Radboud Repository

Radboud University Nijmegen

PDF hosted at the Radboud Repository of the Radboud University Nijmegen

The following full text is a publisher's version.

For additional information about this publication click this link.
http://hdl.handle.net/2066/28586

Please be advised that this information was generated on 2017-12-05 and may be subject to change.

Journal of Biomolecular NMR, 7 (1996) 214–224
ESCOM

J-Bio NMR 343

# Optimisation of metric matrix embedding by genetic algorithms

A.H.C. van Kampen[a], L.M.C. Buydens[a,*], C.B. Lucasius[b] and M.J.J. Blommers[c]

[a]*Laboratory for Analytical Chemistry, Catholic University of Nijmegen, 6525 ED Nijmegen, The Netherlands*
[b]*Department of Chemistry, Dalhousie University, Halifax, NS, Canada B3H 4J3*
[c]*Department of Physics, Ciba Geigy AG, P.O. Box, CH-4002 Basel, Switzerland*

Received 1 September 1995
Accepted 23 February 1996

*Keywords:* Distance geometry; Genetic algorithms

## Summary

To improve the convergence properties of 'embedding' distance geometry, a new approach was developed by combining the distance-geometry methodology with a genetic algorithm. This new approach is called DG-OMEGA (DGΩ, optimised metric matrix embedding by genetic algorithms). The genetic algorithm was used to combine well-defined parts of individual structures generated by the distance-geometry program, and to identify new lower and upper distance bounds within the original experimental restraints in order to restrict the sampling of the metrisation algorithm to promising regions of the conformational space. The algorithm was tested on cyclosporin A, which is notorious for its intrinsic difficult sampling properties. A set of 58 distance restraints was employed. It was shown that DGΩ resulted in an improvement of convergence behaviour as well as sampling properties with respect to the standard distance-geometry protocol.

## Introduction

The elucidation of biomolecular structures is the subject of lively research, as it is generally believed that such knowledge is an extremely important step towards the understanding of macromolecular mechanisms or biological function. Multidimensional NMR spectroscopy has become the state-of-the-art method for the structure determination of biological molecules in solution (for reviews see Wüthrich, 1986,1995; Clore, 1991; Wagner et al., 1992; Roberts, 1993).

In determining a structure of a biomolecule in solution, one has to follow a time-consuming procedure of resonance assignments (Wüthrich et al., 1982). Subsequently, on the basis of these assignments, a list of NOE (distance) restraints – sometimes complemented with information from coupling constants and/or chemical exchange – is used as input for a computer algorithm that converts the experimental information, together with knowledge about covalent bonds, into a three-dimensional structure. Such an algorithm is generally based on the concepts of 'distance geometry', but a variety of implementations have been developed. The structure found after applying a distance-geometry algorithm is often refined with the aid of molecular mechanics/dynamics and/or by a quantitative comparison of the refined structures with the experimental data.

In order to develop NMR towards a broadly and rapidly accessible tool for structure determination of proteins in solution, several computer programs have been designed to assist in resonance assignment (Kraulis, 1989; Van de Ven, 1990; Eccles et al., 1991; Kleywegt et al., 1991), restraint generation, distance geometry (Crippen, 1977; Güntert and Wüthrich, 1991; Havel, 1991) and structure refinement (Boelens et al., 1988,1989; Borgias and James, 1988). Although most of these computer programs still feature a strong interactive component, their development during the past years has contributed to faster and more reliable structure determination by NMR spectroscopy. This paper describes a new distance-geometry algorithm, which is aimed to contribute to the development mentioned above.

One family of distance-geometry programs comprises algorithms based on the embedding of a distance matrix

*To whom correspondence should be addressed.

0925-2738/$ 6.00 + 1.00 © 1996 ESCOM Science Publishers B.V.

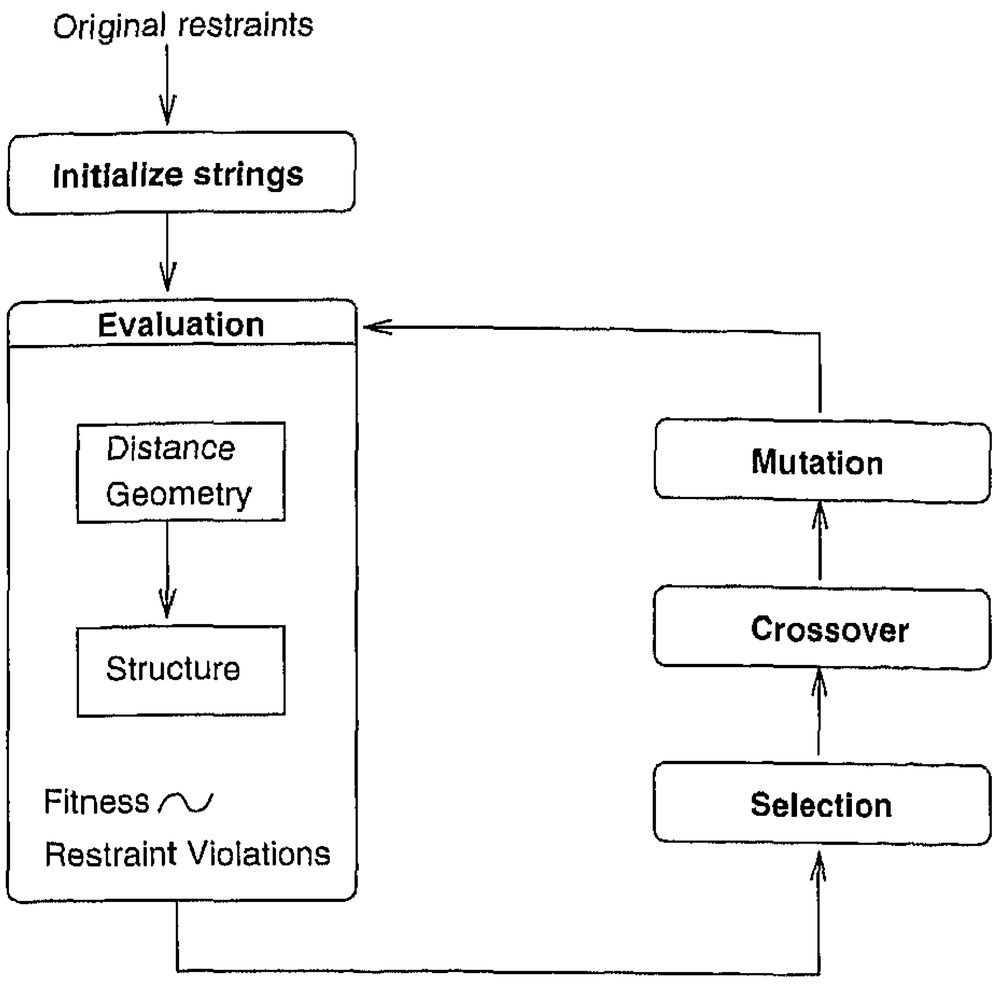



Fig. 1. Distance-geometry OMEGA (DGΩ). A genetic algorithm is combined with distance geometry (DGII). The trial solutions (strings) represent a specific set of restraints. The fitness assigned to each string reflects the number of restraint violations of the conformer calculated by DGII.

and subsequent optimisation of the thus obtained Cartesian coordinates by simulated annealing or distance-driven dynamics (Crippen, 1977; Havel, 1991). An alternative approach uses a so-called variable target algorithm for direct structure optimisation in torsion angle space (Güntert et al., 1991; Güntert and Wüthrich, 1991).

For some time, the distance-geometry methods have been criticised because they appeared to have poor sampling properties (Metzler et al., 1989; Havel, 1990; Kuszewski et al., 1992). Non-optimal sampling of the conformational space will lead to biased, imprecise and sometimes wrong structures, especially when the amount of experimental data is relatively poor. This criticism prompted additional improvements of existing strategies, which are often referred to as 'second-generation distance geometry'.

The second-generation distance-geometry programs obviously have improved sampling properties. However, especially in the absence of sufficient NOE data, it follows from the analysis of distance-geometry structures that the ensemble of structures may converge to the experimental data only to some extent. However, combining parts of these structures might result in additional convergence. In those cases, one is forced to apply the method for much more structures than is normally feasible. In addition, the selection of a set of good-quality structures is a time-consuming task, which requires careful interactive analysis of the data.

In an attempt to improve this aspect of distance geometry, the 'embedding' distance-geometry method was combined with a genetic algorithm. Briefly, the optimisation of an ensemble of distance-geometry structures is carried out in such a fashion that information is exchanged (by an operator called crossover) between the structures in the ensemble during optimisation. The method described in this paper is called DG-OMEGA, or DGΩ for short. OMEGA is an acronym for optimised metric matrix embedding by genetic algorithms. Results obtained in applying DGΩ to experimental data published for cyclosporin A in chloroform solution (Lautz et al., 1987) will be presented. These results are compared to those obtained by the DGII program for the same data set, which indicates that the combination of DGII and a genetic algorithm substantially improves the sampling and convergence properties of distance geometry.

## Methods

*Software and hardware*

DGΩ was developed on a 20 MHz personal IRIS computer (Silicon Graphics, TM) by combining parts of GATES (Genetic Algorithm Toolbox for Evolutionary Search, v. 1.00) (Lucasius and Kateman, 1994a,b) with the DGII program distributed by Biosym (Biosym Technologies, San Diego, CA, 1993). The communication between the genetic algorithm and the DGII package was accomplished via files generated by both the DGII and genetic algorithm programs. In addition, several UNIX shell scripts that are part of the DGII program were modified in order to be able to start DGΩ instead of DGII. The user interface of InsightII and NMRchitect (Biosym Technologies) was used to provide part of the input files needed by DGΩ. Information about the DGΩ software (genetic algorithm written in ANSI C, and the modified UNIX shell scripts) is available from the authors on request.

*Genetic algorithms*

Genetic algorithms (GAs) (Goldberg, 1989; Davis, 1991) comprise a set of optimisation methods especially suited in solving large and complex problems. They derive their name from the fact that they are loosely based on population genetics. GAs were pioneered by John Holland as a possible optimisation method (Holland, 1973, 1992), and ever since many investigations have been reported. More recently, the method raised interest as a tool in chemometric applications (Lucasius and Kateman, 1991; De Weijer et al., 1994), and as an energy minimisation method for molecular modelling and structure determination (Lucasius and Kateman, 1991; Lucasius et al., 1991; Blommers et al., 1992; Schulze-Kremer, 1992; Unger and Moult, 1993; Ring and Cohen, 1994; Sanderson et al., 1994; Ogata et al., 1995; Venkatasubramanian et al., 1995).

The GA maintains a population of strings, where each string represents a trial solution. Each string denotes a set of values for the problem parameters of the optimisation

problem, and is stored in the computer memory. After initialisation, the quality of each trial solution is evaluated. For this purpose an optimisation function, generally called a 'fitness function' in GA terminology, is designed; it assigns a fitness (quality) value to each string in the population.

After the fitness values have been assigned, strings are selected from the best fraction of the current population until an equally sized population results. By selecting above-average strings (strings with a fitness larger than the average fitness of the population), the GA uses information that it has built up in the population during the past iterations, i.e., the GA exploits information from the past.

In addition to the exploitation of previously gathered information, the GA also explores the search space by looking for new information (other solutions) in regions of the search space that were not visited before. In order to explore the search space, modifications are made to the previously selected strings. Two operators are used to that end. The first one is the crossover operator, which recombines two randomly selected strings, with a predefined probability (typically within the range 0.6–0.9). The second one is the mutation operator, which is applied to each string with a predefined probability (typically within the range 0.001–0.05).

If application of recombination and mutation results in improvement of a string, it will be assigned an increased fitness value in the next generation (an iteration in GA terminology), and accordingly may survive again in the selection process. By repeating this cycle, strings may improve every generation until convergence to an optimum results, whereupon the GA is terminated.

*DG-OMEGA (DGΩ)*

The structures resulting from a DGII calculation should normally converge with the experimental data. However, if the latter are incomplete and imprecise (as often is the case), the individual structures are usually of poor quality: they only partially match the structural properties of the true structure, i.e., the best possible solution to the problem. This may be ascribed to the fact that DGII does not systematically search for structures obeying all experimental data, but instead semi-randomly scans the conformational space, and therefore too many structures must be generated to include the true structure. A good solution to this shortcoming seems to be *combining* the good parts of the structures generated by DGII by using an evolutionary optimisation strategy. In order to achieve this, a GA was implemented, which is capable of effectively merging parts of solutions (DGII structures) in order to make the desired improvements. In this way, the search characteristics of DGII are enhanced from semi-randomly to a guided search for improved structures based on previously calculated structures.

The flowchart of DGΩ is presented in Fig. 1. The main idea behind this new approach is to 'optimise' the values of lower and upper bounds in such a way that after metrisation, embedding, and refinement, structures finally emerge that obey the original restraints to a larger extent than those generated with DGII. In DGΩ, a complete set of modified restraints is encoded on each string, which thereby represents a trial structure and replaces the original set of experimental restraints as input for the DGII algorithm. After the DGII calculation and the fitness assignments, the strings are recombined by the crossover operator (i.e., recombination of lower and upper bounds), and the restraints are adjusted by the mutation operator (i.e., the lower and upper bounds are tightened and centered about the corresponding distance calculated from the structure). As a result, the conformational space will shrink towards a region that includes (nearly) optimal structures. Therefore, this process will limit the sampling of the metrisation algorithm to very specific ranges located within the original bounds, and hopefully allows the structure to obey a larger fraction of the experimental input data. It is likely that, within the bounds of the original restraints, several ranges can be identified, which result in different (nearly) optimal structures.

Although is seems that self-consistency is illegally forced between the data (restraints) and the model (DGΩ), i.e., that the data are adjusted to fit the model, this is not the case. The new bounds are only generated to guide the sampling of the metrisation algorithm to promising regions of the conformational space, which were already included by the original restraints. In other words, the new set of bounds is a subset of the original bounds, and therefore does not include new information.

This principle of making modifications to a set of restraints based on a resulting structure can superficially be compared to the REDAC algorithm (Güntert and Wüthrich, 1991), where new bounds on the torsion angles are obtained after inspecting the torsion angle variation in an ensemble of structures.

*Encoding of the restraints*

The encoding of the restraints on the GA strings is shown in Fig. 2, where $L_i^*$ and $U_i^*$ indicate the modified bounds, and each parameter is encoded as a real value. The range assumed by each parameter is dictated by the values of the original restraints, denoted as $L_i$ and $U_i$.

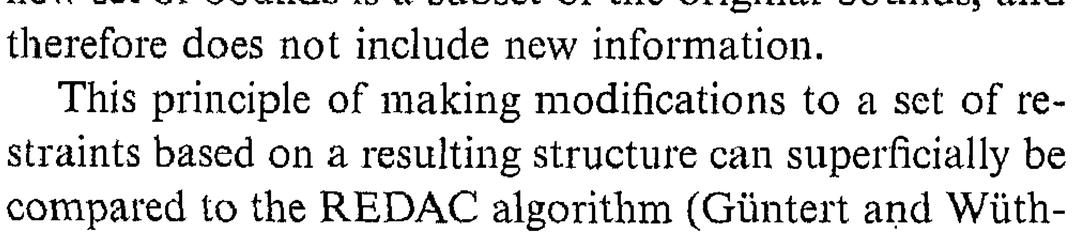



Fig. 2. Example of a trial solution (string). Each restraint is encoded on the string by using the real values of the lower and upper bounds. Each lower and upper bound is constrained by the corresponding original bounds.

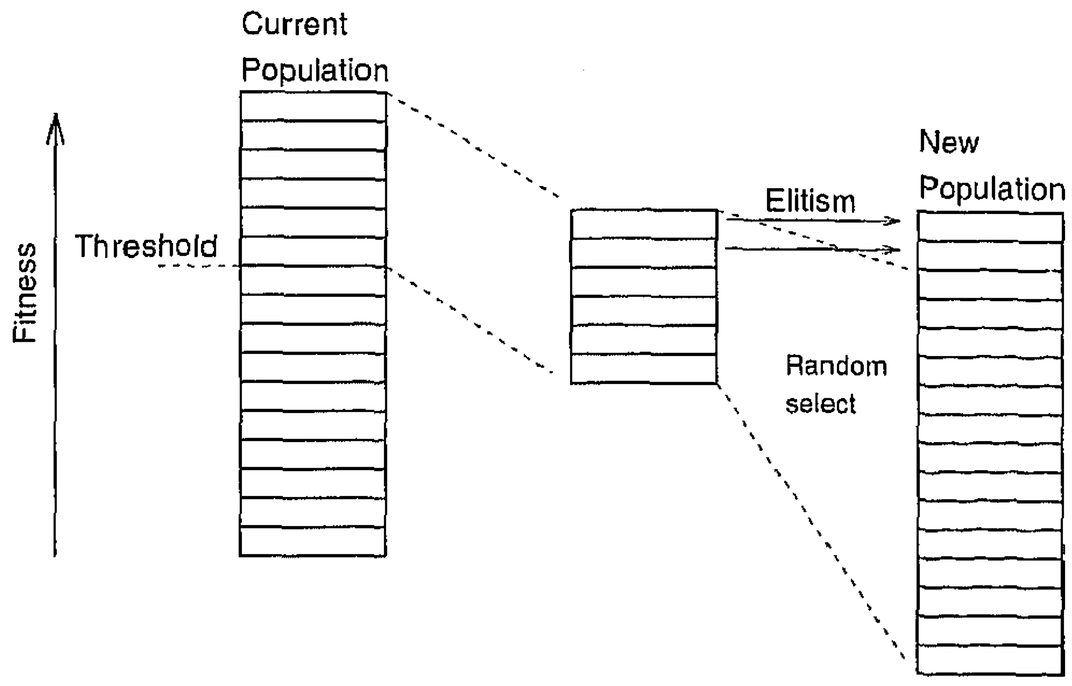



Fig. 3. The compound selection method used by the genetic algorithm in DGΩ. Rank-based threshold selection is combined with elitism for selection of above-average solutions.

Furthermore, $U_i^* \geq L_i^*$, which condition is imposed by the initialisation procedure and mutation operator. This condition is not imposed by the crossover operator, which does operate on the individual values, and therefore the string is repaired (i.e., the lower and upper bounds are switched) whenever this constraint is violated.

*Initialisation of the restraints*

The initialisation of each lower and upper bound on the string cannot be accomplished at random because this will very likely give rise to inconsistent bounds, i.e., to violations of the triangle inequality. Therefore, a procedure was developed which initialises the bounds in such a way that they will represent only small modifications from the original restraints, i.e., each parameter is initialised by:

$$L_i^* = L_i + |N(0,1)| \times I_{dev}$$
$$U_i^* = U_i - |N(0,1)| \times I_{dev} \qquad (1)$$

where N(0,1) denotes the standard normal distribution with zero mean and unit standard deviation. The initialisation parameter $I_{dev}$ is used to control the deviation from the original restraints. A small $I_{dev}$ will ensure only small changes from the original restraints; a potential drawback of this approach resides in the fact that the search space is not spanned optimally, but this turned out not to be a problem in practice. During the initialisation, it is checked whether $U_i^* \geq L_i^*$. If that is not the case, the bounds are switched; otherwise no further action is undertaken.

*The fitness function*

The evaluation of the strings (trial solutions) requires a complete DGII calculation, followed by a fitness assignment of the strings. Note that in contrast to DGII, where an ensemble of distance matrices is generated from one bound matrix, DGΩ first generates a bound matrix from every string. Subsequently, one distance matrix is generated from each bound matrix, which again results in an ensemble of distance matrices. Once the structures are obtained, the assignment of the fitness is straightforward:

$$[\text{fitness}_i]^{-1} = \text{error}_i = \sum_{j=1}^{N} (\text{restraint violation}_j)^2 \qquad (2)$$

where N is the number of restraint violations and i denotes the index of the string. To calculate the restraint violations, the corresponding distances $d_j$ are calculated from the coordinates of the structure:

$$\text{restraint violation}_j = \begin{cases} L_j - d_j, \text{ when } d_j < L_j \\ d_j - U_j, \text{ when } d_j > U_j \end{cases} \qquad (3)$$

Given the fact that DGII is part of the evaluation function, the GA may be regarded as a meta-optimisation method: the simulated annealing (SA) procedure within DGII optimises the embedded structures by minimising the violations of the covalent constraints, experimental restraints, and chirality constraints, whereas the GA optimises the resulting structures by just minimising the restraint violations. Leaving SA out of the DGII calculation, and instead minimising all restraints and constraints by adding the corresponding error terms to the fitness function of the GA, would severely degrade the performance of DGΩ because the structures after embedding would then be too distorted to be assigned a meaningful fitness value.

It is also important to realise that the evaluation function is noisy. This noise is the result of the stochastic effects implied by the DGII algorithm: the metrisation algorithm uses random permutations of the distances to provide for a good sampling of the distance space (Biosym Technologies, 1993); the embedding algorithm uses Tchebychev polynomials starting from a random vector to accelerate convergence (Biosym Technologies, 1993); and finally, simulated annealing is a stochastic optimisation method (Kirkpatrick et al., 1983). As a result of the noise, each string may evaluate to a range of structures of which the fitness values may (strongly) overlap, depending on the magnitude of the noise. If this effect is large compared to the improvements made by the crossover and mutation operator, the selection process of the GA may be severely hindered. However, the results presented in this paper suggest that this is not the case, although the noise can clearly be observed in the error curves.

*Selection of strings*

In DGΩ a rank-based threshold selection (Lucasius and Kateman, 1994c) is used, which is based on the rank of the strings according to their fitness (Fig. 3). A threshold is chosen defining the better fraction of the population and, subsequently, only strings from this fraction are selected at random to be part of the new population. In

Fig. 4. The uniform crossover operator for real encoded parameters. Each block denotes a specific parameter. In DGΩ, each parameter represents a lower or upper bound of a distance restraint.

order to increase the GA performance even further, the rank-based method was combined with elitism selection, which provides for the best strings always to be selected. An elitism fraction is chosen that defines the number of best strings that are copied to the new population.

*Crossover*

In DGΩ the so-called uniform crossover was applied, which selects a predefined number of parameters (real parameter encoding) at random, and exchanges these with the corresponding parameters on the paired string. From this operation, shown in Fig. 4, two new strings result.

*The mutation operator*

A new mutation operator was designed, which, on average, centers and tightens the bounds, subject to confinement to the original range. For each specific string (structure), application of this mutation operator implies the following ($L_i$, $U_i$ define the original restraints; $L_i^*$, $U_i^*$ define the modified restraints):

(i) Calculate the distance $d_i$ corresponding to restraint i ($L_i^*$,$U_i^*$)

(ii) With probability $P_{center}$, center restraint i about $d_i$:

$$L_i^{new} = L_i^* - [(L_i^* + (U_i^* - L_i^*)/2) - d_i]$$
$$U_i^{new} = U_i^* - [(L_i^* + (U_i^* - L_i^*)/2) - d_i] \quad (4)$$

Note that both the lower and upper bound are shifted in the same direction, which is the reason that all signs in these two formulas are identical;

(iii) Check modified restraints:

$$\text{If } (L_i^{new} > U_i) \text{ or } (U_i^{new} < L_i), \text{ then } L_i^{new} = L_i^*,\ U_i^{new} = U_i^*$$
$$\text{If } (L_i^{new} < L_i), \text{ then } L_i^{new} = L_i^* \quad (5)$$
$$\text{If } (U_i^{new} > U_i), \text{ then } U_i^{new} = U_i^*$$

(iv) Generate stochast $x = |N(0,1)| \times D$, where D is used to control the magnitude of the tightening, and then, with probability $P_{tighten}$, tighten restraint i as follows:

$$L_i^{new} = L_i^{new} + x$$
$$U_i^{new} = U_i^{new} - x \quad (6a)$$

else (expand restraint i):

$$L_i^{new} = L_i^{new} - x$$
$$U_i^{new} = U_i^{new} + x \quad (6b)$$

(v) Check the modified restraints described in step iii.

This mutation is applied to a random subset S of the restraints. By adjusting $P_{center}$, $P_{tighten}$, D and the size of subset S, it is possible to control the performance of this operator to some extent. It is important to note that the effect of the mutation depends on the quality of the structure generated, as distances from this structure are used to define new restraints. Ill-defined structures might deceive this operator, i.e., the bounds may converge to non-optimal values.

*Configuration of DGΩ*

For the experiments described in this paper a population of 75 strings was used. Each lower and upper bound

TABLE I
CONFIGURATION OF DGII

| | | |
|---|---|---|
| Smooth | Triangle smoothing | On |
| | Triangle violation tolerance | 20.0 |
| | Tetrangle strategy | None |
| Embed | Uniform probability density | On |
| | Probability coefficient | 0.5 |
| | Eigenvalue iterations | 100 |
| | Eigenvalue criterion | 0.001 |
| | Metrisation | Prospective |
| | Embed dimension | 4 |
| Majorise | Guttman transform | 10 |
| | Linear conjugate gradient iterations | 100 |
| | Linear conjugate gradient criterion | 0.001 |
| | Scale centroid | Off |
| | Calculate Moore–Penrose inverse | On |
| | Moore–Penrose inversion criterion | 0.001 |
| | Weighting scheme | Constant |
| | Overwrite structures | On |
| Optimise | Dimension weight | 0.20 |
| | Chirality weight | 0.1 |
| | Lower maximum | 10.0 |
| | Contact maximum | 1.00 |
| | Dimension scaling | 0.30 |
| | Upper weight limit | 1.00 |
| | Error function form | Sparse matrix |
| | Extra radii | 1.00 |
| Simulated annealing | Initial temperature | 1.00 |
| | Maximum heating | 2.00 |
| | Maximum number of steps | See Results |
| | Calculate initial energy | Off |
| | Initial energy | 1000.0 |
| | Maximum temperature | 200.0 |
| | Fail level | 1.00 |
| | Atom mass | 1000 |
| | Step size | 2e–13 |
| Conjugate gradient | Maximum iterations | 250 |
| | rms gradient | 0.001 |
| Global setup | Generate database | On |
| | Number of structures | 75 (Population size) |
| | Omega wobble | 10 |
| | Increment files | On |

The table lists the configuration of all parameters used within DGII. For a detailed explanation of these variables see the DGII User Guide (Biosym Technologies, 1993).

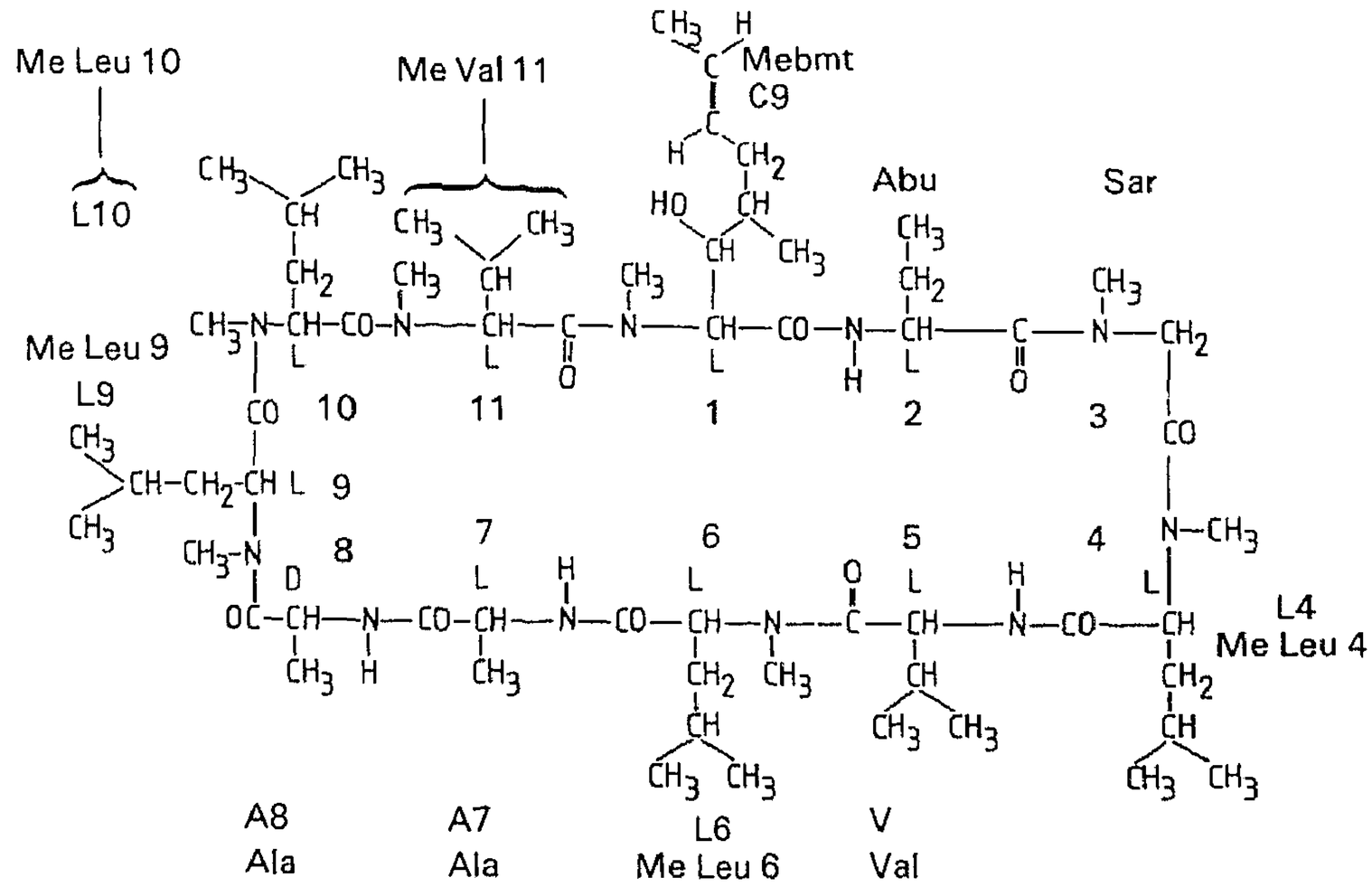



Fig. 5. Chemical structure of cyclosporin A.

was encoded with a precision of 0.001 Å. By considering the ranges that are assumed by the 58 restraints of cyclosporin A, together with the given precision, it can easily be derived that the size of the search space comprises approximatly $10^{34}$ states. The initialisation parameter $I_{dev}$ was set to 0.05. Threshold rank-based selection was used with a threshold fraction of 0.25 (19 strings) and an elitism fraction of 0.02 (2 strings). Uniform crossover was used, applied with a probability of 0.80. The number of parameters to swap was set to 2. The mutation operator was applied to all restraints encoded on the string ($S = 58$). Furthermore, $P_{shift} = 1.0$, $P_{center} = 1.0$, and $D = 0.05$.

The values for all parameters of DGII are listed in Table 1. For a more detailed explanation of these variables, one has to consult the user guide of DGII (Biosym Technologies, 1993). Note that the number of structures in the global setup determines the population size of the GA. The number of SA iterations depends on the experiment and therefore is given in the Results and Discussion.

## Results and Discussion

The performance of DGΩ was compared with that of DGII, using cyclosporin A (CPA) (Lautz et al., 1987, 1989; Kessler et al., 1990) as the target molecule for structure elucidation. The aim of this research was to present the principles underlying DGΩ and to demonstrate that the approach can be used to generate improved structures. The objective was not to reveal the structure of CPA, as the details thereof are already known from other publications.

CPA, an important drug applied during treatment subsequent to organ transplantation because of its unique immunosuppressive properties, is a cyclic undecapeptide, cyclo-(MeBmt[1]-Abu[2]-Sar[3]-MeLeu[4]-Val[5]-MeLeu[6]-Ala[7]-D-Ala[8]-MeLeu[9]-MeLeu[10]-MeVal[11]) (see Fig. 5), with 49 dihedral angles. An X-ray structure is known (Loosli et al., 1985) and, in addition, a structure in apolar solution has been derived on the basis of NMR data (Kessler et al., 1985) by applying static modelling techniques (Lautz et al., 1985). Whereas the cyclic peptide adopts many conformations in equilibrium in polar solvents such as DMSO, no major conformational heterogeneity is observed in chloroform. Therefore, the data set involving CPA measured in chloroform represents an ideal test case.

For the present experiments, a set of 58 distance restraints from Lautz et al. (1987) was used. Because of its inherently difficult sampling properties, this data set has been used in the past to validate new structure optimisation algorithms (Lautz et al., 1987; Schaik et al., 1992).

Using this data set, distance-geometry calculations were performed using the DGΩ algorithm. The DGII algorithm was applied in similar experiments for comparison purposes. The experiments are summarised in Table 2. Although the DGII calculations were carried out with the original set of restraints and not with a set of tighter restraints (e.g., generated by DGΩ), the comparison between these two algorithms can be considered to be fair. From practice it appeared that the sampling of the conformational space is better when the bounds are loose (especially in the SA protocol). Consequently, using tighter bounds for DGII would likely reduce the quality of the resulting structures.

The number of steps of simulated annealing may critically affect the quality of the structures. Therefore, it seems important to investigate to what extent the length of the SA refinement can be reduced in the DGΩ approach; such information can be obtained from the reference experiments involving DGII.

TABLE 2
COMPARISON OF THE NUMBER OF FUNCTION EVALUATIONS FOR DGII AND DGΩ

| Exp. | Algorithm | No. of structures | No. of generations | SA | No. of evaluations |
|---|---|---|---|---|---|
| 1 | DGII | 50 | – | 1000 | 50 000 (75 000) |
| 2 | DGII | 50 | – | 5000 | 250 000 (375 000) |
| 3 | DGΩ | 75 | 18 | 1000 | 1 350 000 |
| 4 | DGΩ | 75 | 13 | 5000 | 4 875 000 |

The number of function evaluations calculated from the size of the structure ensemble, the number of generations (for DGΩ) and the number of steps of simulated annealing (SA) are given. The values within brackets indicate the number of evaluations if 75 structures would be calculated with DGII.

DGΩ calculations were performed with a population size of 75 structures and were set up for either 1000 or 5000 steps of SA. The reference experiments using DGII were performed with 1000, 5000 and 10 000 steps of SA. It appeared that the structures are converged within 5000 steps of SA, and therefore only the results for the first two experiments are depicted in Table 2. When one compares the optimisation by SA (DGII) with that by GA and SA (DGΩ), it would be fair to specify the total number of iterations, i.e., the number of times that the error function of SA is evaluated. This is calculated by the product of steps SA, the number of structures, and the number of generations (where for DGII the number of generations is set to 1). The values given between brackets (Table 2) represent the number of function evaluations if 75 structures would be calculated with DGII, which allows for direct comparison with DGΩ. Note that the number of generations listed in Table 2 corresponds with the point after which no more improvement was observed. The CPU times spent in assembling the input files for DGII from the strings, calculating the fitness values, and application of the genetic operators were not being considered. Consequently, the comparison of the number of function evaluations was not based on CPU times. However, the CPU time involved for these steps was negligible compared to a complete DGII calculation, i.e., an evaluation of the strings. Table 2 shows that the number of evaluations required by DGΩ is larger than for DGII, which is of course due to the fact that in DGΩ the DGII algorithm is iterated by the GA.

From each ensemble of structures resulting from one of the four experiments the minimum, maximum, and the average number of violations were calculated (Table 3); it is obvious from this that DGΩ performs better than DGII.

In addition, for each individual structure in the ensemble the average magnitude of the restraint violations was calculated. From this, the structure with the minimum and maximum average restraint violation was determined; these values, together with the corresponding number of violations, are also shown in Table 3. These quantities allow the calculation of the sum of violations via multiplication. Upon comparing the minimum and maximum average violations it is again clear that DGΩ performs better than DGII. Because the sum of violations for the tabulated structures significantly decreased, it seems fair to conclude that application of DGΩ results in a better convergence compared to DGII. From the average magnitude of violations of all structures, an overall average and standard deviation have been calculated. Comparison of these values reveals a slightly better performance for DGΩ.

Figure 6 illustrates the distributions of the average restraint violations and the number of violations for the ensemble of each experiment. Upon comparing the distribution reflecting the average violations, it is clear that increasing the number of SA iterations decreases the deviation of the distribution and shifts the distribution towards smaller restraint violations, i.e., the structures were found to converge to a larger extent. However, when comparing the differences between DGΩ and DGII, no pronounced effects are observed, although, as already pointed out, the distribution for DGΩ includes structures with decreased average violations. The distributions in-

TABLE 3
COMPARISON OF NUMBER AND MAGNITUDE OF RESTRAINT VIOLATIONS FOR DGII AND DGΩ

| Exp. | Algorithm | No. of violations | | | Magnitude of violations | | | |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Avg | Min (#) | Max (#) | Avg | Std |
| 1 | DGII | 8 | 23 | 16 | 0.077 (12) | 0.275 (8) | 0.176 | 0.041 |
| 2 | DGII | 8 | 21 | 14 | 0.069 (19) | 0.143 (15) | 0.0997 | 0.0188 |
| 3 | DGΩ | 2 | 14 | 9 | 0.046 (2) | 0.230 (5) | 0.1420 | 0.0413 |
| 4 | DGΩ | 3 | 15 | 7 | 0.02 (3) | 0.2140 (3) | 0.0877 | 0.031 |

For each ensemble, the minimum (Min) and maximum (Max) number of violations, and minimum (Min (#)) and maximum (Max (#)) average violations are depicted. For the latter the corresponding number of violations are given in brackets. The average number of violations, average violation (Avg), and standard deviation (Std) denote statistics over the complete ensemble.

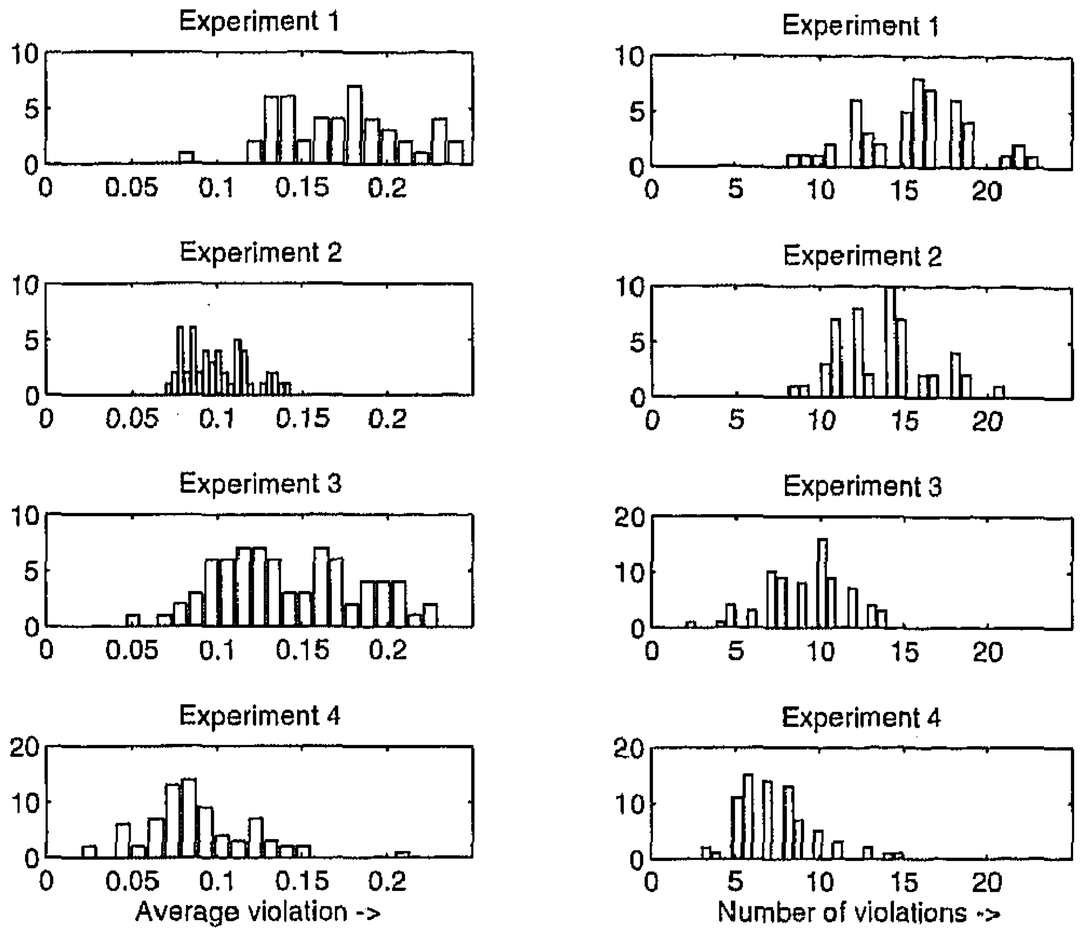



Fig. 6. The distributions of the 'magnitude of violations' and the 'number of violations'. Experiments 1 and 2 comprise the DGII calculations for 50 structures. Experiments 3 and 4 comprise the DGΩ calculations with 75 structures. A decrease in the number of violations can be clearly observed for DGΩ.

volving the number of violations clearly disclose that DGΩ generates structures with a smaller number of restraint violations than DGII, i.e, there was a clear shift to conformers with less violations in comparison to DGII.

Figure 7 illustrates a superposition of seven structures. The average rms deviation of the backbone atoms is 2.2 Å. These structures represent the seven best structures of the ensemble. Each structure has only one, two or three violations of 0.1 to 0.3 Å and the sum of violations is less than 0.6 Å. From a similar selection of the 'best' DGII structures the average rms deviation of the backbone is 1.3 Å. This clearly indicates that apart from the convergence properties, the sampling properties of DGΩ are superior to those of DGII. There is much more variation in structures satisfying the applied restraints to the same extent. In both cases the structures had an average backbone rmsd of 1.5 Å to the previously published structure, which was obtained with restrained molecular dynamics. This indicates that in both cases the resulting family of structures fluctuates about the energy-refined structure.

Figure 8 depicts the error curves for experiment 3 (these are similar to the error curves for experiment 4). They clearly reflect the noise caused by the evaluation function, i.e., despite the use of elitism selection the error of the best string in each generation occasionally increases. Interestingly, the graphs shown in Fig. 8 reveal that the optimisation can be characterised by a very steep optimisation profile during the first 10 generations. Then, within say five generations, there is still significant improvement, but thereafter the error curve fluctuates about the optimal value. These results suggest that, if CPU time is a critical factor, the use of only few generations (i.e., a limited application of the GA) already adds to the convergence of the structures.

Figure 9 shows the evolution for an arbitrary selection of four restraints for 25 generations. This is a clear illustration of the shrinking properties of the mutation operator (see Methods). The values that are plotted correspond to the upper and lower bounds of the best structure generated so far. After initialisation (generation 0), these bounds are close to their original (experimental) value. In the next few generations, the bounds rapidly converge to the same value. After convergence, the lower and upper bounds become about equal, and consequently, the mutation operator can only continue by centering the restraints. Furthermore, at this stage the similarity between the strings was found to increase to such a level

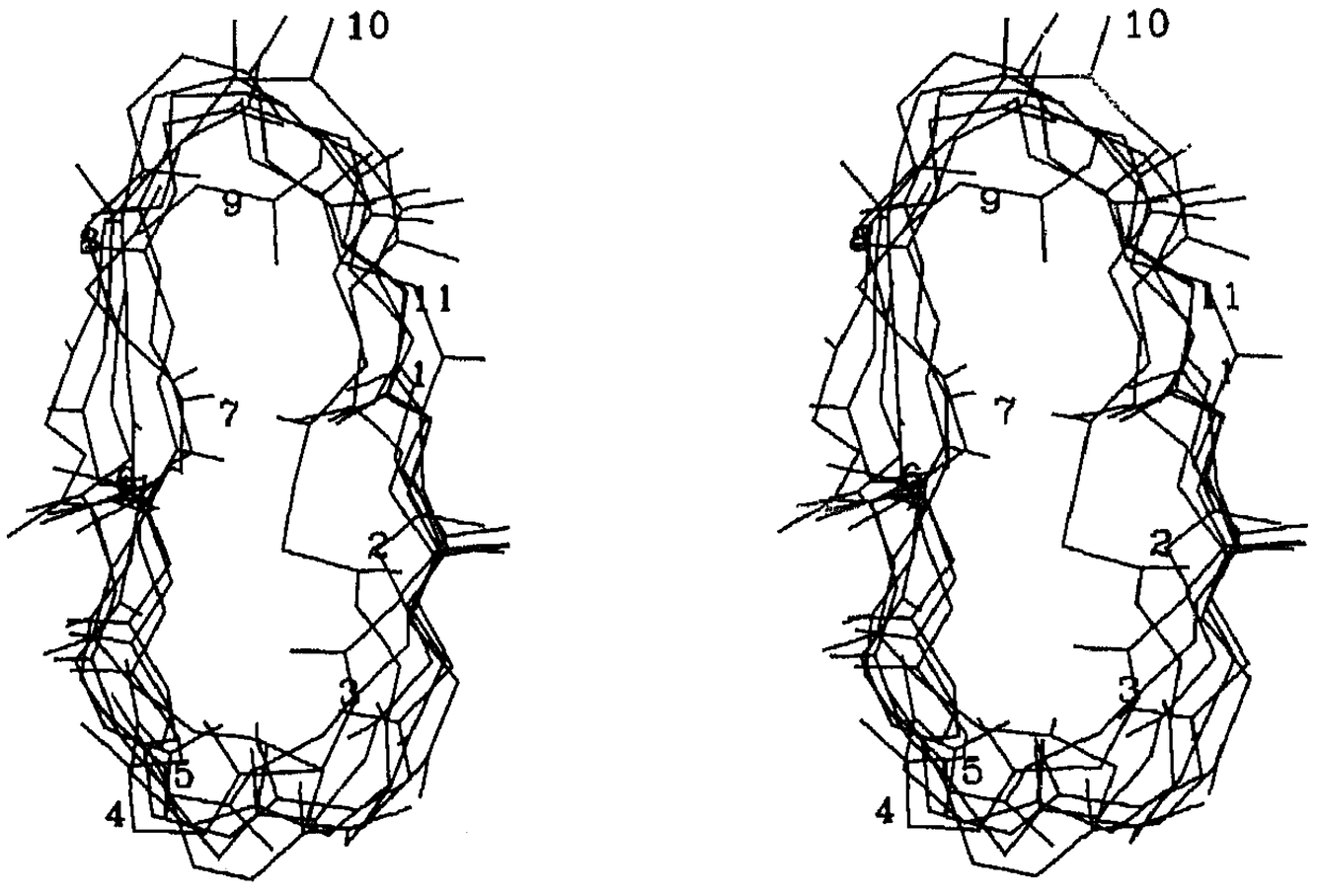
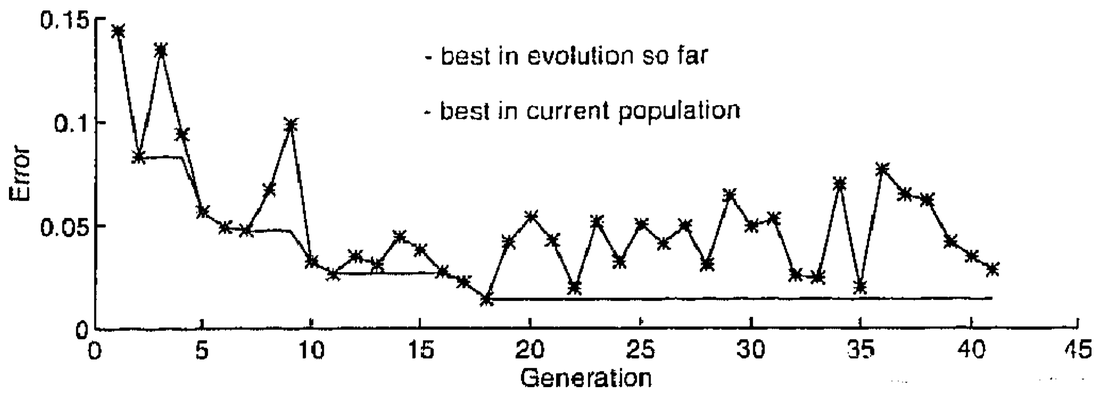



Fig. 7. Selection of the seven best structures generated by DGΩ. Each structure has a maximum of three violations, which amount to less than about 0.3 Å.



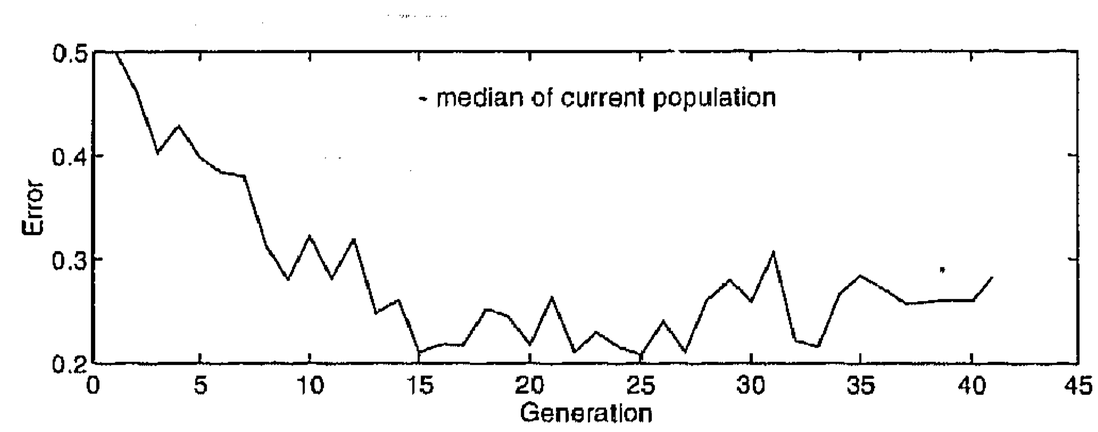




Fig. 8. The performance of the GA is reflected by the evolution of the error values. The curves reflect the median error of the population at each generation, the lowest error found so far and the smallest error of the population at each generation. The noise on the error values can be clearly observed for the smallest error of the current population.

that the effect of crossover largely declined. Accordingly, the fluctuations after convergence may be ascribed mainly to the stochastic effects in DGII, i.e., the best strings evaluate to a different structure and as a result the restraints are recentered. The effect of this on the error was also observed in Fig. 8.

As shown above, the number of function evaluations required by DGΩ to derive the final set of structures was much larger than for DGII. DGΩ needed up to 18 generations, which is comparable to 18 DGII calculations. To make the comparison between DGII and DGΩ more fair, a DGII calculation was performed, which generated 1350 structures (18 generations × 75 structures). This experiment showed a similar performance of DGII compared to the calculation with 75 structures, i.e., no structure of comparable quality of DGΩ was found.

An important shortcoming of DGII resides in the fact that the optimisation strategy used, i.e., simulated annealing, is trajectory based. In other words, the optimisation is started from one conformation and is proceeded by progressive changes towards a conformation that fits better to the experimental data. The GA, on the other hand, seems superior in that it inherently combines partial solutions (substructures) by means of the crossover operator. To investigate the effect of the crossover operator and the use of a population-based search strategy, a DGΩ experiment was performed in which the size of the population was reduced to one. Consequently, no crossover and selection could be applied, and any outcome should thus be caused by the mutation operator alone, i.e., through adjustment of the restraints. The results obtained with this experiment were very poor, because very distorted structures resulted and no improvement of the error values (restraint violations) during subsequent generations was observed. This indicates that both crossover and the use of a population of trial solutions in combination with a

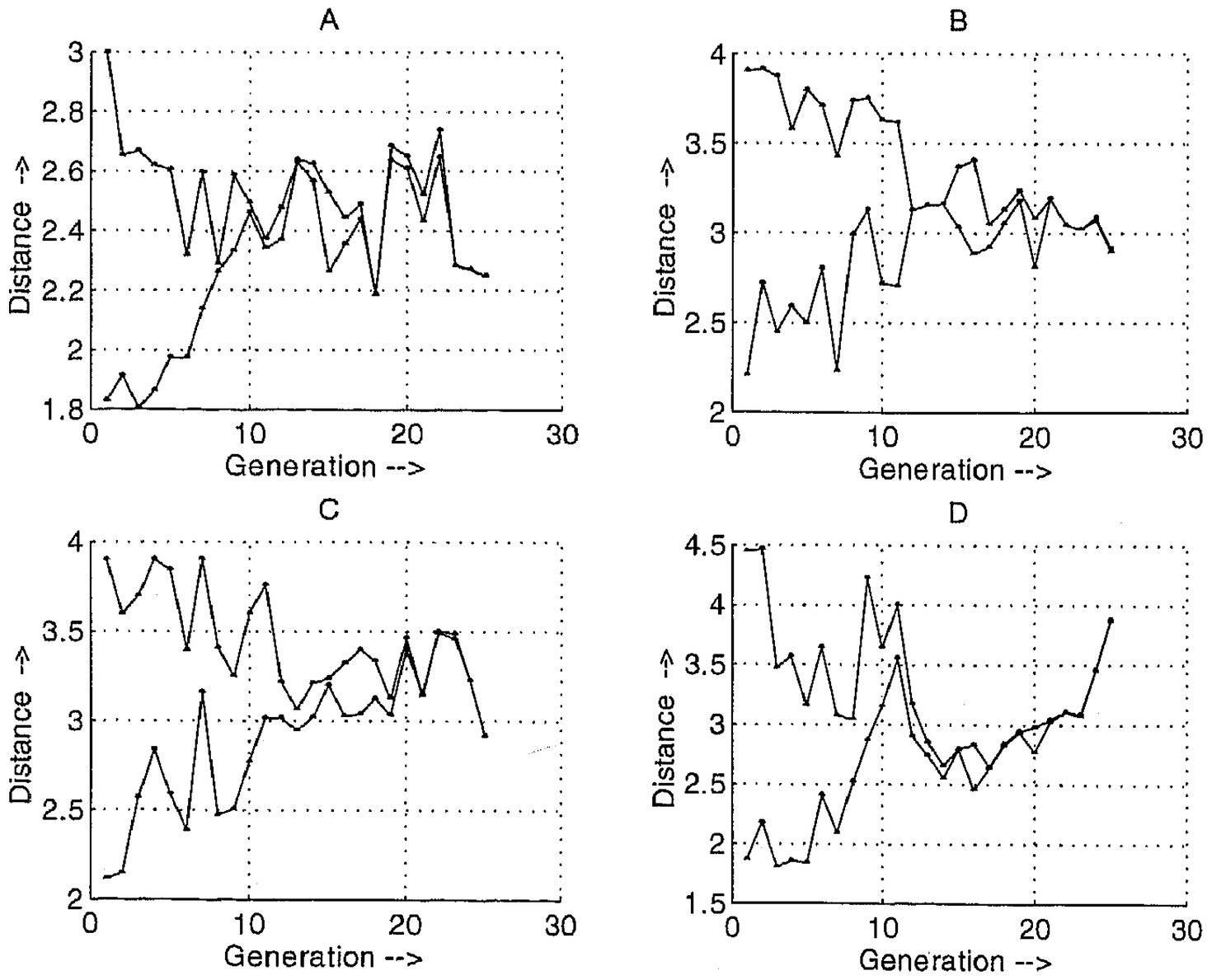



Fig. 9. Evolution of a selection of four restraints for 25 generations. (A) THRn_1:$H^{\alpha}$-ALA_2:HN; (B) LEU_6:$H^{\beta}_1$-LEU_6:CN; (C) VAL_5:$H^{\gamma}_2$-VAL_5:$H^{\alpha}$; (D) VAL_5:$H^{\alpha}$-VAL_5:$H^{\gamma}_1$). The lower and upper bound (lower and upper line, respectively) rapidly converge to an identical value. This may be ascribed mainly to the mutation operator.

selection method contributes significantly to the performance of DGΩ, which justifies the use of the GA methodology.

Although the use of GA is supported by the previous experiment, it may seem possible to use the concepts of DGΩ in a more effective way by only using SA, i.e., without re-embedding of modified bounds. However, after adjustment of the restraints, the present set of structures might not represent an adequate starting set for the next SA iterations. Furthermore, the changes made by SA might be too local to be of value for the DGΩ principles. These problems will, almost certainly, increase the number of SA iterations needed to derive equally good structures as in the current implementation.

Apart from the improvement of convergence and sampling properties, the GA offers the possibility to further develop the quality and efficiency of the structure-determination process. By means of sharing and crowding (Goldberg et al., 1987; Goldberg, 1989) or the use of multiple populations (Stender, 1993), it becomes possible to search more effectively for multiple solutions (structures) with an increased rms value. Another improvement can be obtained by designing an interactive crossover operator. In that case the user would a priori define a most likely 'bad' part of the structure, e.g., a part with many restraint violations. Subsequently, crossover could be applied only to the corresponding bounds, hopefully resulting in better substructures. As a result, the complete protein might converge to a better solution. Another development that makes application of the GA attractive, is direct coupling of structure-quality criteria such as calculated energy or covalent restraint violations, to the fitness function, which should optimise the quality of the conformational pool. The selection process may be further enhanced by using information obtained from the relation between the maximum pairwaise rmsd and the maximum restraint violation error (Widmer et al., 1993). In this way, the sampling that is perceived by DGΩ may be controlled to some extent. In order to increase the convergence rate of DGΩ, the fitness function can be extended with an additional term reflecting the quality of the distance matrix. This term could be determined by calculating the difference between the distance matrices before and after embedding, and the difference between the matrices before and after SA. The former would reflect the extent to which the matrix is embeddable, whereas the second difference would reflect the quality of the structure directly after embedding with respect to the restraints and covalent geometry. This improvement of the fitness function might lead to a reduction of the number of SA iterations. These improvements are currently under investigation.

As is shown in this paper, DGΩ amounts to an improvement over DGII, even without optimisation of the algorithm's configuration and therefore optimisation within this method (especially of the mutation parameters) will very likely further enhance performance.

The integration with the DGII package of Biosym software makes DGΩ an easy to use algorithm. Part of the input files can be set up by using the excellent user interface of InsightII. Additional input files can be obtained with the aid of a text editor.

## Conclusions

A new distance-geometry approach, DGΩ, was presented, which is based on a combination of 'embedding' distance geometry and a genetic algorithm. Application of DGΩ to CPA demonstrated an enhancement of both the convergence and sampling properties with respect to the standard distance-geometry protocol. DGΩ is open for many modifications and extensions that may further improve the sampling and convergence properties, reduce the CPU time required for doing a calculation, or enlarge its applicability.

## Acknowledgements

We would like to thank F.M. van der Kloet for writing parts of the software needed to interface the Biosym software with GATES. Furthermore, we want to acknowledge Dr. B. Rohde (Ciba Geigy AG, Basel, Switzerland) for several valuable discussions and contributions to this project. T.F. Havel (Harvard Medical School, Boston, MA) is acknowledged for providing us with information about the file formats used in the DGII package. Finally, Biosym is acknowledged for software support.

## References

Blommers, M.J.J., Lucasius, C.B., Kateman, G. and Kaptein, R. (1992) *Biopolymers*, **32**, 45–52.

Boelens, R., Koning, T.M.G. and Kaptein, R. (1988) *J. Mol. Struct.*, **173**, 299–311.

Boelens, R., Koning, T.M.G., Van der Marel, G.A., Van Boom, J.H. and Kaptein, R. (1989) *J. Magn. Reson.*, **82**, 290–308.

Borgias, B.A. and James, T.L.Z. (1988) *J. Magn. Reson.*, **79**, 493–512.

Clore, G.M. and Gronenborn, A.M. (1991) *Science*, **252**, 1390–1399.

Crippen, G.M. (1977) *J. Comput. Phys.*, **24**, 96–107.

Davis, L. (Ed.) (1991) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, NY.

Eccles, C., Güntert, P., Billeter, M. and Wüthrich, K. (1991) *J. Biomol. NMR*, **1**, 111–130.

Goldberg, D.E. and Richardson, J. (1987) In *Proceedings of the Second International Conference on Genetic Algorithms* (Ed., Grefenstette, I.I.), Lawrence Erlbaum, New York, NY, pp. 41–49.

Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Güntert, P., Braun, W. and Wüthrich, K. (1991) *J. Mol. Biol.*, **217**, 1–14.

Güntert, P. and Wüthrich, K. (1991) *J. Biomol. NMR*, **1**, 447–456.

Havel, T.F. (1990) *Biopolymers*, **29**, 1565–1585.

Havel, T.F. (1991) *Prog. Biophys. Mol. Biol.*, **56**, 43–78.

Holland, J.H. (1973) *SIAM J. Computing*, **2**, 88–105.
Holland, J.H. (1992) *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA.
Kessler, H., Loosli, H.R. and Oschkinat, H. (1985) *Helv. Chim. Acta*, **60**, 661–681.
Kessler, H., Kock, M., Wein, T. and Gehrke, M. (1990) *Helv. Chim. Acta*, **73**, 1818–1832.
Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983) *Science*, **220**, 671–680.
Kleywegt, G.J., Boelens, R., Cox, M., Llinas, M. and Kaptein, R. (1991) *J. Biomol. NMR*, **1**, 23–47.
Kraulis, P. (1989) *J. Magn. Reson.*, **84**, 627–633.
Kuszewski, J., Nilges, M. and Brünger, A.T. (1992) *J. Biomol. NMR*, **2**, 33–56.
Lautz, J., Kessler, H., Kaptein, R. and Van Gunsteren, W.F. (1987) *J. Comput.-Aided Mol. Design*, **1**, 219–241.
Lautz, J., Kessler, H., Blaney, J.M., Scheek, R.M. and Van Gunsteren, W.F. (1989) *Int. J. Pept. Protein Res.*, **33**, 281–288.
Loosli, H.R., Kessler, H., Oschkinat, H., Weber, H.P., Petcher, T.J. and Widmer, A. (1985) *Helv. Chim. Acta*, **60**, 682–704.
Lucasius, C.B., Blommers, M.J.J., Buydens, L.M.C. and Kateman, G. (1991) In *Handbook of Genetic Algorithms* (Ed., Davis, L.), Van Nostrand Reinhold, New York, NY, pp. 251–281.
Lucasius, C.B. and Kateman, G. (1991) *Trends Anal. Chem.*, **10**, 254–261.
Lucasius, C.B. and Kateman, G. (1994a) *Comput. Chem.*, **18**, 127–136.
Lucasius, C.B. and Kateman, G. (1994b) *Comput. Chem.*, **18**, 137–156.
Lucasius, C.B. and Kateman, G. (1994c) *Chemometrics Intelligent Lab. Syst.*, **25**, 99–146.
Metzler, W.J., Hare, D.R. and Pardi, A. (1989) *Biochemistry*, **28**, 7045–7052.
*NMRchitect User Guide*, 2.2nd edition, Biosym Technologies, San Diego, CA, 1993.
Ogata, H., Akiyama, Y. and Kanehisa, M. (1995) *Nucleic Acids Res.*, **23**, 419–426.
Ring, C.S. and Cohen, F.E. (1994) *Isr. J. Chem.*, **34**, 245–252.
Roberts, G.C.K. (Ed.) (1993) *NMR of Macromolecules, A Practical Approach*, Oxford University Press, Oxford, U.K.
Sanderson, P.N., Glen, R.C., Payne, A.W.R., Hudson, B.D., Heide, C., Tranter, G.E., Doyle, P.M. and Harris, C.J. (1994) *Int. J. Pept. Protein Res.*, **43**, 588–596.
Schulze-Kremer, S. (1992) In *Parallel Problem Solving from Nature*, Vol. 2 (Eds, Manner, R. and Manderick, B.), North-Holland, Amsterdam, The Netherlands, pp. 391–400.
Stender, J. (Ed.) (1993) *Parallel Genetic Algorithms: Theory and Applications*, IOS Press, Amsterdam, The Netherlands.
Unger, R. and Moult, J. (1993) *J. Mol. Biol.*, **231**, 75–81.
Van de Ven, F.J.M. (1990) *J. Magn. Reson.*, **86**, 633–644.
Van Schaik, R.C., Van Gunsteren, W.F. and Berendsen, H.J.C. (1992) *J. Comput.-Aided Mol. Design*, **6**, 97–112.
Venkatasubramanian, V., Chan, K. and Caruthers, J.M. (1995) *J. Chem. Inf. Comput. Sci.*, **35**, 188–195.
Wagner, G., Hyberts, S.G. and Havel, T.F. (1992) *Annu. Rev. Biophys. Biomol. Struct.*, **21**, 167–198.
De Weijer, A.P., Lucasius, C.B., Buydens, L.M.C., Kateman, G., Heuvel, H.M. and Mannee, H. (1994) *Anal. Chem.*, **66**, 23–31.
Widmer, H., Widmer, A. and Braun, W. (1993) *J. Biomol. NMR*, **3**, 307–324.
Wüthrich, K., Widmer, G., Wagner, G. and Braun, W. (1982) *J. Mol. Biol.*, **155**, 311–319.
Wüthrich, K. (1986) *NMR of Proteins and Nucleic Acids*, Wiley, New York, NY.
Wüthrich, K. (1995) *Acta Crystallogr.*, **D51**, 249–270.