Vine Estimation of Distribution Algorithms

Yasser González-Fernández[1], Diana Carrera[2], Marta Soto[3] and Alberto Ochoa[4]

Abstract— **A Vine Estimation of Distribution Algorithm (VEDA) is a recently proposed optimization procedure built on top of a probabilistic graphical model called vine. The first target of vines was uncertainty analysis with high dimensional dependence modeling.**

The aim of this communication is to draw a path through a simple set of experiments, from the Univariate Marginal Distribution Algorithm to VEDA. Four algorithms are investigated in relation to their ability to deal with both weak and strong correlated variables in continuous unconstrained optimization problems. The results show that the models complement each other, although VEDA is the most promising algorithm.

Keywords— **estimation of distribution algorithms, copula, vine.**

I. Introduction

The normal distribution has been commonly used to model real-valued search distributions in Estimation of Distribution Algorithms (EDAs) [1], [2], [3], [4]. Unfortunately, it is often inconsistent with empirical evidence and leads to the construction of wrong models. Copula functions [5] can be used to tackle these problems, because they allow to build more realistic search distributions. However, the use of multivariate copula functions alone suffers from several shortcomings [6] that can be overcome using vines [7], [8]. Vines are powerful probabilistic graphical models that represent a rich variety of patterns of dependence by combining bivariate copulas of different families of distributions.

In this work, various models based on copula theory are used in EDAs: two algorithms are built around the multivariate product and normal copulas while the other two are based on vines. The algorithms are tested on a set of artificial test functions and a protein docking problem from real-world. The numerical results show that vine-based EDAs are better endowed to optimize problems with different patterns of dependence.

The remainder of the paper is organized as follows. Section II introduces the necessary concepts of copulas and vines. It also connects the algorithms investigated in the paper from the perspective of copulas. Section III describes the VEDA. Our empirical investigation is reported in Section IV and the conclusions are given in Section V.

[1]Institute of Cybernetics, Mathematics and Physics, Cuba. ygf@icimaf.cu.

[2]University of Havana, Cuba. d.carrera@lab.matcom.uh.cu, diana@icimaf.cu.

[3]Institute of Cybernetics, Mathematics and Physics, Cuba. mrosa@icimaf.cu.

[4]Institute of Cybernetics, Mathematics and Physics, Cuba. ochoa@icimaf.cu.

II. From UMDA to VEDA

To begin with, we present some definitions from copula theory [5], [9].

Copulas separate the effect of dependence and margins in a joint distribution [8]. In this sense, copulas are functions that link multivariate distributions to their margins. This definition is mathematically supported by the Sklar's theorem [10], which we informally present below.

Let $\mathbf{X} = (\mathbf{X_1}, \ldots, \mathbf{X_n})$ be a vector of continuous random variables with joint density function f and joint cumulative distribution function F, both defined on $\mathbb{R}^{\kappa}$. Also let $\mathbf{x} = (\mathbf{x_1}, \ldots, \mathbf{x_n})$ be an observation of $\mathbf{X}$ and $F_1, \ldots, F_n$ denote the univariate marginal distributions of F. The copula associated with F is a distribution function with uniform margins $C : [0,1]^n \rightarrow [0,1]$ that satisfies

$$F(x_1, \ldots, x_n) = C(F(x_1), \ldots, F(x_n))$$

and

$$C(u_1, \ldots, u_n) = F\left(F_1^{(-1)}(u_1), \ldots, F_n^{(-1)}(u_n)\right).$$

A remarkable result states the uniqueness of $C(u_1, \ldots, u_n)$ for continuous F.

A. Product Copula

An important copula is the product copula, which is given by

$$C_{\mathrm{I}}(u_1, \ldots, u_n) = u_1 \ldots . u_n. \quad (1)$$

Here, the interesting point is that random variables are independent if and only if their underlying copula is the product. This result follows immediately from the Sklar's theorem.

The UMDA proposed in [2] assumes a model of independence of normal marginal distributions. Therefore, an EDA based on the product copula is a generalization of the UMDA, which also supports other types of marginal distributions, including nonparametric distributions [11].

B. Normal Copula

Besides UMDA, in [2] the authors also proposed an EDA based on the multivariate normal distribution. They called it Estimation of the Multivariate Normal Algorithm (EMNA). It turns out that, indeed EMNA can be also reformulated in copula terms: a normal copula + normal margins.

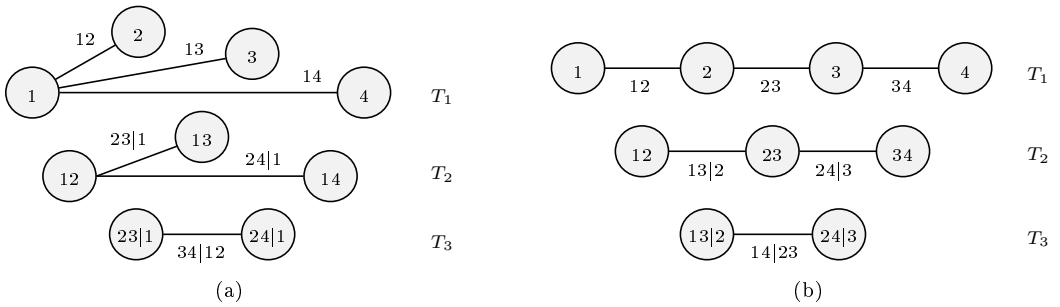

Fig. 1: Four-dimensional C-vine (a) and D-vine (b). In a C-vine, each tree T_j has a unique node that is connected to $n-j$ edges. In a D-vine, no node in any tree is connected to more than two edges.

The Gaussian Copula Estimation of Distribution Algorithm (GCEDA) introduced in [11], [12] is based on the multivariate normal (or Gaussian) copula,

$$C_{\mathrm{N}}\left(u_1,\ldots,u_n;R\right)=\Phi_R\left(\Phi^{-1}\left(u_1\right),\ldots,\Phi^{-1}\left(u_n\right)\right), \tag{2}$$

where Φ_R is the standard multivariate normal distribution function with correlation matrix R, and Φ^{-1} denotes the inverse of the standard univariate normal distribution. This copula allows the construction of multivariate distributions with non-normal margins. If this is the case, the joint density is no longer the multivariate normal, though the normal dependence structure is preserved. Therefore, with normal margins, GCEDA is equal to EMNA, otherwise they are different.

In GCEDA with normal margins, the correlation matrix is estimated by maximum likelihood. The generation of a new individual starts with the simulation of a vector $(u_1,\ldots,u_n)$ from the multivariate normal copula [13]. Then, each component x_i of the new individual is determined as $x_i=\hat{F}_i^{-1}\left(u_i;\hat{\mu}_i,\hat{\sigma}_i^2\right)$.

C. Archimedean Copulas

Several other multivariate copula-based EDAs have been proposed in recent years [14], [15], [16]. In general, the idea underlying these works is the use of the so-called Archimedean copulas [5]. Most of the available parametric copulas are bivariate, but Archimedean copulas can be easily generalized to n dimensions. Unfortunately, in these cases only one scalar parameter quantifies the overall multivariate dependence, which limits the use of this type of copulas in EDAs. The reader is referred to [6] for more details on these shortcomings.

D. Vines

Fortunately, there is an alternative approach to the solution of the above shortcomings: Vines [7], [8]. These are dependence models of a multivariate distribution function based on a decomposition of $f\left(x_1,\ldots,x_n\right)$ into bivariate copulas and marginal densities.

A vine on n variables is a nested set of trees $T_1,\ldots,T_{n-1}$, where the edges of tree j are the nodes of the tree $j+1$ with $j=1,\ldots,n-2$. Two special types of vines are C-vines (canonical vines) and D-vines (drawable vines). Figure 1 shows a four-dimensional C-vine and D-vine. In particular, the C-vine density is given by

$$\prod_{k=1}^{n} f\left(x_k\right)\prod_{j=1}^{n-1}\prod_{i=1}^{n-j} c_{j,j+i|i,\ldots,j-1}, \tag{3}$$

and the D-vine density is given by

$$\prod_{k=1}^{n} f\left(x_k\right)\prod_{j=1}^{n-1}\prod_{i=1}^{n-j} c_{i,i+j|i+1,\ldots,i+j-1}, \tag{4}$$

where j identifies the trees and i denotes the edges in each tree.

The arguments of the pair-copulas in (3) and (4) are conditional distributions of the form $F\left(x\mid\mathbf{v}\right)$ determined by the subscripts of the copula. In [17] it is shown that

$$F\left(x\mid\mathbf{v}\right)=\frac{\partial C_{xv_j|\mathbf{v_{-j}}}\left(F\left(x\mid\mathbf{v_{-j}}\right),F\left(v_j\mid\mathbf{v_{-j}}\right)\right)}{\partial F\left(v_j\mid\mathbf{v_{-j}}\right)}, \tag{5}$$

where $C_{xv_j|\mathbf{v_{-j}}}$ is a bivariate copula distribution function, $\mathbf{v}$ is a n-dimensional vector, v_j is one component of $\mathbf{v}$ and $\mathbf{v_{-j}}$ denotes the $\mathbf{v}$-vector excluding the j component. The recursive evaluation of $F\left(x\mid\mathbf{v}\right)$ yields the expression

$$F\left(x\mid v\right)=\frac{\partial C_{xv}\left(F_x\left(x\right),F_v\left(v\right)\right)}{\partial F_v\left(v\right)}.$$

When x and v are uniform, $F\left(x\mid v\right)$ reduces further to $F\left(x\mid v\right)=\partial C_{xv}(x,v)/\partial v$. Since the bivariate copulas may belong to different distribution families, the $h-$function,

$$h\left(x,v,\theta\right)=F\left(x\mid v\right)=\frac{\partial C_{xv}\left(x,v,\theta\right)}{\partial v}, \tag{6}$$

is defined to facilitate the computation of $F\left(x\mid v\right)$, where θ denotes the set of parameters for the copula of the joint distribution function of x and v. To

use a bivariate copula in a vine we must define the $h-$function and its inverse with respect to the first variable.

We use vines to create a new class of EDAs. The next section gives a short introduction.

III. VEDA

Vine Estimation of Distribution Algorithms (VEDAs) [18], [19], [20] are a class of EDAs that uses vines to model the search distributions. CVEDA and DVEDA are based on C-vine and D-vine, respectively.

The methods for the construction of C-vines and D-vines have been developed in [6]. They consist of the following steps:

1. Select the structure of the C-vines and D-vines.

The construction of a vine begins by determining the empirical Kendall's tau for the bivariate dependencies that will be explicitly modeled in the first tree. This is determined by a variable order, which is chosen by a greedy heuristic.

In a C-vine, the node that maximizes the sum of the weights of its edges to the other nodes is chosen as the root of the first tree. The same applies for the remainder trees.

In a D-vine, the final structure is completely determined by the structure of the first tree. The problem of constructing the first tree consists in finding the maximum weighted sequence of the variables. In [21], this problem is transformed into a traveling salesman problem (TSP) instance. For efficiency, we find an approximate solution of the TSP using the cheapest insertion heuristic [22].

The cost of the construction of these models increases with the number of variables. However, the number of trees might be reduced if conditional independence is detected. In [18] we apply the truncation strategy proposed in [21] to deal with this problem. This strategy carries out a model selection procedure.

2. Select the pair-copula types in the factorization and estimate the copula parameters.

(a) Determine the pair-copulas in the first tree from the original data.

(b) Compute observations by evaluating the conditional distribution functions of the form $F(x|y)$ for the second tree according to the h-functions of the copulas in the first tree.

(c) Determine the pair-copulas in the second tree from observations obtained in step (b).

(d) Repeat steps (b) and (c) for the following trees.

To select a bivariate copula that fits the data appropriately we proceed as follows. The first step is to apply the independence test proposed in [23]. The independence copula is selected if there is not enough evidence against the null hypothesis of independence at a fixed significance level of 0.1. Otherwise, we estimate the parameters of a group of candidate copulas and choose the one that minimizes a Cramér-von Mises statistic [24].

The bivariate copulas that we consider are: normal, Student's t, Clayton, rotated Clayton, Gumbel and rotated Gumbel. The normal copula cannot account for tail dependence, while the Student's t copula restricts upper and lower tail dependence to be equal. Clayton and rotated Clayton are only lower tail dependence, while Gumbel and rotated Gumbel are only upper tail dependence. The copula parameters are estimated using the inversion of Kendall's tau [25]. The degrees of freedom of the Student's t copula are estimated by maximum likelihood with the correlation parameter fixed [26].

Simulation from vines [27], [28], [29] is based on the conditional distribution method [30]. The general algorithm for sampling n dependent uniform [0,1] variables is common for the C- and D-vines. First we sample n independent uniform random numbers $w_i \in (0,1)$ and then we compute

$$\begin{aligned} x_1 &= w_1 \\ x_2 &= F^{-1}_{2|1}(w_2|x_1) \\ x_3 &= F^{-1}_{3|1,2}(w_3|x_1,x_2) \\ &\cdots \\ x_n &= F^{-1}_{n|1,2,\ldots,n-1}(w_n|x_1,\ldots,x_{n-1}) \end{aligned}$$

To determine $F(x_j \mid x_1, x_2, \ldots, x_{j-1})$ for each j, the expressions (5) and (6) are used for both structures. However, choice of the v_j in (5) is different for the C- and D-vines. For details about the implementation of the vine sampling algorithms, the reader is referred to [6]

TABLA I: Test bed.

$$f_{\text{Sphere}}(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

$$f_{\text{Griewank}}(\boldsymbol{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

$$f_{\text{Ackley}}(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1)$$

$$f_{\text{Summation Cancellation}}(\boldsymbol{x}) = \frac{1}{10^{-5}+\sum_{i=1}^{n}|y_i|} \text{ where } y_1 = x_1, \; y_i = y_{i-1} + x_i$$

Docking of 2z5u test system, a 73-atoms molecule with 20 ligand torsions in a box of size $28 \times 32 \times 24$ Å.

IV. Experiments

All above-said about the algorithms and more is implemented in the packages `copulaedas` [31] and `vines` [32] of the statistical environment R [33].

A main goal of the reported experiments is to test whether VEDAs deal appropiately with both weak and strong correlated problems. This is the expected

behaviour due the vine ability to deal with different patterns of dependence. We compare the performance of our vine-algorithms with that of UMDA and GCEDA. Recall the later have the same type of dependence across all pairs of variables. In this paper, all copula-based EDAs deal only with normal margins.

Here we only present results for a few small ($n = 10$) artificial tests functions and one slightly larger ($n = 26$) real-world problem: protein docking. For more details and reports of extensive simulations the reader is referred to [18], [20], [34].

A. Test Functions

Four well-known test functions are considered as benchmark problems: Sphere, Griewank, Ackley and Summation Cancellation [35]. The definition of each function for a vector $\boldsymbol{x} = (x_1, \ldots, x_n)$ is given in Table I. Sphere, Griewank and Ackley are minimization problems with global optimum equal to zero at $\boldsymbol{x} = (0, \ldots, 0)$. Summation Cancellation is a maximization problem which has its global optimum at $\boldsymbol{x} = (0, \ldots, 0)$ with evaluation 10^5.

We find the population size required by each algorithm to reach the global optimum of the function in 30 of 30 independent runs (critical population size). The critical population size is determined using a bisection method [36]. The algorithms stop when the global optimum is found with a precision of 10^{-6} or after 500000 function evaluations. All the algorithms use a truncation selection of 0.3 [37], and no elitism. The initial population is sampled uniformly in the real interval of each variable. As interval we use $[-600, 600]$ in Sphere and Griewank, $[-30, 30]$ in Ackley and $[-0.16, 0.16]$ in Summation Cancellation.

B. Molecular Docking

Molecular docking is a computational procedure to predict the geometry of binding of two molecules. Often, one of these molecules is a protein, while the second one, the ligand, is a small molecule that binds into the protein. The protein-ligand docking problem remains open, since the algorithms for exploring the conformational space and the scoring functions that have been implemented so far, still have significant limitations [38].

Here we present results for the `2z5u` protein-ligand test system, which is available from the Protein Data Bank (PDB) [39] The codification of the individuals contains information about the position (three variables), orientation (three variables), and torsion angles (20 variables) of the ligand. Hence this is a 26-dimensional problem. We use the semiempirical scoring function implemented in Autodock 4.2 [40].

To evaluate the quality of the predicted ligand conformations, we use the root-mean-square deviation (RMSD) between the crystallographic and predicted ligand coordinates of the atoms. A structure with an RMSD within 2Å is classified as successfully docked, while a structure with an RMSD between 2 and 3Å is classified as partially docked.

We first find for each algorithm the population size that yields the lowest energy with the smallest number of evaluations, which ensures a comparison between the algorithms as fair as possible.

For the VEDAs, only the bivariate product and normal copulas are fitted. Besides, the truncation procedure is applied to reduce the number of trees in the vines.

For GA [41] we use a two-point crossover rate of 0.8, a mutation based on Cauchy distribution with parameters $\alpha = 0$ and $\beta = 1$, a mutation rate of 0.2, an elitism value of one, and proportional selection. For PSO (Standard PSO 2007 [42]) we use an inertia weight of $1/2log(2)$, the user defined parameters are $\phi_1 = \phi_2 = 5log\,(2)$, and the number of neighborhoods is $1 - (1 - 1/s)$ where s is the number of particles (swarm size). For DE (variant: DE/local-to-best/1/bin) we use a crossover rate of 0.8, a differential mutation rate of 0.5, and a mutation scale factor of 0.8.

C. Results

The experimental results obtained with UMDA, GCEDA, CVEDA and DVEDA in Sphere, Griewank, Ackley and Summation Cancellation are presented in Tables II, III, IV and V, respectively.

With the first three functions, all algorithms obtain the same final error. However, UMDA has a smaller critical population size and does the job with much fewer function's evaluations than its competitors. The vines algorithms behave similarly but clearly better than GCEDA. CVEDA seems to be a slightly more efficient than DVEDA.

In Summation Cancellation, CVEDA behaves much better than DVEDA, however GCEDA shows excellent results and significantly outperforms both VEDAs. UMDA can not optimize this function,

The experimental results with copula-based EDAs, PSO, DE and GA in the docking example is showed in Table VI.

The tested implementations of PSO, DE and GA do a very poor job. Despite the huge number of function evaluations they end way far from the results of the copula algorithms, both in terms of achieved energy and RMSD. We recall that RMSD values greater than 3Å are not considered successfully docked.

DVEDA achieves the lowest energy with the smallest number of evaluations. GCEDA follows it with respect to evaluations and also has the best RMSD. CVEDA improves the energy obtained by GCEDA, but needs about 15000 more evaluations and gets a worse RMSD.

Aiming to analyze how the number of bivariate normal copulas fitted to the arcs varies during

TABLA II: Results in Sphere with $x_i \in [-600, 600]$, $i = 1, \ldots, 10$.

Algorithm	Success	Population	Evaluations	Best Evaluation
UMDA	30/30	86	3996.1 ± 89.5	$6.9\text{E}-07 \pm 1.9\text{E}-07$
GCEDA	30/30	325	13769.1 ± 248.5	$6.6\text{E}-07 \pm 1.6\text{E}-07$
CVEDA	30/30	188	8033.8 ± 170.5	$6.8\text{E}-07 \pm 2.1\text{E}-07$
DVEDA	30/30	207	8818.2 ± 192.9	$7.0\text{E}-07 \pm 1.8\text{E}-07$

TABLA III: Results in Griewank with $x_i \in [-600, 600]$, $i = 1, \ldots, 10$.

Algorithm	Success	Population	Evaluations	Best Evaluation
UMDA	30/30	113	5179.1 ± 210.0	$7.2\text{E}-07 \pm 1.7\text{E}-07$
GCEDA	30/30	304	12798.4 ± 351.1	$6.6\text{E}-07 \pm 1.7\text{E}-07$
CVEDA	30/30	213	9151.9 ± 452.6	$6.5\text{E}-07 \pm 1.8\text{E}-07$
DVEDA	30/30	225	9630.0 ± 309.2	$6.9\text{E}-07 \pm 1.5\text{E}-07$

TABLA IV: Results in Ackley with $x_i \in [-30, 30]$, $i = 1, \ldots, 10$.

Algorithm	Success	Population	Evaluations	Best Evaluation
UMDA	30/30	88	5426.6 ± 127.2	$8.2\text{E}-07 \pm 1.0\text{E}-07$
GCEDA	30/30	325	18178.3 ± 207.8	$8.0\text{E}-07 \pm 1.5\text{E}-07$
CVEDA	30/30	213	11984.8 ± 184.9	$7.9\text{E}-07 \pm 1.5\text{E}-07$
DVEDA	30/30	213	11920.9 ± 197.6	$7.9\text{E}-07 \pm 1.3\text{E}-07$

TABLA V: Results in Summation Cancellation with $x_i \in [-0.16, 0.16]$, $i = 1, \ldots, 10$.

Algorithm	Success	Population	Evaluations	Best Evaluation
UMDA	0/30	2000	500000.0 ± 0.0	$6.9\text{E}+02 \pm 5.0\text{E}+02$
GCEDA	30/30	325	38848.3 ± 327.6	$1.0\text{E}+05 \pm 1.2\text{E}-07$
CVEDA	30/30	625	84958.3 ± 786.0	$1.0\text{E}+05 \pm 1.1\text{E}-07$
DVEDA	30/30	1400	161840.0 ± 1352.5	$1.0\text{E}+05 \pm 9.3\text{E}-08$

the evolution, we compare CVEDA and DVEDA in terms of the relative proportion between the number of bivariate normal copulas that have been fitted and the total number of edges in the vine. We recall that in the docking example, only the product and normal copulas are fitted. Figure 2 shows that the number of normal copulas fitted by both algorithms increases during the evolution, although DVEDA fits more normal copulas than CVEDA.

It is worth noting that because of the use of a truncation procedure, the number of statistical tests was dramatically reduced. The average number of fitted trees was below eight with CVEDA and below ten with DVEDA.

D. Discussion

We assess the impact of using different types of dependencies and bivariate copulas. The main results are summarized as follow:

1. CVEDA and DVEDA exhibit a good performance in problems with both strong and weak dependencies: While UMDA uses the independence model and GCEDA assumes a linear dependence structure, CVEDA and DVEDA do not assume the same type of dependence across all pairs of variables. The estimation procedures used by the vine-based algorithms select among a group of candidate bivariate copulas, the one that fits the data appropriately. Thanks to this mechanism, CVEDA and DVEDA perform, in general, between UMDA and GCEDA in terms of the number of function evaluations.

2. CVEDA exhibits better results than DVEDA in the easy problems for UMDA (Sphere, Griewank and Ackley): The model used by DVEDA allows a more freely selection of the bivariate dependences that will be explicitly modeled, while the model used by CVEDA has a more restrictive structure. These characteristics enable DVEDA to fit in the first tree a greater number of bivariate copulas that represent dependencies. This may explain why DVEDA requires larger sample sizes than CVEDA, and thus more function evaluations.

3. CVEDA has much better results than DVEDA in Summation Cancellation: Summation Cancellation reaches its global optimum when the sum in

TABLA VI: Comparison of copula-based algorithms with PSO, DE and GA in the 2z5u test system.

Algorithm	Population	Evaluations	Lowest Energy	RMSD
CVEDA	1400	157600 ± 11391	-29.58 ± 1.23	0.58 ± 0.12
DVEDA	1200	125266 ± 11965	-30.16 ± 1.28	0.52 ± 0.12
GCEDA	1600	140966 ± 17835	-29.43 ± 0.56	0.51 ± 0.05
UMDA	1400	171900 ± 11442	-29.14 ± 1.97	0.61 ± 0.18
PSO	300	912510 ± 95512	$+42.52 \pm 27.39$	6.40 ± 2.95
DE	100	972536 ± 47549	$+9.79 \pm 21.48$	7.49 ± 4.19
GA	20	2265600 ± 273062	$+483.50 \pm 538.39$	13.34 ± 4.21

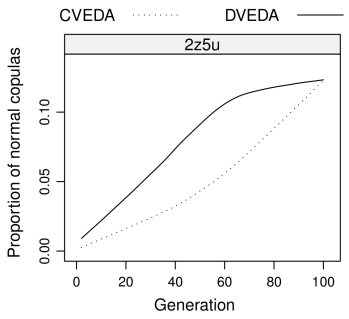

Fig. 2: Comparison of CVEDA and DVEDA for the 2z5u test system, in terms of the relative proportion between the number of fitted normal copulas and the number of arcs in the vine, at different generations. In a 26-dimensional vine, there are 25 trees and 325 arcs.

the denominator of the fraction is zero. The i-th term of this sum is the result of the sum of the first i variables of the function. Therefore, the values of the first variables have a greater influence in the value of the sum. The selected populations reflect these characteristics including stronger associations between the first variables and the next ones. A C-vine structure provides a more appropriate modeling of this situation than a D-vine structure, since it is possible to find a variable that governs the interactions in the sample. However, as it was pointed out before, here the interesting issue is the success of GCEDA. The explanation is simple. On one hand, Summation Cancellation has multivariate linear interactions between the variables [4]. On the other hand, the multivariate normal distribution is indeed, a linear model of interactions.

4. Finally, a few words about the docking example. Following the same line of reasoning of the above paragraph and looking at Table VI, we could conclude that this problem has a certain amount of linear interactions, which explains the difference between UMDA and GCEDA. However, VEDAs have better behavior, since their ability to deal with different patterns of dependence by combining different copulas (in this case, the product and normal copulas). The differences between CVEDA and DVEDA can be explained by the differences of their structures: More flexible regular vines could do a better job. Indeed, although the construction procedure of the C-vine intends to represent explicitly the strongest correlations in the first tree, the constraint that only one variable can be connected to all the others may prevent some strong correlations to be included. As has been emphasized in [6], D-vines allow a more flexible selection of the dependencies to be explicitly modeled, while C-vines might be more appropriate in cases where one of the variables governs the interactions.

V. Conclusions

In this paper we have investigated four copula-based EDAs: UMDA, GCEDA, CVEDA and DVEDA. We have found that the vine-based EDAs are in general more flexible, efficient and robust than the other two, which can be explained by their ability to describe a wider variety of dependence patterns. However, we also have found examples where UMDA and GCEDA outperform the investigated VEDAs. This, once again, points out to the importance of model selection in the context of EDAs. The results also suggest that the use of vines in EDAs open new opportunities to more appropriate modeling of the search distributions.

Referencias

[1] H. Mühlenbein and G. Paas, "From Recombination of Genes to the Estimation of Distributions I. Binary Parameters," *Lecture Notes in Computer Sciences, Parallel Problem Solving from Nature PPSN IV*, vol. 1141, pp. 178–187, 1996.

[2] P. Larrañaga and J. A. Lozano, Eds., *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*, Kluwer Academic Publisher, 2002.

[3] S. Kern, S. D. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos, "Learning probability distributions in continuous evolutionary algorithms – A comparative review," *Natural Computing*, vol. 3, pp. 77–112, 2003.

[4] P. A. N. Bosman and D. Thierens, "Numerical optimization with real-valued estimation of distribution algorithms," in *Scalable Optimization via Probabilistic Modeling. From Algorithms to Applications*, M. Pelikan,

K. Sastry, and E. Cantú-Paz, Eds., pp. 91–120. Springer-Verlag, 2006.

[5] R. B. Nelsen, *An introduction to copulas*, Springer-Verlag, second edition, 2006.

[6] K. Aas, C. Czado, A. Frigessi, and H. Bakken, "Pair-copula constructions of multiple dependence," *Insurance: Mathematics and Economics*, vol. 44, no. 2, pp. 182–198, 2009.

[7] T. Bedford and R. M. Cooke, "Vines — a new graphical model for dependent random variables," *The Annals of Statistics*, vol. 30, no. 4, pp. 1031–1068, 2002.

[8] D. Kurowicka and R. M. Cooke, *Uncertainty analysis with high dimensional dependence modelling*, John Wiley & Sons, 2006.

[9] H. Joe, *Multivariate models and dependence concepts*, Chapman & Hall, 1997.

[10] A. Sklar, "Fonctions de répartition à n dimensions et leurs marges," *Publications de l'Institut de Statistique de l'Université de Paris*, vol. 8, pp. 229–231, 1959.

[11] M. Soto, A. Ochoa, and R. J. Arderí, "Gaussian copula estimation of distribution algorithm," Tech. Rep. ICIMAF 2007-406, Institute of Cybernetics, Mathematics and Physics, Cuba, June 2007, ISSN 0138-8916.

[12] R. J. Arderí, "Algoritmo con estimación de distribuciones con cópula gaussiana," Bachelor thesis, University of Havana, Cuba, June 2007.

[13] G. Fusai and A. Roncoroni, *Implementing Models in Quantitative Finance: Methods and Cases*, chapter Structuring Dependence using Copula Functions, pp. 231–267, Springer-Verlag, 2008.

[14] L.-F. Wang, J.-C. Zeng, and Y. Hong, "Estimation of Distribution Algorithm based on copula theory," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2009)*, May 2009, pp. 1057–1063.

[15] Y. Gao, "Multivariate Estimation of Distribution Algorithm with Laplace transform Archimedean copula," in *Proceedings of the International Conference on Information Engineering and Computer Science (ICIECS 2009)*, W. Hu and X. Li, Eds., Dec. 2009.

[16] A. Cuesta-Infante, R. Santana, J. I. Hidalgo, C. Bielza, and P. Larrañaga, "Bivariate empirical and n-variate Archimedean copulas in Estimation of Distribution Algorithms," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2010)*, July 2010, pp. 1355–1362.

[17] H. Joe, "Families of m-variate distributions with given margins and $m(m-1)/2$ bivariate dependence parameters," in *Distributions with fixed marginals and related topics*, L. Rüschendorf, B. Schweizer, and M. D. Taylor, Eds., 1996, pp. 120–141.

[18] M. Soto, Y. González-Fernández, and A. Ochoa, "Modeling with copulas and vines in estimation of distribution algorithms," Submited for publication to IEEE Transactions on Evolutionary Computation.

[19] M. Soto and Y. González-Fernández, "Vine Estimation of Distribution Algorithms," Tech. Rep. ICIMAF 2010-561, Institute of Cybernetics, Mathematics and Physics, Cuba, May 2010, ISSN 0138-8916.

[20] Y. González-Fernández, "Algoritmos con estimación de distribuciones basados en cópulas y vines," Bachelor thesis, University of Havana, Cuba, June 2011.

[21] E. C. Brechmann, "Truncated and simplified regular vines and their applications," Diploma thesis, Technische Universität München, Oct. 2010.

[22] D. J. Rosenkrantz, R. E. Stearns, and II M. Lewis Philip, "An analysis of several heuristics for the Traveling Salesman Problem," *SIAM Journal on Computing*, vol. 6, no. 3, pp. 563–581, 1977.

[23] C. Genest and B. Rémillard, "Tests of independence or randomness based on the empirical copula process," *Test*, vol. 13, no. 2, pp. 335–369, 2004.

[24] C. Genest, B. Rémillard, and D. Beaudoin, "Goodness-of-fit tests for copulas: A review and a power study," *Insurance: Mathematics and Economics*, vol. 44, pp. 199–213, 2009.

[25] C. Genest and A. C. Favre, "Everything you always wanted to know about copula modeling but were afraid to ask," *Journal of Hydrologic Engineering*, vol. 12, no. 4, pp. 347–368, 2007.

[26] S. Demarta and A. J. McNeil, "The t copula and related copulas," *International Statistical Review*, vol. 73, no. 1, pp. 111–129, 2005.

[27] T. Bedford and R. M. Cooke, "Monte Carlo simulation of vine dependent random variables for applications in uncertainty analysis," in *Proceedings of ESREL 2001*, 2001.

[28] T. Bedford and R. M. Cooke, "Probability density decomposition for conditionally dependent random variables modeled by vines," *Annals of Mathematics and Artificial Intelligence*, vol. 32, no. 1, pp. 245–268, 2001.

[29] D. Kurowicka and R. M. Cooke, "Sampling algorithms for generating joint uniform distributions using the vine-copula method," in *3rd IASC world conference on Computational Statistics & Data Analysis*, 2005.

[30] L. Devroye, *Non-Uniform Random Variate Generation*, Springer-Verlag, 1986, ISBN 0-387-96305-7.

[31] Y. González-Fernández and M. Soto, *copulaedas: Estimation of Distribution Algorithms based on copula theory*, 2011, R package version 1.0.3.

[32] Y. González-Fernández and M. Soto, *vines: Multivariate dependence modeling with vines*, 2011, R package version 1.0.3.

[33] R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2010, ISBN 3-900051-07-0.

[34] M. Soto, A. Ochoa, Y. González-Fernández, Y. Milanés, A. Álvarez, and E. Moreno, "Vine Estimation of Distribution Algorithms with application to Molecular Docking," in *Markov networks in Evolutionary Computation*, R. Santana and S. Shakya, Eds. Springer-Verlag, 2011.

[35] E. Bengoetxea, T. Miquélez, J. A. Lozano, and P. Larrañaga, "Experimental results in function optimization with EDAs in continuous domain," in *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*, P. Larrañaga and J. A. Lozano, Eds., pp. 181–194. Kluwer Academic Publisher, 2002.

[36] M. Pelikan, *Hierarchical Bayesian Optimization Algorithm. Toward a new generation of Evolutionary Algorithms*, Springer-Verlag, 2005.

[37] H. Mühlenbein and T. Mahnig, "FDA — a scalable evolutionary algorithm for the optimization of additively decomposed functions," *Evolutionary Computation*, vol. 7, no. 4, pp. 353–376, 1999.

[38] G. L. Warren, C. W. Andrews, A. M. Capelli, B. Clarke, J. LaLonde, M. H. Lambert, M. Lindvall, N. Nevins, S. F. Semus, S. Senger, G. Tedesco, I. D. Wall, J. M. Woolven, C. E. Peishoff, and M. S. Head, "A critical assessment of docking programs and scoring functions," *Journal of Medicinal Chemistry*, vol. 49, pp. 5912–5931, 2006.

[39] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The protein data bank," *Nucleic Acids Research*, vol. 28, pp. 235–242, 2000.

[40] R. Huey, G. M. Morris, A. J. Olson, and D. S. Goodsell, "A semiempirical free energy force field with charge-based desolvation," *Journal Computational Chemistry*, vol. 28, pp. 1145–1152, 2007.

[41] G. M. Morris, D. S. Goodsell, M. E. Pique, W. Lindstrom, R. S. Halliday, R. Huey, S. Forli, W. E. Hart, R. K. Belew, and A. J. Olson, *Automated Docking of Flexible Ligands to Flexible Receptors. User Guide AutoDock. Version 4.2*, 2010.

[42] A. Auger, T. Blackwell, D. Bratton, M. Clerc, S. Croussette, A. Dattasharma, R. Eberhart, N. Hansen, H. Keko, J. Kennedy, R. Krohling, W. Langdon, W. Li, A. Liu, V. Miranda, R. Poli, P. Serra, and M. Stickel, *Standard PSO 2007*, 2007, http://www.particleswarm.info/.