Parameter Optimisation for Parameterised-Response Traders with Differential Evolution

Paolo Mura
Department of Computer Science
University of Bristol
Bristol, BS8 1UB, U.K.
rk19763@bristol.ac.uk

***Abstract*—Differential evolution algorithms iteratively adapt their solution while running, with the aim of solving an optimisation problem. These algorithms have at least two main parameters of interest, a population size NP and a mutation factor F. This paper explores these parameters in one specific optimisation problem: trader profit in simulated financial markets. In general, NP determines how many trading strategy values are considered at a time, with higher values of NP leading to increased evaluation periods. Higher values of F increase the amount of mutation applied to generate new strategies. Experiments are presented that aimed to find the optimal parameter values. Although smaller values of k and F values centred around 1.0 seemed to generate the most profit, the results were largely inconclusive. More success was found in developing an adaptive form of differential evolution, which automates the process of choosing F. This alternative algorithm was shown to significantly outperform the standard differential evolution algorithm.**

***Index Terms*—parameterised-response, zero intelligence, adaptive differential evolution, parameter optimisation**

I. Introduction

Throughout the twentieth century, developments in technology have allowed financial markets to transition from a physical space towards an increasingly digital platform. In today's world, traditional human agents have largely been replaced by autonomous robot traders. These robot traders are provided with limit prices by their client and follow algorithms to submit quote prices on their client's behalf, with the aim of maximising profit margin (the difference between their quote price and limit price).

One class of such trading algorithms is the zero-intelligence traders, so called because of their minimally-constrained decision-making strategies. For example, the ZIC (Zero Intelligence Constrained) algorithm simply quotes prices from a uniform random distribution, constrained only by their limit price so as not to incur a loss. Gode and Sunder showed in 1993 that in simulated market experiments, ZIC behaved surprisingly similarly to human traders [1].

The contemporary PRZI (Parameterised-Response Zero Intelligence) algorithm was introduced by Cliff in 2021 [2]. It extends ZIC through the inclusion of a strategy parameter, $s \in [-1.0, +1.0]$, which modifies the shape of the probability distribution from which the trader generates its quote prices. A value of $s = 0$ gives a uniform distribution, causing PRZI to behave identically to ZIP. Increasingly negative values relax the strategy until at $s = -1.0$, it behaves like the SHVR algorithm, which simply "shaves" a single unit off the best quote price each iteration. On the contrary, increasingly positive values raise the urgency of the strategy until at $s = +1.0$, it behaves like the GVWY algorithm, which quotes at the trader's limit price. In other words, a strategy value of -1 encourages the trader to maximise the profit of a transaction, whereas a value of $+1$ encourages it to make a transaction as soon as possible. It is clearly useful to find the optimal strategy value that maximises overall profit given a specific market.

In the following year, Cliff introduced another algorithm called PRDE (Parameterised-Response Differential Evolution), which automates the process of finding this optimum [3]. The PRDE algorithm is inspired by differential evolution, an iterative process that aims to solve an optimisation problem by evaluating different possible solution values. Specifically, it implements the DE/rand/1 variant, which works as follows. Each trader maintains its own population of strategy values $s_1, s_2, ..., s_{NP}$, with size $NP \geq 4$. With each iteration i of the DE process, it evaluates the fitness of the corresponding strategy $s_{i,x}$. It then randomly chooses three other, distinct strategies from its population to form a new mutant strategy $s_{i,y}$. In the DE/rand/1 approach, this mutation step uses the formula:

$$s_{i,y} = s_{i,a} + F \cdot (s_{i,b} - s_{i,c}) \tag{1}$$

Where $s_{i,a}$, $s_{i,b}$ and $s_{i,c}$ are the randomly chosen strategies and F is a mutation coefficient. If the fitness of $s_{i,y}$ is greater than that of $s_{i,x}$, then $s_{i,y}$ replaces $s_{i,x}$ in the subsequent round. There are several distinctions between Cliff's implementation found in [4] and the original DE algorithm proposed in [5]. These include the order in which steps are carried out; the absence of a crossover step (since it deals with 1-dimensional strategy values rather than vector solutions); the current strategy chosen at random rather than iteratively; and a vector-perturbation method to minimise the risk of strategy convergence within the population.

Although PRDE automates the selection of s , it introduces two new parameters: NP and F . The function of both parameters is clear. NP determines the size of the population, i.e. the number of strategies that can be maintained at any given time, while F determines the mutation coefficient. A small value of F will result in a mutant strategy that is very similar to an existing strategy, specifically $s_{i,a}$ in (1), whereas a large value gives a higher multiplication factor to the offset and consequently may help to generate a wildly new value. This paper provides an investigation into the specific effect these parameters have on the performance of the PRDE algorithm, as well as how best to optimise these values. It then introduces an adaptive differential evolution algorithm implementation, which automates the process of selecting a value of F and constantly updates it alongside the strategy value itself.

II. Parameters of Differential Evolution

A. Description of the Parameters and Approach

In order to test the effect of the NP and F parameter values, I made use of simulated agent-based modelling techniques via BSE (the Bristol Stock Exchange) [4]. BSE simulates a CDA (continuous double auction), in which buyers make ascending bids while sellers simultaneously make descending offers (or "asks") until a quote price "crosses the spread". This is the situation where a buyer quotes a price that is higher than the best ask, at which point a transaction occurs at the best ask price, or vice versa. BSE is an abstraction of real-world financial trading platforms that also commonly implement a CDA system. Some of these simplifying assumptions include no delay in communications, only one tradable asset and full availability of the latest state of the LOB (limit order book) to all traders. At each of its cycles, a trader is randomly given the chance to submit an order, after which all traders are able to update their internal state. It is at this point where PRDE performs the DE algorithm to adapt its strategy. In BSE, the population size of PRDE traders is denoted k since this is comparable with the k parameter used in the PRSH (Parameterised-Response Stochastic Hill-climber) trader, which is itself inspired by k-armed bandits. From this point forwards, k may be used interchangeably with NP .

There is disagreement in the literature as to the best values of k and F [6] [7] for differential evolution. DE is an approach that generalises well across many different optimisation problems but the optimal values of k and F will be different depending on both the given optimisation problem and the specific (market) conditions. Generally, suggested values for k can range up to 50, while F values are broadly recommended between 0 and 2.

The BSE implementation of PRDE evaluates each strategy's fitness—in this case, profit per second—for a fixed time period δ_e . This means that the total time for the entire population of k strategies to be evaluated at least once each has a lower bound expressed by:

$$t_e \geq 2 \cdot \delta_e \cdot k \tag{2}$$

There is a factor of 2 since both the strategy and its potential mutant replacement must each be evaluated at every step of the DE algorithm. It is a lower bound because PRDE's DE algorithm selects the next strategy randomly rather than iteratively, so not all k strategies may get evaluated in k generations of DE. In BSE, the default evaluation period is $\delta_e = 7200$ i.e. two simulated trading hours, giving

$$t_e \geq 4 \cdot k \tag{3}$$

hours. It is therefore evident that choosing a large value of k requires many simulated days in order to give the strategies a good chance of being evaluated. Due to limited compute time, I chose to investigate f values of $[0.2, 0.6, 1.0, 1.4, 1.8]$ and K values of $[5, 10, 15, 20, 25]$. This allows experiments to cover a good portion of the recommended ranges while being practical to perform simulations with.

In a similar approach to [8] (but with different values), I used a simple market configuration with stepped supply-demand schedules in the range $[50, 150]$ for both supply and demand with 10 buyers and 10 sellers. I performed different experiments with both homogeneous populations of PRDE traders, as well as mixed populations of both PRDE and ZIC traders. As with the experiments in [8], these were done one-to-many and balanced group (i.e. a 50:50 ratio of PRDE to ZIC traders). In all experiments, the values of k and F were kept constant across all traders, but varied across independent experiments. The reason for such simplistic configurations was to limit the amount of noise that may interfere with the signal (distinguishing behaviour of traders with different parameter values). The use of ZIC traders was done to provide the simplest benchmark possible; introducing more sophisticated trader types would be an unnecessary variable to consider when the objective is merely to test between the different parameters. Cliff's original PRDE experiments ran for 300 days, but profits were shown to converge within approximately 25 days [3]. This, together with the limited compute resources available, seemed to suggest that 50 simulated trading days would be sufficient per experiment. I repeated each experiment to give 10 independent trials per test.

B. Results

To make some comparison on the outcomes of the simulations, I aggregated results by finding the sum of the profits for each trial. I then visualised these data using box plots in order to capture any trend and variance. This can be seen in Fig. 1 to Fig. 6, which exclusively plot the data from PRDE traders only, even in markets co-populated by ZIC traders. This is because it is only the comparison between different values of PRDE's k and F parameters that we are interested in.

In the homogeneous trials (Fig. 1 and Fig. 4), visible trends can be seen. An F value of 1.0 gives the highest mean profit across the trials, with lower profits for more extreme values of F. Standard deviation is much smaller for higher values of F. In Fig. 4, there is a clear inverse relationship between k and profit, though less variation in the standard deviation as with F.

The difference in values was much less well defined for the 1-many and balanced group trials. Performing statistical tests via an Anova for each experiment revealed that in each of these four experiments, the p-value was greater than 0.05. This confirmed that there was not enough evidence to make conclusions on the effect of k or F on the profit.

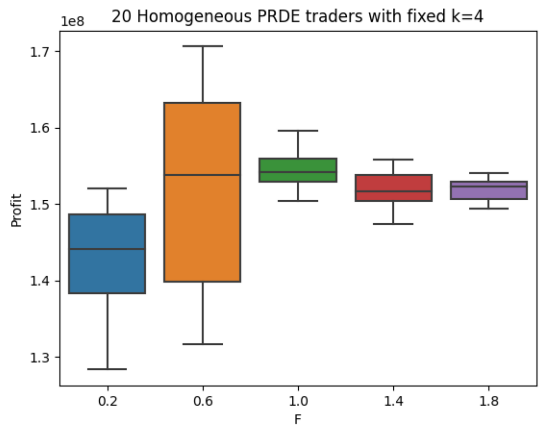

Fig. 1. The results of 5 experiments with different F values, each with 20 Homogeneous PRDE traders with fixed k=4.

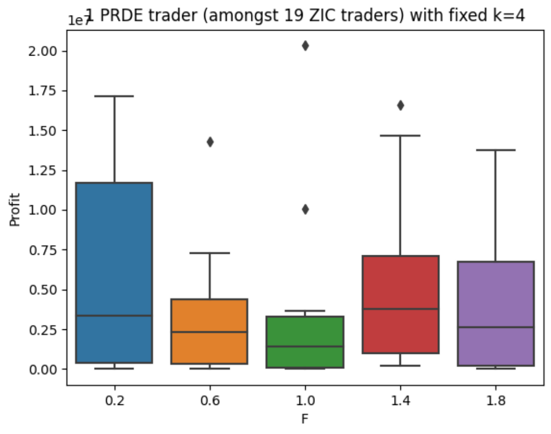

Fig. 2. The results of 5 experiments with different F values, each with 1 PRDE trader amongst 19 ZIC traders with fixed k=4.

TABLE I
AVERAGE PROFIT FOR HOMOGENEOUS PRDE WITH FIXED K

Experiment	p-value
Homo F	0.00551
Homo k	5.99e-6
1toM F	0.761
1toM k	0.842
BG k	0.405
BG F	0.545

For Table I, the experiment names are coded as follows: ”Homo” refers to homogeneous, ”1toM” refers to 1-to-many and ”BG” means balanced group, while k and F are the independent variable.

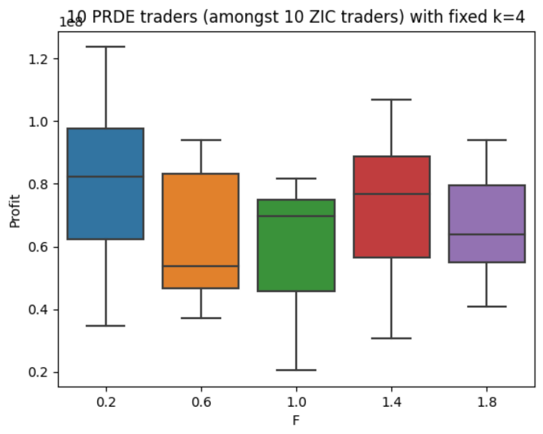

Fig. 3. The results of 5 experiments with different F values, each with 10 PRDE traders and 10 ZIC traders with fixed k=4.

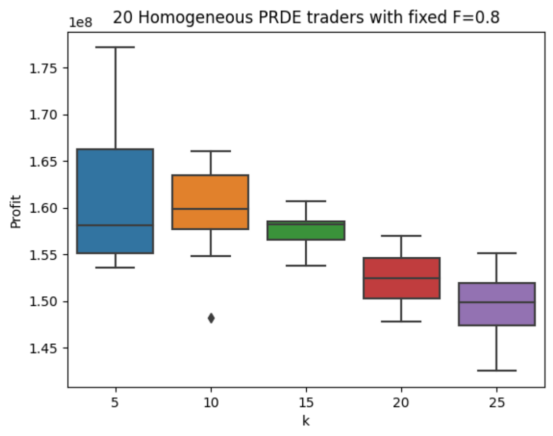

Fig. 4. The results of 5 experiments with different F values, each with 20 Homogeneous PRDE traders with fixed k=4.

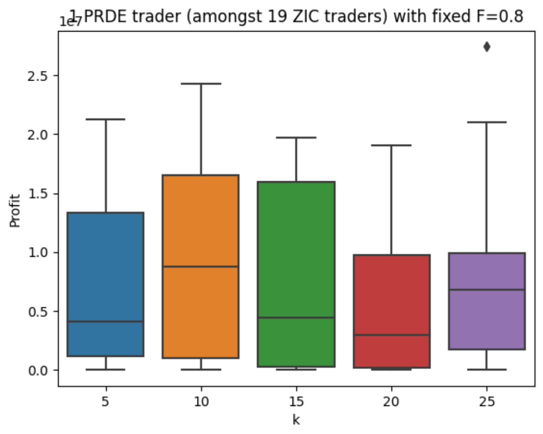

Fig. 5. The results of 5 experiments with different F values, each with 1 PRDE trader amongst 19 ZIC traders with fixed k=4.

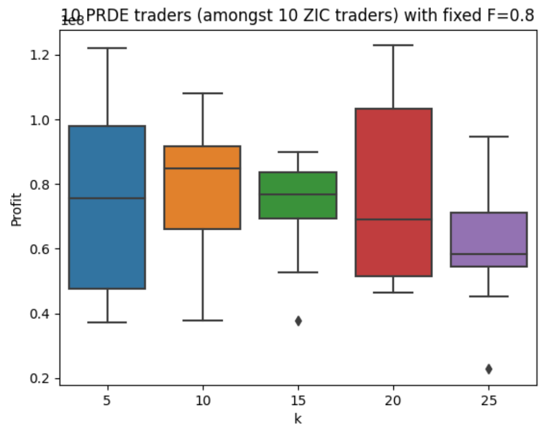

Fig. 6. The results of 5 experiments with different F values, each with 10 PRDE traders and 10 ZIC traders with fixed k=4.

III. Adaptive Differential Evolution

A. *Algorithm Implementation*

The experiments in section II only scratch the surface of the parameter space for k and F. Finding the optimal choice of parameters using this approach would take much more compute time. I tried several approaches, including via an 8-core MacBook Air with M1 Apple Silicon, single-core t2.micro AWS instances and University of Bristol lab machines. The t2.micros were found to be highly unreliable and the MacBook was surprisingly slow (though it was running day-to-day tasks as well). The University lab machines provided fastest performance but were publicly accessible to all students. Even with compute resource availability aside, [6] also raise the point that parameter optimisation through trial-and-error is specific to the particular problem; their suggestion is to use adaptive algorithms instead.

The motivation for extending PRDE was to develop an adaptive algorithm that can tune the DE parameters automatically throughout the market simulation. This prevents the need to perform trial-and-error experiments and gives rise to a trader that can adapt its DE parameters to different market configurations. Comparisons in [7] showed that the JADE algorithm consistently performed at least as well as alternative adaptive algorithms. Despite the fact that they were using minimisation problems on structural integrity as benchmarks—as opposed to a profit maximisation problem in dynamic markets—they claim that the advantage of JADE is that it is a generalised solution that should be adaptable for a variety of optimisation problems.

In 2009, Zhang and Sanderson proposed the JADE algorithm in [9]. It follows the same generic structure of the standard DE algorithm with three main differences. The first is that rather than using DE/rand/1, its mutation strategy is DE/current-pbest, which makes use of a randomly selected strategy $s^p_{i,best}$ from the top $100p\%, p \in [0,1]$ solutions when generating a mutant strategy $s_{i,y}$. The complete mutation formula is:

$$s_{i,y} = s_{i,x} + F_i \cdot (s^p_{i,best} - s_{i_x}) + F_i \cdot (s_{i,r} - s_{i_a}) \quad (4)$$

Where $s_{i,x}$ is the current strategy being evaluated, $s_{i,r} \neq s_{i,x}$ is a strategy randomly chosen from the population and $s_{i,a} \neq s_{i,x} \neq x_{i,r}$ is a strategy randomly chosen from the union of the population and the optional archive. The second difference is the inclusion of this archive; a set of previously beaten strategies. Its use in the formula is to encourage exploration in the opposite direction to these failed strategies, towards a more promising region of the solution space. The final difference is that it adapts the control parameters F_i and CR_i. CR_i is a parameter used in crossover – since we are only concerned with scalar strategy values rather than vectors, we disregard this crossover step. It adapts its F_i value from a Cauchy distribution before each DE step, using a location parameter μ_F.

$$F_i = rand_{C_i} \cdot (\mu_F, 0.1) \quad (5)$$

The location parameter itself is updated every k generations based on a set of the best performing F_i values. The inclusion of the p best solutions and archive help with population diversity and therefore strategy space exploration. This in turn means the algorithm is less likely to get stuck at local minima and converges to optimal results faster.

I modified the PRZI trader in BSE to include a JADE optimiser. PRAD (Parameterised-Response Adaptive Differential evolution) seemed a fitting name. Generally speaking, I aimed to integrate the JADE algorithm as closely as possible to the description in [9] while retaining the existing style of the BSE approach. The notable distinctions are as follows. Although both the original DE algorithm and JADE stopped parameter evolution after some time, PRAD continues its DE process indefinitely, just like PRDE, since it needs to be able to adapt to a dynamic market. Rather than iterating through each strategy in turn, PRAD randomly chooses the current strategy to evaluate, again, to keep in line with PRDE. However, it does maintain a counter so that after k lots of DE generations, it can perform the same update step of μ_F as in JADE. Finally, PRAD also kept PRDE's vector-perturbation method to avoid population convergence. Thanks to JADE, this event is even less likely, but it can't hurt to have an extra layer of safety.

B. *Experiments*

In order to test the performance of PRAD, I chose recommended values of the JADE constants $p = c = 0.1$ where p determines how many of the best strategies to include in DE/current-pbest and c is a constant used in the generation of F. I also used an initial F_i value of 0.8 although this shouldn't have an effect; PRAD generates a new F_i before each DE generation.

I followed the same market configuration as [3] as far as possible, in order to best compare the performance of PRAD to PRDE. As with Cliff's original experiments, there was a perfectly elastic supply and demand schedule of 60 and 140 respectively (i.e. all sellers were given the same, unchanging limit price of 60 and similarly with 140 for buyers). The traders were made up of a homogeneous population of 60 PRAD traders with a 50:50 split between buyers and sellers. Each PRAD trader had a fixed k value of 4, just like the original PRDE experiments. All other market configurations were kept the same as the default BSE settings except for the number of simulated trading days. This was set to 50 rather than 300, which is long enough for PRDE profits to converge as demonstrated in [3].

Although this first experiment was a good way to directly compare PRAD to historic PRDE results, it failed to properly make use of PRAD's adaptive algorithm. This is because with a strategy population size of $k = 4$ and $p = 0.1$, the top p best strategies to choose from are those from $i = 0$ to

$$\lfloor p \cdot k \rfloor = \lfloor 0.1 \cdot 4 \rfloor = 0 \quad (6)$$

In other words, only the very best strategy gets used in the mutation step each time, reducing the strategy to DE/current-best rather than DE/current-pbest. To rectify this, I ran more trials, this time with $k = 15$ and $p = 0.2$, which allows the top three best strategies to be sampled from in the mutation step.

C. Results

Fig. 7 displays the results of these three simulations, each repeated over 15 independent trials. The graph plots profit per second (PPS) against time, in order to directly compare with those results first published in [3]. Although the PRDE lines follow the same trend as in [3] (early convergence followed by some variation in PPS over time), the range was significantly higher. Despite following the market configuration details as closely as possible, it is likely there are still some discrepancies between the setups that caused this difference.

Despite this, it is clear to see the improvement of PRAD performance, even with the same population size of 4. It converges to higher profit margins much faster and displays lower variation. This minimised variance is made even more extreme by setting $k = 15$. The convergence is slower, but this is likely explained by the algorithm taking more time in having to iterate over a larger strategy population.

As with the previous experiments in section II-B, I performed statistical analysis on the total profits extracted by the traders. Again, this involved performing an Anova between the total profits of the three experiments (PRDE, PRAD with $k = 4$ and PRAD with $k = 15$). As might be expected from the visualisations in Fig. 7, the variation across trials was confirmed to be very low. The pvalue generated by the Anova was $3.41e - 29$, which also demonstrated that the null hypothesis could be rejected, suggesting that the PPS between each experiment was indeed significantly different. Table II shows that the average profit extracted by PRAD traders outperformed PRDE.

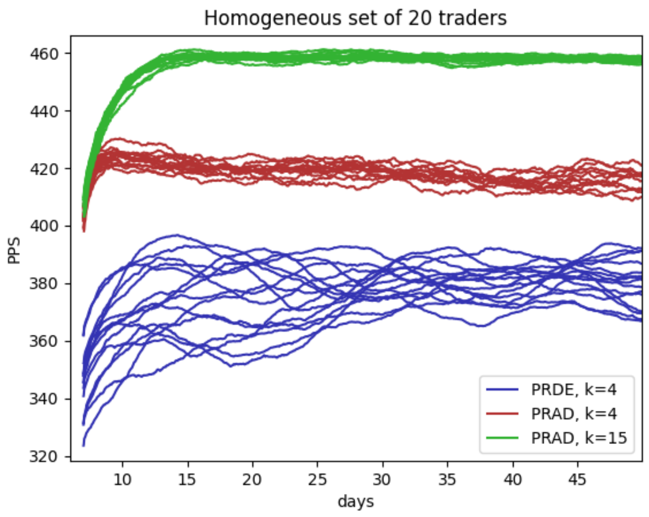

Fig. 7. The results of 3 distinct experiments, each testing homogeneous populations of 20 traders across 15 independent trials. Each line represents a single trial and each colour corresponds to a different experiment. Profit per second is measured along the y-axis and time in days along the x-axis.

TABLE II
AVERAGE PROFIT FOR 20 HOMOGENEOUS TRADERS

PRDE	PRAD, k=4	PRAD, k=15
1.80	1.62	1.95
(±0.00548)	(±0.0260)	(±0.00187)

IV. CONCLUSIONS

The parameters of the differential evolution algorithm used in PRDE determine its behaviour with respect to adapting its strategy value over time. A theoretical analysis of the algorithm sheds light on their expected effects. In particular, a higher value of the population size k increases the variety of strategies considered at any time at the expense of higher evaluation periods. The F parameter affects the amount of mutation applied when generating potential replacement strategies. Higher values of F encourage a greater exploration of the strategy space but may fail to converge to an optimum if too high.

The majority of comparison experiments introduced in section II-B were inconclusive with the exception of the homogeneous populations of traders. One possible reason for this is the small number of trials performed. The range in parameter values explored was also limited. With higher compute time it may be possible to draw better conclusions on the optimal values of k and F in these alternative market systems. Despite the experiments' shortcomings, this in itself helped illustrate the need for an adaptive approach.

Generally speaking in the homogeneous cases, values of F centred around 1.0 performed best while smaller values of k gave higher profits.

The use of PRAD with its adaptive JADE implementation removed the need to fine tune parameters through trial-and-error. Instead, the algorithm clearly demonstrated its ability to outperform PRDE by adapting its F value throughout the course of the simulation. This improvement was further exaggerated with a higher population size of $k = 15$, where PRAD was given the chance to make full use of its DE/current-pbest mutation strategy.

Future extensions to this paper could explore a larger volume of trials on the k and F parameters, potentially through the use of more powerful cloud compute resources or a supercomputer. Other experiments could explore the effect of these parameters in more dynamic markets, such as with market shocks. Some of the specifics of the BSE implementation could also be investigated, such as whether the vector-perturbation method is actually needed for PRAD. PRAD is an adaptive algorithm purely for its F value. Further investigations could be carried out into the use of self-adaptive algorithms such as SPDE or DESAP where the control parameters themselves undergo mutation. One shortcoming is that PRAD only adapts its F value; perhaps research could be performed into algorithms that dynamically resize the strategy population throughout a simulation.

REFERENCES

[1] D. Gode and S. Sunder, "Allocative Efficiency of Markets with Zero-Intelligence Traders: Market as a Partial Substitute for Individual Rationality," Journal of Political Economy, 101(1):119–137, 1993.

[2] D. Cliff, "Parameterized-Response Zero-Intelligence Traders," SSRN:3823317, 2021.

[3] D. Cliff, "Metapopulation Differential Co-Evolution of Trading Strategies in a Model Financial Market," SSRN:4153519, 2022

[4] ——,BristolStockExchange:open-sourcefinancialexchangesimulator. https://github.com/davecliff/BristolStockExchange, 2012–2022.

[5] R. Storn, K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces", Journal of Global Optimization, 11, 341 359, 1997.

[6] G.Wu,R.Mallipeddi,P.Suganthan,R.Wang,andH.Chen,"Differential evolution with multi-population based ensemble of mutation strategies," Information Sciences, vol. 329, pp. 329–345, 2016.

[7] M. Georgioudakis and V. Plevris, "A comparative study of differential evolution variants in constrained structural optimization," Frontiers in Built Environment, vol. 6, no. 102, pp. 1–14, 2020.

[8] Tesauro, G., and Das, R. 2001. High-performance bidding agents for the continuous double auction. In Third ACM Conference on Electronic Commerce, 206–209.

[9] J. Zhang and A. Sanderson, "JADE: Adaptive Differential Evolution with Optional External Archive," IEEE Transactions on Evolutionary Computation, vol. 13, no. 5, pp. 945–958, 2009.