other approximations depart from the Gaussianity assumption. Examples of these approaches include the application of independent component analysis (ICA) [3], [18], [29], histogram based probabilistic modeling [24], Cauchy distribution [19] and Copula methods [20], [27], [28]. While probabilistic modeling using Gaussian distributions has proved to be an effective alternative for many optimization problems, there are situations where Gaussian models fail. Therefore, it is a relevant question to investigate non-Gaussian approaches to probabilistic modeling in EDAs. In this paper we focus on probabilistic modeling using Copulas.

Copula functions describe the dependence structure of two or more random variables associated by a joint probability distribution function. In other words, they provide an scale-free description of how a number of random variables are distributed. Once the copula is unveiled, the whole joint probability distribution function can be found. Moreover we can use the copula function to generate new samples with such a joint probability distribution.

Bivariate, a.k.a. 2-copulas, are well known. A comprehensive text about the subject is due to Nelsen [14]. There are numerous methods to find the copula that best fits a data set [4], [5], [6] and they are being applied in many different fields such as finance [15], [16], signal processing [2] or networked systems [22]. But constructing n-copulas is difficult. Research has focused on Elliptic and Archimedean copulas because of their good properties [12].

Wang et al. [27] introduced two different bivariate copulas, and the approximation of the marginals was made by means of Gaussian distributions. Nine functions with two variables were used as benchmark and no comparison with other EDAs was included. On the other hand Salinas-Gutiérrez et al. [20], use the Archimedean family of Frank copulas and Gaussian copulas to model n-dimensional distributions. The introduced algorithm is inspired in the MIMIC [9] which learns a chain-shaped structure. To learn the chain, the introduced copula-based EDAs finds a permutation that minimizes the Kullback-Leibler divergence between the empirical density function and copula-based approximation. To this end, the copula entropy between each pair of variable is estimated. These copula-based EDAs were tested on five functions with n = 10 dimensions and results were compared with MIMIC [9].

Our approach is different to previous uses of copulas in EDAs in various aspects. Firstly, we use not only Archimedean or Gaussian copulas as in previous research [20], [27], [28]. Instead, we introduce empirical copulas with the purpose of not being constrained a priori by a set of candidate copulas. This makes necessary to develop a methodology for generating random variates no matter the dimensionality of the search. The procedure proposed in this paper does not provide neither a proper n-copula nor a joint probability distribution function of the best fitted. Instead it focuses only on obtaining a new set of points with a similar distribution, following Vapnik’s motto of “trying to solve only the problem that one has, and not a more general one”. Secondly, while copulas and other alternative
probabilistic modeling methods can be used together with Gaussian distributions, the approach we follow does not consider Gaussian modeling at any step. In addition we extend the use of Archimedean copulas to \( n \) variables. And finally, we conduct an extensive evaluation on a large set of benchmark functions for different dimensions.

The paper is organized as follows. Section II introduces the main concepts from copula theory. In Section III different variants of copula learning and sampling methods are proposed. The experimental framework and the numerical results from our experiments are presented in Section IV. Finally conclusions and trends for future work are outlined in Section V.

II. COPULA FUNCTION THEORY

Definition 1: A function \( C(u, v) : [0, 1]^2 \rightarrow [0, 1] \) is a copula if and only if it satisfies the three following conditions:

- For every \( 0 \leq u \leq 1 \) and every \( 0 \leq v \leq 1 \)
  \[
  C(0, v) = C(u, 0) = 0
  \]

- For every \( 0 \leq u_1 \leq u_2 \leq 1 \) and every \( 0 \leq v_1 \leq v_2 \leq 1 \)
  \[
  C(u_2, v_2) - C(u_2, v_1) - C(u_1, v_2) + C(u_1, v_1) \geq 0.
  \]

Copulas therefore satisfy the conditions of zero-gapped bivariate distribution functions of \( U \) and \( V \) with uniform marginals. Hence a probabilistic interpretation may be given in the same way as any other joint cumulative distribution function (JCDF):

\[
C(u, v) = \Pr(U \leq u, V \leq v).
\]

Then the unique joint probability density function (JPDF) \( c(u, v) \) associated to \( C \) is such that:

\[
c(u, v) = \int_{-\infty}^{u} \int_{-\infty}^{v} c(v, \nu) d\nu dv.
\]

The relevance and utility of copulas is due to Sklar’s theorem [23].

Theorem 1: (Sklar’s Theorem) Let \( H_{XY} (x, y) \) be the JCDF of \( X \) and \( Y \) with margins \( F_X(x) \) and \( F_Y(y) \) and such that \( u = F_X(x) \) and \( v = F_Y(y) \). Then there exists a copula \( C \) such that for all \( x, y \) in \((-\infty, +\infty)\),

\[
H_{XY} (x, y) = C(u, v)
\]

(4)

If \( F_X \) and \( F_Y \) are continuous then \( C \) is unique; otherwise \( C \) is uniquely determined on \( \text{Range}(F_X) \times \text{Range}(F_Y) \). Conversely if \( C \) is a copula and \( F_X \) and \( F_Y \) are cumulative distribution functions (CDF) then the function \( H_{XY} \) defined by (4) is the JCDF with margins \( F_X(x) \) and \( F_Y(y) \).

Thus, it is possible to separate 1) the marginal behavior due to the individual contributions of the random variables \( X, Y \), described by its margins \( F_X \) and \( F_Y \) respectively, and 2) the dependence structure, which is given by the copula \( C \) (couples \( X \) and \( Y \)). Moreover, a key feature of copulas is that they are invariant under strictly monotone transformations of their random variables (\( U \) and \( V \)). In other words, the way \( X \) and \( Y \) move together is modelled by the copula, whatever scales of \( X \) and \( Y \) were.

Once the copula is known, one can use the conditional distribution method to generate new samples.

Definition 2: Let \( U \) and \( V \) be two random variables whose JCDF is the copula \( C \). Then the conditional copula for \( V \) given \( U = u \) is

\[
C_u(v) = \frac{\partial}{\partial u} C(u, v).
\]

(5)

Again, the probabilistic interpretation of the conditional copula is the same as any conditional distribution function:

\[
C_u(v) = \Pr(V \leq v | U = u).
\]

Then, the following procedure generates 2 random variates \((u, v)\) whose JCDF is the copula \( C \):

1) Draw \( u \) and \( t \), two samples uniformly distributed in the unit interval.

2) Set \( v = C_u(1)(t | u_0) \).

Here \( C_u(1)(t | u_0) \) denotes the quasi-inverse of the conditional copula \( C_u \) for a given \( u = u_0 \), defined as follows.

Definition 3: Let \( F \) be a CDF. Then the quasi-inverse of \( F \) is any function \( F^{-1} \) with domain \([0, 1]\) such that:

\[
F(F^{-1}(t)) = t,
\]

for all \( t \) in the Range(\( F \)). Otherwise

\[
F^{-1}(t) = \inf \{ x | F(x) \geq t \} = \sup \{ x | F(x) \leq t \}.
\]

III. COPULA-BASED EDAS

In this section we describe two variants of copula-based EDAs (CEDA) that are proposed in this paper. The first one considers empirical copulas and the second the Archimedean family. A general pseudocode is shown in Algorithm 1. Both variants differ only in the type of learning and sampling methods used (steps 5 and 6).

Algorithm 1: CEDA

1: Generate an initial population \( D_0 \) of individuals and evaluate them
2: \( t \leftarrow 1 \)
3: do
4: \( D_{t-1}^{\text{se}} \leftarrow \) Select \( N \) individuals from \( D_{t-1} \) using truncation selection
5: Using \( D_{t-1}^{\text{se}} \) as the data set, learn a copula based approximation \( C \)
6: \( D_t \leftarrow \) Sample \( M \) individuals from \( C \) using a copula sampling method
7: until Stopping criterion is met
A. EDAs based on empirical copulas

A straightforward approach is to use empirical distributions and numeric methods both for the derivative and the inverse functions of the copula as well as the margins. The advantage is not to be bounded to a set of candidate JCDFs nor copulas out of which one has to choose the best fitted. Thus rare distributions are also captured. On the other hand they have the shortcoming of rough outcomes in the extremes that numerical methods provide.

**Definition 4:** The empirical CDF of a data set \( \{x_i\} \subset \mathbb{R} \) with \( i = 1, \ldots, N \) is the function \( F(x_i) = t_i \), being
\[
t_i = \frac{\# \{x_j : x_j < x_i\}}{N - 1}
\]
(6)
where the symbol \( \# \) stands for the cardinality of the set.

This definition is extended to its continuous version \( F(x) : \mathbb{R} \rightarrow [0, 1] \) by doing:
\[
F(x) = \begin{cases} 
LIP(x_i, t_i, x) & \text{if } \inf\{x_i\} \leq x \leq \sup\{x_i\} \\
0 & \text{if } x < \inf\{x_i\} \\
1 & \text{if } x > \sup\{x_i\},
\end{cases}
\]
(7)
where \( LIP(x_i, t_i, x) \) is the linear interpolation of the point \( x \) given the pairs \( (x_i, t_i) \) computed with the definition 4. The quasi-inverse of the empirical CDF is then obtained as follows:
\[
F^{-1}(t) = \begin{cases} 
LIP(t_i, x_i, t) & \text{if } \inf\{t_i\} \leq t \leq \sup\{t_i\} \\
\inf\{x_i\} & \text{if } t < \inf\{t_i\} \\
\sup\{x_i\} & \text{if } t > \sup\{t_i\}.
\end{cases}
\]
(8)
Regarding the empirical copula, the starting point is to obtain its empirical density.

**Definition 5:** The empirical density copula of a data set \( \{u_j, v_j\} \subset [0,1]^2 \) with \( j = 1, \ldots, N \), is the function \( c(u^*_j, v^*_j) = t_i \), with \( i = 1, \ldots, N_b \), being \( N_b = 2^{\lfloor \log_2 \sqrt{N} \rfloor} \) and \( t_i \) the relative frequency of the pair \( (u^*_i, v^*_i) \) in a 3D histogram of \( N_b \times N_b \) bins, each one of them centred on \( (u^*_i, v^*_i) \).

The empirical copula \( C(u^*_i, v^*_i) = T_i \) is constructed doing the cumulative sum of the empirical density, first in one variable and then in the other. The extension to the continuous unit square is:
\[
C(u, v) = \begin{cases} 
LIP(u^*_i, v^*_i, T_i, u, v) & \text{if } \inf\{u^*_i\} \leq u \leq \sup\{u^*_i\} \text{ and } \inf\{v^*_i\} \leq v \leq \sup\{v^*_i\} \\
0 & \text{if } u < \inf\{u^*_i\} \text{ or } v < \inf\{v^*_i\} \\
u & \text{if } u > \sup\{u^*_i\} \text{ or } v < \inf\{v^*_i\} \\
v & \text{if } u > \sup\{u^*_i\},
\end{cases}
\]
(9)
where \( LIP(u^*_i, v^*_i, T_i, u, v) \) is the linear interpolation in two dimensions of the point \( (u, v) \) given the triples \( (u^*_i, v^*_i, T_i) \) computed with Definition 5.

Finally, for the conditional copula, rather than approximating the derivative with respect to the first argument of the empirical copula it is faster to take the cumulative sum of the empirical density in the direction of the second argument, already computed in the steps for obtaining the empirical copula. The outcome of this operation is a succession of CDFs indexed by \( \{u^*_i\} \) and denoted as \( C_{u^*_i}(v^*_j) \). Then, alike with empirical CDFs, the extension of the inverse conditional copula to the continuous is defined by
\[
C_u ^{-1}(t) = EICDF(C_{u^*_i}(v^*_j), v^*_j, t) \text{ for } u^*_i \sim u,
\]
(10)
where the function \( a_i \sim b \) returns the element of the set \( \{a_i\} \) closest to \( b \) and \( EICDF(C_{u^*_i}(v^*_j), v^*_j, t) \) is computed with (8) using the given parameters in the inner function \( LIP \).

1) Learning empirical CEDA: Let \( S = [a, b]^n \) be the search space. Let \( P = \{a_1, \ldots, a_m\} \subset S \) be the population that is being evaluated in the objective function \( f \). Let \( f_0 \) be the threshold that defines the subset of best fitted \( X = \{x_1, \ldots, x_\ell\} \subset P \) such that \( f(x_1) \geq f_0 \). Representing \( x_i = [x_{i1}^1, \ldots, x_{iN}^j]^T \), for \( i = 1, \ldots, \ell \), where the subscript \( T \) denotes the transpose, the subset \( X \) takes the form of a matrix
\[
X = \begin{pmatrix}
x_1^1 & x_1^2 & \cdots & x_1^j & \cdots & x_1^N \\
x_2^1 & x_2^2 & \cdots & x_2^j & \cdots & x_2^N \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
x_\ell^1 & x_\ell^2 & \cdots & x_\ell^j & \cdots & x_\ell^N \\
\end{pmatrix}
\]
Thus, the \( i \)-th column of \( X \) will be \( x_i \), as defined above, whereas the \( j \)-th row of \( X \) will be denoted as \( z_j = [x_{j1}^1, \ldots, x_{jN}^j] \) for \( j = 1, \ldots, n \).

Let us define the new variable \( y_i \) in the following recursive way:
\[
y_i = \begin{cases} 
z_1 & \text{for } i = 1 \\
H_{i-1}(y_{i-1}, z_1) & \text{for } i = 2, \ldots, n - 1,
\end{cases}
\]
(11)
where \( H_{i-1}(y_{i-1}, z_1) \) is the JCDF of \( y_{i-1} \) and the \( i \)-th row of \( X \). Therefore, there must be an underlying copula \( C_{i-1,i} \) that constructs its dependence structure. Thus, according to Sklar’s theorem:
\[
y_1 = z_1; \\
y_2 = H_{1,2}(y_1, z_2) = C_{1,2}(G_1(y_1), F_2(z_2)) = C_{1,2}(u_1, v_2); \\
\vdots
\]
In general
\[
y_i = H_{i-1,i}(y_{i-1}, z_1) = C_{i-1,i}(G_{i-1}(y_{i-1}), F_i(z_1)) = C_{i-1,i}(u_{i-1}, v_{i}),
\]
with \( i = 1, \ldots, n - 1 \); being \( G_j \) the CDF of \( y_j \) and \( F_j \) the CDF of \( z_j \), for \( j = 1, \ldots, n \).

2) Sampling with empirical CEDA: Now using the inverse of the conditional copula is possible to generate a whole new population \( \tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_m\} \) with the same distribution than \( X \) following the Algorithm 2.

B. EDAs based on Archimedean copulas

As an alternative to empirical copulas one can attempt to use any of the known closed forms both for a given copula and its inverse conditional. That way neither generates a new population with a similar distribution of the old one nor obtains a real copula describing the dependence structure of the population. However it is possible to achieve the second objective by choosing Archimedean copulas.
Algorithm 2: New population from Emp. CEDA

1. Draw \( \hat{u}_1 \) and \( t \), both uniformly distributed in \([0,1]\) and obtain
   \[
   \hat{u}_2 = \frac{\partial C_{1,2}^{-1}}{\partial u}(t|\hat{u}_1).
   \]
2. Set \( \hat{p}_1^1 = C_1^{-1}(\hat{u}_1) \) and \( \hat{p}_2^1 = F_2^{-1}(\hat{u}_2) \).
3. For \( k = 2 \) to \( n-1 \):
   4. Set \( \hat{y}_k = C_{k-1,k}(\hat{u}_{k-1}, \hat{u}_k) \).
   5. Set \( \hat{u}_k = G_k(\hat{y}_k) \).
   6. Draw a new \( t \) as in step 1 and obtain
      \[
      \hat{u}_{k+1} = \frac{\partial C_{k,k+1}^{-1}}{\partial u}(t|\hat{u}_k).
      \]
7. Set \( \hat{p}_i^{k+1} = F_k^{-1}(\hat{u}_{k+1}) \).
8. End
9. Repeat from step 1 until completing a whole new population, i.e. for \( i = 1, \ldots, m \).

Algorithm 3: New population from Arch. CEDA

1. Draw \( \hat{w}_1 = \hat{u}_1 \) and \( t \), both uniformly distributed in \([0,1]\) and obtain
   \[
   \hat{u}_2 = \frac{\partial C^{-1}}{\partial u}(t|\hat{u}_1).
   \]
2. Set \( \hat{p}_1^1 = F_1^{-1}(\hat{w}_1) \) and \( \hat{p}_2^1 = F_2^{-1}(\hat{u}_2) \).
3. For \( k = 2 \) to \( n-1 \):
   4. Set \( \hat{w}_k = C(\hat{u}_{k-1}, \hat{y}_k) \).
   5. Draw a new \( t \) as in 1 and obtain
      \[
      \hat{u}_{k+1} = \frac{\partial C^{-1}}{\partial u}(t|\hat{w}_k).
      \]
6. Set \( \hat{p}_i^{k+1} = F_k^{-1}(\hat{u}_{k+1}) \).
7. End
8. Repeat from step 1 until completing a whole new population, i.e. for \( i = 1, \ldots, m \).

A. Experimental framework

We compare a number of EDAs grouped into three sets. The first and second sets are the CEDAs proposed in the paper whereas the third one is a group of well-known EDAs whose performance has been already proved. All of them follow the general scheme shown in Algorithm 1 but steps 4 and 5, where sampling and learning algorithms are implemented, are modified according to the set used.

The first set of CEDAs is based on three popular Archimedean copulas for which there are closed forms of the inverse conditional copula, namely Frank, Clayton and HRT, which can be found in [25] for instance. All of them depend on a parameter \( a \) that determines the degree of association of the copula and its structure of dependence as Figure 1 shows.

(A,B) Copula Frank with \( a = \{-5,5\} \). It is symmetric, allows negative dependence between the variables and tends to the independence as \( a \) goes to 0. They will be also represented as CEDA\(^{Frank}_{a\in[-5,5]}\).

(C,D) Copula HRT with \( a = \{0.2,5\} \). It is asymmetric, with greater dependence in the positive tail than in the negative, and tends to the independence as \( a \) increases. They will be also represented as CEDA\(^{HRT}_{a\in(0,2.5)}\).

(E,F) Copula Clayton with \( a = \{0.2,5\} \). It is asymmetric, with greater dependence in the negative tail than in the positive, and tends to the independence as \( a \) approaches to 0. They will be also represented as CEDA\(^{Clayton}_{a\in(0,2.5)}\).

The second set is the CEDA based on empirical copulas.

(G) Obtained by the procedures given in Section III-A and also represented as CEDA\(^{Emp}_{a\in(0,2.5)}\).

Finally, the third set consists of the following EDAs based on Gaussian distributions.

(H) UMDA\(_n\): A univariate marginal distribution algorithm where each variable is independently modeled using a univariate Gaussian distribution [9].
EDAs from a global perspective, computing on average the number evaluations required to achieve with 5 variables. The second one gives a closer look to the case 10 experiments. The first one focuses on the accuracy attained functions for all the procedures used in this paper.

hoc is the Black-Box Optimization Benchmarking both on parameter Archimedean copulas showing the way they are concentrated depending for Empirical and Archimedean copulas were developed in Matlab using the MATEDA-2.0 software [21]. Methods Fig. 1. Scatterplot of (I) EMNA: A multivariate EDA where the density function is modeled using a multi-variate Gaussian distribution. To avoid (likely but unfrequent) numerical errors in the computation of the covariance matrix, the population is restarted when the variance goes below a given threshold [9].

(J) EDDA: The eigenspace EDA is based on an eigenspace analysis of the covariance matrix of the population. The algorithm is similar to EMNA but after an eigenspace decomposition of the covariance matrix is computed, the minimum eigenvalue is reset to the value of the maximum eigenvalue [26].

As a function benchmark we use the 24 functions from the Black-Box Optimization Benchmarking \(^1\) at GECCO-2009. These functions as well as the experimental setup are described in [7].

The population size is \(M = 1000\) for all functions and problem dimensions. The truncation selection parameter used is \(T = 0.5\) and the stop criterion is reaching a maximum of 100 generations. All algorithms have been implemented in Matlab using the MATEDA-2.0 software [21]. Methods for Empirical and Archimedean copulas were developed \textit{ad hoc}; nevertheless Matlab Statistical Toolbox 7.2 incorporates functions for all the procedures used in this paper.

B. Numerical Results

In order to evaluate the algorithms we carried out three experiments. The first one focuses on the accuracy attained as the dimension of the problem increases from 2 to 5 and to 10 variables. The second one gives a closer look to the case with 5 variables comparing also the number of successes and the number evaluations of evaluations required to achieve them. Finally, we investigate the behavior of the different EDAs from a global perspective, computing on average the fraction of successful trials as a function of the number of evaluations and of the optimization accuracy.

1) Accuracy: We consider three different dimensions: 2, 5 and 10 variables. The aim is to determine the order of magnitude of \(\Delta f\), the difference between the optimum \(f_{opt}\) and the closest value of \(f\) attained, measured in ten to the power of \(k \in \{-8, -4, -2, 1\}\). Results are shown in Table I where columns A to J are the values of the smallest \(k\) for each one of the EDAs used.

For each row, the smallest value is highlighted in yellow. In addition, if a CEDA (columns A to G) performs equal to the smallest non-CEDA (columns H,I,J) it is highlighted in cyan. Finally if a CEDA performs better than the best non-CEDA, it is highlighted in green.

Comparing CEDA and non-CEDA for 2 variables only for function 5 is impossible to find a CEDA that performs at least equal than a non-CEDA. The rest of functions can be split into two groups, considering the number of CEDAs that perform equal. Thus for functions \(\{1, 2, 3, 4, 7, 15, 16, 17, 20, 21, 23\}\), almost half of the benchmark set, all the CEDAs attain the highest accuracy, \((10^{-8})\), the same than at least one non-CEDA and sometimes even better; whereas for the rest only some CEDAs perform well. As the number of variables increases to 5 and 10 non-CEDAs become outranked in 6 functions for 5 variables and in 4 for 10 variables, and matched in other 5 functions for 5 variables and 7 for 10 variables.

2) Successful trials and number of function evaluations: We now consider not only the order of magnitude of the accuracy attained \((k)\) but also the number of successful trials (NT) and the median number of function evaluations to reach the best function value \((RT_{suc})\) in thousands. Additional parameters are: 5 variables, 30 trials and 198000 as the maximum number of function evaluations. Results are shown in Table II, where rows are sorted first in the decreasing order of \(k\), then in the increasing order of NT and finally in the decreasing order of \(RT_{suc}\). Thus it is clear that CEDAs outrank non-CEDAs in a 25\% of the benchmark set corresponding to functions \(\{3, 4, 8, 9, 16, 23\}\), and have a quite similar performance in functions \(\{1, 2\}\), which altogether cover one third of the benchmark set.

3) Empirical distribution of trials: In the last experiment we evaluate the general behavior of all the algorithms for all the functions. Figure 2 shows the empirical cumulative distribution functions (ECDFs) plotting the fraction of trials versus the number of function evaluations divided by search space dimension \(D = 5\), to fall below \(f_{opt} + \Delta f\) with \(\Delta f = 10^k\), where \(k\) is the first value in the legend. The second value in the legend indicates the number of functions that were solved in at least one trial.

By analyzing the number of functions that were solved in at least one trial we get a perspective of the general behavior of the different EDAs. It can be seen that the best performance is achieved by EMNA, which is able to solve, for all levels of accuracy 10 or more functions. However, the behavior of the CEDAs is acceptable for the first level of accuracy. They are able to solve 23 out of the 24 functions. Particularly relevant is the behavior of CEDA\(^{\text{emp}}\) which is

\(^1\)http://coco.gforge.inria.fr/doku.php?id=bbob-2009
Accuracy attained by the EDAs proposed, represented in \(10^k\) where \(k\) is the value shown for each function (rows 1 - 24) and each EDA (columns A-J). Blank spaces mean that the algorithm did not achieve an accuracy with order of magnitude \(10^4\). Tables are shown considering 2, 5 and 10 variables in the objective functions \(f\). The smallest value of each row is highlighted in yellow. Values in columns A to G equal to the smallest of columns H, I, and J are highlighted in cyan. Values in columns A to G smaller than the smallest of columns H, I, and J are highlighted in green.

the only algorithm that solves 15 out of the 24 for the second level of accuracy. There are important differences between the Archimedean CEDAs, particularly for the last level of accuracy where CEDA\(\alpha = -5\) and CEDA\(\alpha = 5\) are the only Archimedean CEDAs that solve at least one function.

Considering the fraction of successful trials (those were the desired accuracy was reached), the CEDAs using Frank copulas reach better results than EMNA and UMDAc, but in this case the best contender is EDDA.
<table>
<thead>
<tr>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>S</th>
<th>T</th>
<th>U</th>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Performance of the proposed EDAs for each objective function (in all cases $n = 5$). Tables are sorted first by increasing $\Delta f$, measured in $10^5$ and represented by $k$. The second sorting criterion is the number of successful trials (NT) in decreasing order. Finally we also consider the number of function evaluations necessary to reach the optimum (RTsucc) in increasing order.

V. CONCLUSIONS

In this paper we first introduce a simple method to use bivariate empirical copulas in $n$-dimensional EDAs. Afterwards we change the learning method by using an Archimedean copula instead of the empirical one. Three Archimedean copulas, with different parameters, have been used to model search distributions in EDAs. We have compared CEDA algorithms with classical EDAs based on Gaussian distributions in a wide set of functions, representing different domains of difficulty.

The use of copula functions in EDAs is still a large unexplored area. Previous works in this line either consider only 2-dimensional problems or attempt to find an underlying $n$-copula that captures the real dependence structure. Our proposal is to consider bivariate empirical copulas first. Using them we do not provide the real underlying copula, but a method to generate new populations considering the marginal behaviour of each variable and the dependence between variable $x_j$ with the distribution of the copula $C_{j-1,j-2}$. Then, due to Archimedean copulas are associative, using one of them instead of the empirical copulas turns the method proposed into a way to construct real $n$-copolas.

Again it is not the real underlying copula because it needs to be prefixed beforehand. In that sense, our proposal is similar to standard EDAs that rely on Gaussian multivariate distribution to obtain new generations, which is seldom the one that models the real distribution. The advantage of our proposal consists of not only allowing the incorporation of marginal behaviours but also the provision of closed forms of the inverse conditional copula, so faster and more accurate objectives are expected.

Our hypothesis is that there are situations where the assumption of Gaussian distributions can be outranked by better fitted distributions. The exhaustive experimental results show that CEDAs proposed outperformed Gaussian EDAs for a noticeable 20% of the tested functions, whereas in the remaining 80% it is still possible to find better performance in some of the CEDAs proposed with respect to at least one non-CEDA. Another result derived from our analysis is the evidence that the parameters of the Archimedean copulas can have an important effect in the optimization results.

Next steps in our research are: 1) To determine which characteristics of CEDAs make them particularly suitable to optimize a given function. 2) Neither a study of which
bivariate Archimedean copula nor which parameter is best for conducting the CEDA has been done. Thus, it is interesting to investigate whether tuning copula parameters may improve CEDA performance and the computational burden that it involves. 3) To incorporate advanced sampling strategies in order to avoid search stagnation.

This research is part of the CajalBlueBrain project. It has been partially supported by TIN-2008-06815-C02-02, TIN2007-62626 and TIN 2008-00508; Consolider Ingenio 2010 - CSD2007-00018 and 2010 2007/2011 projects (Spanish Ministry of Science and Innovation); by Interligare Institute for Innovation in Intelligence (I4) and also by the Real Colegio Complutense at Harvard.