The merits of Boltzmann selection in differential evolution.

SD van Deventer
20273401
20273401@sun.ac.za

***Abstract*—Differential evolution (DE) is defined as a stochastic, population based search strategy developed by Storn and Price [1], [2]. DE makes use of information about the distance and direction of the population to guide the search process. This sets it aside from other evolutionary algorithms. The original intent of the DE was to be applied to continuous-valued landscapes. This report explores an alternative approach to the selection process when selecting an individual to be used in the next generation through the use of Boltzmann selection. Additionally the notion of dynamically adjusting the control parameters of the DE is explored in an attempt to address the time consuming task of finding the best control parameter combination.**

I. Introduction

Differential evolution (DE) was first introduced in 1995 as an evolutionary algorithm (EA) developed to be used over continuous spaces. Since then it has been successfully applied to multiple real world multidimensional global optimization problems. A DE algorithm consists of a number of individuals that form a population. Each individual is defined by a vector that represents a solution to some optimisation problem. Through the use of distance and directional information about the current population the search process is guided to find optima [4]. After each iteration a new generation is created. This refers to a new population comprised out of a subset of the current population and some new individuals. EAs traditionally apply crossover and/or mutation to form a new generation. Mutation is a genetic operator used to introduce genetic diversity between generations by altering one or more genes of an individual. Similarly crossover is the genetic process of combining the genetic material of two individuals to produce a new offspring individual. If both of these steps are applied crossover is applied first and then followed by the mutation process. In contrast to this, DE first applies mutation to produce a trial vector that is then used by the crossover operator to create the offspring of a parent individual.

As stated DE uses information regarding the distance and direction of the current population when forming a new generation. This is enforced by the use of difference vectors to influence the step size during the mutation process. Difference vectors, in this report, refers to the element wise difference between two vectors. This mutated trial vector is then used in addition to a parent individual in the crossover operator to create an offspring.

In the standard DE a parent individual is replaced by its offspring only if the offspring is more fit than the parent individual. Fitness is determined by some fitness function. This is called elitism, the notion that only the best individual is used in the next generation. This could however limit the exploration ability of DE and as such there may be a more suitable approach. The proposed approach is to make use of Boltzmann selection. This allows the offspring and the parent individual to have the same probability of being used in the next generation in the early stages of the algorithm and then progressively become more elitist in nature, meaning that the more fit individual will be chosen in the later stages of the algorithm. This could allow for more exploration which then gradually turns into exploitation as the algorithm moves to an elitist approach.

DE has three control parameters – namely, the population size, the scaling factor and the recombination probability. These control parameters are problem dependent and the task of finding optimal combinations can be very time consuming. This report explores a self adapting approach to optimise the control parameters with each iteration. In addition a new approach is also presented that attempts to mimic the exploration vs exploitation trade off.

This report considers all the above mentioned approaches and evaluates them against a number of optimization benchmark problems of varying complexity and dimension. The aim of this report is to explore Boltzmann selection as a selection operator to conclude whether or not it provides better performance than an elitist approach.

II. Background

A. *Differential evolution*

DE is a stochastic, population based search strategy. It forms part of evolutionary algorithms, but differ greatly from others in the same paradigm. This is due to the fact that it incorporates the mutation operator before the crossover operator and uses information about the current populations direction and distances. It has been successfully applied to many multidimensional global optimisation problems.

DE consists of a population of individuals. Each individual is expressed as a vector that represents a solution to the optimization problem. DE navigates the search space by creating new generations of individuals based on the direction and distance of the current population. This means that each individual in the population in turn is seen as a parent individual. A trial vector is calculated based on the mutation of three randomly selected individuals. This trial vector is then used along with the parent vector in the crossover operator to produce an offspring. This is done by selecting some elements of the parent vector and some elements of the trial vector to form the new offspring vector. The offspring and parent individuals are evaluated by the fitness function and the most fit individual is used in the next generation. Algorithm 1 provides the standard DE algorithm. The scaling factor and recombination probability is given by β and p_r respectively. $C(t)$ denotes the population in generation t and n_s represents the number of individuals in the population. A fitness function f calculates the fitness of an individual. The i^{th} individual in generation t is represented as $\mathbf{x}_i(t)$.

Algorithm 1 Differential evolution

```
Initialize the control parameters, $\beta$ and $p_r$
Create and initialize the population, $C(0)$, of $n_s$ individuals
repeat
    for each individual $\boldsymbol{x}_i \in C(t)$ do
        Evaluate the fitness, $f(\mathbf{x}_i(t))$
        Create the trial vector, $\mathbf{u}_i(t)$, through mutation
        Create an offspring, $\mathbf{x}'_i(t)$, through crossover
        if $f(\mathbf{x}'_i(t))$ is better than $f(\mathbf{x}_i(t))$ then
            Add $\mathbf{x}'_i(t)$ to $C(t+1)$
        else
            Add $\mathbf{x}_i(t)$ to $C(t+1)$
        end if
    end for
until stopping condition is true
Return the individual with the best fitness
```

As can be seen the offspring individual will be used in the new generation instead of the parent individual only if it is more fit than the parent. Otherwise the parent will persist and the offspring is discarded. This is the notion of elitism, only the most elite individual is deemed usable in the next generation.

B. Control parameters

Differential evolution makes use of three control parameters,

- population size, n_s
- scaling factor, β
- recombination probability, p_r

Each of these parameters have an impact on the performance of the DE algorithm. Unfortunately these are problem dependent and finding optimal values require a very time consuming trial and error approach.

The population size n_s has a lower bound: $n_s > 2 \cdot n_v + 1$, where n_v denotes the number of difference vectors used. This report only considers the use of one difference vector and so this lower bound is larger than 3. This means that for the algorithm to work at least 4 individuals are needed. This is due to the fact that the mutation operator requires 3 unique individuals aside from the parent individual under consideration.

The scaling factor β is used within mutation and scales the difference vector or step size. A larger value causes larger step sizes and results in more exploration. A smaller value leads to smaller step sizes and focuses more on exploitation. The value of β is in $(0, \infty)$.

The recombination probability p_r defines the probability that an element in the trial vector will be used in the offspring and $1 - p_r$ is the probability that an element in the parent vector will be used in the offspring. Once again a larger value for p_r leads to more of the trial vector being present in the offspring and can be seen as the enforcement of exploration. This is because the offspring will differ more from the parent and will likely produce a much different solution. Similarly a smaller value will lead to less of the trial vector being present in the offspring. This means that the offspring will be closer to the parent and therefore produce a solution closer to that of the parent. As this is a percentage the value of p_r is in $[0.0, 1.0]$.

C. Mutation operator

Mutation is a way to introduce variation in the population and as such can be used to explore the search space. Mutation makes use of a target vector and scaled difference vectors. This report only considers the use of one difference vector. The difference vector is the element wise difference between two random unique individuals from the population. It is scaled by the control parameter β. Mutation is the process of adding this scaled difference vector to the target vector to obtain a new trial vector. Doing so adds genetic diversity to the algorithm. This trial vector is used in the crossover operator.

D. Crossover operator

Crossover is the process of combining the genetic material to produce a new offspring from the parent and the trial vectors. There are two main approaches to this – namely,

- binomial crossover
- exponential crossover

This report only considers binomial crossover. Algorithm 2 shows the procedure of binomial crossover.

Algorithm 2 Binomial crossover

```
$\mathbf{j}^* \sim U(1, n_x)$
$J \leftarrow J \cup \{\mathbf{j}^*\}$
for each $\boldsymbol{j} \in \{1, \ldots, n_x\}$ do
    if $U(0,1) < p_r$ and $\mathbf{j} \neq \mathbf{j}^*$ then
        $J \leftarrow J \cup \{\mathbf{j}\}$
    end if
end for
```

The variable $\mathbf{j}^*$ denotes an initial random number sampled from a uniform distribution between 1 and n_x (the vector size). J denotes the set of dimensions in the offspring vector that originates from the trial vector. As can be seen the initial random element $\mathbf{j}^*$ is added to J to ensure that the offspring differs from the parent in at least one dimension. Thereafter a random floating point value is sampled from a uniform distribution between 0 and 1 for each dimension except the dimension denoted by $\mathbf{j}^*$. If this floating point value is less than that of the recombination probability p_r the dimension $\mathbf{j}$ is added to the set J. It should be noted that a larger value of p_r results in a larger number of dimensions added to the set J. The offspring is then created. The dimensions reflected in J are added to the offspring from the trial vector and the remaining dimensions are added from the parent vector.

E. Elitism

After an offspring is produced from a parent the individual to be used in the next generation must be selected. Both individuals are evaluated against some fitness function f. For the standard selection scheme used by DE the most fit individual will move on to the next generation and the other will be discarded. This is referred to as elitism. This selection scheme may not be optimal as it limits the exploration of DE in that the direction of a solution is only explored if it is definitely better than the current solution. This approach focuses more on exploitation.

F. Boltzmann selection

Boltzmann selection is based on the thermodynamic principles of simulated annealing. Simulated annealing is an optimization process based on the cooling process of a liquid or solid [3]. It can however be used to compute selection probabilities and thus provides an alternative selection scheme to elitism. The selection probability is given as,

$$\varphi(\mathbf{x}_i(t)) = \frac{1}{1 + e^{f(\mathbf{x}_i(t))/T(t)}} \tag{1}$$

where $\varphi(\mathbf{x}_i(t))$ is the probability of selecting $\mathbf{x}_i(t)$ and $T(t)$ is the temperature parameter. T becomes smaller as time progresses. This selection scheme can be altered to incorporate both individuals under consideration and allows both individuals to have the same probability of being selected in early generations but moves steadily to an elitist approach for later generations.

G. Fitness evaluation

The fitness of an individual indicates the quality of the found solution. In the case of a minimisation problem an individual with a smaller fitness value is seen as a better solution than an individual with a larger fitness value. In this report the fitness function is defined by five classical benchmark functions of differing complexity and dimension. Only minimization problems are considered.

III. Implementation

A. Initialisation

The basic structure of algorithm 1 is followed. The control parameters n_s, β and p_r are initialised to values of 100, 1.9 and 0.9 respectively. The initial population is initialised with n_s random individuals. Each individual consists of n_x dimensions each sampled from a uniform distribution between the upper and lower bounds set by the classical benchmark function under consideration.

Fitness evaluation is simply done by applying the individual to the benchmark function.

B. Mutation

The trial vector $\mathbf{u}_i(t)$ is created through mutation. The following steps are taken in the mutation process:

1) Pick a random individual, that is unique from the parent under consideration, from the current population. This individual is referred to as the target vector, $\mathbf{x}_{i_1}(t)$.
2) Pick two more unique random individuals from the population. Call them $\mathbf{x}_{i_2}(t)$ and $\mathbf{x}_{i_3}(t)$.
3) Calculate the difference vector as $\mathbf{x}_{i_2}(t) - \mathbf{x}_{i_3}(t)$ and scale it using the scaling factor β.
4) Add this scaled difference vector to the target vector to create the trial vector.

We can therefore define the trial vector to be,

$$\mathbf{u}_i(t) = \mathbf{x}_{i_1}(t) + \beta \cdot (\mathbf{x}_{i_2}(t) - \mathbf{x}_{i_3}(t)) \tag{2}$$

The individuals are picked randomly according to a uniform distribution between 1 and n_s.

C. Crossover

The crossover operator creates the offspring individual, $\mathbf{x}'_i(t)$. This is done by binomial crossover as defined in algorithm 2. The dimension $\mathbf{j}^*$ is sampled from a uniform distribution between 1 and n_x to ensure that at least one dimension of the trial vector is used within the offspring. Thereafter the other dimensions are considered for the offspring through the use of the crossover rate, p_r. The vector resulting from the parent and trial vector crossover forms the offspring individual.

D. Selection

This report considers two selection schemes – namely,

- Elitism
- Boltzmann selection

The selection scheme in algorithm 1 is referred to as elitism and is the standard selection scheme for DE. This scheme selects the individual that produces the most fit solution and discards the other.

Boltzmann selection is based on simulated annealing. To incorporate this selection scheme in DE a small adjustment is made. Equation (3) shows the Boltzmann selection used in this report.

$$\varphi(\mathbf{x}_i(t)) = \frac{1}{1 + e^{(f(\mathbf{x}_i(t)) - f(\mathbf{x}'_i(t)))/T(t)}} \tag{3}$$

This results in both individuals, the parent and offspring, having equal probability of being selected during the early generations and during the later generations the more fit individual is selected. To enforce this behaviour the variable T must decay over time. This is done by using exponential decay,

$$T(t+1) = \alpha T(t) \tag{4}$$

where α is the factor with which T decays. For this report the α value is set at 0.5 and $T(0)$ is set at 5000, which is the total

number of iterations.
This selection probability is then used to select an individual by sampling a random floating point value uniformly between 0.0 and 1.0. If the value is lower than the selection probability the offspring is chosen over the parent.

E. *Self adaptive control parameters*

The control parameters need to be optimised for each new problem and requires a very time consuming trial and error process. As such Zhao, Zhiwei, Jingming Yang, Ziyu Hu, and Haijun Che. proposed a self adaptive DE approach [5]. Among other aspects of the DE they propose self adaptive behaviour for the scaling factor β and the recombination probability p_r. The proposition is that at the end of each generation the values for β and p_r change as follows:

- $\beta \sim \mathcal{N}(\mu_\beta, 0.1)$
- $p_r \sim \text{Cauchy}(\theta_{p_r}, 0.1)$

where μ_β and θ_{p_r} is initialised to 0.5 and updated as
$\mu_\beta = a \cdot \mu_\beta + (1-a) \cdot \frac{\sum_{\beta \in w\beta} \beta}{n_s}$ and
$\theta_{p_r} = a \cdot \theta_{p_r} + (1-a) \cdot \frac{\sum_{p_r \in wp_r} p_r}{n_s}$
The sets $w\beta$ and wp_r are introduced to keep track of successful values of β and p_r for every parent respectively. This is done by adding the values of β and p_r to their sets if the offspring was chosen over the parent. The update functions for μ_β and θ_{p_r} are called after all parents have been evaluated and the sets $w\beta$ and wp_r are set to the empty set. If these sets are empty at the time of calling the update functions, the values for μ_β and θ_{p_r} are retained. The variable a is a positive constant within the range (0,1). A value of 0.9 is given to a.

F. *Dynamic control parameters*

To contrast the use of the self adaptive control parameters this report presents an alternative approach to dynamically update the control parameters. The control parameters have a direct impact on the exploration and exploitation trade off. The value of β impacts the step size or difference vector directly. It is easy to see that a large β results in a larger step size and enforces more exploration. In contrast to this a small β results in a smaller step size and therefore more exploitation. Similarly the value of p_r is the probability that a dimension of the trial vector is used in the creation of the offspring. Therefore a larger p_r results in a more diverse offspring. This behaviour favours exploration and similarly a smaller p_r favours exploitation as the offspring is closer to the parent. It is therefore proposed to update the values of β and p_r as follows:

$$\beta(t+1) = \beta(0) \cdot e^{-(\frac{t}{1000})} + 0.1 \tag{5}$$

$$p_r(t+1) = p_r(0) \cdot e^{-(\frac{t}{1000})} + 0.1 \tag{6}$$

These update functions allow the values of β and p_r to dynamically decrease from the first generation to the last generation to enforce the exploration-exploitation trade off.

IV. Empirical procedure

A. *Benchmark functions*

To evaluate the performance of each approach a set of classical benchmark functions are used to represent the fitness function. These benchmark functions vary in complexity and dimension and are minimised through the DE algorithm. They are defined as follows:

- f_1, the absolute value function
$$f_1(\mathbf{x}) = \sum_{i=1}^{n_x} |x_i| \tag{7}$$
with each $x_i \in [-100, 100]$.
- f_5, the elliptic function
$$f_5(\mathbf{x}) = \sum_{i=1}^{n_x} (10^6)^{\frac{i-1}{n_x-1}} \tag{8}$$
with each $x_i \in [-100, 100]$.
- f_8, the michalewicz function
$$f_8(\mathbf{x}) = -\sum_{i=1}^{n_x} \sin(x_i) \left(\sin\left(\frac{i x_i^2}{\pi}\right)\right)^{2m} \tag{9}$$
with each $x_i \in [0, \pi]$ and $m = 10$.
- f_{12}, the rastrigin function
$$f_{12}(\mathbf{x}) = 10n_x + \sum_{i=1}^{n_x} (x_i^2 - 10\cos(2\pi x_i)) \tag{10}$$
with each $x_i \in [-5.12, 5.12]$.
- f_{24}, the vincent function
$$f_{24}(\mathbf{x}) = -\left(1 + \sum_{i=1}^{n_x} \sin(10\sqrt{x_i})\right) \tag{11}$$
with each $x_i \in [-0.25, 10]$.

Once again n_x refers to the number of dimensions in the vector. As can be seen each of these benchmark functions are defined within certain bounds, for example f_1 is defined for $x_i \in [-100, 100]$. If an individual moves outside these bounds it is seen as infeasible and not considered for the next generation.

B. *Performance measures*

The following performance measures are used to evaluate the DE algorithm:

- average diversity of the population
- average difference vector magnitude
- average best solution quality
- average number of boundary violations

The diversity of the population gives an indication of whether the population is very spread out across the search space and therefore exploring or if the population is moving towards a singular point on the search space and therefore exploiting. The desired outcome of the DE is that individuals move closer to each other to find a single optimal solution.

The difference vector can be seen as a step size and indicates with how much the trial vector will differ from the target vector. The difference vector magnitude gives a good indication of how much this step size changes over time as a large value indicates a large step size and a small value a small step size. This value is expected to decrease over time as that is indicative of the population moving from exploration towards exploitation and finding a single optimal solution.

The quality of the best solution is obtained by the fitness function and indicates the effectiveness of the DE algorithm in minimising the benchmark function. This value is expected to decrease over time as the population finds better solutions to minimise the benchmark function.

The number of boundary violations gives an indication of how much time is spent exploring infeasible space. This value is expected to decrease with a decline in difference vector magnitude. This is due to the fact that individuals will search closer to themselves which is more likely to be within feasible space.

C. *DE configuration*

The standard DE described in algorithm 1 is used as a basis for all the variants discussed in this report. Firstly the standard DE is implemented and optimal control parameters are found through trial and error. Table I shows the values considered for function f_1. Note that the values $\beta = 0.5$ and $p_r = 0.1$ is found to be the most optimal combination.

TABLE I
CONTROL PARAMETER INITIALISATION

n_s	β	p_r
100	1.9	0.9
100	1.0	0.5
100	0.5	0.5
100	0.1	0.1
100	0.5	0.1

The second DE variant incorporates the self adaptive control parameter approach discussed in section 3 (E). Table II depicts the control parameter initial values considered. $\beta = 1.9$ and $p_r = 0.9$ was found to be the best fit.

TABLE II
CONTROL PARAMETER INITIALISATION

n_s	β	p_r	μ_β	θ_{p_r}
100	1.9	0.9	0.5	0.5
100	1.0	0.5	0.5	0.5
100	0.5	0.5	0.5	0.5
100	0.1	0.1	0.5	0.5

The third DE variant incorporates the dynamic control parameters presented in section 3 (F). The initial values of the control parameters are set as $n_s = 100$, $\beta = 1.9$ and $p_r = 0.9$. The values of β and p_r changes dynamically with each generation as can be seen in figures 1 and 2 respectively.

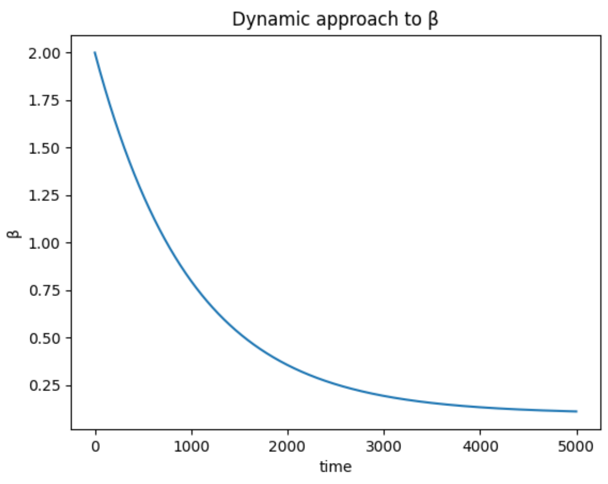

Fig. 1. Dynamic decrease of the control parameter β

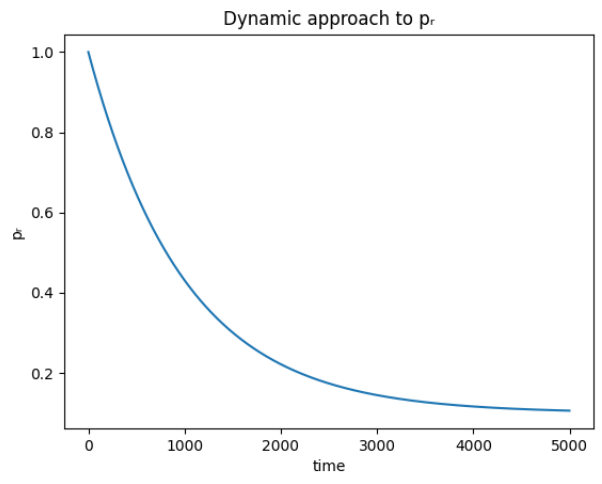

Fig. 2. Dynamic decrease of the control parameter p_r

In addition to elitist individual selection, boltzmann selection is applied to each of the above mentioned DE variants along with the same initialisation values.

Each DE algorithm is run for 5000 iterations (generation) and the average performance measures are recorded over 30 independent simulations.

V. Research Results

First consider the standard trial and error approach to selecting the values of β and p_r. Figure 3 shows the quality of the best solutions obtained by different combinations of β and p_r. These results are for the benchmark function f_1 and follow a standard elitism approach to selecting the individual that proceeds to the next generation. It can be seen that the combination of $\beta = 0.5$ and $p_r = 0.1$ yields the best results as it converges fairly quickly.

Figure 4 depicts the quality of the best solution of the three different DE approaches when applied to benchmark function f_1. The first approach is the standard DE as can be seen in

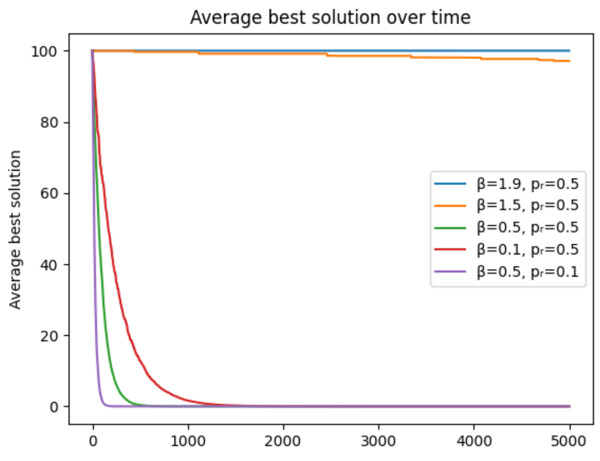

Fig. 3. Quality of the best solution of a standard elitism DE for f_1

figure 3 with $\beta = 0.5$ and $p_r = 0.1$ obtained by trial and error. Secondly the dynamic DE refers to the DE that dynamically changes the values of β and p_r starting at a value of 1.9 and 0.5 respectively. Lastly the self adapting DE approach is depicted as is defined in section 3 (E). It can clearly be seen that the self adapting DE falls short and fails to produce an optimal value relative to the other approaches. It should also be noted that while the dynamic DE takes longer to converge than the standard DE the quality of the best solution is higher. The standard DE produces on average a best solution of 5.91×10^{-24} while the dynamic DE produces a value of 1.85×10^{-50}. This combined with the time saved by bypassing the need to refine the control parameters results in an overall improved approach. Hence the dynamic DE is used to obtain the following results.

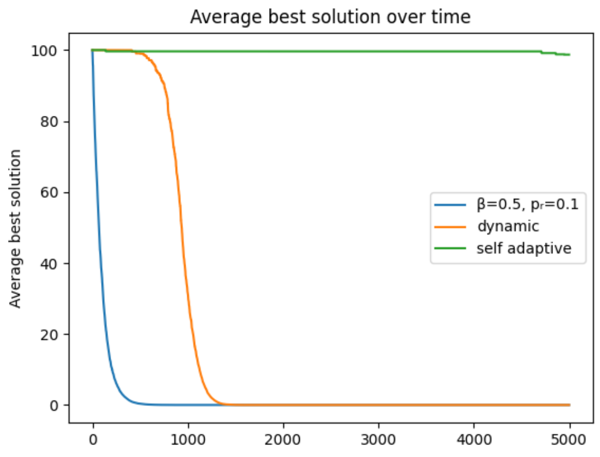

Fig. 4. Quality of the best solution of a standard, dynamic and self adapting elitism DE for f_1

Figure 5 displays the behaviour of the search process when applying the dynamic elitism DE approach to benchmark function f_1. It can be noted that individuals are initially far apart but move closer to one another as they find more optimal solutions. Individuals become very close to one another indicating that there is very little difference between their respective solutions.

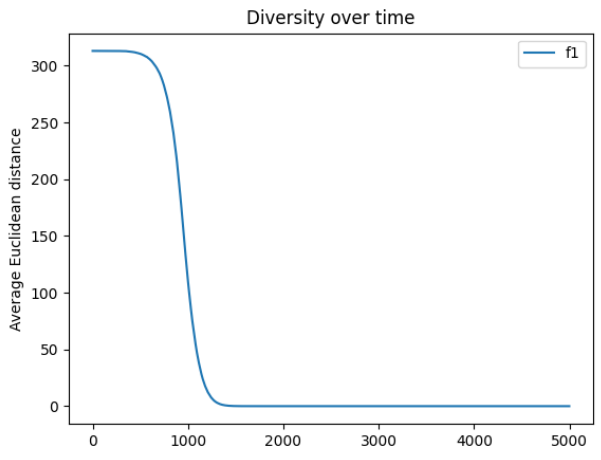

Fig. 5. Diversity of the population of a dynamic elitism DE for f_1

Figure 6 displays the average difference vector magnitudes. This displays noteworthy behaviour as it shows that the step size fluctuates rapidly in early generations and maintain a high value. The step size then decreases over generations and become extremely close to zero. This implies that there is very little difference between the individuals of the population and that individuals have evolved to obtain a single solution to the benchmark problem.

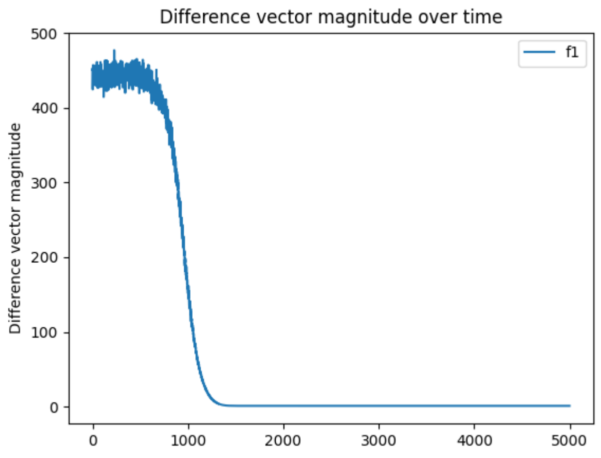

Fig. 6. Difference vector magnitude of a dynamic elitism DE for f_1

The quality of the best solution produced by the population is depicted in figure 7. This is a minimisation problem and as such a lower value indicates a superior solution. This shows that the quality of the best solution improves over generations and that an optimal solution is found close to zero. This is the desired outcome and prove the effectiveness of DE to optimise the benchmark function at hand.

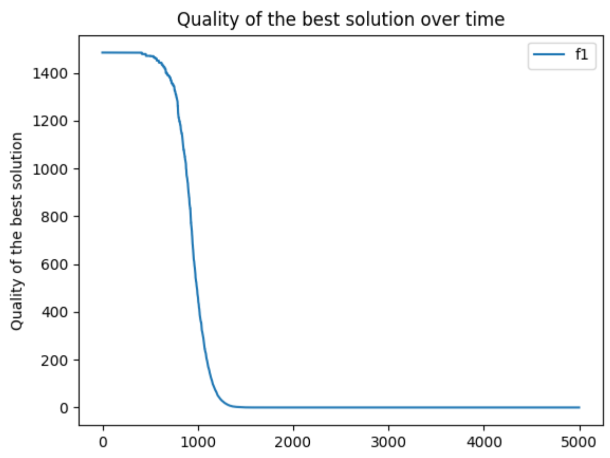

Fig. 7. Quality of the best solution of a dynamic elitism DE for f_1

Figure 8 shows the number of boundary violations or individuals searching in infeasible space. This gives a clear indication that the DE search explores infeasible space during the early generations. This is good since there might be optimal feasible solutions that might not be obtainable by only exploring the current feasible area. The search moves toward feasible space as time progress until it is no longer exploring any infeasible space.

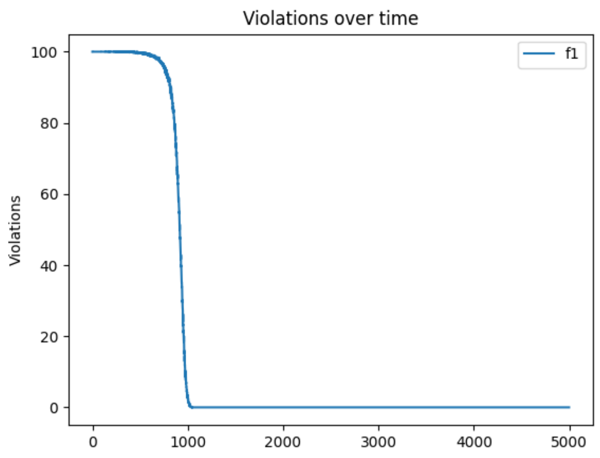

Fig. 8. Number of boundary violations of a dynamic elitism DE for f_1

It can be noted from figure 9 that the elitist and boltzmann selection schemes produce very similar DEs. This figure depicts the search behaviour of the two DEs when applied to all benchmark functions and clearly show that they have very similar search patterns. It can be noted that the solutions are similar as well and is depicted in figure 10. The results observed in figures 9 and 10 are similar to the results seen for the standard DE where the values of β and p_r are refined through trial and error.

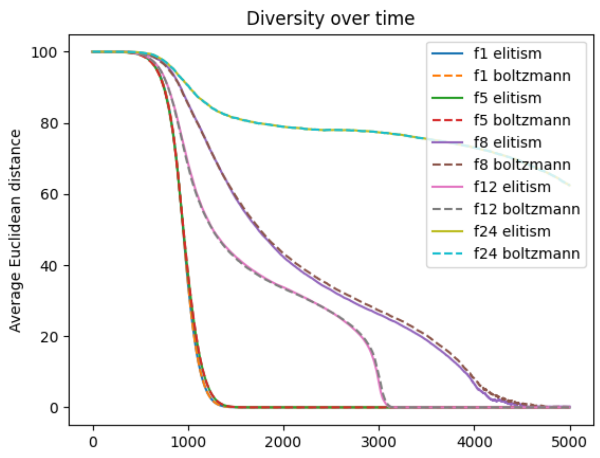

Fig. 9. Diversity of the population for the elitist and boltzmann DE applied to all benchmark functions

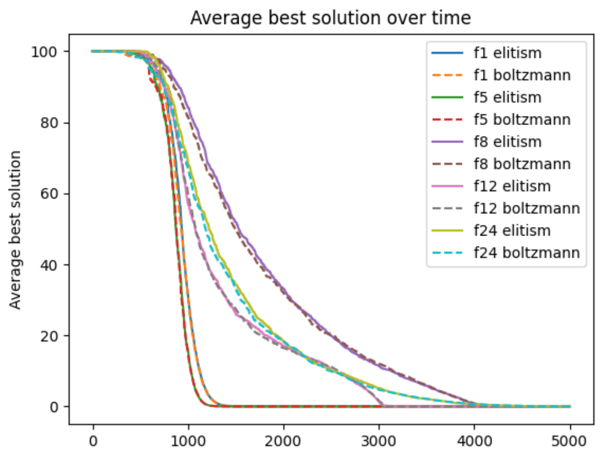

Fig. 10. Quality of the best solutions of the population for the elitist and boltzmann DE applied to all benchmark functions

VI. Conclusion

From the results observed the difference between the elitist and Boltzmann selection schemes seem to be negligible and produce very similar results. The use of Boltzmann selection did not produce better quality solutions or different search behaviour.

It is also established that the self calibration technique did not produce a good DE algorithm and is not effective in finding optimal solutions. The proposed dynamic approach to refining the control parameters β and p_r produced meaningful results that are comparable to those obtained from a trial and error procedure. This approach does have a delay in convergence speed, but reduces the time needed for control parameter refinement.

Future research

- *Exponential crossover*. Replacing binomial crossover with that of exponential crossover could result in a different offspring composition. The offspring could be comprised out of a more or less diverse sample of the parent and trial vector. This could potentially impact the searching behaviour of the DE.
- *Parent selection*. For this report a random parent selection scheme is implemented. This refers to the manner in which the individuals are selected that form part of the difference vector during mutation. While the random selection of these individuals have resulted in good search behaviour different schemes should also be studied. Some of these could include Roulette wheel sampling, tournament selection and some rank-based selection.
- *Other benchmark functions*. This report only considers five classical benchmark problems. There are however a plethora of functions that could lead to additional insights.

References

[1] R. Storn and K. Price. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. Journal of Global Optimization,11(4):431–359, 1997.

[2] K.V. Price, R.M. Storn, and J.A. Lampinen. Differential Evolution: A Practical Approach to Global Optimization. Springer, 2005.

[3] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of State Calculations by Fast Computing Machines. Journal of Chemical Physics, 21:1087–1092, 1958.

[4] Andries P. Engelbrecht. *Computational Intelligence: An Introduction*, Second Edition, John Wiley & Sons, University of Pretoria, South Africa, 2007. pp. 237.

[5] Zhao, Zhiwei, Jingming Yang, Ziyu Hu, and Haijun Che. “A Differential Evolution Algorithm with Self-adaptive Strategy and Control Parameters Based on Symmetric Latin Hypercube Design for Unconstrained Optimization Problems.” European Journal of Operational Research 250.1 (2016): 30-45. Web.