Vine Estimation of Distribution Algorithms with Application to Molecular Docking

Marta Soto, Alberto Ochoa, Yasser González-Fernández, Yanely Milanés, Adriel Álvarez, Diana Carrera, and Ernesto Moreno

Abstract
Four undirected graphical models based on copula theory are investigated in relation to their use within an estimation of distribution algorithm (EDA) to address the molecular docking problem. The simplest algorithms considered are built on top of the product and normal copulas. The other two construct high-dimensional dependence models using the powerful and flexible concept of vine-copula. Empirical investigation with a set of molecular complexes used as test systems shows state-of-the-art performance of the copula-based EDAs in the docking problem. The results also show that the vine-based algorithms are more efficient, robust and flexible than the other two. This might suggest that the use of vines opens new research opportunities to more appropriate modeling of search distributions in evolutionary optimization.

1 Introduction

Estimation of distribution algorithms (EDAs) [26, 29] have been used successfully in several Bioinformatic problems [3], such as *de novo* peptide design [7], protein folding [34] and protein side chain placement [35]. In this chapter we investigate the

Marta Soto, Alberto Ochoa, Yasser González-Fernández
Institute of Cybernetics, Mathematics, and Physics, Cuba.
e-mail: {mrosa,ochoa,ygf}@icimaf.cu

Yanely Milanés, Adriel Álvarez, Diana Carrera
University of Havana, Cuba.
e-mail: {y.milanes,a.mosquera,d.carrera@lab.matcom.uh.cu}

Ernesto Moreno
Center of Molecular Immunology, Cuba.
e-mail: emoreno@cim.sld.cu

performance of several EDAs based on undirected copula models in the molecular docking problem.

Molecular docking consists in predicting the binding geometry of small flexible ligands that bind a large macromolecular entity, in most cases a protein. Currently, docking has become an important component of structure-based drug design, with the aim of optimizing the initial screening phases of the long and costly process for developing a new drug.

In this chapter, several undirected graphical models based on copula theory are combined into four different estimation of distribution algorithms to address the molecular docking problem. Two algorithms are built around the product and normal copulas, while the other two are based on vines, which constitute a powerful and flexible class of high-dimensional dependency models able to represent a rich variety of dependencies and marginal distributions. We pursue two goals: to test the potential of EDAs in the docking problem and evaluate the relative performance of EDAs based on vines in comparison to unstructured copula models.

The outline of the chapter is as follows. Section 2 gives a necessary background on copula theory and describes two EDAs, UMDA and GCEDA, which are based on the product and normal copulas, respectively. Section 3 describes two algorithms based on the copula-vine models, DVEDA and CVEDA. Section 4 provides a brief introduction on the molecular docking problem and the fitness function model. The empirical investigation is reported in Section 5. Numerical results with the genetic, particle swarm optimization and differential evolution algorithms are included in Section 6. The conclusions are given in Section 7.

2 Two Continuous EDAs based on Undirected Models

Nowadays, there is an increasing interest on EDA approaches based on copula theory. Several copula-based EDAs have been proposed in the literature, for example, [12, 40, 41]. In this section we briefly describe two multivariate copula-based EDAs that are relevant to this work.

A multivariate normal distribution forms a Markov random field (MRF) with respect to a graph $G = (V,E)$ if the missing edges correspond to zeros on the precision matrix (the inverse covariance matrix) $X = (X_i)_{i\in V} \sim N(\mu,\Sigma)$, so that $\left(\Sigma^{-1}\right)_{ij} = 0$ if $\{i, j\} \notin E$.

As important subclasses of this Gaussian MRF one can list the models associated with the independence graph, trees and join trees, and also several models based on copula theory. In this chapter we focus on the later class of algorithms. We start with some definitions from the copula theory.

A comprehensive treatment on copula theory is given in [22, 31]. The notion of copulas separates the effect of dependence from the effect of margins (scale, localization) in a joint distribution [25]. In this sense, copulas are functions that join multivariate distributions to their margins. The theory supports this definition with the Sklar's theorem [36].

Let $\mathbf{X} = (X_1, \ldots, X_n)$ and $\mathbf{x} = (x_1, \ldots, x_n)$ denote a continuous random vector and one of its possible configurations, respectively. Let also f, F and $F_1, \ldots, F_n$ be the joint density function, the joint cumulative distribution function and the margins associated to $\mathbf{X}$. The basic relations supported by the Sklar's theorem are

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

and

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

where C is a unique cumulative distribution function with uniform marginal distributions in $[0,1]$.

An immediate consequence of Sklar's theorem is that the random variables in a vector are independent if and only if their underlying copula corresponds to the product copula, which is given by

$$C_I(u_1, \ldots, u_n) = u_1 \ldots . u_n. \qquad (1)$$

The famous Univariate Marginal Distribution Algorithm (UMDA) [26] for continuos variables is obviously connected to (1) because it entails independence among all variables. The UMDA has proven to be very successful even in situations where it was not expected to behave like this.

Another important copula is the multivariate normal copula,

$$C_N(u_1, \ldots, u_n; R) = \Phi_R\left(\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_n)\right), \qquad (2)$$

where Φ_R is the standard multivariate normal distribution function with linear correlation matrix R and Φ^{-1} is the inverse of the standard univariate normal distribution. Although in (2), Φ is the standard normal distribution, this copula can also be associated with other types of marginal distributions. If that is the case, the created joint density is not longer the multivariate normal, although the normal dependence structure is preserved.

With non-normal margins, the correlation matrix is estimated using either $\hat{R}_{ij} = sin\left(\pi/2\hat{\tau}_{ij}\right)$ or $\hat{R}_{ij} = 2sin\left(\pi/6\hat{\rho}_{s_{ij}}\right)$ $(i, j = 1, \ldots, n$ for each pair of variables $i, j)$. Here $\hat{\tau}$ and $\hat{\rho}_s$ are the non-parametric estimators Kendall's tau and Spearman's rho, respectively. If the resulting $\hat{R}$ is not positive-definite the correction proposed in [33] is applied.

Our second algorithm, the Gaussian Copula EDA (GCEDA) [38], was designed based on the multivariate normal copula.

As it was pointed out, a key advantage of the models based on copula is their ability to deal with different types of marginal distributions. In this work, this property is used to accommodate the range constraints of the problem. Indeed, the algorithms use the truncated normal distribution [23], which is confined to a given real interval $[a, b]$. The density function with mean μ and standard deviation $\sigma \geq 0$ is given by

$$f(x;\mu,\sigma,a,b)=\frac{\frac{1}{\sigma}\phi(\frac{x-\mu}{\sigma})}{\Phi(\frac{b-\mu}{\sigma})-\Phi(\frac{a-\mu}{\sigma})},$$

and the cumulative distribution by

$$F(x;\mu,\sigma,a,b)=\frac{\Phi(\frac{x-\mu}{\sigma})-\Phi(\frac{a-\mu}{\sigma})}{\Phi(\frac{b-\mu}{\sigma})-\Phi(\frac{a-\mu}{\sigma})},$$

where ϕ and Φ denote the probability density and the cumulative distribution of the standard univariate normal, respectively.

3 Vine Approach in EDAs

This section provides a brief description of the C-vine and D-vine models and the motivation for using them to construct the search distributions of the EDAs. Furthermore, we introduce CVEDA and DVEDA, our third and fourth algorithms.

3.1 From Multivariate Copulas to Vines

Due to the ability of copulas to split join densities into marginal information and a dependence structure, they allow the modeling of more realistic search distributions. This overcomes the constraints imposed by the multivariate normal.

Nevertheless, the multivariate copula approach has also several shortcomings, which have been clearly identified in the literature [1, 25]. Firstly, the number of tractable copulas when more than two variables are involved is rather limited. In fact, the majority of the available parametric copulas are bivariate. Secondly, the multivariate copula approach is not appropriate when not all the pairs of variables have the same dependence structure. Finally, most of the existing multivariate extensions have only one parameter to describe the overall dependence, which is a serious issue when not all pairs of variables share the same type of dependencies.

An alternative to the multivariate copula approach is the pair-copula decomposition or vine [11]. This technique permits to extend bivariate copulas to higher dimensions by using them as building blocks. The ability of these methods to model a rich variety of dependencies by combining bivariate copulas of different families has motivated a growing research activity in the area. Additionally, when modeling with vines, we do not need to make any assumption about the type of dependence between the variables, since the estimation procedure will choose the bivariate copula that fits the data appropriately.

3.2 Vine Modeling Approach

Vines are dependency models of multivariate distribution functions based on the decomposition of $f(x_1,\ldots,x_n)$ into pair-copulas and marginal densities [5, 6]. 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 $j=1,\ldots,n-1$) and each tree has the maximum number of edges. Regular vines constitute a special case of vine in which two edges in tree j are joined by an edge in tree $j+1$ only if these edges share a common node. Two types of regular vines are C-vines (canonical vines) and D-vines (drawable vines). Figure 1 shows a four-dimensional C-vine and D-vine. Each of these graphical models gives a specific way of decomposing the density into pair-copulas and marginal densities [25]. In particular, the C-vine density is given by

$$\prod_{k=1}^{n} f(x_k) \prod_{j=1}^{n-1}\prod_{i=1}^{n-j} c_{j,j+i|i,\ldots,j-1}\left(F\left(x_j|x_1,\ldots,x_{j\text{-}1}\right), F\left(x_{j+i}|x_1,\ldots,x_{j-1}\right)\right), \tag{3}$$

and the D-vine density is given by

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

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

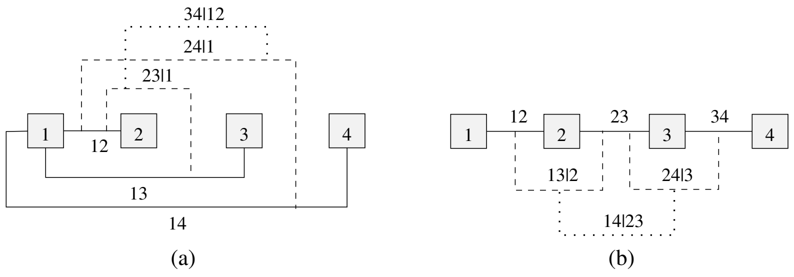

Fig. 1: Four-dimensional C-vine (a) and D-vine (b). Two edges in T_j, which become nodes in T_{j+1}, are joined by an edge in T_{j+1} if these edges in T_j share a common node. In a C-vine, each tree T_j has a unique node that is connected to $n-j$ edges. In a D-vine, no node in any tree is connected to more than two edges.

Note that in (3) and (4) the pair-copulas are evaluated at conditional distributions of the form

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

where $C_{xv_j|\mathbf{v}_{-j}}$ is a bivariate copula distribution function [21], $\mathbf{v}$ 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 $F(x \mid \mathbf{v})$ yields the expression

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

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

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

is defined to facilitate de computation of $F(x \mid v)$, where θ denotes the set of parameters for the copula of the joint distribution function of x and v. Moreover, h^{-1} is the inverse of h with respect to the first variable (the inverse of $F(x \mid v)$). The derivation of h^{-1} for different bivariate copulas can be found in [1].

3.3 Vine Estimation of Distribution Algorithm

Vine Estimation of Distribution Algorithm (VEDA) [16, 37] is a class of EDA that uses vine-copula to model the search distributions. CVEDA and DVEDA are VEDAs based on C-vine and D-vine, respectively. Next we describe the particularities of the estimation and simulation steps of these algorithms.

3.3.1 Estimation

The methods for the construction of C-vines and D-vines have been developed in [1]. They consist of the following steps:

1. Selection of a specific factorization.
   In the current implementation, when constructing a C- or D-vine, the strongest relationships between the nodes define the first tree. Then, the structure of the subsequent trees can be defined arbitrarily starting from the first tree. The sum of certain weights assigned to each edge of the first tree (for instance, the absolute value of the empirical Kendall's τ between a pair of variables) is used by a heuristics to select the decomposition. In particular:
   - The construction of a C-vine begins by determining the weights between each variable (the possible root) and the others. The node of the tree that maximizes the sum of the weights of its edges is chosen as the root node of the first tree. In the subsequent trees, any edge in T_j is chosen as the root in T_{j+1}.
   - In a D-vine the final structure is uniquely determined by the structure of the first tree. D-vines are more flexible than C-vines since we can select more

freely the pairs to model. The problem of constructing the first tree consists in finding the maximum weighted sequence of the variables. In [9] this problem is transformed into a Traveling Salesman Problem (TSP) by adding a dummy node with weight zero on all edges to the other nodes. To find a solution of TSP we use the cheapest insertion greedy heuristic [19, 32].

A n-dimensional vine has $n-1$ trees and requires the fitting of copulas in $n(n-1)/2$ edges. Assuming conditional independence may reduce the number of trees (if X and Y are conditionally independent given $\mathbf{V}$, then $c_{xy|\mathbf{v}}\left(F_{x|\mathbf{v}}(x \mid \mathbf{v}), F_{y|\mathbf{v}}(y \mid \mathbf{v})\right) = 1$). In order to simplify the construction of the model we apply the truncantion strategy presented in [9, 10]. According to this strategy, truncating a vine at level K means that all pair-copulas with conditioning set equal to or larger than K are replaced by independence copulas.
To identify the most appropriate truncation level, a greedy model selection procedure based on Akaike Information Criterion (AIC) [2] is used: The tree T_{j+1} is expanded if the AIC value calculated up to tree T_{j+1} is smaller than the AIC value obtained up to the previous tree; otherwise, the vine is truncated at T_j. It is worth noting that assuming conditional independence from an arbitrary tree level may seriously damage the performance of VEDA [37].

2. Choice of the pair-copula types in the factorization and estimation of the copula parameters.

   a. Determine the pair-copulas in the first tree from the original data.
   b. Compute observations (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.

   In this work, both the bivariate product and normal copulas are used in the vine. To fit these copulas, we first apply an independence test based on the distance between the empirical and the product copulas measured by a Cramér-von Mises statistic [14]. Then the product copula is fitted if there is not enough evidence against the null hypothesis of independence at a fixed significance level of 0.01. Otherwise, the normal copula is fitted using the inversion of Kendall's τ [13].

3.3.2 Simulation

The simulation procedure of the C-vines and D-vines is based on the general sampling algorithm for n dependent Uniform$(0,1)$ variables, allowing us to create a sample from the joint distribution function. The algorithm first samples n independent uniform random numbers $w_i \in (0,1)$, and then computes $x_1, x_2, \ldots, x_n$ according to $x_1 = w_1,\ x_2 = F^{-1}_{2|1}(w_2|u_1), \ldots, x_n = F^{-1}_{n|1,2,\ldots,n-1}(w_n|u_1,\ldots,u_{n-1})$.

4 Molecular Docking with VEDA

Molecular docking is a computational procedure to predict the geometry of binding of two molecules. Often, one of these molecules, the "receptor", is a protein, while the second one, the "ligand", is a small molecule that binds into the protein active site, usually a pocket or a groove on the protein surface. 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 [42]. In this work we address the conformational sampling problem.

In the chapter the protein is treated as a rigid body and the ligand as fully flexible. Therefore, the individuals represent the position, orientation and torsion angles of the ligand. The first three variables account for the ligand position in the three dimensional space. The 3D sampling is limited by a box enclosing the receptor binding site. For each test system, this box was constructed based on the minimum and maximum values of the ligand coordinates in its crystal conformation, plus a padding distance of $\pm 5A$ in each space dimension.

To represent the ligand orientation, we use three variables, the Euler angles α, β, γ, that take values in $[0, 2\pi]$, $[-\pi/2, \pi/2]$ and $[-\pi, \pi]$, respectively. Thus, the position and orientation are defined using in total six variables. Finally, each flexible torsion angle in the ligand accounts for one variable, which takes values in $[-\pi, \pi]$. Hence, the dimension of the optimization problem is six plus the number of torsion angles.

We use the semiempirical scoring function implemented in Autodock 4.2 [20]. The overall docking energy of a given ligand molecule is expressed as the sum of the pairwise interactions between receptor and ligand atoms (the intermolecular interaction energy) and the pairwise interactions between ligand atoms (the ligand intramolecular energy) as given by the expression

$$E = \Delta G_{vdv} E_{vdw} + \Delta G_{hbond} E_{hbond} + \Delta G_{elec} E_{elec} + \Delta G_{solv} E_{solv}$$

The first three terms consider dispersion/repulsion, hydrogen bonding and the electrostatic potential. The last term accounts for desolvation effects. All terms are scaled empirically. To save calculation time, the scoring function is evaluated using grids defined within the box enclosing the binding site, which account for the contributions from receptor atoms.

To evaluate the quality of the predicted ligand conformations, they are compared with the experimental crystal structure using the Cartesian root-mean-square deviation, $RMSD = \sqrt{\Sigma_{i=1}^{n} \left(dx_i^2 + dy_i^2 + dz_i^2\right)/n}$, where n is the number of ligand atoms and dx_i, dy_i and dz_i are the deviations between the crystallographic and predicted ligand coordinates of the i^{th} atom. A structure with an $RMSD$ within $2A$ is classified as successfully docked, while a structure with an $RMSD$ between 2 and $3A$ is classified as partially docked.

5 Experiments

This section presents the experimental design and reports the numerical results of our empirical investigation.

5.1 Experimental Design

Table 1 presents the four protein-ligand complexes used as test systems, solved by X-ray crystallography and available from the Protein Data Bank (PDB) [8]. They were selected taking into account the large number of torsion angles in the ligand molecule.

Table 1: Description of the test systems used in the experiments.

PDB code	Protein receptor	Number of ligand atoms*	Box dimensions (A)	Number of ligand torsions
`1adb`	Alcohol dehydrogenase	56	$28 \times 20 \times 22$	15
`1bmm`	Alpha-Thrombin	43	$17 \times 19 \times 22$	10
`1cjw`	Serotonin N-acetyltransferase	71	$26 \times 22 \times 30$	26
`2z5u`	Lysine-specific histone demethylase 1	73	$28 \times 32 \times 24$	20

* Non-polar hydrogens are not counted.

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. It is worth noting that the optimal energy values for the test problems are unknown. To accomplish this, each algorithm was run 30 times using population sizes in the range between 200 and 2000 individuals with an increment of 200.

The EDAs stop when the standard deviation of the energy in the population is less than 0.01. The algorithms use a truncation selection of 0.3 and no elitism.

At this point, it is worth noting that we are not interested in the study of any sort of memetic strategy. We are aware of the possibilities of such methods and recommend their use in practical applications. However, for the sake of clarity of our results, we concentrate ourselves in the global exploration capabilities of the algorithms.

The algorithms were implemented using two packages programmed in R: copulaedas [17] and vines [18], which provide tools to build EDAs based on copula functions.

5.2 Numerical Results and Discussion

In this section we analyze the relative performance of the algorithms. For the EDAs based on vines we analyze how the number of bivariate normal copulas fitted to the arcs varies during the evolution. Finally, several considerations about the adequacy of the fitness function model are given.

Table 2 summarizes the results obtained for each test system and each algorithm, in terms of lowest energy ("best" solution), the number of function evaluations and the *RMSD* of the best solution with respect to the experimental conformation. The numbers shown in the table are the average values and their standard deviations over 30 independent runs. Figure 2 illustrates how the minimum energy decreases as the number of function evaluations increases during the optimization process.

As shown in Table 2 and Figure 2, GCEDA, CVEDA and DVEDA perform better than UMDA, except for the 1bmm system. On the other hand, DVEDA and GCEDA behave slightly better than CVEDA. In general, CVEDA and DVEDA are able to perform satisfactorily both in problems with weak and strong interactions between the variables. While UMDA assumes independence and GCEDA postulates a normal dependency structure, CVEDA and DVEDA assign the bivariate copula that fits best the data; in this particular application, either an independence or a normal copula is chosen. Figure 3 illustrates, as example, correlations that DVEDA finds in the selected populations in 1bmm and 2z5u at generations 60 and 90, respectively (these are the average generations were DVEDA fits more normal copulas in 1bmm and 2z5u (see Figure 4)). In 1bmm, only weak correlations are observed, while in 2z5u, strong correlations between several pairs of variables (torsion angles) are evident.

We now turn our attention to a comparison of 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. Figure 4 presents the results for the 1bmm and 2z5u test instances.

The number of bivariate copulas fitted by the algorithms increases during the evolution for the 2z5u instance. For 1bmm we observe the same behaviour up to the generation 60, and then decreases. It is interesting to note that after this point CVEDA fits slightly more normal copulas than DVEDA. Otherwise DVEDA fits more normal copulas than CVEDA, which is particularly significant in 2z5u. 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 [1], 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.

It is worth noting that because of the use of a truncation procedure (see Section 3.3.1) the number of statistical tests was dramatically reduced. In 2z5u the average number of fitted trees was below eight with CVEDA and below ten with DVEDA, while in 1bmm, only a maximum of nine trees were fitted by both algorithms.

Table 2: Average performance of copula-based EDAs.

PDB code	Algorithm	Population	Evaluations	Lowest Energy	*RMSD*
1adb	UMDA	1800	157933 ± 11286	-16.55 ± 2.09	3.17 ± 0.76
	CVEDA	1400	114258 ± 17585	-17.01 ± 2.18	2.63 ± 0.98
	DVEDA	1400	112133 ± 13920	-17.61 ± 1.79	2.50 ± 1.03
	GCEDA	1400	108200 ± 20162	-18.69 ± 0.84	1.44 ± 0.81
1bmm	UMDA	600	48633 ± 6099	-9.42 ± 1.37	4.58 ± 0.40
	CVEDA	800	65766 ± 10500	-9.11 ± 1.39	4.88 ± 0.49
	DVEDA	800	62266 ± 9776	-9.41 ± 1.33	4.77 ± 0.71
	GCEDA	1000	72166 ± 18632	-8.31 ± 1.10	4.99 ± 0.90
1cjw	UMDA	1200	180000 ± 13341	-12.19 ± 1.46	6.24 ± 1.04
	CVEDA	1200	180233 ± 13103	-12.85 ± 1.69	6.35 ± 1.33
	DVEDA	1200	165400 ± 18872	-14.40 ± 2.40	5.20 ± 1.34
	GCEDA	1600	201033 ± 37696	-15.55 ± 2.02	4.97 ± 1.26
2z5u	UMDA	1400	171900 ± 11442	-29.14 ± 1.97	0.61 ± 0.18
	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

The ultimate test for a docking algorithm concerns the ability to reproduce the experimentally observed ligand conformations. This is usually evaluated in terms of the *RMSD* between the predicted and the crystal ligand coordinates, and the results heavily depend on the scoring function. Although such type of analysis involving the scoring function is outside the scope of this investigation, we illustrate here, in Figures 5 and 6, the results obtained from this test for the 1adb system.

Figure 5 shows a plot of energy versus *RMSD* for the best solutions obtained at generations 26, 31, 40 and 85. Because of the limitations of the scoring function, there is no correlation between the energy scores and the ligand RMSDs, so that solutions with higher energies may have better RMSDs. Figure 6 illustrates such a situation for the ligand molecule in 1adb. The right half of the ligand is buried in the protein binding site, so its conformation is very restricted by the surrounding protein atoms. On the contrary, the left half of the ligand is exposed to the solvent and therefore is less affected by the steric constraints imposed by the protein (and so by the scoring function). The best solution at generation 85 (panel b), although it has a higher overall *RMSD*, shows a better fit for the buried part (right half), which explains its better energy as compared to solution at generation 26 (panel a).

6 AG, PSO and DE in the Molecular Docking Problem

In this section we include, for comparison with the EDAs, the results obtained with simple versions of a genetic algorithm (GA) [15] and a particle swarm optimization

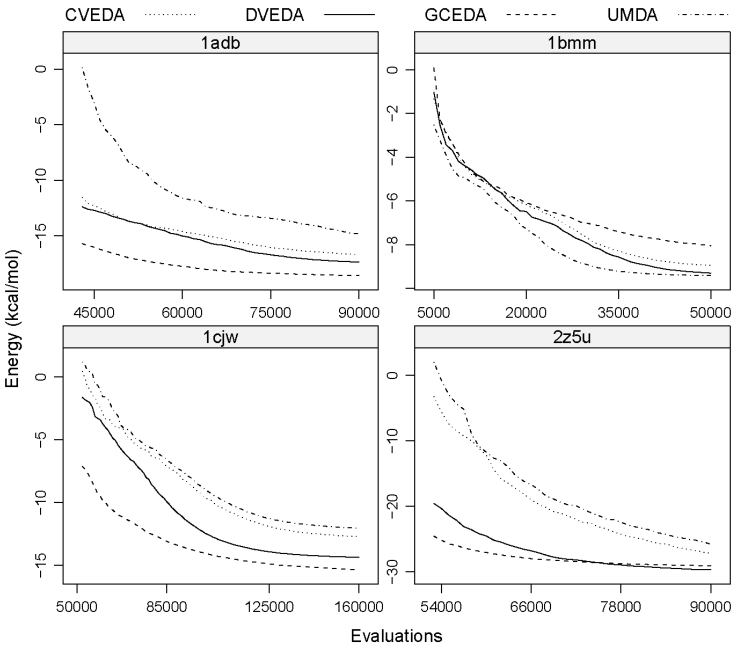

Fig. 2: Energy *versus* number of function evaluations during the evolution in the four protein-ligand test instances.

algorithm (PSO) [24]. These two population-based algorithms have been used in molecular docking applications [27, 30]. We also consider a differential evolution algorithm (DE) [39].

Similarly, as we did for the EDAs, we determined the population size that yields the lowest energy with the smallest number of evaluations. Each algorithm was run 30 times using population sizes in the range between 20 and 200 individuals with an increment of 20 for GA, and between 50 and 400 individuals with an increment of 50 for PSO and DE.

For GA [28] 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 [4]) we use an inertia weight of $1/2log(2)$, the user defined behavioral 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, differential mutation rate of 0.5, and a mutation scale factor of 0.8.

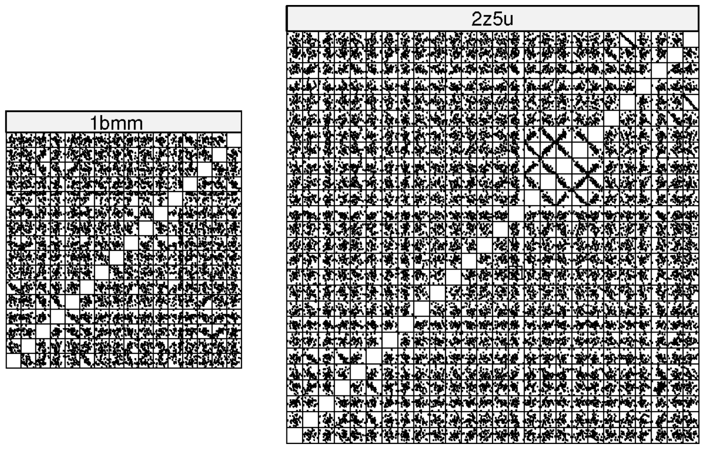

Fig. 3: Correlations in selected populations with DVEDA in 1bmm and 2z5u at generations 60 and 90, respectively. In 1bmm, weak correlations are observed, while in 2z5u the strong correlations between 11 pairs of variables (where seven torsion angle variables are involved) are evident.

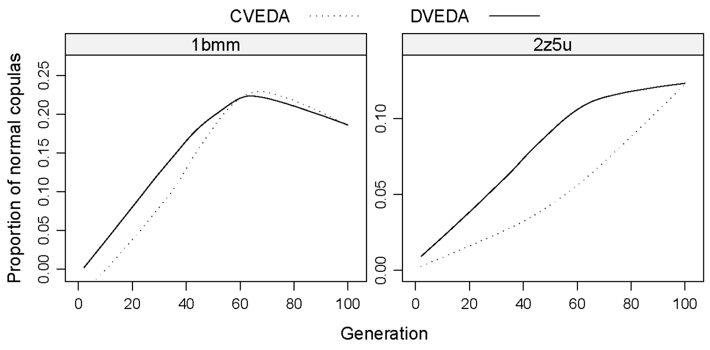

Fig. 4: Comparison of CVEDA and DVEDA for the 1bmm and 2z5u test instances, in terms of the ratio between the number of fitted normal copulas and the number of arcs in the vine, at different generations. For 1bmm, there are 15 trees and 120 arcs in a 16-dimensional D-vine and C-vine. For the 2z5u, there are 25 trees and 325 arcs in a 26-dimensional D-vine and C-vine.

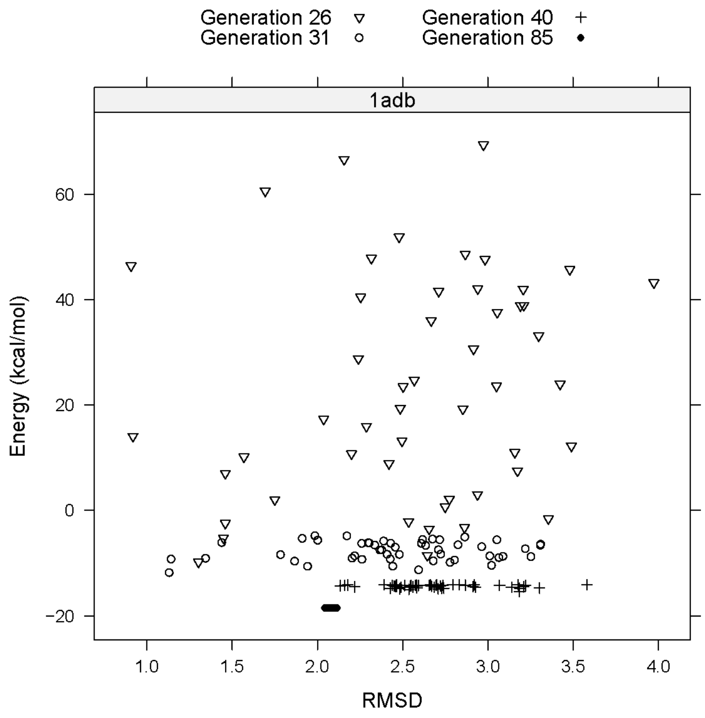

Fig. 5: Energy *versus RMSD* for the best solutions generated by DVEDA at different generations, for the 1adb system. Solutions with higher (worse) energies may have better RMSDs.

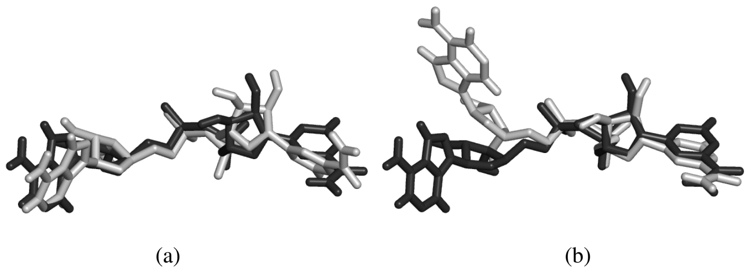

Fig. 6: Comparison of the experimental ligand conformation in 1adb (in black) with two predicted structures generated by DVEDA (in white). At generation 26 (a) the solution with the lowest *RMSD* ($0.8A$) has an energy value of $+48.7 kcal/mol$, while at generation 85 (b) the solution with the lowest energy ($-17.6 kcal/mol$) has a *RMSD* of $2.5A$.

The GA stops after a maximum number of 2.5 million energy evaluations, while PSO and DE stop after one million evaluations (the alternative stopping condition: if the standard deviation of the energy in the population is less than 0.01, is never achieved by these algorithms in the test suite used).

A comparison of the results shown in Table 2 with those in Table 3, confirms that copula-based EDAs outperform GA, PSO and DE, which is particularly significant for the GA. Only in 1bmm, PSO and DE perform better than EDAs. However, it is worth noting the big difference on the number of evaluations.

Table 3: Average performance of GA, PSO and DE.

PDB code	Algorithm	Population	Evaluations	Lowest Energy	*RMSD*
1adb	GA	20	1808000 ± 432005	-8.72 ± 5.30	10.71 ± 1.91
	PSO	100	863366 ± 140913	-8.20 ± 1.54	9.19 ± 2.23
	DE	200	921073 ± 93455	-12.61 ± 1.66	4.94 ± 2.47
1bmm	GA	140	2141200 ± 301527	-8.84 ± 0.81	6.62 ± 0.92
	PSO	300	936630 ± 132021	-11.88 ± 2.06	3.98 ± 2.25
	DE	300	933060 ± 78343	-14.40 ± 1.29	2.69 ± 1.98
1cjw	GA	40	2235400 ± 225130	-11.09 ± 1.79	11.00 ± 2.89
	PSO	100	915123 ± 101167	-7.39 ± 1.49	9.20 ± 1.68
	DE	100	955600 ± 48426	-12.94 ± 0.52	7.28 ± 1.57
2z5u	GA	20	2265600 ± 273062	$+483.50 \pm 538.39$	13.34 ± 4.21
	PSO	300	912510 ± 95512	42.52 ± 27.39	6.40 ± 2.95
	DE	100	972536 ± 47549	9.79 ± 21.48	7.49 ± 4.19

7 Conclusions

We have described an EDA approach to protein docking, which is based on undirected graphical models that use copulas. The empirical studies carried out on a test set of experimentally determined structures of protein-ligand complexes suggest that this approach performs competitively with the solutions of the state-of-the-art.

We have found that vine-based EDAs constitute promising tools for designing molecular docking algorithms, since they are endowed with mechanisms for building search distributions with different dependencies and with normal and non-normal margins. Vine-based EDAs are indeed more powerful and flexible than their predecessors UMDA and GCEDA. The results also suggest that the use of vines in EDAs open new opportunities to more appropriate modeling of the search distributions.

Acknowledgements The authors thank the anonymous referees and the editors for the valuable comments that helped to improve this chapter.

References

1. K. Aas, C. Czado, A. Frigessi, and H. Bakken. Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, 44:182–198, 2009.
2. H. Akaike. A new look at statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.
3. R. Armañananzas, I. Inza, R. Santana, Y. Saeys, Flores J. L., Lozano J. A., Y. Van de Peer, R. Blanco, V. Robles, C. Bielza, and P. Larrañaga. A review of estimation of distribution algorithms in bioinformatics. *BioData Mining*, 1(6), 2008.
4. 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, V. Liu A. Miranda, R. Poli, P. Serra, and M. Stickel. *Standard PSO 2007*, 2007. http://www.particleswarm.info/.
5. T. Bedford and R. M. Cooke. Probability density decomposition for conditionally dependent random variables modeled by vines. *Annals of Mathematics and Artificial Intelligence*, 32:245–268, 2001.
6. T. Bedford and R. M. Cooke. Vines – a new graphical model for dependent random variables. *The Annals of Statistics*, 30:1031–1068, 2002.
7. I. Belda, S. Madurga, X. Llorá, M. Martinell, T. Tarragó, M. Piqueras, E. Nicolás, and E. Giralt. ENPDA: An evolutionary structure-based de novo peptide design algorithm. *Journal of Computer-Aided Molecular Design*, 19(8):585–601, 2005.
8. 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*, 28:235–242, 2000.
9. E. C. Brechmann. Truncated and simplified regular vines and their applications. Diploma thesis, Technische Universität München, 2010.
10. E. C. Brechmann, C. Czado, and K. Aas. Truncated regular vines in high dimensions with application to financial data. Note SAMBA/60/10, Norwegian Computing Center, NR, 2010.
11. R. M. Cooke. Markov and entropy properties of tree- and vine-dependent variables. In *Proceedings of the American Statistical Association Section on Bayesian Statistical Science*, 1997.
12. 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)*, 2010.
13. C. Genest and A. C. Favre. Everything you always wanted to know about copula modeling but were afraid to ask. *Journal of Hydrologic Engineering*, 12:347–368, 2007.
14. C. Genest and B. Rémillard. Tests of independence or randomness based on the empirical copula process. *Test*, 13:335–369, 2004.
15. D. E. Goldberg. *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley, Reading, MA, 1989.
16. Y. González-Fernández. Algoritmos con estimación de distribuciones basados en cópulas y vines. Diploma thesis, University of Havana, July 2011.
17. Y. González-Fernández and M. Soto. *copulaedas: Estimation of distribution algorithms based on copula theory*, 2011. R package version 1.0.1. http://CRAN.R-project.org/package=copulaedas.
18. Y. González-Fernández and M. Soto. *vines: Multivariate dependence modeling with vines*, 2011. R package version 1.0.1. http://CRAN.R-project.org/package=vines.
19. M. Hahsler and K. Hornik. TSP – Infrastructure for the traveling salesperson problem. *Journal of Statistical Software*, 23:1–21, 2007.
20. 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*, 28:1145–1152, 2007.
21. 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*, pages 120–141, 1996.
22. H. Joe. *Multivariate models and dependence concepts*. Chapman & Hall, 1997.

23. N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous univariate distributions*, volume 1. John Wiley & Sons, 2 edition, 1994.
24. J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks IV*, pages 1942–1948, 1995.
25. D. Kurowicka and R. M. Cooke. *Uncertainty analysis with high dimensional dependence modelling*. John Wiley & Sons, 2006.
26. P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*. Kluwer Academic Publisher, 2002.
27. G. M. Morris, D. S. Goodsell, R. S. Halliday, R. Huey, W. E. Hart, R. K. Belew, and A. J. Olson. Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry*, 19(14):1639–1662, 1998.
28. 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.
29. H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions i. Binary parameters. In *Parallel Problem Solving from Nature — PPSN IV*, pages 178–187. Springer-Verlag, 1996.
30. V. Namasivayam and R. Günther. Flexible peptide-protein docking employing pso@autodock. In *From Computational Biophysics to Systems Biology (CBSB08)*, volume 40, pages 337–340, 2008.
31. R. B. Nelsen. *An introduction to copulas*. Springer-Verlag, 2 edition, 2006.
32. D. J. Rosenkrantz, R. E. Stearns, and M. Lewis Philip, II. An analysis of several heuristics for the Traveling Salesman Problem. *SIAM Journal on Computing*, 6:563–581, 1977.
33. P. Rousseeuw and G. Molenberghs. Transformation of nonpositive semidefinite correlation matrices. *Communications in Statistics: Theory and Methods*, 22:965–984, 1993.
34. R. Santana. *Advances in Probabilistic Grahical Models for Optimization and Learning. Applications in Protein Modeling*. PhD thesis, University of the Basque Country, 2006.
35. R. Santana, P. Larrañaga, and J. A. Lozano. Side chain placement using estimation of distribution algorithms. *Artificial Intelligence in Medicine*, 39:49–63, 2007.
36. A. Sklar. Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut de Statistique de l'Université de Paris*, 8:229–231, 1959.
37. M. Soto and Y. González-Fernández. Vine Estimation of Distribution Algorithms. Technical Report ICIMAF 2010-561, Institute of Cybernetics, Mathematics and Physics, May 2010. ISSN 0138-8916.
38. M. Soto, A. Ochoa, and R. J. Arderí. Estimation of distribution algorithm based on Gaussian copula. Technical Report ICIMAF 2007-406, Institute of Cybernetics, Mathematics and Physics, June 2007. ISSN 0138-8916.
39. R. Storn and K Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.
40. L.-F. Wang, Y. Wang, J.-C. Zeng, and Y. Hong. An Estimation of Distribution Algorithm based on Clayton copula and empirical margins. In K. Li, X. Li, S. Ma, and G. W. Irwin, editors, *Life System Modeling and Intelligent Computing*, pages 82–88. Springer-Verlag, 2010.
41. 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)*, 2009.
42. 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*, 49(20):5912–5931, 2006.