On Diversity and Artificial Immune Systems: Incorporating a Diversity Operator into aiNet

Paul S. Andrews[1] and Jon Timmis[2]

[1] Department of Computer Science, University of York, UK
psa@cs.york.ac.uk
[2] Departments of Electronics and Computer Science, University of York, UK
jtimmis@cs.york.ac.uk

Abstract. When constructing biologically inspired algorithms, important properties to consider are openness, diversity, interaction, structure and scale. In this paper, we focus on the property of diversity. Introducing diversity into biologically inspired paradigms is a key feature of their success. Within the field of Artificial Immune Systems, little attention has been paid to this issue. Typically, techniques of diversity introduction, such as simple random number generation, are employed with little or no consideration to the application area. Using function optimisation as a case study, we propose a simple immune inspired mutation operator that is tailored to the problem at hand. We incorporate this diversity operator into a well known immune inspired algorithm, aiNet. Through this approach, we show that it is possible to improve the search capability of aiNet on hard to locate optima. We further illustrate that by incorporating the same mutation operator into aiNet when applied to clustering, it is observed that performance is neither improved nor sacrificed.

1 Introduction

The development of algorithms in the field of Artificial Immune Systems (AIS) has grown rapidly over the past 10 years. It is only recently, however, that the research methodologies employed to design AIS have begun to be examined [1, 2, 3]. Highlighted by each of these methodologies, is the need to consider carefully the target applications and biological inspirations when designing AIS. By doing this, it is hoped that the designer will maximise their chance of producing an improved AIS. As a preliminary piece of work, we revisit an existing AIS called aiNet [4], and consider how it introduces diversity into its population of candidate solutions for the task of function optimisation. We then incorporate into aiNet, a diversity operator that has been tailored for function optimisation and inspired by an immune process called receptor editing. Function optimisation experiments are carried out to assess the performance of the algorithm with, and without, the diversity operator. The results show improved performance with the diversity operator. This process of adapting a bio-inspired algorithm to a particular problem, via the introduction of additional functionality such as the diversity operator outlined in this paper, is a common task within the evolutionary computation community. However, the adaptations made to improve

B. Apolloni et al. (Eds.): WIRN/NAIS 2005, LNCS 3931, pp. 293–306, 2006.

performance in one problem domain may not be suitable when applied to a different domain. Thus, consideration must again be taken when attempting to re-apply specialised algorithm modifications for different purposes. To highlight this, we incorporate the diversity operator into a version of aiNet designed for data clustering. Experiments show that the performance of aiNet for clustering is not affected by the diversity operator for the chosen test case. This results is to be expected as the diversity operator, which was designed for a different problem area, was copied blindly without consideration for the clustering application.

2 Artificial Immune Systems

AIS have been defined by de Castro and Timmis [1] as being:

> "adaptive systems, inspired by theoretical immunology and observed immune functions, principle and models, which are applied to problem solving."

This very general definition reflects the fact that the term AIS is used to describe a collection of systems taking inspiration from different ideas and processes in immunology. Typically, however, most AIS have been inspired by the notion that the immune system protects the body from potentially harmful micro-organisms called pathogens, and by the immune mechanisms used to achieve this. One such mechanism utilises the production of molecules called antibodies (Ab) that can recognise other molecules, called antigens (Ag), via a process of chemical binding. An Ag can either be part of a pathogen or of the organism's own cells, known as self Ag. The recognition of self Ag by Ab in the body is considered undesirable, and so additional immune mechanisms exist ensuring only Ab that recognise non-self Ag are produced. In addition to taking inspiration from such immune mechanisms, AIS designers typically use the immunological terminology to describe the components of the system. Examples of AIS include those based on the selection of immune cells in thymus and the negative selection principle [5], the maturation of immune cells via the process of clonal selection [6], and the interaction of immune cells described by the immune network theory [4]. A comprehensive overview of AIS and immunology is beyond the scope of this paper and the interested reader is referred to [1] and [7] for more details.

2.1 Designing AIS

As the field of AIS has matured, a better understanding of how they work is currently being sought. From these investigations, a number of ideas have arisen concerning how AIS should be designed. de Castro and Timmis [1] introduced a layered framework for engineering AIS that identifies three basic system design elements: a representation for the components of the system, a set of mechanisms (affinity measures) to evaluate the interactions of individuals within the environment and with each other, and procedures for adaptation. Additionally, Freitas and Timmis [2] highlighted the need to consider the problem for which the AIS

is being designed, and note that the selection of a representation scheme and evaluation functions in AIS can bias the results. Consequently, when designing a suitable AIS, it needs to be tailored to the problem at hand. Similarly, Hart and Ross [8] investigated the effects of different affinity measurement mechanisms with respect to immune networks and highlight the importance of getting the shape-space and matching rule right when designing AIS. Recently, Stepney et al. [3] proposed that bio-inspired algorithms, such as AIS, are best developed within a conceptual framework. This framework takes an interdisciplinary approach to the problem, and involves designing AIS through a series of observational and modelling stages in order to identify the key characteristics of the immunological process on which the AIS will be based. The same authors also highlight five areas for consideration when designing bio-inspired algorithms that can affect their behaviour: openness, diversity, interactions, structure and scale.

Common to all of the works above, is the requirement on the designer to consider both the biology on which the AIS is based, and the AIS application area. By doing this, it is hoped that the designer will take a more principled approach to algorithm design, leading to a better suited and performing algorithm. In this paper, we have chosen to revisit the design of an existing AIS, the Artificial Immune NETwork (aiNet) that was first introduced by de Castro and Von Zuben [4] in 2000. The design of this algorithm follows the ideas later presented as the layered framework for engineering AIS [1]. By taking into account additional AIS design ideas highlighted above, we investigate mechanisms for introducing population diversity in aiNet for the specific task of function optimisation, and take inspiration from diversity mechanisms in the human immune system.

2.2 aiNet

aiNet was originally designed for the task of data clustering [4], but has also been adapted for multi-modal function optimisation [9]. The algorithm is an example of a discrete immune network model in which network elements, called Ab, adapt to match a population of input elements, called Ag, that are to be recognised. Typically, the Ab and Ag elements are represented as D-dimensional vectors of real numbers, and are matched for similarity using a distance metric such as Euclidean distance. aiNet proceeds by an iterative procedure where each member of the Ag population is presented to each Ab in turn. Members of the Ab population then adapt to better match the Ag according to a mechanism inspired by Burnet's clonal selection principle. Once the Ab have adapted to the Ag, the same distance metric is used to calculate the similarity between each of the Ab. Ab that are similar to each other are eliminated due to a pre-defined threshold. It is this interaction between the Ab elements that forms the immune network, which can produce a dynamic behaviour of its own without the need for Ag interactions. Diversity is introduced into the population of Ab at the end of each iteration by the addition of a number of new Ab that have been randomly generated to a legal value in the problem's search space. de Castro and Timmis [1] provide the following pseudocode for a generalised version of the aiNet algorithm:

1. *Initialisation*: create an initial random population of network Ab;
2. *Antigenic Presentation*: for each Ag, do:
   (a) *Clonal Selection and Expansion*: for each Ab, determine its affinity with the Ag presented. Select a number of high affinity Ab and reproduce (clone) them proportionally to their affinity;
   (b) *Affinity Maturation*: mutate each clone inversely proportional to affinity. Re-select a number of the highest affinity clones and place them into a clonal memory set;
   (c) *Metadynamics*: eliminate all memory clones whose affinity with the Ag is less than a pre-defined threshold;
   (d) *Clonal Interactions*: determine the network interactions (affinity) among all the Ab of the clonal memory set;
   (e) *Clonal Suppression*: eliminate those memory clones whose affinity with each other is less than a pre-specified threshold;
   (f) *Network Construction*: incorporate the remaining clones of the clonal memory with all network Ab;
3. *Network Interactions*: determine the similarity between each pair of network Ab;
4. *Network Suppression*: eliminate all network Ab whose affinity is less than a pre-specified threshold;
5. *Diversity*: introduce a number of new randomly generated Ab into the network;
6. *Cycle*: repeat Steps 2 to 5 until a stopping condition is reached.

The version of aiNet for data clustering (clust-aiNet) is presented in [10], and follows the above pseudocode. The function optimisation aiNet version (opt-aiNet) is presented in [9], and differs from the pseudocode in a number of ways. In opt-aiNet, there is no distinct Ag population for the Ab to interact with, but an optimisation function instead, and therefore the Ab will represent candidate solutions to the optimisation function. As there is no Ag population, step 2 of the pseudocode above is updated to perform a localised search whereby all Ab are evaluated against the optimisation function, and are cloned proportionally to their fitness. This local search stops when the average Ag population fitness doesn't change significantly from one iteration to the next dependant on a threshold parameter.

3 Diversity

As highlighted in step 5 of the aiNet pseudocode above, diversity is introduced into the Ab population by randomly generating Ab vectors over the possible search space. This process is similar to the main diversity mechanism according to the clonal selection principle from which aiNet takes inspiration. Within the immune system, however, a mechanism called receptor editing exists that introduces diversity to the Ab population in a slightly different way.

3.1 Receptor Editing

In the human immune system, Ab are Y-shaped molecules formed of 2 linked molecular chains, called the light chain and the heavy chain. Each of these chains consists of a constant region, similar for all Ab, and a variable region, which is unique to each Ab. This variable region is the primary site where binding, and hence recognition, of Ag occurs. So in order for the immune system to be able to recognise unseen Ag, a large number of random Ab are initially generated. This generation process uses gene libraries and DNA rearrangement to produce a vast number of different Ab with unique variable regions. Any of these Ab that react with self Ag are then rejected, leaving a population of Ab that can go through an adaptation process of cloning and mutation of their variable regions to produce Ab that better match an Ag. The receptor editing mechanism can take Ab that were initially rejected for recognising self Ag, and apply a process whereby the variable region of the light chain of the Ab will re-arrange. The variable region of the heavy chain, however, stays the same, therefore the new Ab receptor is different to the original one, whilst retaining part of the old structure that is known to be able to recognise Ag [7].

In a computational sense, we can view the receptor editing process as re-using information that is known to be beneficial. Comparing this to the generation of Ab at random, we see that receptor editing can bias the occurrence of a new solution to an area of the search space where beneficial results are known to exist. Based on this observation, we extrapolate from this idea and develop an operator that re-uses the information stored within an existing Ab, whilst adding a biased variability.

3.2 A Diversity Operator

As our application is function optimisation, we investigated the sort of diversity operator that could introduce new Ab individuals into aiNet that beneficially bias the search, thus improve performance. By inspecting the results achieved with the standard opt-aiNet algorithm, it was apparent that the optima on the boundaries of the function are the most difficult to find, therefore our diversity operator was designed so that it would increase the chance of locating these optima. This property is present in a non-uniform mutation operator presented by Michalewicz in [11] for use in genetic algorithms. The operator is called non-uniform as it works dependant on the current algorithm iteration, producing less variability the higher the iteration number, thus allowing better local exploration in the later stages of the algorithm. The operator is described by equations 1 and 2 below:

$$x' = \begin{cases} x + \Delta(t, u - x) & \text{if a random digit is 0} \\ x - \Delta(t, x - l) & \text{otherwise} \end{cases} \tag{1}$$

$$\Delta(t, y) = y \left(1 - r^{\left(1 - \frac{t}{T}\right)^b}\right) \tag{2}$$

where x is a real number between a lower bound, l, and an upper bound, u, t is the iteration number, T is the maximum number of iterations, b is a parameter

determining the degree of non-uniformity, and r is a uniformly generated random number in the range [0,1]. The iteration dependent nature of this operator is not advantageous to a diversity operator in an AIS as it is being used for a different purpose. Mutation and local exploration are carried out by the 2nd step in the aiNet pseudocode presented above, and so we do not wish the diversity of new Ab to be restricted at the end of the algorithm's operation. So, by removing the iteration dependant nature of function $\Delta(t, y)$ we are left with the following uniform operator described by equations 3 and 4:

$$x' = \begin{cases} x + \Delta(u - x) & \text{if a random digit is 0} \\ x - \Delta(x - l) & \text{otherwise} \end{cases} \tag{3}$$

$$\Delta(y) = ry \tag{4}$$

This operator has the advantage of biasing the generated number towards the boundary of the values of x. This is illustrated by comparing Figure 1(a) with Figure 1(b), which show the distribution of values generated with this operator for $x = 0$ and $x = 7$ respectively, and $l = -10$ and $u = 10$. Here, as 0 is in the middle of the range of values x can be in, there is an equal probability for any value between -10 and 10 occurring. With $x = 7$, however, the chance of moving between 7 and 10 is a lot higher than between -10 and 7.

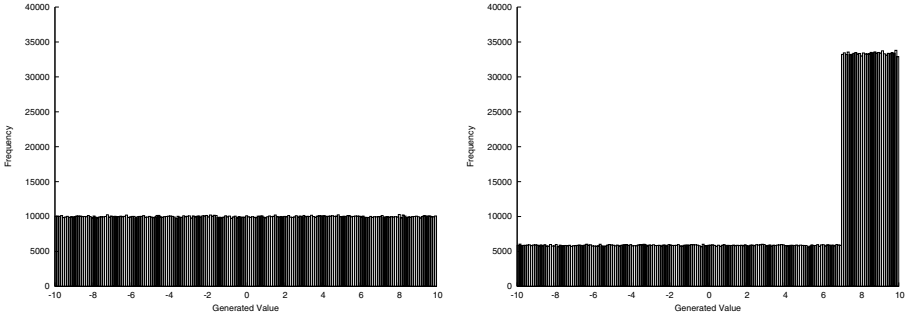
We have taken this operator and adjusted it further taking inspiration from the receptor editing process. The new operator is given by equations 5 and 6:

$$x' = \begin{cases} x + \Delta(u - x) & \text{if a random digit is 0} \\ x - \Delta(x - l) & \text{otherwise} \end{cases} \tag{5}$$

$$\Delta(y) = \begin{cases} ry & \text{if } y < p(u - l) \\ rp(u - l) & \text{otherwise} \end{cases} \tag{6}$$

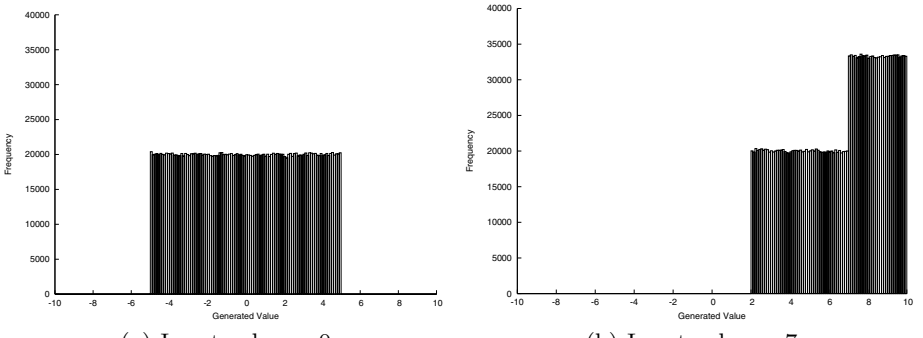
where p is a parameter that sets the range of values x' can be to a proportion of $[l,u]$. To illustrate this, Figure 2(a) shows the distribution of values generated when $x = 0$, $l = -10$, $u = 10$ and $p = 0.25$. Here we see that the new value will have an equal probability of falling between -5 and 5, but cannot be further away. Figure 2(b) shows the distribution of values generated when $x = 7$, demonstrating that the new operator still biases the search towards the boundaries of x when it is nearby. It is noted that this operator is only suitable on static functions where the boundaries of the function dimensions are known.

This diversity operator, described in equations 5 and 6, was incorporated into aiNet at the diversity introduction stage of the algorithm (stage 5 in the pseudocode above). The new diversity mechanism works by randomly selecting existing Ab vectors from the Ab network, cloning them, and then subjecting each dimension of the clone to the diversity operator. A user defined parameter is used to set the ratio of new Ab vectors to be introduced by the diversity operator and the original random introduction respectively.



(a) Input value = 0 (b) Input value = 7

Fig. 1. Histograms showing frequency of generated values using the uniform Michaelwicz operator



(a) Input value = 0 (b) Input value = 7

Fig. 2. Histograms showing frequency of generated values using the diversity operator

4 Function Optimisation

To test our approach at designing a diversity operator for opt-aiNet, the modified algorithm was applied to two function optimisation problems using the following three different ratios of diversity operator and random Ab introduction:

1. 100% Random Ab
2. 50% Random Ab, 50% Diversity Operator
3. 100% Diversity Operator

In the following results, we refer to these as versions 1, 2 and 3 of the algorithm respectively. For all experiments, the parameter p is set to an empirically determined value of 0.25, and all other algorithm parameters remain constant. Note that version 1 is identical to the original aiNet algorithm, and is used for comparison purposes to ascertain the effectiveness of the diversity operator.

When applied to a multi-modal function optimisation problem, opt-aiNet is capable of finding not only the global optimum, but a large number of the local

optima. Therefore, to evaluate the performance of opt-aiNet, we measure the number of optima found, and the speed at which it finds them. In accordance with [9] and [12], we calculate the number of optima found by recording the number of Ab in the final network. However, we differ from [9] and [12] by measuring the algorithm's performance by recording the number of times the objective function is evaluated during a run, rather than algorithm iterations. For the following experiments, our hypothesis is that we will be able to better locate the hard to find optima using the diversity operator when compared to the original random Ab introduction, as we are biasing the search in areas of the search space where opt-aiNet appears to struggle.

4.1 Experiments

We experimented with two 2-dimensional function optimisation problems, thus the Ab vectors consist of 2 values, x and y. This means that the fitness landscape of the functions can be represented on a 3 dimensional plot. The first optimisation function, used in accordance with [9], is given by equation 7:

$$f(x, y) = x.sin(4\pi x) - y.sin(4\pi y + \pi) + 10 \tag{7}$$

where x and y are in the range [-2,2]. A graphical depiction of this function is given in Figure 3, which shows that there are 100 maxima in this function.

For the second optimisation function, we have adapted $f(x, y)$ to introduce an irregular fitness landscape with a large area containing no local maxima. This type of irregular fitness landscape should provide a different challenge for the opt-aiNet algorithm. This function, shown graphically in Figure 4, is given by:

$$g(x, y) = \begin{cases} f(x, y) & \text{if } x < 0.5 \\ -x^2 + 7 & \text{otherwise} \end{cases} \tag{8}$$

where x and y are in the range [-2,2]. There are 70 maxima for this function.

Two experiments were performed with functions $f(x, y)$ and $g(x, y)$ using the 3 versions of opt-aiNet mentioned above. The first experiment used the algorithm

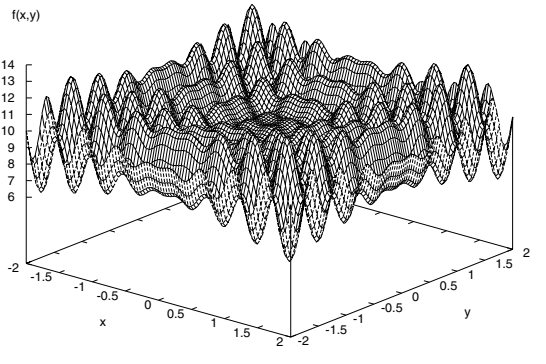


Fig. 3. Regular multi-modal function $f(x, y)$

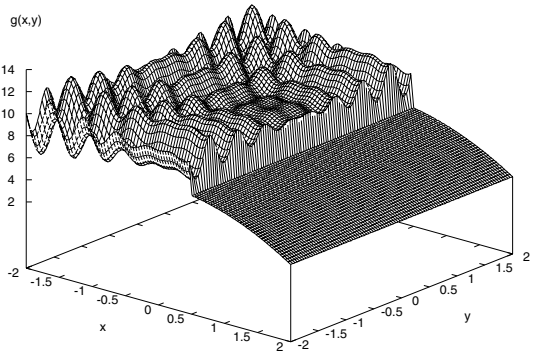


Fig. 4. Irregular multi-modal function $g(x, y)$

stopping condition described in [9], where the algorithm stops when the size of the Ab network does not change between iterations. This experiment was run 100 times for each algorithm version, and the mean, median and range (lowest and highest) values for the number of maxima found and function evaluations performed were recorded. The second experiment sets the stopping condition to a fixed number of maxima, so the algorithm only stops when this number of maxima have been found. For the function $f(x, y)$, 100 algorithm runs were made with the stopping conditions 90 maxima and 100 maxima and for $g(x, y)$, 100 runs were made with the stopping condition set to 70 maxima. Again the mean, median and range values were recorded.

4.2 Results

The results for the first experiment with $f(x, y)$ are shown in Table 1. These show that the more the diversity operator is used, the more maxima the algorithm will find, however, this is at the expense of function evaluations, with the number of evaluations increasing the more the diversity operator is used. Dividing the mean number of function evaluations by the mean number of maxima found for each algorithm version, we see that the number of evaluations required to find each maximum works out at around the same value ($\approx$ 4600 evaluations). This

Table 1. Results showing number of maxima and function evaluations for 100 runs of opt-aiNet using the original stopping condition when applied to $f(x, y)$

Algorithm Version	Maxima Mean	Maxima Median	Maxima Range	Evals Mean	Evals Median	Evals Range
1	87.68	89	[68 : 98]	400131	393019	[195932 : 663058]
2	91.13	92	[76 : 100]	416715	405020	[198396 : 649462]
3	92.12	94	[65 : 99]	428251	437162	[151635 : 689975]

Table 2. Results showing number of maxima and function evaluations for 100 runs of opt-aiNet using the original stopping condition when applied to $g(x, y)$

Algorithm Version	Maxima Mean	Maxima Median	Maxima Range	Evals Mean	Evals Median	Evals Range
1	51.86	55	[12 : 65]	435999	449537	[26136 : 812966]
2	55.83	58	[10 : 68]	378820	382250	[21032 : 767019]
3	58.59	62	[8 : 70]	316343	321035	[11792 : 625460]

Table 3. Results showing number of evaluations for 100 runs of opt-aiNet to find 90 maxima when applied to $f(x, y)$

Algorithm Version	Evals Mean	Evals Median	Evals Range
1	356807	346148	[245333 : 578523]
2	364084	348931	[261624 : 576818]
3	350657	343552	[235554 : 576444]

Table 4. Results showing number of evaluations for 100 runs of opt-aiNet to find 100 maxima when applied to $f(x, y)$

Algorithm Version	Evals Mean	Evals Median	Evals Range
1	2215077	1947781	[897314 : 9415846]
2	1344824	1154989	[660264 : 3911380]
3	966430	910349	[449955 : 2279970]

Table 5. Results showing number of evaluations for 100 runs of opt-aiNet to find 70 maxima when applied to $g(x, y)$

Algorithm Version	Evals Mean	Evals Median	Evals Range
1	1675100	1642025	[677820 : 2761473]
2	924242	888019	[488257 : 1823107]
3	643913	612656	[307362 : 1122979]

shows that the main effect of the diversity operator for this experiment is to delay the termination of opt-aiNet, allowing more maxima to be found but at cost the of function evaluations. Looking at the range values for the number of maxima and evaluations, we see that there is a large disparity between the lowest and the highest number, and thus the performance of the algorithm is widely varying. This observation agrees with observations of [12]. The results for the first experiment with $g(x, y)$ shown in Table 2, enhance and exaggerate those seen with $f(x, y)$. Again, we see that the the more the diversity operator is used, the

more maxima are found. However, in contrast to the $f(x, y)$ results, it is observed that the number of function evaluations required to find these maxima reduces the more the diversity operator used, making the average number of evaluations to find each maxima smaller. This function also enhances the unpredictable nature of when the algorithm will stop, with the difference between the lowest and highest number of maxima found being very large for all algorithm versions.

The results for the second experiment with $f(x, y)$ in Tables 3 and 4 and show conclusively where the benefit of the diversity operator lies. When the stopping condition is set at 90 maxima, all three algorithm versions perform virtually identically. However, when the algorithm is required to find all 100 maxima, the more the diversity operator is used, the lower the number of function evaluations required, with algorithm version 1 requiring over twice as many function evaluations on average than algorithm version 3. Upon visual inspections of the maxima found using the 90 maxima stopping condition, it was seen that the 10 maxima not found were located on the boundary of the search space. From this we conclude that the diversity operator enhances the ability of opt-aiNet to find these difficult maxima, thus showing our approach at designing a biased mutation operator was effective. The results of the second experiment to find all 70 maxima in $g(x, y)$ are shown in Table 5. Again, this shows clearly that the advantage of the diversity operator, with algorithm version 1 requiring over two and half times more function evaluations than algorithm version 3 to find all the maxima.

From these experiments, we can see that the diversity operator gives us more benefit on the function $g(x, y)$ than $f(x, y)$. This is due to the large area of the function $g(x, y)$ that contains no local maxima, so there is a far smaller chance of introducing a new Ab into this area with the diversity operator than there is with random introduction. We can also see that our hypothesis that the diversity operator will improve the performance of opt-aiNet was shown to be correct for these two functions.

5 Data Clustering

Having established the advantages of the diversity operator for function evaluation, we blindly incorporated it into clust-aiNet and applied it to a data clustering problem to observe its performance in a problem domain it was not explicitly designed for. The task of clustering data differs in many ways to the task of function optimisation, thus the experiments and performance measures suitable for clust-aiNet also differ. clust-aiNet is an unsupervised learning algorithm that performs a data compression to produce a map of the data representing the centres of data clusters. After running the algorithm, the final Ab network represents an internal image of the data set to which it was exposed [10]. Thus to determine whether this internal image is a representative solution, de Castro and Von Zuben [10] analyse their solutions using a number of statistical and graph theory techniques. We feel that, although valid, this approach is beyond the scope of this paper, so we make an assumption that the final output of

our experiments with clust-aiNet are all equally valid. We have verified this assumption by visually inspecting a subset of the output of our experiments. This assumption means that we can focus on assessing whether the diversity operator improves the performance of clust-aiNet by measuring speed of convergence of the different algorithm versions.

5.1 Experiments

For the clust-aiNet experiments, the same three algorithm versions were used with p set to 0.25 and all other parameters remaining constant. The task was to cluster a set of randomly generated data shown in Figure 5. This data consists of six clusters of fifty 3-dimensional data items, where each dimension falls in the range [-10,10]. Each of the 3 versions of clust-aiNet was run 100 times, with the algorithm stopping after 50 iterations. During each run, the number of Ab in the Ab network was recorded at each iteration. The mean number of Ab for each iteration was then calculated for the 3 versions of clust-aiNet.

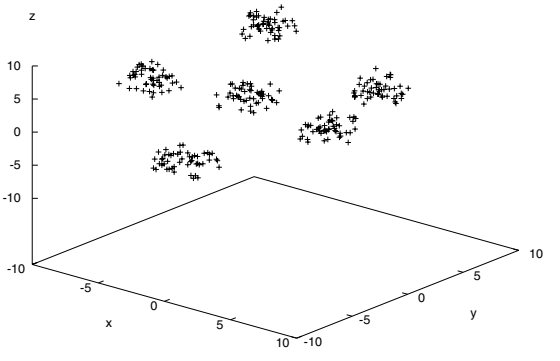


Fig. 5. Randomly generated clustered data points

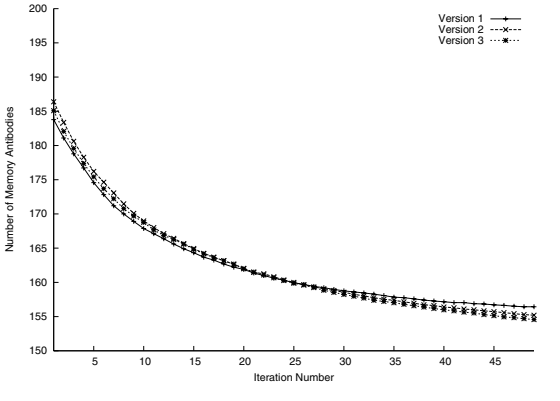


Fig. 6. Results showing mean number of memory Ab per iteration for 100 runs with each version of clust-aiNet when applied to data clustering

5.2 Results

The results of the experiment with clust-aiNet shown in Figure 6, show virtually no difference in the speed of convergence of the algorithm for all three versions of the algorithm. Although the experiments with clust-aiNet were not extensive, these results show that the diversity operator neither enhances nor harms its performance. This is not surprising, however, as the operator was designed to improve the performance of function optimisation, not data clustering.

6 Conclusion

There are a number of general conclusions we can draw from our results. Our original intention was to take a structured approach at designing a diversity operator to improve the performance of aiNet for the task of function optimisation. The results with opt-aiNet clearly show that our diversity operator does this, and so we conclude that our design approach was successful. By incorporating the diversity operator into clust-aiNet and using it for a purpose it was not designed for, we were able to show that the bias involved in the design of the diversity operator was not appropriate in improving performance for this problem domain. This type of result is to be expected and backs up the suggestion of taking a problem oriented approach [2] in the design of AIS for a specific engineering problems. We have also been successful in showing that the way diversity is introduced into AIS does affect the performance of the algorithm, as highlighted by Stepney et al. [3]. As a final comment, we suggest that if the fitness landscape of the problem to which you are applying an AIS is unknown, an appropriate method for introducing diversity is to use a mixture of randomly introduced Ab and an alternative diversity mechanism that tries to bias the newly introduced individuals dependant on the nature of the problem.

References

1. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer-Verlag (2002)
2. Freitas, A.A., Timmis, J.: Revisiting the foundations of artificial immune systems: A problem-oriented perspective. In: Proceedings of the 2nd International Conference on Artificial Immune Systems (ICARIS 2003), Springer (2003) 229–241
3. Stepney, S., Smith, R.E., Timmis, J., Tyrell, A.M.: Towards a conceptual framework for artificial immune systems. In: Proceeding of the 3rd International Conference on Artificial Immune Systems (ICARIS 2004), Springer (2004) 53–64
4. de Castro, L.N., Von Zuben, F.J.: An evolutionary immune network for data clustering. In: Proceeding of the IEEE Brazilian Symposium on Artificial Neural Networks. (2000) 84–89
5. Forrest, S., Perelson, A., Allen, L., Cherukuri, R.: Self-nonself discrimination in a computer. In: Proceedings of the IEEE Symposium on Research in Security and Privacy. (1994) 202–212

6. de Castro, L.N., Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation **6** (2002) 239–251
7. Janeway, C.A., Travers, P., Walport, M., Shlomchik, M.: Immunobiology: The Immune System in Health and Disease. 5th edn. Garland Publishing (2001)
8. Hart, E., Ross, P.: Studies on the implications of shape-space models for idiotypic networks. In: Proceedings of the 3rd International Conference on Artificial Immune Systems (ICARIS 2004), Springer (2004) 413–426
9. de Castro, L.N., Timmis, J.: An artificial immune network for multimodal function optimization. In: 2002 Congress on Evolutionary Computation. (2002) 699–704
10. de Castro, L.N., Von Zuben, F.J.: ainet: An artificial immune network for data analysis. In Abbass, H.A., Sarker, R.A., Newton, C.S., eds.: Data Mining: A Heuristic Approach. Idea Group Publishing (2002) 231–259
11. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag (1996)
12. Timmis, J., Edmonds, C.: A comment on opt-ainet: An immune network algorithm for optimisation. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004). (2004) 308–317