Automated Innovization in Drug Discovery using Multiattribute Inverse Design

Eric Gaertner

Leiden Institute of Advanced Computer Science
Leiden University of Science
P.O. Box 9500
2300 RA Leiden – The Netherlands
gaertner_eric@yahoo.fr

Abstract. In the context of drug design, where multi-objective optimization technics try to find best solutions but often loose diversity in the process, findind alternative effective solutions can be an interesting way to improve the diversity.
In this papier, a method is developped in order to find such alternative solutions, based on an evolutionnary algorithm approach. It uses different subpopulations which evolve simultaneously to their best solution, while one subpopulation which also evolves is used as reference. As we look for maximally different alternative solutions, a distance measure is also used.
Moreover, the method is planned to work in pair with a molecular platform, Pipeline Pilot.

Keywords: Evolutionnary algorithm, generate alternatives, inverse design, molecules, drug design.

1 Introduction

Drug design is subject to huge constraints, as candidate molecules have to validate an incredible number of points, such as maximizing efficiency, minimizing side effects, satifying differents kinds of constraints, from an incredibly large search space. Consequently, only very solutions are selected, which result is a loose in diversity.
However, in real life, and particularly in drug design, objectives can be hard to define. In that case, many inferior solutions could be almost as interesting as the optimal solutions. Moreover, the evolution of the needs can cause to add or modify objectives, which could result in upgrading good solutions into best solutions.
Then, an interesting way is to focus on alternative solutions which present similarities with the best solutions, even if they arent as efficient as the best one. In that case, losing some efficiency can lead to more diversity. This problem can be considerated as Inverse design, because we look for solutions in input space, knowing solutions in output space. Then many methods have been developped, each of them presenting different aproaches.
Our method, based on evolutionary algorithm, uses different subpopulations. A reference population evolves to the best solution, which will be the reference solution. Other subpopulations evolve in their own to alternatives solutions, but have to follow some rules depending on the reference solution and the distance from other subpopulations.
Consequently, our paper is decomposed as follows:
We first start presenting a state of art on finding alternatives, with some definitions. Then we will present our methods and some results. Lastly we will conclude with the remaining work to finish.

2 State of art

2.1 Definitions

We start with few definitions.

– Optimization problem

  The main problem is:

  $Minimize\ \ Zp = fp(X) \quad \forall p = 1, ..., P.$
  $subject\ to\ \ gi(X) \leq b \quad \forall i = 1, ..., M.$

  $f1(X)..fp(X)$ are the objective functions,
  $X = \{xj,\ j = 1, ..., N\}$ is the decision vector,
  N is the number of decision variables,
  M is the number of constraints.

P is the number of objectives.

– Evolutionnary algorithm

  An evolutionary algorithm is a set of methods for solving optimization problems. By analogy with natural biological evolution, the main idea is "the survival of the fittest". Candidate solutions play the role of individuals in a population, and the fitness function defines the environnment in which they live. The candidates evolves during the process, and are subject to natural technics like reproduction, mutation, recombination, and selection.
  Consequently, the process keeps individuals that are better suited to their environment.

2.2 Current methods

There are many current methods on findind alternative solutions which can be similar or totally different, using tools such as evolutionnary algorithms, genetic algorithms, niching, or using specific molecular properties.

In the methods which are not applied to molecules, E.M Zechamn and S. Ranji Ranjithan use an evolutionnary algorithm based on evolving a set of instructions or rules to minimize the prediction error, which be used as fitness function [2]. Different subpopulations evolve with the supervision of a reference subpopulation and depend on the fitness function but also on how much the constraints are satisfied.

On their side, D.H. Loughlin and S.Ranjithan [3] have developped a method based on genetic algorithm, which sorts the population after each generation according to a difference measure, and a small set of the most different solutions are placed strategically in the solution vector. It uses also some fitness sharing and niches, which have restrictions in order to minimize migration of a solution from one niche to another.

From an other point of view, E. Zitler and S.Knzli [4] focus on the question: how to deal with 2 conflictig goals: minimize distance to the pareto set, and maximize diversity at the same time? Their method try to integrate preference information of the decision maker, by using pareto based ranking. Fitness assignment and binary tournament method are also implemented.

T. Park and K.Ryel Ryu [7] use a totally different approach with genetic algorithms. Indeed, 2 populations are used. The main population evolves in order to find best solution, and the reserve population evolves in order to offer diversity to the main population. In order to fulfill that, it has a fitness function which is based on how far (on average) is the individual from the other population. The exchanges between the 2 populations are done by crossbreeding.

Also, a good example of non trivial application of this kind of inverse design is the paper [8], in which public healht is involved.
Water distribution is subject to important controls, possible contamination has to be evaluated continuously. As the the contaminant source has to be detected from some components found in the water, the problem is an inverse problem. It is necessary to identify more than one unique soltion in order to let the automatic detectors find with success when there is contamination.

In an other hand, general tools can be applied with success on molecules. A good example is the paper [6], in wich genetic algorithm with sorting combined to niching can lead to find alternative solutions. The idea behind is that genetic algorithms are an interesting way to find efficient solutions, but most of time loose diversity. In that case niching extention allow to maintain diversity by converging into multiple optimas. About the sorting, best ranking is assignated to non dominated solutions, and equal dominance based rank are done looking at the contribution of the solution in the diversity in objective space, using dissimilarity measure. Niching radius can be adjusted in order to get the best diversity/quality ratio.

In addition to these methods, we can find in the molecular domain molecule based methods [9] or fragment based methods [10].
Virtual screening which can be either molecular or fragment based, is another example of the main molecular methods. It consists in searching for fragments or molecules similar to a particular one, in a large database. An example of this method has been done by H. Geppert, M. Vogt, J. Bajorath [5]. They combine virtual screening to the question of the chemical space representation. Indeed, there is always a gap between the chemical domain and the computational domain that chemoninformatics research try to fill in. Therefore, representaion possibilities give birth do many different approaches. Some examples are the way to study similarity, such as distance metric, or 2 or 3 dimensional comparison.

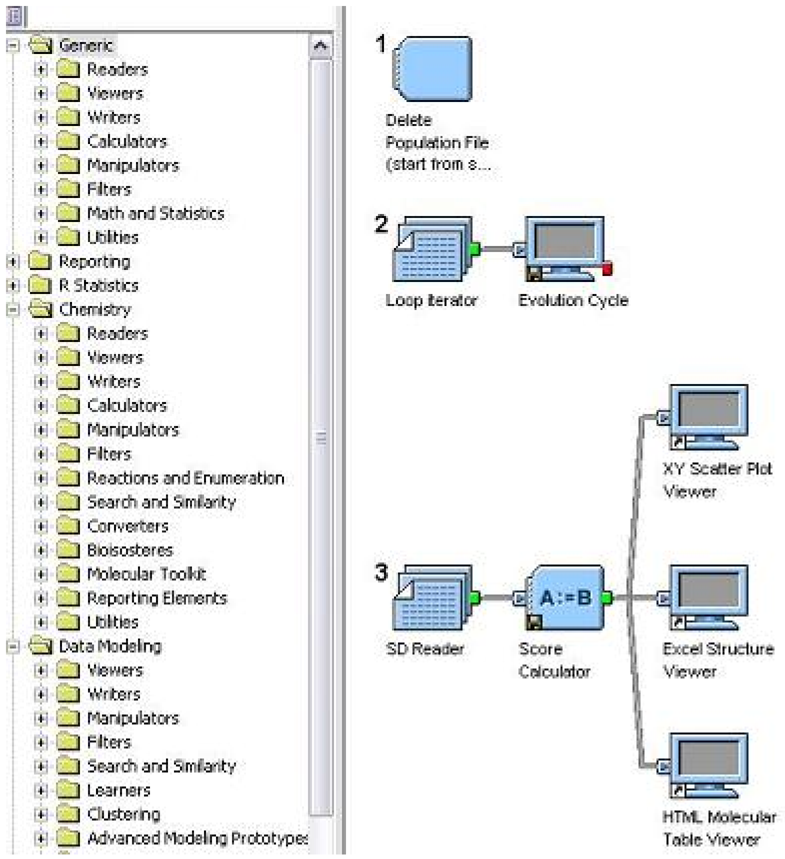
Lastly, a method has been developped based on "geometrical reshuffling" [11]. Using the fact that the behaviour of the molecule is dictated by its geometrical form and its electrostatic properties, a reference molecule (which can be a best solution from a problem) is modified by elementary transformations, such as operations on groups of atoms. The result is a large set of molecules wich present similar geometrical and electrostatic properties to the reference molecule so they fulfill the same roles, even if they are different.

In our case we will use E.M Zechman S. Ranji Ranjithan paper, "An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems" [1] as a basis for our method.

3 Methods

3.1 Language and Platform

The method is programmed in cpp.
The Platform used is Pipeline Pilot, which is a set of applications for modeling and simulation. It allows to connect easily components such as sources, databases, and applications together by a graphical interface. We select components, we drag them into the main window, connect them. Then we run the obtained procotol and it gives the result. We have access to the parameters of each component in another window.
Pipeline Pilot currently possesses a complete set of components dedicated to the chemistry, which allows to manipulate molecular data.
An example of what does it look is below.



3.2 General idea

Our method uses as a basis the work in [1]. The main idea is to do in a given population a minimization in order to get the best solution, then make relax-

ation of the best solution, to identify potential alternatives solutions which will be sufficiently efficient. Then we focus in the obtained set from relaxation on finding maximally different solutions.
To realize that, we divide the initial population into many subpopulations, and construct an evolutionary algorithm which will make them evolve. A subpopulation is used as reference, and the objective value of it best solution, which can change during evolution, will be used as reference for the other subpopulations, in the sense that candidate solutions in other subpopulations will be considered as feasible if they are in the relaxation constraint (choosen by us or the decision maker) and unfeasible otherwise. In the other subpopulations, feasible candidate best solutions will be chosen by their distance from the other populations, identified by their centroids. In case there are no sufficient feasible candidates, unfeasible candidates best solutions will be chosen by best objective value. Evolution process also includes some mutation and crossover.
We develop the method step by step, from very generic and simple case to bi objective case adapted to molecules with some improvements.
The input is a population of individuals with their characteristics represented in the decision variables.
The output is a set of best solution and alternative best solutions.

3.3 1st algorithm: 1-objective

Let's see the optimization problem in one objective case:

$$\begin{array}{ll} Minimize & Z = f(X) \\ subject\ to & g_i(X) \leq b \quad \forall i = 1, ..., M. \end{array}$$

$f_1(X)..f_p(X)$ are the objective functions,
$X = \{x_j,\, j = 1, ..., N\}$ is the decision vector,
N is the number of decision variables,
M is the number of constraints.

Adding alternative solutions issue to the optimization problem adds the following objective:
Let X* be the optimal solution with an objective value Z*.
In order to generate an alternative solution which is maximally different from X*, we have to solve:

$$\begin{array}{ll} Maximize & D = \sum_j |x_j - x_j*| \\ subject\ to & g_i(X) \leq b \quad \forall i = 1, ..., M \\ and & f(X) \leq T(Z*). \end{array}$$

T represents the relaxation of the optimal solution in order to find good solutions close to the optimal.

The algorithm works as follows:
- *Step 1:*
Create an initial population with p subpopulations, each with a population size of K.
- *Step 2:*
In the reference subpopulation, evaluate and identify the best solution with respect to the modeled objective.
- *Step 3:*
In each subpopulation, evaluate all solutions with respect to the modeled objective. In addition, assign feasible flag to the candidates which satisfy the relaxation constraint, and assign unfeasible otherwise.
- *Step 4:*
.For each subpopulation other than the reference, compute the associated centroid. The formula is $c_i^p = (1/K)\sum_k x_i^(k,p)$; $i = 1, ..., N$, where c_i^p is the centroid location of decision variable i in subpopulation p, $x_i^(k,p)$ is the decision variable i of solution k in subpopulation p, and N is the number of decision variables.
.For each candidate of each subpopulation other than the reference, compute its distance from the centroids of the other subpopulations. The formula is $D^(k,p) = \min \sum_i |x_i^(k,q) - c_i^p|; p = 1, ..., P; p \neq q$.
.For each subpopulation other than the reference, compute the best candidate: if all solutions in the given subpopulation are unfeasible, the best candidate will be the one who is the best with respect to the modeled objective. Otherwise, the best candidate will be the feasible solution which is the most distant to the other subpopulations through centroids.
- *step 5:*
If the number of wanted iterations is reached, stop the algorithm, otherwise go to step 6.
- *step 6:*
.Binary tournament selection is done between candidates of their same subpopulation.
For the reference subpopulation: best candidate is chosen with respect to the modeled objective.
For the other subpopulations:
If both candidates are feasible, then best candidate will be the one who is the most distant to the other subpopulations.
If only one candidate is feasible, then the feasible candidate will be selected.
If no candidate is feasible, then best candidate is chosen with respect to the modeled objective.
.Winner are stocked in order to be used in next step.
- *step 7:*
.Apply recombination operators to the selected candidates from step 6.
.Repeat from step 2.

A reflexion was done in order to be as clear as possible, as the algorithm is

going to be complexified. Thus a class candidate was created, whose characteristics are:
x: first decision variable of the candidate.
y: second decision variable of the candidate.
p: subpopulation where the candidate belongs.
flag: feasibility flag.
Z: objective value.

2 algorithms have been implemented, one for very basic tests, and one more robust.

3.4 2nd algorithm: 2-objective

Let's see the optimization problem for 2 objective case:
$Minimize \;\; Zp = fp(X) \quad \forall p = 1, 2.$
$subject\ to \;\; gi(X) \leq b \quad \forall i = 1, ..., M.$

$f1_(X)..fp(X)$ are the objective functions,
$X = \{xj,\, j = 1, ..., N\}$ is the decision vector,
N is the number of decision variables,
M is the number of constraints.
P is the number of objectives.

Adding alternative solutions issue to the optimization problem objective becomes now more complicated. We can still say:
Let X* be the optimal solution with an objective value Z*.
In order to generate an alternative solution which is maximally different from X*, we have to solve:

$Maximize \;\; D = \sum^{j} |xj - xj^*|$
$subject\ to \;\; gi(X) \leq b \quad \forall i = 1, ..., M$
$and \qquad f(X) \leq T(Z^*).$

T represents the relaxation of the optimal solution in order to find good solutions close to the optimal.

But now the difference is about the choice of X*, because we have instead of one best solution a set of best solutions. Some approaches can be done in order to solve that:

Desirability function :
First solution is to use desirability funtion.
It is a multiplicative weighted objective operation, where we can choose to give more importance to an objective. We have the following function:

$D = \prod^i \mathrm{d}_i(x)^{\gamma i}$ with $i = \{1, 2\}$ (γ^i can be equal to 1)

Each objective is associated with an desirability index $d_i()$.
Desirability function is a multiplication of desirability indexes of the objective functions. It is also possible to deal with a factor γ.
Desirability index $d_i()$ takes a value between 0 and 1. 0 means that the objective is not satisfied at all, while 1 means that the objective is fully satisfied (we don't look for other solutions even if they are better).

Pareto approach :
Second solution is to use epsilon dominance instead of relaxation.
Addictive $\epsilon - dominance$:
x $\prec^\epsilon x'$ iff
$\forall\ i \in 1, ..., m,\ fi(x) \leq\ fi(x') +\ \epsilon$
$\exists\ i \in 1, ..., m,\ fi(x) \prec\ fi(x') +\ \epsilon$

Best solution Z*:
Z* $=\ y^1, y^2, ..., y^k\} \subset R^m$
$\forall y* \in Z*, y* \prec\neq^\epsilon f(x)$

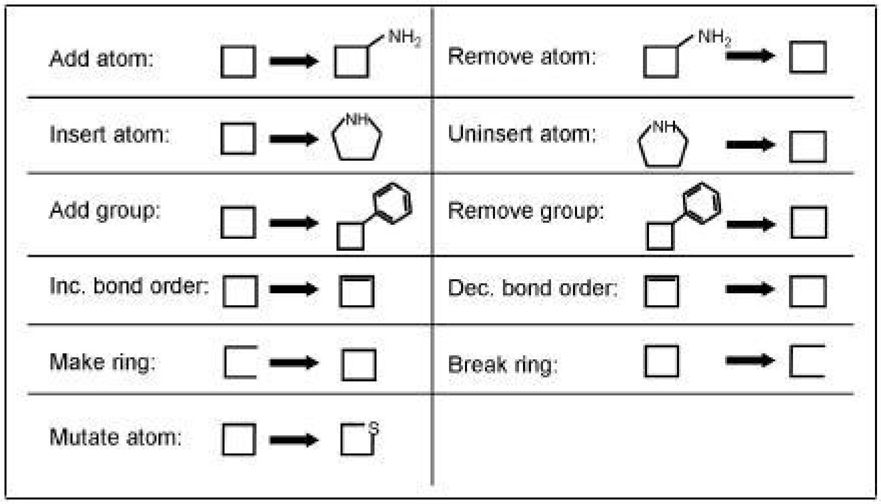
Pareto approach seems more interesting, we are thinking about it.

Regarding the algorithm, it is in current developpment.
We use the developped algorithm and we adapt it to the 2 objective case. That implies:
- First subpopulation gives now a set of optimal solutions.
- A candidate solution is feasible if it is within the relaxation tolerance range of each objective for one pareto optimal solution of the reference subpopulation, otherwise it is unfeasible.
- When the selection is supposed to be by best objective value, we can know sort candidates by non dominance rank, then sort equal candidates by crowding distance. (a crowding distance d_i of a solution i is a measure of the search space around i which is not occupied by any other solution in the population).
- If the binary tournament method of step 8 doesn't adapt well, we can search for another method, like $(\mu + \mu)$ selection for the reference subpopulation, or all the subpopulations.

3.5 3nd algorithm: 2-objective, adapted to molecules

After the 2 objective general case, we will adap the method to the chemical domain. Instead of using points in 2 dimensional space, we will use molecules. The main used molecule format is the chemical data file format sdf ("'structure

data file"').
We will use the molecular tools that have been developped in Pipeline Pilot in order to make the transformation between general case and molecular case.
About the mutations, we will use the ones from the paper [6], which can be seen below.



3.6 3nd algorithm: improvements

We have though about some improvements we will test as soon the 2 objective algorithm adaptated to molecules will be implemented.
The main improvement we want to do is about the mutations. From [11], we know that using elementary transformations can greatly reduce redundancy, even avoid it, on obtained molecules, and so improve efficiency and time processing. These elementary transformation have been done in [11], so we would like to transpose them in this problem.
There are also some ideas to think about:
- Change measure distance.
- Change the centroid formula.
- Use candidates wich are best with respect to one particular objective to explore their neighbourhood in search space, in order to get solutions who would be very good in that objective, even better than alternative solutions that have been found, but weren't selected because they have their other objective value just below the threshold which depends on the choice of the relaxation constraint.

4 Results

The following parameters have been used:
Constraint: $x < 10$

Relaxation on objective value: 20%
Mutation: $-+5\%$, with 50% probability to occur.
Crossover: 50% probability to occur.
Iterations: different values.
Population size: different values.

Basic constraint is used in order to avoid interesting solutions not to be selected, as we are testing the method. Also even if real constraints are more interesting and realistic, puting them will not change the efficiency of the algorithm.

The notation is:
x: first decision variable of the candidate.
y: second decision variable of the candidate.
p: subpopulation where the candidate belongs.
flag: feasibility flag.
Z: objective value.

4.1 1 objective algorithm: 1st dataset and objective function

The function used is $f(x) = x + y$.
The population is composed of 3 subpopulations, each composed of 3 candidates for a total of 9 candidates.
We have chosen initial candidates: c1(1,2), c2(2,1), c3(2,2), c4(0,1), c5(1,0), c6(1,1), c7(2,3), c8(3,2), c9(3,3).
3 iterations have been done in tables 1,2,3.
50 iterations have been done in tables 4,5,6.

- 3 iterations:

Table 1. 1st results for 1st dataset and objective function, 3 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 1.805	x = 0.9975	x = 1.805
y = 0.9025	y = 0.9975	y = 2.7075
p = 1	p = 2	p = 3
flag = 1	flag = 0	flag = 0
Z = 2.7075	Z = 1.995	Z = 4.5125

Table 2. 2nd results for 1st dataset and objective function, 3 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 0.9975	x = 1.1025	x = 1.995
y = 1.995	y = 1.1025	y = 2.9925
p = 1	p = 2	p = 3
flag = 1	flag = 0	flag = 0
Z = 2.9925	Z = 2.205	Z = 4.9875

Table 3. 3rd results for 1st dataset and objective function, 3 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 0.9975	x = 0.9975	x = 1.995
y = 1.995	y = 0.9975	y = 2.9925
p = 1	p = 2	p = 3
flag = 1	flag = 0	flag = 0
Z = 2.9925	Z = 1.995	Z = 4.9875

- 50 Iterations:

Table 4. 1st results for 1st dataset and objective function, 50 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 1.46831	x = 0.894609	x = 2.20247
y = 0.734157	y = 0.894609	y = 1.46831
p = 1	p = 2	p = 3
flag = 1	flag = 1	flag = 0
Z = 2.20247	Z = 1.78922	Z = 3.67079

Table 5. 2nd results for 1st dataset and objective function, 50 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 1.08476	x = 0	x = 1.62714
y = 0.542381	y = 0.776676	y = 1.08476
p = 1	p = 2	p = 3
flag = 1	flag = 1	flag = 0
Z = 1.62714	Z = 0.776676	Z = 2.71191

Table 6. 3rd results for 1st dataset and objective function, 50 iterations

reference subpopulation	subpopulation 2	subpopulation 3
x = 1.54947	x = 1.47558	x = 2.3242
y = 0.774734	y = 1.47558	y = 1.54947
p = 1	p = 2	p = 3
flag = 1	flag = 1	flag = 0
Z = 2.3242	Z = 2.95115	Z = 3.87367

- Analysis:
Tests have been done with 3 and 50 iterations.
We can do the following remarks:
- Correct solutions are found for both number of iterations: final solutions are better than initial solutions. The candidates in subpopulations 2 and 3 evolve correctly in order to be closer to the best solution of the reference subpopulation.
- 50 iterations allow to find better solutions than 3 iterations.
- The possibilities are limited because of the low number of canditates, and the fact that 20% relaxation with these results bring to unfeasible solutions most of the time.

4.2 1 objective algorithm: 2nd dataset and objective function

The function used is Himmelblau function: $f(x) = (x + y - 11) + (x + y - 7)$, which has 4 local minimums, all equals to 0: $f(3; 0) = f(-2, 805118; 3, 131312) = f(-3, 779310; -3, 283186) = f(-3, 779310; -3, 283186) = f(3, 584428; -1, 848126) = 0$. Thus this function is ideal for testing the problem of findind alternatives, as the suppopulations can evolve to different local minimum, satisfying at the same time the relaxation constraint.
The population is composed of 4 subpopulations, each composed of 20 candidates for a total of 80 candidates.
As for the first dataset, the candidates are the same for each test, but now they have "random" values beewten -15 and 15.

- 3 iterations:

Table 7. 1st results for 2nd dataset and objective function, 3 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = -3.99	x = -1.995	x = -3.99	x = 3.61
y = -2.9925	y = 2.9925	y = -2.9925	y = 0.9025
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 0	flag = 1	flag = 0
Z = 7.85663	Z = 16.2222	Z = 7.85663	Z = 15.245

Table 8. 2nd results for 2nd dataset and objective function, 3 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = 3.99	x = 1.995	x = -2	x = 0.9975
y = 1.995	y = 2.9925	y = 4	y = -0.9975
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 0	flag = 1	flag = 0
Z = 48.7595	Z = 31.8235	Z = 58	Z = 146.13

- 100 iterations:

Table 9. 1st results for 2nd dataset and objective function, 100 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = 2.52103	x = 2.52103	x = -1.37925	x = -3.75352
y = -0.840341	y = 0	y = 2.75849	y = 2.50235
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 1	flag = 1	flag = 1
Z = 44.3167	Z = 41.6319	Z = 40.7782	Z = 51.438

Table 10. 2nd results for 2nd dataset and objective function, 100 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = -3.7152	x = 2.7864	x = -3.7152	x = 2.7516
y = -2.7864	y = 0	y = -2.7864	y = 0.687899
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 0	flag = 1	flag = 0
Z = 8.7098	Z = 28.2261	Z = 8.7098	Z = 21.7643

- 500 iterations:

Table 11. 1st results for 2nd dataset and objective function, 500 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = -2.78582	x = -2.0527	x = -2.0527	x = 1.78268
y = 3.71442	y = 3.07906	y = 3.07906	y = 2.67402
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 1	flag = 1	flag = 0
Z = 16.315	Z = 13.9275	Z = 13.9275	Z = 30.239

Table 12. 2nd results for 2nd dataset and objective function, 500 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = 2.28045	x = 2.23528	x = 1.7689	x = 1.67646
y = -0.76015	y = 0.55882	y = -0.884449	y = 1.39705
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 1	flag = 0	flag = 1
Z = 60.1835	Z = 49.4689	Z = 96.4502	Z = 57.5059

- 1000 iterations:

Table 13. 1st results for 2nd dataset and objective function, 1000 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = -1.64681	x = 4.24037	x = -1.23358	x = 2.26068
y = 2.19575	y = 0	y = 2.46715	y = 3.39102
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 1	flag = 1	flag = 1
Z = 51.7501	Z = 56.3463	Z = 53.7646	Z = 51.9347

Table 14. 2nd results for 2nd dataset and objective function, 1000 iterations

reference subpopulation	subpopulation 2	subpopulation 3	subpopulation 4
x = 2.45758	x = 2.45758	x = 1.10352	x = 1.8392
y = -0.819192	y = 0	y = 1.10352	y = 1.8392
p = 1	p = 2	p = 3	p = 4
flag = 1	flag = 1	flag = 0	flag = 0
Z = 48.3901	Z = 45.2384	Z = 97.2105	Z = 36.5485

- Analysis:

We can do the following remarks:
- The difference in convergence between 3 iterations and the others iterations is notable. Furthermore, few iterations work well if it exists candidates in the subpopulation which are already near to the best solutions, otherwise we need more iterations in order to get closer to the best solutions.
- More iterations improves efficiency of the method, but after some number of iterations it no longer improves, so we have already reached converge. For example 1000 iterations are not better than 500.
- Correct solution is found in reference population and good alternative solutions are found in the other subpopulations. Indeed the alternatives solutions satisfy the relaxation contraint, but are sufficiently differents (for example one will have

negative x while the other one will have positive one).
- Time: the algorithm is fast, as it takes less thant 30 seconds to execute 1000 iterations.

4.3 Future results

The 2-objective algorithm is in current developpment.
In case we can put weights in the objective functions, we can use the previous algorithm in order to get results from a second objective case.
For the test of the algorithm we can try some examples from [12] and [13], such as superspheres problem with multiple local optima.
The adaptation of the method with molecules and improvements will be the next step.

5 Conclusion and perspectives

In the context of molecules, findind alternatives to the best solution is an idea which is taking more and more interest.
The method developped here, which is still in developpment, is in a good way to find maximally different alternatives solutions in molecular domain. We have shown that it can find distant alternative solutions to the reference solution on sufficiently big sets of candidates, with a good number of iterations. These distant alternative solutions are also most of the time distant from themselves.
However, there are some limitations, for example the number of alternative solutions depend on the number of supbopulations, so looking for a supplementary alternative solution is equal to adding an entire subpopulation, which could be annoying with more complex computations. But in counterpart, it presents good particularities, like interesting diversity in the alternative solutions, not only in comparison with the reference solution, but also between them, and high modularity. For example we can set differents parameters such as the choice of the relaxation, mutations, crossovers, to try to obtain different interesting results, or we can also let the decision maker fix himself some of these parameters.
To conclude, we think the method has some perspectives, like using alternative solutions as references in order to explore similar solutions in new sets of molecules.
For the next weeks we will finish the method for the 2 objective case, and then we will focus on adapting the algorithm on molecules, and also try some improvements we have though about. By this way we will show that evolutionary algorithm to generate alternatives works in the context of molecular evolution. More perspectives will come after the improvements part.

References

1. Zechman, E.M., Ranji Ranjithan S.: An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems. Engineering Optimization 36(5),539-553 (2004)
2. Zechman, E.M., Ranji Ranjithan S.: Multipopulation cooperative coevolutionary programming (MCCP) to enhance design innovation. GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computationACM Press, 1641-1648 (2005)
3. Loughlin, D. H., Ranjithan, S., Baugh, J. W. and Brill, E.D., Jr : Generic algorthm approches for addressing unmodeled objectives. Engineering Optimization 33(5), 549-569. (2001)
4. E. Zitzler and S. Knzli.: Indicator-Based Selection in Multiobjective Search. Conference on Parallel Problem Solving from Nature (PPSN VIII), 832–842, 2004. Springer.
5. H. Geppert, M. Vogt, J. Bajorath Current Trends in Ligand-Based Virtual Screening: Molecular Representations, Data Mining Methods, New Application Areas, and Performance Evaluation. J. Chem. Inf. Model., 2010, 50 (2), pp 205216
6. J. W. Kruisselbrink, A. Aleman, M. T.M. Emmerich, Ad P. IJzerman, A. Bender, T. Baeck, Eelke van der Horst Enhancing search space diversity in multi-objective evolutionary drug molecule design using niching. Proceedings of the 11th Annual conference on Genetic and evolutionary computation 217-224 (2009), ACM
7. T. Park, K.Ryel Ryu, A dual population genetic algorithm with evolving diversity. 2007 IEEE Congress on Evolutionnary Computation (CEC2007)
8. Kumar, J., E. M. Zechman, E. D. Brill, G. Mahinthakumar, S. Ranjithan, J. Uber, Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods, Proceedings of the ASCE/EWRI World Environmental and Water Resources Congress, Tampa, FL, May 2007.
9. Schneider G, Fechner U.: Computer based de novo design of drug like molecules. Nat Rev Drug Discov. 4(8) 649-63 (2005 Aug)
10. Rees DC, Congreve M, Murray CW, Carr R.: Fragment-Based Lead Discover Nat Rev Drug Discov. 3(8) 660-72 (2004 Aug)
11. E. Gaertner : Réarrangements géomtriques pour la création de novo de molécules, (2009)
12. K. Deb, S. Agrawal, A. Pratap, T; Meyarivan: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization Parallel Problem Solving from Nature PPSN VI, Vo 1917/2000, 849-858 (2000), Springer
13. O. M. Shir, M. Preuss, B. Naujoks, and M. Emmerich Enhancing Decision Space Diversity in Evolutionary Multiobjective Algorithms Proceedings of the 5th International Conference on Evolutionary Multi-Criterion Optimization, 95 - 109 (2009)