Degeneration Recognizing Clonal Selection Algorithm for Multimodal Optimization

Nan Xu, Yongsheng Ding, *Senior Member, IEEE*, Lihong Ren, and Kuangrong Hao

*Abstract*—In this paper, a computing speed improvement for the clonal selection algorithm (CSA) is proposed based on a degeneration recognizing (DR-) method. The DR-CSA is designed for solving complex engineering multimodal optimization problems. On each iteration of CSA, there is a large amount of eliminated solutions which are usually neglected. But these solutions do contain the knowledge of the non-optimal area. By storing and utilizing these data, the DR-CSA is aimed to identify part of the new population as degenerated and eliminate them before the evaluation operation, so that a number of evaluation times can be avoided. This pre-elimination operation is able to save computing time because the evaluation is the main reason for the time cost in the complex engineering optimization problem. Experiments on both test function and a real-world engineering optimization problem (wet spinning coagulating process) are conducted. The results show that the proposed DR-CSA is as accurate as regular CSA and is effective in reducing a considerable amount of computing time.

*Index Terms*—clonal selection algorithm, degeneration recognizing, multimodal optimization, wet spinning coagulating process, complex engineering optimization

# I. INTRODUCTION

E

ngineering is the creative application of scientific principles to design or develop useful systems. For many reasons, such as cost saving, efficiency increasing, or markets expanding, one engineering process seeks not only a feasible method, but also the best design in some ways. Selecting the best design of an engineering process from some available alternatives is called engineering optimization.

With the development of modern industrial technologies and computer science, a lot of real-world engineering processes become more and more complicated. Consequently, the optimization process is facing more challenges nowadays. A new research topic of Complex Engineering Optimization (CEO) has risen at the moment and drawn wide attention [1, 2]. Complex engineering optimization usually subjects to various difficulties such as uncertainties, computational expensiveness, system complexity, multiple criteria/ objectives/decision variables, and so on [2]. In recent years, a great deal of research effort has been devoted to solving CEO problems [3]. A number of new research areas have emerged including evolutionary optimization in dynamic and uncertain environments, such as surrogate-assisted evolutionary optimization [4], multi- and many-objective optimization [5], large-scale optimization, and integrated optimization [6, 7].

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This work was supported in part by the Key Project of the National Natural Science Foundation of China (no. 61134009), the National Natural Science Foundation of China (nos. 61473077, 61473078, 61503075), Cooperative research funds of the National Natural Science Funds Overseas and Hong Kong and Macao scholars (no. 61428302), National Key Research and Development Plan from Ministry of Science and Technology (2016YFB0302701), Program for Changjiang Scholars from the Ministry of Education, and International Collaborative Project of the Shanghai Committee of Science and Technology (no. 16510711100).

All the authors are with the Engineering Research Center of Digitized Textile & Apparel Technology, Ministry of Education, together with the College of Information Science and Technology Donghua University, Shanghai 201620, China (corresponding to Prof. Y.-S. Ding, phone: 86-21-67792301; fax: 86-21-67792353; e-mail: ysding@dhu.edu.cn).

Among all different research areas of the CEO problems, multimodal optimization is a specific type of optimization problem which deals with optimization tasks that involve finding all or most of the multiple solutions (as opposed to one single best solution) to a problem [8, 9]. Different from multi-objective optimization, multimodal optimization aims at solving the multiple best solutions of one single objective.

Multimodal optimization is quite common in real-word engineering problems, and its solutions always have great practical values. To an engineering optimization task, the knowledge of multiple solutions is usually very helpful. For example, sometimes due to physical (and/or cost) constraints, the best results may not always be realizable. In such a scenario, if multiple solutions (local and global) are known, the implementation can be quickly switched to another solution and still obtain an optimal system performance [9].

Evolutionary computing techniques play a major role in solving multimodal optimization problems, which mostly involve metaheuristic optimization algorithms such as Evolutionary Algorithms (EAs), Swarm Intelligence (SI), Differential Evolution (DE), and Artificial Immune System (AIS). These methods have proven to be powerful for global optimization of a wide range of problems [10-18].

The challenge of using the evolutionary computing techniques for multimodal optimization is that the multimodal problem is against the natural tendency of evolution, which will always converge to the best (or a sub-optimal) solution [10]. To overcome such difficulty, Niching is a technique of finding and preserving multiple stable niches, or favorable parts of the solution space possibly around multiple solutions, so as to prevent convergence to a single solution [11]. Also, Petrowski’s clearing method [12] and Goldberg’s sharing function approach [13] are two very well studied and respected methods in the Genetic Algorithm (GA) community for multimodal problem. DE based local selection and global selection approaches have also been attempted for solving multimodal problems. A cluster-based DE method for multimodal problems can be seen in [14]. A scatter learning PSO algorithm was proposed for multimodal problems in [15].

In [19], the Clonal Selection Algorithms (CSA) was shown to have a great ability in solving multimodal problems. It is capable of locating a larger number of local and global optima solutions. In [20] a synergetic immune CSA method was proposed for fiber drawing process. In [21], a surrogate-assisted CSA was proposed to improve the computing speed for expensive optimization problems.

In order to explore the fitness landscape, the clone amount is usually more than a few. For example, the clone number is usually 10 in the references. So this means on each generation, comparing to other metaheuristic methods, CSA has a 10 times population which requires a 10 times evaluations. This makes CSA a lot slower than other methods. If the original population is 100, and the iteration times is 30, then there will be a 3e5 times of evaluation in total. If it takes a few seconds to evaluate some real-life engineering model, then the overall optimization time for this task will take days to finish.

But in many of the real-life multimodal optimization tasks which CSA is required, the speed of the CSA is very important. For example, during developing the control system for a real-life production process, the CSA is adopted to search for all of the equivalent optimal control plans. By doing this, the system can be quickly switched to another control plan if the running one is not optimal anymore (which maybe cause by a sudden change on physical constraints). In such cases, the speed of the CSA determines the time costs of the system development tasks. For an industrial company, it would be better if such time costs is low, which requires the CSA to be fast. And for online optimization problems, the speed is even more important for the system to have a good real-time ability.

In this paper, a novel computing speed improvement is designed for the CSA to solve time-consuming multimodal optimization problems. The proposed method is called Degeneration Recognizing Clonal Selection Algorithm (DR-CSA). The method sacrificed a small amount of optimization accuracy in return for a great improvement of the calculation speed. In a lot of real-life engineering optimization tasks, the optimization objectives such as temperature, pressure, concentration, etc., don’t require an accuracy to three decimal places. But for these companies saving time costs is much more important. Efficiency determines the competitiveness of an industry. So the proposed speed-improved method has great practical value, it is significant for real-life industrial problems.

The main contributions of this paper are as follows: 1) A novel idea of improving calculation speed of CSA by accumulatively collecting and studying the eliminated solutions is proposed; 2) The proposed idea is achieved by a few benchmark functions, and the test results demonstrated that DR-CSA has excellent performance on calculation speed; 3) The optimization task for polymer fiber wet spinning coagulating process is accomplished by adopting DR-CSA, about 60% of the calculation time cost is saved.

This paper is organized as follows. Section II introduces the basic idea of CSA for multimodal optimization. Section III states the design details of the DR-CSA. Section IV conducts test experiments on DR-CSA and analyzes the results. In Sections V, the DR-CSA is applied to the engineering problem of wet spinning process. Section VI concludes the paper with some concluding remarks and suggestions for future work.

# II. Clonal Selection Algorithm

## A. The principle of clonal selection algorithm

AIS are a class of computationally intelligent systems inspired by the principles and processes of the vertebrate immune system. For optimization, AIS usually has a population of antibodies (candidate solutions), one (or a few) antigen (optimization objective), and one function of affinity calculation (objective function). The attempt is for the antibodies to reach or match the antigens (optimize) [22-24].

In AIS family, CSA is a class of algorithms inspired by the clonal selection of acquired immunity that explains how B and T lymphocytes improve their response to antigens over time called affinity maturation. These algorithms focus on the Darwinian attributes of the theory where selection is inspired by the affinity of antigen-antibody interactions, reproduction is inspired by cell division, and variation is inspired by somatic hyper-mutation [21]. The CSA was proposed in [19].

According to the clonal selection theory, only cells that are capable of recognizing an antigen will reproduce and be maintained as memory cells; differentiation happens along with proliferation; diverse antibody patterns are formed by accelerated somatic mutation [25]. Based on such phenomenon, antibodies with higher affinity are selected and cloned, then the hyper-mutated clones are re-selected again.

Fig. 1 (the solid line part) presents the block diagram of the computational procedure for the CSA. And these general steps shown in Fig. 1 (also proposed in [19]) of the CSA to solve optimization problems are defined as follows:

Step 1: Initialization: randomly initialize a population *N* of antibodies.

Step 2: Evaluation: determine the affinity of each antibody.

Step 3: Selection and cloning: select a number (*n*) of the highest affinity antibodies and generate clones independently and proportionally to their affinities.

Step 4: Hyper-mutation: generating matured clones. The higher the affinity, the smaller the mutation rate.

Step 5: Clone evaluation and re-selection: determine the affinity of the matured clones in relation to antigen. Select the antibody with the highest affinity from the matured clones and form the new population *N*.

Step 6: Repeat Steps 2-5 until termination criterion is met.

To sum up, the main characteristics of the CSA are: 1) Instead of using recombination operators (such as crossover in GA), the CSA generates the new antibodies by cloning so that all the antibodies are independent; 2) The mutation rate is usually higher (compares with other evolutionary computing techniques) and inversely proportional to the affinity.

## B. The CSA for multimodal problem

As to multimodal optimization problem, the objective is to find all or most of the equally/approximately best solutions. The multiple best solutions can locate in different independent extremes or a continuous extremal area. In order to locate all of them, the evolutionary optimization algorithms need to evolve all or most of the population to let the solutions evenly distributed on the optimal peaks/areas.

For multimodal optimization problems, two parameters of the CSA may assume default values [19]:

### 1) Assign n = N, i.e., all antibodies from the population will be selected for cloning in Step 3.

### 2) The number of clones generated for each of the N antibodies should be the same.

With these assumptions, the CSA is capable of exploring the fitness landscape individually and evolving the entire population simultaneously. The population tends to spread on the optimal peaks/areas. As a result, the CSA is a feasible method for solving multimodal optimization problems.

However, since the CSA for multimodal optimization clones all of the population (*n*=*N*) on every iteration, so the cloning operation in CSA highly increases the evaluation times. As a result, CSA is difficult to be applied to some real-world complex engineering problems.

# III. Degeneration Recognizing Clonal Selection Algorithm

## A. General idea of degeneration recognizing method

There are few ways to improve the computing speed of the CSA, such as the surrogate-assisted [4] [21] or detecting promising areas [16]. The methods may be different, but the principle is mainly the same. This basic principle is to reduce the times of direct calculation of the affinity via the mechanism model (evaluation), because the evaluation is the major reason of the time cost.

The intention here is to design a fast and simple method which aims at reducing the times of evaluation as well. The selection operation of CSA of each generation selects part of the population and eliminates the rest. We noticed that the selected ones may or may not be the final optimal solutions, but the eliminated ones are definitely not the optimal solutions. These eliminated antibodies along with theirs mark of “eliminated” are of some useful information.

To be specific, first, one can be sure that an antibody, which was eliminated once before due to poor affinity, should be eliminated again if it reappeared. Therefore, using a database to store all of the eliminated antibodies and match every new antibody with this database to see if it is a once-eliminated antibody before its evaluation should reduce some evaluation times. As a most preliminary idea, we can foresee that such method could avoid a few times of model calculation and save a few time.

To further develop the method (assuming the problem is continuous and differentiable), if one antibody is eliminated before, all other antibodies located inside a very small neighbourhood of this antibody should be eliminated. This determination is made under the assumption that two very close antibodies have approximately same affinities. For most of the real-word engineering problems, this assumption is acceptable because real-word problems are usually continuous and smooth. Then, if the size of the neighbourhood is properly chosen, a considerable amount of calculation times would be avoided. In this paper, a new generated antibody (by clone and hyper-mutation) falling into a small neighbourhood of a once-eliminated antibody is called ***degeneration***. This antibody is called ***degenerated antibody*** and the small neighbourhood is called ***degenerated area***. Once a degenerated antibody is recognized, it will be eliminated immediately. This method is called Degeneration Recognizing Clonal Selection Algorithm.

The surrogate-assisted kind of method usually uses a faster approximation to replace some of the evaluations, it is effective in saving evaluation time but the approximation may affect the accuracy of the result. However, the optimization accuracy of the proposed DR-CSA method is as accurate as the CSA. This is due to the fact that all the approximations in the DR-CSA are used on the soon-to-be-eliminated antibodies, while all the selected antibodies were evaluated.

To sum up, there are three reasons to believe that the proposed DR-CSA is feasible:

1) Since the clone operation of the CSA creates a huge population, which means the eliminated population is also huge. Even though each eliminated antibody only has a very small neighbourhood (degenerated area), when the amount of the eliminated antibody is great, the combined degenerated area will be large enough to make a difference.

2) Due to the hyper-mutation process, which is a great help for exploring the function landscape, a large number of degenerated solutions is created. This pre-elimination method has a relatively large probability in recognizing the degenerated solutions and the calculation time can be obviously saved.

3) The accuracy of the multimodal optimization results is not affected by the proposed degeneration recognizing method.

Comparing to surrogate-assisted methods in [4] [21], the DR-CSA method is not facing the problem that surrogates usually introduce approximation errors to the solution. And comparing to the detecting promising areas method in [16], the DR-CSA does not need to presume probabilistic models for each problem, so that the undesirable performance in [16] cause by presumed model not fitting the structure of the problem will not appear. One of the strengths of the DR-CSA is that it runs with historical data only and does not need any prior knowledge of the problem. This makes the DR-CSA easier to achieve in real-life applications. However there are still some aspects of DR-CSA to be improved, for example, it hasn’t been extended to many-objective and high-dimensional problems. A full discussion of future works is given in Section VI.

## B. DR-CSA algorithm procedure and computational complexity analysis

For the DR-CSA, two more steps are added into the regular CSA, show in Fig. 1 (dashed line part). One is degeneration recognizing and pre-elimination, and the other one is degeneration database storage and management.



Fig. 1. Flowchart of the CSA and DR-CSA

For the degeneration recognizing and pre-elimination step, every new matured clone is strategically compared with the database. The recognized degenerated antibodies will be eliminated immediately. The pseudo-code of this step is shown as follow:

After this degeneration recognizing and pre-elimination step, the final result (*Cnds*) is the rest of the population that survive pre-elimination. The next steps are the standard CSA evaluation and selection. And then, the selected part of the *Cnds* becomes the new generation and the rest (eliminated) part becomes the input (*elmCnds*) of the following step:

As shown, the database expands on each iteration by accumulating eliminated solutions. On large-scale problems, some proper database management can be applied to prevent overflow. For example, the data in the database with the smallest neighbourhood can be deleted.

|  |
| --- |
| **Algorithm**: Degeneration Recognizing and Pre-Elimination  **Input**: new population (*newCnds*); degeneration database (*dgBase*), which is made of degeneration antibodies (*dgBase.points*) and their respective degeneration neighbourhood radius (*dgBase.radius*).  **Output**: non-degenerated antibodies (*Cnds*). |
| 1. *nc* = the number of *newCnds*  2. *nd* = the number of *dgBase.points*  3. **for** *j* = 1 to *nc*  4. *dgMark* = 0;  5. **for** *i* = 1 to *nd*  6. *distance* = the Euclidean distance between *newCnds*(*j*) and *dgBase.points*(*i*)  7. **if** *distance* < *dgBase.radius*(*i*)  8. *dgMark* = 1;  9. **break**;  10. **end if**  11. **end for**  12. **if** *dgMark* = 0  13. Store *newCnds*(*j*) into *Cnds*  14. **end if**  15. **end for**  16. **return** *Cnds* |
| **Algorithm**: Degeneration Database Storage & Management  **Input**: eliminated antibodies (*elmCnds*); degeneration database (*dgBase*).  **Output**: renewed degeneration database (*dgBase*). | |
| 1. *ne* = the number of *elmCnds*  2. **for** *i* = 1 to *ne*  3. *radius* = the degeneration neighbourhood radius of antibody *elmCnds*(*i*), detailed calculation method is introduced in the next segment  4. Store *elmCnds*(*i*) into *dgBase.points*  5. Store *radius* into *dgBase.radius*  6. **end for**  7. **return** *dgBase* | |

According to the code structure introduced above, the computational complexities of the two algorithms that are added for the DR-CSA are and respectively. Obviously, the computational complexity is increased comparing to the regular CSA. In Degeneration Recognizing and Pre-Elimination Algorithm, is the population size which is a fixed number. is the number of all the eliminated antibodies (pre-eliminated antibodies are not included) and keeps increasing along with iteration times. In Degeneration Database Storage and Management Algorithm, the is the number of the eliminated antibodies in each iteration, it is the increment of in each iteration. is relative small and stable with certain randomness.

As a result, besides evaluation, the major computational expensiveness of the DR-CSA is dominated by and the computation time is also increasing along with iteration. In complex engineering optimization, only when the evaluation time cost is much higher than the DR-CSA algorithm itself, the proposed method is able to reduce considerable computation time. The increasing pattern of is monitored and plotted in the following experiments.

## C. Degeneration neighbourhood size calculation method

The determination of the neighbourhood size is important in the DR-CSA. If the neighbourhood size is too small, the probability of a new antibody’s falling into this neighbourhood will become too small to effectively save time. On the other hand, if the neighbourhood size is too large, the actual optimal peaks/areas may accidentally be covered by some degenerated antibody’s neighbourhood, causing the algorithm unable to locate these optimal peaks/areas. In this segment, three ways of calculating a proper neighbourhood size is introduced by using affinity as guidance.

Assuming that the optimization objective is and the definition of the affinity is

where is the complex engineering model and is the engineering input variables. The multimodal optimization task is to achieve the that minimize .

Such optimization task is commonly appeared in the engineering optimization problem. For example, when the optimization objective is a certain property index of the product/process and expected to be stable around a set value, the set value is the here.

Let be the radius (defined by Euclidean distance) of the neighbourhood area for an antibody with affinity . Three different ways of neighbourhood radius calculation method are proposed as follow:

### 1) Fixed size method

This method used a fixed value for the neighbourhood size, which means the radius of the neighbourhood is a constant. The calculation formula can be written as,

where is the set value for the neighbourhood radius. is a threshold for the affinity. The threshold is used to protect the optimal peaks/areas. A simple diagrammatic sketch of the neighbourhood size relating to affinity is shown in Fig. 2(a).

### 2) Linear increasing method

This method is a size-varied method. is increasing along with the affinity. The method is defined as,

where is parameter of the increased gradient of the . A diagrammatic sketch of this method is shown in Fig. 2(b).

### 3) Non-linear increasing method

This method is a development of method 2. It is defined as,

The parameter is an exponent. is necessary. With the exponent smaller than 1, increases faster when is small, but won’t be too large when is big. A simple diagrammatic sketch of this method is shown in Fig. 2(c).

If the optimization objective value is not known in advance for the optimization problem, these neighbourhood size calculation methods need some further improvement because the range of the affinity is unknown. The improvement method can be a recalculation of the affinity. For example, a normalization within all the known affinity (the range that have been found) would solve the problem. This can be a further improvement for the DR-CSA in the future research.

*f*

*rnbh*

*δ*

*f*

*rnbh*

*δ*

*f*

*rnbh*

*ε*

*δ*

(a)

(b)

(c)

0

0

0

Fig. 2. Sketch of three neighbourhood size methods

# IV. Experimental Studies

## A. Regular CSA multimodal optimization experiment

In complex engineering optimization problems, there may be more than one optimal solution for an objective, and sometimes the optimum can even be continuous area. So, the test function that adopted first is an altered Schaffer’s F6 function. The Schaffer’s F6 function is

The term is set to zero so that the fluctuation of the function is not dampened. Thus, there are multiple optimal (maximum) areas of the function (see Fig. 4). The feasible region is . The amount of the initial solutions is 100.

First, a pair of contrast experiment is conducted between the traditional CSA and the traditional GA. These two methods are both used to optimize the test function. In order to observe clearly, the optimization objective is set as: . The parameters of these two evolutionary algorithms are: antibody/ individual population *N*=100; number of the best solutions to be clone *n*=100; number of clone *M* = 10; iteration/generation times *I*, *G=*30; binary coding length 22; crossover probability is0.50; mutation rate is 0.01; hyper-mutation rate is0.10.

The optimization results are shown in Fig. 3 and Fig. 4. Fig. 3 gives the maximum and mean values of the entire population of the first 20 generations/iterations. Both of the traditional GA and traditional CSA can reach the maximum solution rapidly (within 3 generations) and GA maybe faster. However, since the traditional GA only focuses on selecting the best solutions among the population, the mean value of the population has a certain randomness. But as to the traditional CSA, the entire population is evolving and gradually reaching the maximum solutions. So the mean value of the population also tends to reach the maximum.

Fig. 4 shows the entire population of the 30th iteration of the traditional CSA. Apparently, the CSA is able to locate all the maximum areas and spread the solutions evenly on these areas. However the traditional GA is not able to evenly spread them on all of the fittest areas like CSA. The explanation for this phenomenon is that the traditional GA always selects the individuals with better fitness. If one fittest individual appeared, it will be kept in the population till the end. And due to the crossover operation, this fittest individual will start affecting other individuals. As a result, the entire population will converge towards this fittest individual. But for the traditional CSA, each antibody is independently evolving. Since *n* = *N*, each antibody has its own clones. The selection is always among the origin with its own clones respectively. So, each antibody has the opportunity to evolve independently so that the entire population does not converge while evolving. As a result, the entire population is randomly located on the best-affinity areas at the end.

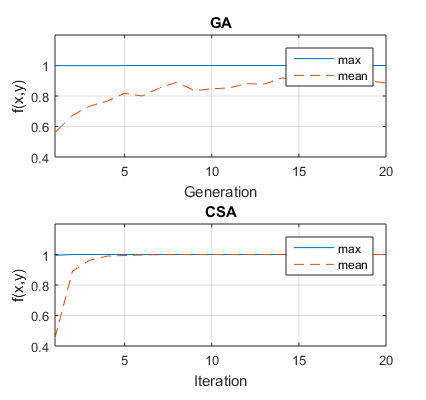


Fig. 3. Maximum and mean results of the GA and CSA experiments

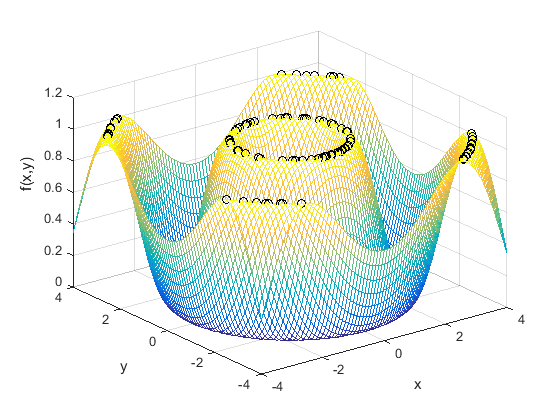


Fig. 4. Results of the 30th iteration of the CSA experiment

From these results we can see that the traditional CSA is a favorable method for solving multimodal optimization problem. However, in this experiment, the affinity evaluation that CSA carried out is ten times more than GA in order to obtain multimodal solutions. When it comes to complex time-consuming engineering problem, the large amount of evaluation times will cause a severe problem.

## B. Experiments of the DR-CSA with three different neighbourhood size methods

After proving the effectiveness of the CSA on multimodal optimization problem, a set of test experiments of the proposed DR-CSA is conducted. Meanwhile, the three neighbourhood size calculation methods are tested and compared with each other.

The test function is the same as the former experiment, so is the optimization objective and the CSA parameters. For the following experiments, to reveal the DR-CSA’s abilities of solution-evolving and time-saving, the termination criterion for iteration is set as . Three sets of parameters for the three neighbourhood size calculation methods are listed in Table I. To be fair, these parameters are roughly chosen.

Table I. Parameters for three neighbourhood size methods

|  |  |  |
| --- | --- | --- |
| Fixed Size Method | Linear Increasing Method | Non-linear Increasing Method |
| = 0.1 | = 0.3 | = 0.04 |
| = 0.01 | = 0.005 | = 0.50 |
|  |  | = 0.001 |

In order to observe how the three methods perform, a set of the same 100 random initial antibodies is given to all three experiments. A quarter of the initial antibodies (dots) and their degenerated areas (shadows) are shown in Fig. 5. The degenerated area size is calculated by the three methods respectively after a pre-evaluation of the initial antibodies’ affinities. In the following DR-CSA steps, any antibody that falls into these shadows will be regarded as degenerated.

Fig. 5 shows that the three different methods have distinctive performances. First of all, all three methods have a few antibodies which do not have any neighbourhood because of . It means that these antibodies have very good affinities. Any other antibodies which are near these antibodies may also have very good affinities so that evaluation is required to ensure the method’s accuracy. And for all the other antibodies which have an obvious degenerated neighbourhood, their neighbourhood sizes are various. For the fixed size method, as its name, all the sizes are identical. And for the linear increasing method, some antibodies have an unusually large neighbourhood because these antibodies’ affinity is relatively bad. And for the non-linear increasing method, the neighbourhood sizes are more appropriate. Its neighbourhood sizes are not too big nor too small, within a reasonable range.

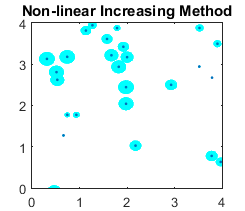
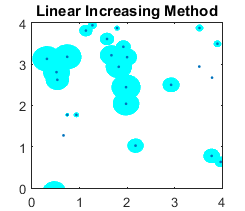
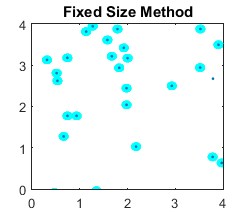
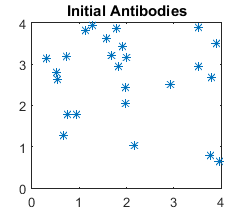


Fig. 5. Initial antibodies and initial degenerated areas of the three methods

After initialization, as the iteration continues, more and more antibodies will be eliminated and then enter the degeneration database. Consequently, the degenerated areas will gradually expand so that the probability of a degenerated antibody being recognized will gradually increase. As a result, the DR-CSA method will be more and more efficient as the iteration continues.

In order to evaluate the timesaving ability of the DR-CSA, an evaluating indicator is introduced. Since the DR-CSA is aimed to save time by avoiding a number of evaluations, two evaluating indexes for timesaving are defined as

where is the number of actual evaluation times using the DR-CSA, and is the number of the evaluation times using the traditional CSA. So and here represents the percentage of the time cost of the DR-CSA over the time cost of the traditional CSA. But the difference is, stands for the overall time cost of all iterations together, and stands for the time cost of each single iteration. Therefore, a value of and a curve of can both help understanding the method’s performance of timesaving.

Meanwhile, the database usage, the mean value of the entire population, and the iteration times to reach the target are also monitored to show the optimization performance. Due to the randomness in heuristic algorithm, all of the following experiment results are an average result of 10 times repetition of the same experiment.

As shown in Fig. 6, the curves of the population’s mean value of the three methods are very similar. And the iteration times are also very close to each other (around 13-15 times). As a result, the three different methods do not affect the method’s accuracy and effectiveness.

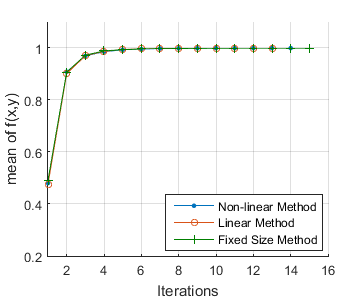


Fig. 6. Test function mean value of three methods

However, when it comes to the time cost and database usage, the three different methods have some obvious differences. Table II provides the index of the three methods. For timesaving, under the listed parameters, the fixed size method is better than the linear method and the non-linear method is better than the fixed size method.

Table II. Experimental results of the three neighbourhood size methods

|  |  |  |  |
| --- | --- | --- | --- |
| Methods | Iterations times | Database usage | Ω |
| Fixed Size | 14.9 | **2913.3** | 50.13% |
| Linear | **13.0** | 3882.3 | 61.81% |
| Non-linear | 13.8 | 3745.4 | **38.98%** |

Fig. 7 shows the curves of index . For the initial antibodies, the entire population needs to be calculated because there is no antibody in the eliminated database yet. So the time cost for the first iteration is always 100%. Then, after the first selection, a lot of antibodies are eliminated and stored to recognize degenerated antibodies, so the time cost is decreasing. From Fig. 7, we can see that the time cost is dropping remarkably within the first few iterations. And then the time cost is changing slowly with certain randomness. This is because, after a few iterations, a big part of the population are gathering around the best-affinity areas and protected from pre-elimination by . And when most of the population are gathering in such best areas, the mean value of this iteration is close to the optimum. As a result, an inverse similarity of Fig. 7 and Fig. 6 can be observed. According to and , under the chosen parameters, the non-linear increasing method has a better performance on timesaving. And looking at the final database usage in Table II and its changing curve in Fig. 8, the data storage is also increasing fast at first, and then gradually slowing down.

The fixed sized method’s data storage is smaller than the other two methods. This is because the neighbourhood size of the fixed size method is not continuously changing starting from 0. So it doesn’t have tiny neighbourhood cases and the amount of the data storage is relatively small. And the other two methods, the non-linear method is better than the linear method because the neighbourhood size of the non-linear method is increasing faster than the linear method when close to 0. So the non-linear method has less tiny neighbourhood cases than linear method.

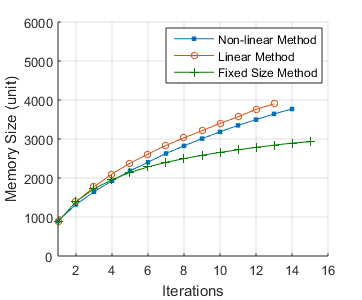
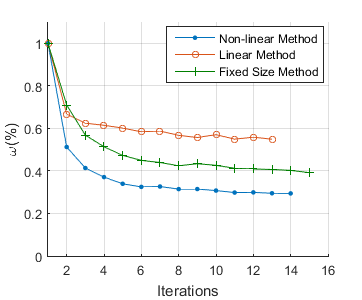


Fig. 7. Curves of index Fig. 8. Database usage

Another way to observe the DR-CSA’s performance is to see if the degenerated area can gradually expand and cover most of the not-best-affinity areas and not covering the best-affinity areas. Therefore Fig. 9 gives the degenerated area of the 13th iteration in non-linear increasing method experiment and a partial enlarged detail. As shown, the shadows cover almost all of the degenerated areas and does not cover the optimal areas. As a result, the method is proved to be valid.

## C. Experiments of the non-linear increasing method with different parameters

The neighbourhood size determination method is important for the DR-CSA, and its parameters affect the method’s performance. In this segment, a set of test experiments is conducted to show how the method is affected by these parameters. And the method chosen for this experiment is the non-linear increasing method.

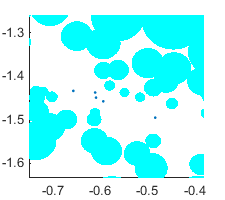
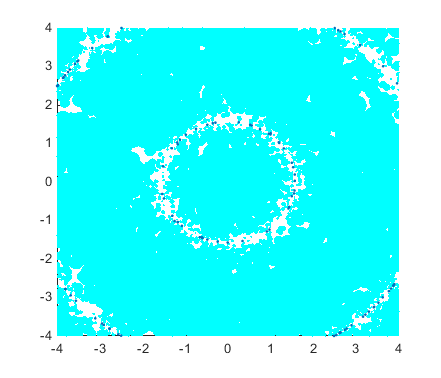


Fig. 9. Degenerated area of the 13th iteration of non-linear method

For the non-linear increasing method, there are three parameters, i.e., , , and . The parameter values chosen for the experiment are listed on the left side of Table III. And the experiments results are listed on the right side. Each result is an average of 10 times of the same experiment.

First of all, all the experiments reach the termination criterion around 12-15 iterations. The accuracy of the optimization is not much affected by the parameters. Then, several comparison curves of are selected to form plots that demonstrate the influences of the parameters in the method.

In Fig. 10, the three curves (experiment 1, 2, and 3) have the same and but different . According to the definition of the method, is the general proportion of the neighbourhood size so it determines the average neighbourhood size. As shown in the Fig. 10, the time cost decreases with an increasing .

In Fig. 11, the three curves (experiment 3, 6, and 9) have the same and but different . When and are the same, a smaller corresponds to a lower time cost. That is because as a less-than-one exponential term, the smaller is, the faster the neighbourhood size grows when the affinity is close to zero. However, unlike , only affects part of the neighbourhood size (when affinity is small). In fact, the main function of is to prevent the neighbourhood size from becoming too large and causing a failure of the algorithm.

In Fig. 12 the three curves (experiment 1, 10, and 11) have the same and but different . protects the optimal areas by keeping a zero neighbourhood zone around the optimal areas. So the performances of the first few iterations is not affected by because the population average is not close to optimum. The three curves are mostly overlapped. But after that, most of the population is evolved and entered the optimal nearby area, the starts to affect the curves. As shown in Fig. 12 and Table III, a relatively smaller helps improving the computation speed but bigger helps decreasing data storage.

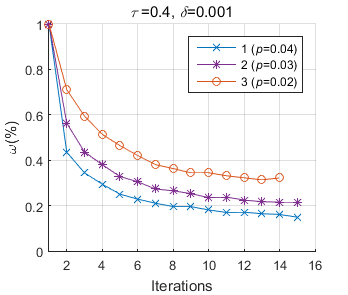
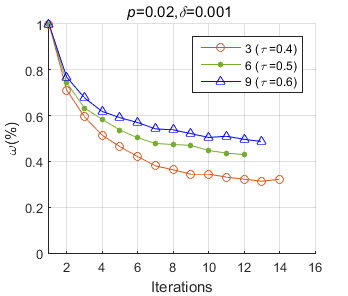
 

Fig. 10. Comparison of Exp. 1, 2, & 3 Fig. 11. Comparison of Exp. 3, 6, & 9

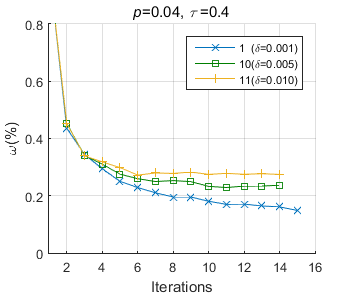


Fig. 12. Comparison of exp. 1, 10, and 11

Table III. Experiment parameters and their results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Parameters | | | Results | | |
|  |  |  | Iterations times | Database usage | Ω |
| 1 | 0.04 | 0.4 | 0.001 | 14.9 | 3106.0 | **27.96%** |
| 2 | 0.03 | 0.4 | 0.001 | 14.7 | 3833.7 | 34.86% |
| 3 | 0.02 | 0.4 | 0.001 | 13.7 | 4915.3 | 46.76% |
| 4 | 0.04 | 0.5 | 0.001 | 14.0 | 3782.9 | 38.73% |
| 5 | 0.03 | 0.5 | 0.001 | 13.1 | 4286.2 | 45.77% |
| 6 | 0.02 | 0.5 | 0.001 | **12.3** | 5068.3 | 56.00% |
| 7 | 0.04 | 0.6 | 0.001 | 13.7 | 4099.4 | 46.14% |
| 8 | 0.03 | 0.6 | 0.001 | 13.4 | 4589.0 | 51.52% |
| 9 | 0.02 | 0.6 | 0.001 | 13.4 | 5089.5 | 59.98% |
| 10 | 0.04 | 0.4 | 0.005 | 13.6 | 2759.9 | 32.93% |
| 11 | 0.04 | 0.4 | 0.010 | 14.0 | **2548.4** | 35.10% |

At last, by comparing all the experiment results in Table III, it is pleasant to find that when parameters are properly chosen, the time saving of the DR-CSA method is more than 82%.

## D. Experiments on multiple multimodal test functions comparing with different optimization algorithms

In order to prove the effectiveness of the proposed DR-CSA method, the method was tested on eight different benchmark multimodal functions, comparing with several well-developed heuristic optimization methods. All of the chosen multimodal functions are frequently used as test function. They are Schaffer’s F6 function, Branin function, Himmelblau function, Rosenbrock function, Six-Hump Camel Back function, Cross-in-Tray function, Shubert function, and Rastrigin function.

The eight test functions cover various kinds of multimodal functions. Within the designated solution space, the Schaffer’s F6 function contains several global optimal areas (see Fig. 4); the Branin and the Himmelblau function have several global optimal peaks and no local peak; the Rosenbrock function has one flat global peak and no local peak; the Six-Hump Camel Back function has 2 global peaks and 4 local peaks; the Cross-in-Tray and the Rastrigin function have a few global peaks and a lot of local peaks; the Shubert function has 2 global peaks, a few local peaks, and a lot of very small fluctuations.

The chosen comparison methods are all multimodal optimization methods. They are regular CSA, Niching GA, Niching PSO, and opt-aiNet [26]. Parameters of these methods for the following test experiments are listing in Table IV.

Table IV. Parameters of each comparison method

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CSA/DR-CSA | | | Niching GA | | Niching PSO | | opt-aiNet | |
| *N* | 100 | *N* | | 100 | *N* | 100 | *N* | 100 |
| *I* | 30 | *I* | | 200 | *I* | 200 | *I* | 300 |
| *L* | 22 | *L* | | 22 | *wmax* | 1 | - | - |
|  | 0.1 |  | | 0.05 | *wmin* | 0 | - | - |
| *M* | 10 | cross-over | | 0.9 | *c1* | 2 | *M* | 10 |
|  | 0.03 | distance | | 0.01 ~0.1 | *m* | 0.1 | *τ* | 0.1 |
|  | 0.5 | *n* | 3 | *d* | 0.4 |
|  | 0.001 | - | | - | - | - | *β* | 50 |

Listed in Table IV, *N* is the population size, *I* is the maximum iteration (generation), *L* is the binary code length, *M* is the clone amount, is the mutation rate, is the hyper-mutation rate, and all other parameters are with each method itself. The distance parameter of Niching GA is chosen within 0.01 - 0.1 according to function’s range. All of the parameters are chosen carefully so that all of the methods perform well.

In the neighbourhood size calculation of the DR-CSA, in order to employ the same parameters (, , and ) to different test functions, this part carries out an additional normalization of the function and the solution space. But normalization is not necessary if the parameters can chose freely. The experiments calculated the best result error (the error between experimental and theoretical best), mean value error (the error between population mean and theoretical best), the standard deviation of the final population, and the number of evaluation times.

Amount these results, the number of evaluation times reflects the computation speed of the optimization method. Besides the numerical results, the multimodal performance (including the number of the optimal peaks/areas that found and the distribution of the solutions) of each method is observed and recorded. As a computation speed improving method, these results are the most important results.

To evaluate the evaluation times, besides the termination criterion of mean value error , another termination criterion which is the maximum iteration times is added. This criterion is adopted because some optimization methods will converge before satisfying the first criterion, so without a second criterion of maximum iteration times the computation will run forever and the judgment of its computation speed will not be fair. The results of the comparison test experiments are listed in Table V. Every listed result is an average value of the same experiment repeating ten times.

As shown in Table V, the computation speed (the number of fitness evaluation times) of the DR-CSA is clearly better than all other methods. The best result appeared in the Rastrigin function experiment, the evaluation times of the DR-CSA is only **11.5%** of opt-aiNet and **18%** of the regular CSA. While the computation speed is highly improved, all the other optimization results are also comparable with other methods.

When look at the multimodal optimization performance, the DR-CSA is the same as the regular CSA. They are able to locate all of the global optimal results and if the optimal result is an area (like Schaffer’s F6 function), these two methods can evenly (with randomness) distribute the results on this area. If the function has suboptimal peaks which are almost as good as the global peaks, the two methods can also locate such suboptimal peaks, such as the Shubert function and the Rastrigin function. The multimodal optimization ability of Niching GA is relatively weaker than other methods. If there are many peaks in the function, the result of Niching GA may miss several of the peaks. Besides, the distribution of the final population on the peaks of Niching GA is usually not even. The peak-locating ability of Niching PSO is not very stable and the result distribution is not even either. The opt-aiNet method has an excessive peak-locating ability, it can locate all of the peaks in the function, including global and local peaks and even the very small fluctuations. If there is no local peaks in the function, such as the Himmelblau function, this opt-aiNet method has an excellent result. But if the function has many very small peaks, such as the Shubert function, this method’s result will be very unsatisfactory.

When look at the result of standard deviation, the Niching GA usually has the smallest standard deviation due to its convergence property. However the strong convergence affects the distribution ability in multimodal optimization problem. Considering both the standard deviation and the distribution performance, the DR-CSA and the CSA have a better overall performance.

As to the best result and the mean result, DR-CSA is not always better than other methods. This is because in order to reduce calculation time, DR-CSA reduces a large number of evaluation times. And reducing evaluation times leads to a lowering of the chance to locate the optimal. In other words, DR-CSA is sacrificing accuracy for speed improvement. An accuracy index can be calculated to help with comparing the accuracy between DR-CSA and all the methods. The index is defined as:

where, - the Error of Best Result of DR-CSA; - best (smallest) Error of Best Result amount all methods; - worst (biggest) Error of Best Result amount all methods. So indicates the percentage of accuracy that DR-CSA sacrificed. means the DR-CSA is the best (no accuracy is sacrificed) method and means the DR-CSA is the worst. Based on the experiment data listed in Table IV, for 8 experiments, the are: 20.73%, 0%, 4e-4%, 13.51%, 3e-4%, 3.6%, 0%, and 2.95e-5%. As shown in the results, only experiment 1 (20.73%) and experiment 4 (13.51%) have relatively unsatisfactory accuracy. However in these two experiments, the DR-CSA can save about 68.9% and 65.6% of the calculation time respectively. And in other experiments, DR-CSA sacrificed almost no accuracy but saved up to 88.5% of the calculation time.

In conclusion, for most of the problems, the proposed DR-CSA is able to improve the optimization speed significantly while maintaining a favorable optimization result.

Table V. Experiment results of different optimization methods on multimodal benchmark functions

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Optimization Function | Objective | Algorithm | Error of Best Result | | Error of Mean Result | | Standard  Deviation | Evaluation  Times | | Located  (Real) Peaks |
| Schaffer’s F6  *x, y*[-4,4] | maximum  (1) | DR-CSA | 3.98e-11 | 6.91e-5 | | 2.77e-4 | | **6344** | 5 (5) | |
| CSA | 1.92e-10 | **5.30e-5** | | **2.14e-4** | | 16400 | 5 (5) | |
| Niching GA | 2.54e-13 | 7.28e-5 | | 2.78e-4 | | 18830 | 5 (5) | |
| Niching PSO | **0** | 2.95e-2 | | 9.59e-2 | | 20400 | 5 (5) | |
| opt-aiNet | 8.39e-14 | 8.04e-5 | | 3.68e-4 | | 20737.3 | 5 (5) | |
| Branin  *x,y*[-10,10] | minimum  (0.39789) | DR-CSA | **3.23e-6** | 0.4466 | | 1.0605 | | **11537** | 3 (3) | |
| CSA | 3.67e-6 | **0.4187** | | 1.1174 | | 30000 | 3 (3) | |
| Niching GA | 5.56e-5 | 0.2059 | | **0.1479** | | 20100 | 2.8 (3) | |
| Niching PSO | 4.42e-3 | 1.6047 | | 1.2145 | | 20400 | 3.9 (3) | |
| opt-aiNet | 4.19e-6 | 0.5770 | | 0.5026 | | 63777 | 3 (3) | |
| Himmelblau  *x, y*[-6,6] | minimum  (0) | DR-CSA | 5.47e-6 | 0.8392 | | 1.7482 | | **14877** | 4 (4) | |
| CSA | 1.04e-5 | 0.7981 | | 1.6689 | | 30000 | 4 (4) | |
| Niching GA | 1.70e-4 | 0.7273 | | 0.5303 | | 20100 | 3.6 (4) | |
| Niching PSO | 1.02e-2 | 3.5099 | | 3.4065 | | 20400 | 4 (4) | |
| opt-aiNet | **1.39e-6** | **9.44e-5** | | **8.71e-5** | | 30094.4 | 4 (4) | |
| Rosenbrock  *x*[-2,2]  *y*[-1,3] | minimum  (0) | DR-CSA | 2.56e-4 | 0.7689 | | 1.0482 | | **11539** | 1 (1) | |
| CSA | 4.85e-4 | 0.8049 | | 1.0614 | | 30000 | 1 (1) | |
| Niching GA | 3.07e-4 | 0.6198 | | **0.5402** | | 20100 | 1 (1) | |
| Niching PSO | 1.89e-3 | 1.2478 | | 1.7092 | | 20400 | 1 (1) | |
| opt-aiNet | **8.39e-7** | **0.2787** | | 0.5846 | | 33510 | 1 (1) | |
| Six-Hump Camel Back  *x*[-2,2]  *y*[-1,1] | minimum  (-1.0316) | DR-CSA | 6.65e-6 | 0.0227 | | 0.0328 | | **9094** | 2 (2) | |
| CSA | **6.51e-6** | **0.0218** | | 0.0336 | | 30000 | 2 (2) | |
| Niching GA | 2.75e-5 | 0.0266 | | **0.0183** | | 20100 | 2 (2) | |
| Niching PSO | 4.74e-4 | 0.4116 | | 0.3475 | | 20400 | 3.9 (2) | |
| opt-aiNet | 2.84e-5 | 0.5244 | | 0.9400 | | 29032.4 | 6 (2) | |
| Cross-in-Tray  *x, y*[-4,4] | minimum  (-2.0626) | DR-CSA | 1.07e-6 | 9.06e-4 | | 2.14e-3 | | **11205** | 4 (4) | |
| CSA | **1.04e-6** | 8.81e-4 | | 1.97e-3 | | 30000 | 4 (4) | |
| Niching GA | 1.86e-6 | **2.62e-4** | | **1.82e-4** | | 20100 | 3.8 (4) | |
| Niching PSO | 1.66e-6 | 2.14e-2 | | 2.97e-2 | | 20400 | 4.7 (4) | |
| opt-aiNet | 1.87e-6 | 7.91e-2 | | 9.76e-2 | | 32342.8 | 13.1 (4) | |
| Shubert  *x, y*[-2,2] | minimum  (-186.73) | DR-CSA | **1.54e-4** | **13.03** | | 24.99 | | **6561** | 4.7 (2) | |
| CSA | 2.01e-4 | 13.04 | | 25.38 | | 30000 | 4.8 (2) | |
| Niching GA | 2.15e-3 | 13.55 | | **10.15** | | 20100 | 1.7 (2) | |
| Niching PSO | 9.06e-2 | 112.17 | | 51.64 | | 20400 | 13.4 (3) | |
| opt-aiNet | 6.02e-4 | 130.76 | | 51.09 | | 21645 | 31.4 (2) | |
| Rastrigin  *x, y*[-5 5] | minimum  (0) | DR-CSA | 1.39e-5 | 2.2656 | | 2.2853 | | **5403** | 18.5 (1) | |
| CSA | 6.75e-5 | **2.0638** | | 1.9986 | | 30000 | 17.3 (1) | |
| Niching GA | **1.32e-7** | 2.1829 | | **0.9085** | | 20100 | 9 (1) | |
| Niching PSO | 0.4797 | 6.5484 | | 3.7553 | | 20400 | 27.2 (1) | |
| opt-aiNet | 2.95e-4 | 9.4255 | | 5.5570 | | 46925.4 | 58.9 (1) | |

# V. Wet Spinning Coagulating Process Modeling and Optimization

## A. Wet Spinning Coagulating Model

Wet spinning is one of the most traditional producing methods of polymer fibers. This process is used for polymers that need to be dissolved in a solvent to be spun. It is the key processes during the whole fiber production [27, 28].

A spinneret is submerged in a chemical bath where the dissolved polymer is extruded. The polymer solution is a mix of polymer, solvent, and non-solvent (coagulant). The bath liquid contains mostly the coagulant for the polymer fibers to precipitate and solidify. A diagrammatic sketch of a polymer fiber’s sectional view is shown in Fig. 13. The axis is the direction from the spinneret to the take-up wheel along bath. The axis is the direction from the fiber center to the outside.

is the velocity at each point of the fiber. is the outer radius of the fiber. As the polymer solution gradually solidifies, the gel phase grows from the outside to the center. is the radius of solution-gel interface. The solvent and non-solvent are diffusing between the fiber and the bath as the polymer moves inside the bath. The concentrations of solvent and non-solvent inside the fiber are and .

*r*

*z*

*R*(*z*)

Solution

Solvent

Non-solvent

*V*(*z*)

*R*0(*z*)

Gel

Fig. 13. Enlarged diagrammatic sketch of polymer fiber’s sectional view

By studying the physical relationships between the variables and process parameters, the mechanism of the coagulating process can be divided into four major parts: 1) double diffusion; 2) phase inversion; 3) mass balance; 4) force balance. Details of the model please see [29-32]. By combing all four sub-models, the coagulating model is written as:

As we can see, this coagulating mechanism model is quite complicated. It contains two PDEs, one ODE, one program block, and one equation. The relation among the four sub-models is shown in Fig. 14. Due to the complicatedness, the computation of this model is relatively time-consuming.

Double Diffusion

Phase Inversion

Mass Balance

Force Balance

*D*

*CS*

*CNS*

*R*

*R0*

*V*

*V*

*V*

Fig. 14. Structure block of coagulating mechenism model

## B. Numerical Computation Method for Wet Spinning Coagulating Model

The equations in wet spinning coagulating model need to be solved numerically. It is tricky when the two PDEs and one ODE need to be combined and calculated simultaneously.

One way to combine PDE with ODE is to adopt the Method of Lines (MOL). The basic idea of the MOL is to replace the spatial (boundary-value) derivatives in the PDE with algebraic approximations. Once this is done, the spatial derivatives are no longer stated explicitly in terms of the spatial independent variables. Thus, in effect, only the initial-value variable, typically time in a physical problem, remains. With only one remaining independent variable, there is a system of ODEs that approximate the original PDE [33]. The details of the MOL method to change PDE into ODE can be found in [33].

## C. Model Numerical Computation Results

The polymer wet spinning coagulating system that is studied in this paper is the PAN-DMSO-water system. Which means in the process the polymer is polyacrylonitrile, the solvent is dimethyl sulphoxide, and the non-solvent is water. According to [30-31, 34], coefficients of this PAN-DMSO-water system are adopted. The numerical computation of the model is conducted by MATLAB (Version R2014b).

Some of the model computation results are shown in Fig. 15 and Fig. 16. Fig. 15 shows the concentration curves of the solvent and the non-solvent inside the fiber from the beginning of the bath to the end. The different curves are the different layer-positions inside the fiber relating to different radiuses. The solvent inside the fiber is gradually reducing, and the outer layer (lower curves) is reducing faster than the inner layer (upper curves). At the same time, the non-solvent is gradually increasing insider the fiber. The outer layer is increasing faster than the inner layer. Fig. 16 shows the results of fiber radius and the solution-gel interface radius. As shown in the figure, after the point *z* = 0.48 the fiber is fully gelled. The point is named***fully gelled point***.

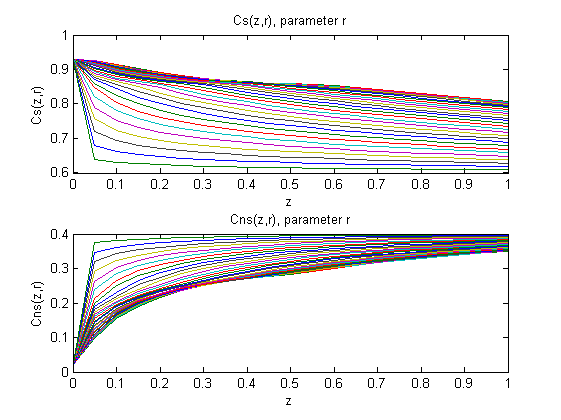
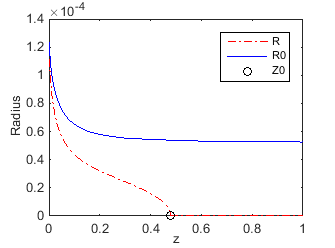
 

Fig. 15. Concentration of solvent Fig. 16. Curves of fiber radius and

and non-solvent solution-gel interface radius

The computer environment of this experiment is 3.40 GHz Intel Core i7-3770 CPU, 8.00 GB RAM, and 64-bit operating system. The overall time cost of one complete computation is 2.93 seconds. If an optimization evaluates this model thousands of times, the overall time cost would be days.

## D. Wet Spinning Coagulating Process Optimization Based on DR-CSA

By observing and studying the model outcomes, the fully gelled point appears to be a significant result. The outcome of this point is the result of all the process parameters such as solvent diffusivity, non-solvent diffusivity, polymer viscosity, bath concentration, fiber concentration, fiber extrusion speed, polymer solution flow rate, take up speed, spinneret orifice diameter, bath length, and so on. At the same time, this fully gelled point is related to all kinds of fiber performances. It directly reflects the gelation degree. And the gelation degree can reflect the fiber’s qualities such as strength, evenness, viscoelasticity, elongation at break, et al. As a result, this fully gelled point can be a good choice as an optimization objective.

Instead of maximizing or minimizing the objective, the optimization objective of the fully gelled point should be a desired value, i.e. . But eventually, it can be converted into a minimize problem as . And when the focus is only on the output of , the system can be regarded as a multiple-input single-output (MISO) system. By a few test runs of this MISO system, it is realized that the solution of a desired output is not unique. There will be multiple different combinations of inputs that have the same output. So this optimization task is a multimodal optimization problem. Also, it is expected that its solutions are located on one or more extremal intervals instead of independent extremes. The optimization result should be a set of equally best solutions for the technicians to choose.

Due to the effectiveness of the DR-CSA in solving time-consuming multimodal optimization problem. It is adopted for the coagulating process optimization.

Within all the influencing factors of the wet spinning coagulating process mentioned above, several factors are considered as constants because they are impossible or hard to change. These factors include: specifications of the spinneret, specifications of the bath, physical properties of the polymer, solvent and non-solvent, parameters of diffusion, Reynolds number, surface tension, and so on. And then, some other controllable factors are chosen as the optimization input variables (decision variables). The chosen input variables are:

### 1) Bath liquid proportion: includes solvent concentration CS\_bath, non-solvent concentration CNS\_bath. Since CS\_bath + CNS\_bath = 100%, only one variable needs to be used as an input factor. In this experiment, the CS\_bath is chosen.

### 2) Original polymer solution proportion: includes polymer concentration Cpolymer, solvent concentration CS\_solu, and non-solvent concentration CNS\_solu. Since Cpolymer + CS\_solu + CNS\_solu = 100% and Cpolymer is a constant, CS\_solu is chosen to be the input that represents this factor.

### 3) Fiber extrusion speed V0 and polymer solution flow rate G. Since , only G is chosen as a input.

### 4) Take up speed Vend .

With the inputs all selected, the optimization objective for this experiment is set as = 0.70 (*m*). And the computing parameters are: antibody population *N* = 100; number of the best solutions to be clone *n* = 100; number of clone *M* = 10; iteration times *I =* 15; binary coding length is 22; hyper-mutation rate is0.10; parameters for non-linear neighbourhood size increasing method: =0.03, =0.6, and =0.001. Both of the CSA and DR-CSA are applied.

The max and mean values of the optimization population are shown in Fig. 17, and the curve of timesaving index is shown in Fig. 18. As shown in the figures, both of the CSA and the proposed DR-CSA is able to fulfill the task of optimizing this time-consuming wet spinning coagulating model. DR-CSA and CSA have the approximately same performance on accuracy. However, the DR- improvement results of DR-CSA are: the overall time costs index = **39.60%**; the final database usage is **5025** units. Which means the overall running time of DR-CSA is only about 40% of the traditional CSA, and it can be smaller if the parameters are optimal. The first 10 best solutions are listed in Table VI. The optimization statistical results are listed in Table VII.

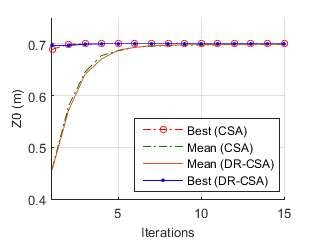
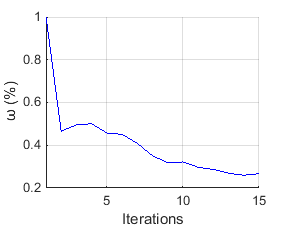
 

Fig. 17. Results of max and mean Fig. 18. Timesaving index changing

values of *Z**0* optimization curve of *Z**0* optimization

Based on the listed data, DR-CSA sacrificed 0% of the Error of Best and 8.7% of the Error of Mean in exchange for 60.4% of the calculation time. The original 5.6 hours task is shorten to a 2.3 hours task, which is a great help for this optimization task.

Table VI. First 10 best solutions of *Z**0* optimization

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *CS\_solu* (%) | *CS\_bath* (%) | *G* (*kg/s)* | *Vend* (*m/s*) | *Z0* (*m*) |
| 93.6 | 68.2 | 5.8e-07 | 0.048 | 0.70 |
| 93.7 | 58.4 | 6.6e-07 | 0.060 | 0.70 |
| 92.9 | 55.1 | 6.8e-07 | 0.068 | 0.70 |
| 93.1 | 65.6 | 6.9e-07 | 0.053 | 0.70 |
| 93.9 | 65.4 | 5.2e-07 | 0.051 | 0.70 |
| 93.7 | 64.9 | 5.1e-07 | 0.052 | 0.70 |
| 93.6 | 58.2 | 7.2e-07 | 0.060 | 0.70 |
| 92.6 | 60.2 | 6.6e-07 | 0.063 | 0.70 |
| 94.0 | 68.4 | 6.6e-07 | 0.046 | 0.70 |
| 93.3 | 60.8 | 6.1e-07 | 0.059 | 0.70 |

Table VII. Statistical results of *Z**0* optimization

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *Error of Best* | *Error of Mean* | *Std* | *Evaluate Times* | *Running Time (s)* |
| CSA | 9.06e-7 | **4.51e-4** | **8.34e-4** | 15000 | 20185.6 |
| DR-CSA | **8.53e-7** | 4.94e-4 | 9.62e-4 | **5936** | **8129.2** |

# VI. Conclusions

In this paper, a degeneration recognizing clonal selection algorithm is proposed for time-consuming multimodal optimization problems. The basic idea of this method is to utilize the eliminated solutions to identify and eliminate the degenerated solutions before evaluation operation. As a result, a number of the solutions no longer need to be evaluated by objective function so that the overall calculation time is reduced. The novelty of this idea lies in the utilizing of the eliminated solutions (which are usually neglected) to improve calculation speed. In the future, this idea is promising to be extended and applied on other metaheuristic optimization algorithms. One major contribution of this paper is that it opens up a new possibility for solving time-consuming metaheuristic optimization problems.

The method is effective because it is designed specifically to target the CSA’s characteristics, namely, clone and hyper-mutation operations. In order to explore the affinity landscape for locating the best solutions, the CSA uses clone and hyper-mutate operations to create a large amount of mutations. Within these mutations, there are evolved solutions together with degenerated solutions. And the probability of the degeneration is usually (after the first few iterations) a lot higher than the evolution. As a result, the DR-CSA is able to save a considerable computing time, benefiting from recognizing and pre-eliminating these high-probability degenerated solutions. In this paper, the method is tested by a group of test function experiments and then applied to a real-world engineering problem, which is the optimization of the wet spinning coagulating process. As shown in the results, the DR-CSA is well effective in saving calculation time.

There are still a few problems of the DR-CSA that worth further studying in our next step. As for now, our method is mainly focused on reducing evaluation times in the degenerated area to improve speed. In our next step, refinements of the searching method for the non-degenerated area can be made. So that the accuracy of the method can be further improved. Also, since the proposed method requires a relatively large database, some proper searching, filtering, and managing method for database which can increase the method’s efficiency are worth study. In addition, the extension of DR-CSA to multi- and many- objective optimization, binary optimization, and high-dimensional optimization problems are worth further research.

# References

[1] Rao S S, and Rao S S. Engineering optimization: theory and practice [B]. John Wiley & Sons, 2009.

[2] Chai T Y, Jin Y C, and Sendhoff B. Evolutionary complex engineering optimization: opportunities and challenges [J]. IEEE Computational Intelligence Magazine, 2013, 8(3): 12-15.

[3] Jin Y C and Branke J. Evolutionary optimization in uncertain environments—A survey [J]. IEEE Transactions on Evolutionary Computation, 2005, 9(3): 303–317.

[4] Jin Y C. Surrogate-assisted evolutionary computation: Recent advances and future challenges [J]. Swarm and Evolutionary Computation, 2011, 1(2): 61–70.

[5] Marler R T and Arora J S. Survey of multi-objective optimization methods for engineering [J]. Structural and Multidisciplinary Optimization, 2004, 26(6): 369-395.

[6] Chai T Y. Challenges of optimal control for plant-wide production processes in terms of control and optimization theories [J]. ACTA Automatica Sinica, 2009, 35(6): 641-649.

[7] Liang X, Ding Y S, Wang Z D, Hao K R, Hone K, and Wang H P. Bidirectional optimization of the melting spinning process [J]. IEEE Transactions on Cybernetics, 2014, 44(2): 240-251.

[8] Woldemariam K M and Yen G G. Vaccine-enhanced artificial immune system for multimodal function optimization [J]. IEEE Transactions on Cybernetics, 2010, 40(1): 218-228.

[9] Das S, Maity S, Qu B Y, and Suganthan P N. Real-parameter evolutionary multimodal optimization—A survey of the state-of-the-art [J]. Swarm and Evolutionary Computation, 2011, 1(2): 71-88.

[10] Roy S, Islam S M, Das S, and Ghosh S. Multimodal optimization by artificial weed colonies enhanced with localized group search optimizers [J]. Applied Soft Computing, 2013, 13(1): 27-46.

[11] Deb K. An efficient constraint handling method for genetic algorithms [J]. Computer Methods in Applied Mechanics and Engineering, 2000, 186(2): 311-338.

[12] Pétrowski A. A clearing procedure as a niching method for genetic algorithms [C]. Proceedings of IEEE International Conference on Evolutionary Computation 1996, 1996, 798-803.

[13] Goldberg D E and Richardson J. Genetic algorithms with sharing for multimodal function optimization [C]. Proceedings of the Second International Conference on Genetic Algorithms, Hillsdale, NJ: Lawrence Erlbaum, 1987, Genetic algorithms and their applications: 41-49.

[14] Gao W, Yen G G, and Liu S. A cluster-based differential evolution with self-adaptive strategy for multimodal optimization [J]. IEEE Transactions on Cybernetics, 2014, 44(8): 1314-1327.

[15] Ren Z G, Zhang A M, Wen C Y, and Feng Z R. A Scatter learning particle swarm optimization algorithm for multimodal problems [J]. IEEE Transactions on Cybernetics, 2014, 44(7): 1127-1140.

[16] Yang P, Tang K, Lu X. Improving estimation of distribution algorithm on multimodal problems by detecting promising areas [J]. IEEE Transactions on Cybernetics, 2014, 45(8): 1438-1449.

[17] Zhang G, Li Y. A memetic algorithm for global optimization of multimodal nonseparable problems [J]. IEEE Transactions on Cybernetics, 2015, 46(6): 1375-1387.

[18] Hui S, Suganthan P N. Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization [J]. IEEE Transactions on Cybernetics, 2015, 46(1): 64-74.

[19] De Castro L N and Von Zuben F J. Learning and optimization using the clonal selection principle [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(3): 239-251.

[20] Chen J J, Ding Y S, Jin Y C, and Hao K R. A synergetic immune clonal selection algorithm based multi-objective optimization method for carbon fiber drawing process [J]. Fibers and Polymers, 2013, 14(10): 1722-1730.

[21] Bernardino H S, Barbosa H J C, and Fonseca L G. Surrogate-assisted clonal selection algorithms for expensive optimization problems [J]. Evolutionary Intelligence, 2011, 4(2): 81-97.

[22] Dasgupta D. Advances in artificial immune systems [J]. IEEE Computational Intelligence Magazine, 2006, 1(4): 40-49.

[23] Hofmeyr S A and Forrest S. Architecture for an artificial immune system [J]. Evolutionary Computation, 2000, 8(4): 443-473.

[24] Zhang W, Yen G G, and He Z. Constrained optimization via artificial immune system [J]. IEEE Transactions on Cybernetics, 2014, 44(2): 185-198.

[25] Luh G C and Chueh C H. Multi-modal topological optimization of structure using immune algorithm [J]. Computer Methods in Applied Mechanics and Engineering, 2004, 193(36), 4035-4055.

[26] De Castro L N, Timmis J. An artificial immune network for multimodal function optimization[C]. Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), IEEE, 2002:699-704.

[27] Ding Y S, Liang X, Hao K R, and Wang H P. An intelligent cooperative decoupling controller for coagulation bath in polyacrylonitrile carbon fiber production [J]. IEEE Transactions on Control Systems Technology, 2013, 21(2): 467-479.

[28] Xiao L, Ding Y S, Wang Z D, et al. Bidirectional optimization of the melting spinning process [J]. IEEE Transactions on Cybernetics, 2014, 44(2): 240-51.

[29] Ding Y S, Xu N, Ren L H, and Hao K R. Data-driven neuroendocrine ultrashort feedback-based cooperative control system [J]. IEEE Transactions on Control Systems Technology, 2015, 23(3): 1205-1212.

[30] Kalabin A L and Pakshver E A. Simulation of kinetics and heat and mass transfer for the spinning of chemical fibers from polymer solutions [J]. Theoretical Foundations of Chemical Engineering, 1997, 31(6): 520-525.

[31] Walczak Z K. Processes of fiber formation [B]. Elsevier, 2002.

[32] Yilmaz L and McHugh A J. Analysis of nonsolvent–solvent–polymer phase diagrams and their relevance to membrane formation modeling [J]. Journal of Applied Polymer Science, 1986, 31(4): 997-1018.

[33] Vande Wouwer A, Saucez P, and Schiesser W E. Simulation of distributed parameter systems using a Matlab-based method of lines toolbox: chemical engineering applications [J]. Industrial & Engineering Chemistry Research, 2004, 43(14): 3469-3477.

[34] Sano Y. Drying behavior of acetate filament in dry spinning [J]. Drying Technology, 2001, 19(7): 1335-1359.