A Comment on opt-AiNET: An Immune Network Algorithm for Optimisation

Jon Timmis and Camilla Edmonds

Computing Laboratory, University of Kent. Canterbury. Kent. CT2 7NF. UK
j.timmis@kent.ac.uk, camillaedmonds@hotmail.com

Abstract. Verifying the published results of algorithms is part of the usual research process. This helps to both validate the existing literature, but also quite often allows for new insights and augmentations of current systems in a methodological manner. This is very pertinent in emerging new areas such as Artificial Immune Systems, where it is essential that any algorithm is well understood and investigated. The work presented in this paper results from an investigation into the opt-aiNET algorithm, a well-known immune inspired algorithm for function optimisation. Using the original source code developed for opt-aiNET, this paper identifies two minor errors within the code, propose a slight augmentation of the algorithm to automate the process of peak identification: all of which affect the performance of the algorithm. Results are presented for testing of the existing algorithm and in addition, for a slightly modified version, which takes into account some of the issues discovered during the investigations.

1 Introduction

This paper investigates the now popular aiNET algorithm, first proposed in [1]. The initial system was initially designed for data clustering, but later adapted for optimisation [2,6]. This investigation forms part of a larger on-going project into the usefulness and viability of the AIS approach: the investigations call for a detailed re-implementation and testing of existing algorithms. To this end, a detailed testing of an algorithm called opt-aiNET was undertaken. This investigation employed both the existing code written by the authors of [2] (obtained with their permission) and a reimplementation of the system in Java by the authors of this paper. Through the process of revisiting the algorithm, testing the original system and undertaking reimplementation, existing results can be verified and augmentations and improvements can be proposed. It was found that results from [2] were not exactly reproducible, whilst this is to be expected to some degree (as this is a stochastic algorithm) we observed that when averaged over a reasonable number of independent runs (in this case 50), the results were slightly different than first reported in [2]. This is not to say that results in [2] were inaccurate, merely incomplete, as results for multiple independent results were not reported. A reimplementation of opt-aiNET was then undertaken, to further investigate the performance of the algorithm and try to identify reasons for these differences. Reimplementation is a useful tool and this is especially true in new paradigms such as AIS [3]. Indeed, work as already shown problems with algorithms such as AINE [4] another immune network approach. Work by [5] demonstrated premature convergence of the algorithm, which was not observed in the original work. This in itself, acted to some degree to motivate this work and validate a similar style immune network algorithm. It is felt that only through rigorous investigations can the field of AIS hope to grow and be taken as a serious competitor and viable alternative to other techniques.

This paper presents the original opt-aiNET, and identifies a number of minor issues relating to the actual implementation of that algorithm. The paper then proposes slight modifications to the algorithm and results testing both the original and re-implemented versions of opt-aiNET. This paper assumes knowledge of the AIS area in general, if the reader is not familiar with the area, they are directed to [3] for further information.

2 Artificial Immune Systems in Optimisation

There is a natural parallel between the immune system and optimisation. Whilst the immune system is not specifically an optimiser, the process of the production of antibodies in response to an antigen is evolutionary in nature: hence the comparison with optimisaiton, the location of better solutions. The process of clonal selection (a theory widely held by many immunologists) describes how the production of antibodies occurs in response to an antigen, and also explains how a memory of past infections is maintained. This process of clonal selection has proved to be a source of inspiration for many people in AIS and there have been a number of algorithms developed for optimisation inspired by this process [3,6,12,13].

One algorithm that has received much attention is aiNET [1,6]. AiNET has the capability to not only perform unimodal search, but also multimodal, without the need for any enhancements, unlike hybrid genetic algorithms [2]. The rest of this paper focuses on the algorithm proposed in [1], then extended in [2]. The reason for this is two fold: (1) It is becoming more widely used, so it is important to ascertain that the algorithm behaves the way it is reported and (2) it allows for greater confidence in the area of AIS if results are verified.

2.1 AiNET

The aiNET algorithm is a discrete immune network algorithm that was developed for data compression and clustering [1], and was also extended slightly and applied to optimization to create the algorithm opt-aiNET [2]. This has subsequently been developed further and applied to areas such as bioinformatics [7] and even modeling of simple immune responses [8].

Opt-aiNET, proposed in [2], evolves a population, which consists of a network of antibodies (considered as candidate solutions to the function being optimised). These undergo a process of evaluation against the objective function, clonal expansion, mutation, selection and interaction between themselves. Opt-AiNET creates a memory set of antibodies that represent (over time) the best candidate solutions to the objective function. Opt-aiNET is capable of either unimodal or multimodal optimisation and can be characterised by five main features:

- The population size is dynamically adjustable;
- It demonstrates exploitation and exploration of the search space;
- It determines the locations of multiple optima;
- It has the capability of maintaining many optima solutions;
- It has defined stopping criteria.

2.1.1 The Algorithm

Assuming the following terminology:

Network Cell	Individual of the population. Opt-aiNET does not employ any mechanism of encoding – real values are used in a Euclidean shape space.
Fitness	Measure of how good a particular cell is performing in relation to the objective function
Affinity	The Euclidean distance between two network cells
Clone	Offspring cells that are identical copies of their parent cell.
Mutated Clone	A clone that has undergone somatic hypermutation

```
1. Randomly initialise population

2. While (stopping criteria is not met) do

        I. Determine fitness of each network cell against
           objective function

        II. Generate Nc clones for each network cell

        III. Each clone undergoes somatic hypermutation
           in proportion to the fitness of the parent
           cell (see Eq. 1)

        IV. Determine the fitness of all network cells
           (including new clones and mutated clones)
        V. For each clone select the most fit and remove
           the others

        VI. Determine average error (distance from
           solution), if different from previous
           iteration, repeat from step 2

  3. Determine highest affinity network cells and perform
     network suppression.

  4. Introduce a percentage d of randomly generated
     network cells.
```

In step 1, the initial population consists of N network cells (randomly created). Each cell is a real value vector, which represents a candidate solution. During steps I-V each network cell undergoes a process of clonal expansion (N x Nc) and affinity maturation. Clones of each cell are mutated according to the affinity of the parent cell. The fitness represents the value of the function for the specific candidate solution. The affinity proportion mutation is performed according to the following equation:

$$C' = c + \alpha N\,(0.1) \qquad \textbf{(1)}$$

$$\alpha = (1/\beta)\,\exp\,(-f^*)$$

where α is the amount of mutation, c is the parent cell, c' is the mutated clone of c, N (0,1) is a Gaussian random variable of zero mean and standard deviation of 1, β is a parameter that controls the decay of the inverse exponential function and f* is the fitness of c normalised in the interval [0..1]. As c' represents a candidate solution, it must be within the range of the functions specified domain. If c' exceeds that, then it is rejected and removed from the population.

The fitness of each clone (and parent cell) is evaluated, then the fittest individual being selected to become a memory cell and the algorithm adopts an elitist approach to achieve this by always selecting the most fit. This is an iterative process that continues unto the average error value (distance from objective function) stabilises (this must be less than 0.0001). Once stablilisation occurs, the algorithm then proceeds to steps 3 and 4. Network suppression removes any similar or non-stimulated antibodies and antibodies that fall below the pre-determined suppression threshold σ. By removing similar cells, opt-aiNET prevents antibodies clustering on a single peak. This reduces the amount of cells maintained in the memory set,

It should be noted at this point, that the network interactions within opt-aiNET are only suppressive in nature and they do not contribute to the stimulation of the cells in any way. This algorithm is not faithful to the traditional immune network theory by Jerne [9], which proposes interactions of suppression and stimulation between B-cells.

2.2 Observations on opt-AiNET

A detailed investigation of the code for opt-aiNET both employing the existing code (implemented by the authors of [2]) and a reimplementation of the algorithm by authors of this paper. These investigations uncovered a number of minor issues within the code implanted in [2]. The implication of this was that if one were to implement a version directly from the paper, then the results reported there would not be obtained. Uncovering this minor points, whilst not dramatically altering the algorithms performance, does have some impact, however. This paper proposes the small fixes required to be implemented, which we feel more accurately represents the algorithm as it was intended. This work also extends the current system by employing an automated mechanism for the location of peaks in search space, a mechanism which was lacking from the original. Discrepancies between the original results and results obtained using the same code were also identified. It is suggested that the original results reported were not in any way meant to be misleading, but the discrepancy is more likely to be caused by a lack of independent runs being reported in [2]. A technique which has been employed in this paper.

The observations we would like to make are:

1. *Population fitness is calculated assuming that the overall fitness has increased and both values are positive.*

Every 5 iterations, the degree of similarity between the present average fitness (*Avfit*) and the previous average fitness, is evaluated (this number is left out from any literature and is hard coded into the algorithm. Experimental evidence also suggested tat this could play an important role in the convergence of the algorithm). If the similarity is less than 0.0001 then the algorithm the algorithm can proceed to the suppression function.
The average fitness delta is calculated using the following equation:

$$\text{If I} - \mathit{avefitold}/\mathit{avefit} < 0.0001; \qquad (\mathbf{2})$$

Calculating the value in this manner assumes that Avfit > avefitold are positive and that *Avfit* is greater than *avefitold*. This creates a problem if both values are negative, as the absolute value of avefitold is greater than the absolute value of *Avfit* (as in the existing code, it is the absolute value that is taken). This would result in the criteria for similarity being met even if there is a large difference between the two values. In order to fix this problem, these assumptions have been lifted.

2. *The selection function within opt-aiNET was not elitist.*

In the original system, the suppress function, calls the DIST function in MATLAB. This calculates the Euclidean distance between two vectors. If the distance is less than the suppression threshold, then the first vector is suppressed. No attempt is made to evaluate which of the two possess the greater fitness. This of course can result in the deletion of a potential optimum solution. The greater the threshold, the greater the chance of this occurring.

In order to overcome this, a simple ordering was placed on the network cells (they were sorted by fitness), to ensure that the least fit was always removed, as opposed to another candidate solution that had a higher fitness.

3. The calculation of peaks was done manually

This is very important for the reliability of the results. Upon inspection of the code taken from [2], it was clear that there was no reliable mechanism for the analysis of peaks within the system. Therefore, a more reliable automated approach was created that counted the number of optima found within the population, so as to accurately record the number. This is done simply by comparing the fitness of the candidate solution with the fitness value of its neighbors (determined by stimulation threshold). The candidate solution with the highest fitness is determined to be the highest point in that neighborhood.

3 Results

3.1 Experimental Protocol

In order to assess the opt-aiNET, we were obliged to follow as far as possible the same experimental protocol as the authors of [2]. Therefore, we used the same three functions presented by those authors, namely:

Multi Function: Range [-2,2]

$$G(x,y) = x.\sin(4\Pi x) - y.\sin(4\Pi y + \Pi) + 1 \qquad (3)$$

This function has a single global optimum solution, with many local optima distributed non-uniformly

Roots Function: Range [-2,2]

$$g(z) = 1/1 + |z^6 - 1| \qquad (4)$$

Where z is a complex number z = x + iy

This function has six maxima, which are located on slender peaks that rise to a height of 1 (g(z) ==1) from a plateau of 0.5 centered on (0,0). These maxima are at the six roots of the unity of the complex plane.

Schaffer's Function: Range [-10,10]

$$g(z) = 0.5 + \frac{\sin^2\left(\sqrt{x^2 + y^2}\right) - 0.5}{\left(1 + 0.001(x^2 + y^2)\right)} \quad (5)$$

This function has as single global optima and an infinite number of local optima, which form concentric rings expanding out from the optima. The global optimum is hard to locate due to the similarity with the best local optima.

The parameters employed by the original authors were:

Parameter	Value
Suppression threshold	0.2
Initial population size	20
Number of clones generated	10
Percentage of random new cells each iteration	40%
Scale of affinity proportion selection	100
Maximum iteration number	500

The results in the following section record the number of peaks located (Peaks), ItG is the number of iterations required to reach the global optima and ItC is the number of iterations required for convergence. Each experiment was performed 50 times, and the standard deviation is presented. Results are presented for the published version of the results (taken from [2]) identified as **published**, testing of the original code, identified as **recorded** and taken from the reimplementation, identified as **new**.

3.2 Multi Function

Opt-aiNET (Published)			**Opt-aiNET (Recorded)**		
Peaks	*ItG*	*ItC*	*Peaks*	*ItG*	*ItC*
56.10±4.36	53.50±47.19	278.50±70.09	57.2±13.83	58.8±45.18	410±145.36

	Opt-aiNET (New)	
Peaks	*ItG*	*ItC*
56.5±17	212.8±140	384.6±143

Table 1. Resutls for opt-aiNet

Analysis the results in detail revealed some interesting results. First, there is an issue of extreme convergence, which affects the average value of convergence (ItG) and also the number of peaks found. The empirical evidence would suggest that on this function, opt-aiNET failed to converge at least 50% of the time, but premature convergence was less frequent. When there is a failure to converge, it was observed that the suppress function was never activated which lead to a large number of candidate solutions to be located on a single peak. The authors of [2] comment that opt-aiNET avoids a "waste of resources" by positioning only a single individual on a peak. This assumes that it converges before reaching the maximum number of iterations allowed and as peaks were counted manually, that process was very prone to error.

Secondly, the global optima solution was found to be very unstable. Although in each instance the algorithm did reach a point that was considered the global maximum, as many as 90% of the time, this optimum solution was lost (due to the problems highlighted earlier).

Overall, the obtained and previously published results are very similar, with the exception of the number of iterations. As the work in [2] does not report how many independent runs were undertaken, one can only assume this figure is somehow affected in the work presented here, due to the number of experiments undertaken.

However, when one observes the results using the version with the enhancements (which were highlighted above), then the results appear different. The ability to find peaks has not been affected greatly, but the time taken to find the first optima solution, has. In addition, the overall convergence rate increases. This is most likely explained by the elitist mechanism now implemented in the new version of the algorithm. This may well lead to an overall quicker convergence rate being achieved. It is also worth pointing out that the standard deviations are rather large, even for a stochastic search algorithm, this might indicate a lack of reliability and robustness in the system.

3.3 Roots Function

Opt-aiNET (Published)			**Opt-aiNET (Recorded)**		
Peaks	*ItG*	*ItC*	*Peaks*	*ItG*	*ItC*
6.0	86.89±34.31	295±129.74	5.9±0.32	93.2±31.99	308.8±112.62

	Opt-aiNET (New)	
Peaks	*ItG*	*ItC*
5.2±0.79	149.2±87.1	344±99

Table 2. Results comparing the Roots Function

Overall, with this function, the results that were obtained were consistent with those published, but did vary with the reimplementation of the algorithm. The time it takes to first locate the optima solution is significantly longer than the original versions and indeed, average convergence is also longer. In some ways this is surprising, as the new version operates with an elitist approach. Clearly the average fitness within the network is staying lower for longer, hence the increase in time to convergence. Also, the ability to identify peaks has been diminished slightly Whilst not overly significant, we fell that it is worthy of note at least as the standard deviation is quite high and often the maximum number of peaks was not obtainable.

3.4 Schaffer's Function

For this function, things were a little more complicated. According to the published results [2], the algorithm should not converge within the 500 iterations. However, this was found not to be the case, where on average, the number of iterations for convergence was 219.2±199.97 It is proposed that the cause for this is the unstable average fitness level within the population. The instability is such that convergence occurs frequently.

After further investigations, it was discovered that the reason that convergence is occurring so frequently is that with a suppression threshold of 0.2 (as taken from the already published work) it is highly probable that no network cells will undergo the suppression. Upon examination of the actual original code, if this does occur, than the algorithm will terminate. A simple remedy to this problem is increasing the suppression threshold, but this will reduce the number of peaks that are located by the search. An alternative to that would be to increase the size of the initial population. This would hope to reduce the chances of no suppression taking place (as there will be more members and a higher probability that some will be close enough to be below the suppression threshold). However, if this approach is adopted that there will be an adverse affect on the running time, as a larger population increases the execution time.

4 Conclusions

This paper has revisited an immune inspired algorithm called opt-aiNET. A number of minor modifications to the original system have been proposed, which, it is felt more accurately reflect the intended and previously described system.

An identical test protocol based on the original work was established, with systematic tests using the existing code and newly developed code, being undertaken. It was observed that the results obtained from testing the system were not quite the same as reported in [2], but this may well be due to more tests being produced. Minor problems with existing code were identified, and these corrected in a new implementation. Whilst not affecting the performance significantly, the algorithm does behave differently, and is subject (independent to the alterations) to a very high standard deviation when experiments are performed multiple times. We are confident that the system that has been developed as part of this investigation is faithful to the original intended system and the code is available from the authors upon request.

References

[1] De Castro, L.N and Von Zuben, F. (2001). "aiNET: An Artificial Immune Network for Data Analysis", in Data Mining: A Heuristic Approach. Abbas, H, Sarker, R and Newton, C (Eds). Idea Group Publishing.

[2] De Castro, L.N and Timmis, J. (2002) An Artificial Immune Network for Multimodal Function Optimisation. Proc. Of IEEE World Congress on Evolutionary Computation. Pp. 669-674

[3] De Castro, L.N and Timmis, J. (2002) Artificial Immune Systems: A New Computational Intelligence Approach. Springer-Verlag.

[4] Timmis, J and Neal, M. (2001). A Resource Limited Artificial Immune System for Data Analysis. Knowledge Based Systems, 14(3-4):121-130.

[5] Knight, T and Timmis, J (2001). AINE: An Immunological Approach to Data Mining. In Cercone, N, Lin, T and Wu X. (Eds) IEEE International Conference on Data Mining. Pp. 297-304, San Jose. CA.

[6] De Castro, L. N. & Von Zuben, F. J. (2002), Learning and Optimisation Using the Clonal Selection Principle. IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems, **6**(3), pp. 239-251.

[8] Jerne, N (1975). Towards a Network theory for the Immune System. Annals of Immunology., Inst. Pasture.

[9] (name removed for blind review) Artificial Immune Networks and Multimodal Optimisation. MSc Thesis. (place removed for blind review)

[10] Bezerra, B and De Castro, L.N. (2003) Bioinformatics data analysis using an artificial immune network. Proceedings of ICARIS 2003, eds. Timmis, J., Bentley, P. and Hart, E. Lecture Notes in Computer Science 2787, pp. Springer-Verlag, 2003.

[11] De Castro, L.N. (2003). The Immune response of an Artificial Immune Network (aiNET). Congress on Evolutionary Computation (CEC) pp. 1273-1280. IEEE press.

[12] Walker, J and Garrett, G (2003). Dynamic Function Optimisation: Comparing the Performance of Clonal Selection and Evolutionary Strategies. LNCS 2787. 273-284. Timmis, J, Bentley, P and hart, E. (Eds.)

[13] Kelsey, J, Timmis, J and Hone, A. (2003). Chasing Chaos. Proceedings of the Congress on Evolutionary Computation (CEC). Pages 413-219. IEEE.