Chapter 4

NOVEL EXTENSIONS TO DYNDE

The purpose of this chapter is to propose and evaluate new adaptations to the DynDE algorithm. These adaptations include: an approach that allows sub-populations to compete for fitness evaluations based on performance and an approach aimed at improving the effectiveness of the exclusion component of DynDE. The two adaptations are combined to form a third approach. The new approaches are empirically compared to DynDE and analysed in terms of their performance characteristics on a large set of benchmark problems.

4.1 Introduction

The previous chapter described the DynDE algorithm as a multi-population, DE-based algorithm aimed at DOPs. The literature review found that DynDE employs strategies that are generally used by other algorithms to optimise dynamic environments (refer to Table 3.1). DynDE is a comparatively simple algorithm which has been shown to be effective in solving DOPs [Mendes and Mohais, 2005].

A potential problem with DynDE is the fact that function evaluations are equally distributed to all the sub-populations that are used to track optima. Function evaluations are wasted on optimising local optima, whereas merely locating the general location of

local optima would suffice for the purpose of tracking the optima. The algorithms of Kamosi *et al.* [2010b] and Novoa-Hernández *et al.* [2011], which were developed concurrently with the algorithms presented in this study, periodically remove badly performing sub-populations from the evolution process to allow more function evaluations for the better performing sub-populations. The competitive population evaluation (CPE) approach that is presented in this chapter proposes that sub-populations should compete for fitness evaluations. Lower error values can be found using fewer function evaluations, by evolving sub-populations sequentially based on performance, rather than in parallel.

DynDE makes use of exclusion to prevent sub-populations from converging to the same optima by reinitialising a sub-population that moves within the exclusion radius of another. The exclusion approach ignores the possibility that multiple optima may exist within the exclusion radius, which would result in only one of the optima being tracked. This problem is addressed by a second novel adaptation to DynDE, called reinitialisation midpoint check (RMC). RMC changes the mechanism by which exclusion is implemented in DynDE to avoid reinitialisation of sub-populations that are not positioned on the same optimum. The fitness of the midpoint between the best individuals in the sub-populations is checked to determine whether the sub-populations are located on the same optimum. Exclusion is not performed if multiple optima are detected.

This chapter is structured as follows: Section 4.2 describes how the DynDE exclusion threshold calculation can be changed so that information regarding the number of optima in the fitness landscape is not required. CPE is presented in Section 4.3. Section 4.4 describes the RMC approach. The combination of CPE and RMC to form competing differential evolution (CDE) is discussed in Section 4.5. Section 4.6 describes experiments to investigate the performance and scalability of DynDE, CPE, RMC and CDE. Specific research questions are formulated and the experimental procedure is given in Sections 4.6.1 and 4.6.2. Section 4.6.3 investigates appropriate settings for the sub-population size and number of Brownian individuals for DynDE. The scalability of DynDE, CPE, RMC and CDE as the change period, number of dimensions, severity of changes, change types, and underlying function are varied, is investigated in Section 4.6.4. Section 4.6.5 investigates whether the novel approaches proposed in this chapter significantly improve DynDE. The convergence behaviour of CDE is compared and contrasted with that of DynDE in Section

4.6.6. The performance of CDE is compared in Section 4.6.7 to that of DynDE, using the average lowest error before changes in the environment as performance measure.

A comparison of CDE to other algorithms in the literature is given in Section 4.7. The general applicability of the approaches presented in this chapter is investigated by incorporating the approaches into *jDE* in Section 4.8. Final conclusions are drawn in Section 4.9.

4.2 Alternative Exclusion Threshold Approach

This thesis considers dynamic environments where information regarding the number of optima is not available to the optimisation algorithm. Section 3.4.1.2 described the calculation of the exclusion threshold as used by Mendes and Mohais [2005] and Blackwell and Branke [2006]. The exclusion threshold, as calculated using equation (3.1), assumes that the number of optima is known. Knowledge of the number of optima is generally not available when solving a DOP, consequently, this thesis proposes that the exclusion threshold rather be calculated using the number of sub-populations as follows:

$$r_{excl} = \frac{V_{max,F} - V_{min,F}}{2{n_k}^{\frac{1}{n_d}}} \tag{4.1}$$

where n_k is the number of sub-populations, and $V_{max,F}$ and $V_{min,F}$ are the upper and lower search range of function F in the n_d dimensions (assuming equal ranges for all dimensions). The threshold is thus dependent on the number of available sub-populations. A small number of sub-populations results in a large exclusion threshold, thus assigning a large area of the search space to each sub-population. Conversely, a large number of sub-populations results in a small exclusion threshold, consequently assigning a relatively small area of the search space to each sub-population. Equation (4.1) was also used by Blackwell [2007] in the self-adapting multi-swarms algorithm discussed in Section 3.3.3.2.

Note that, for the moving peaks benchmark (MPB) experiments discussed in this chapter, the same number of populations as the number of optima was used. The results can thus be directly compared to results from researchers that made use of equation (3.1) to calculate the exclusion threshold, since the same threshold value was used.

4.3 Competitive Population Evaluation

The effectiveness of an algorithm on static optimisation problems is usually measured by the error of the best solution found at the end of the optimisation process. Although many approaches aim to reduce the execution time of optimisation algorithms (i.e. reach a low error value as soon as possible), the error during the course of the optimisation process is of secondary concern. In contrast, optimisation in dynamic environments implies that a solution is likely to be required at all times (or at least just before changes in the environment occur), not just at the termination of the algorithm. In these situations, it is imperative to find the lowest error value as soon as possible after changes in the environment have occurred.

The offline error performance measure (discussed in Section 2.5.5) measures an algorithm's efficiency at finding solutions quickly by calculating the performance of an optimisation algorithm as the average of the lowest error value over every function evaluation, as opposed to averaging errors immediately prior to changes in the environment. The offline error can thus be reduced by finding solutions earlier (without necessarily finding a better solution). An algorithm that finds solutions faster will also be beneficial in situations where evaluation criteria are only concerned with the error value immediately prior to changes in the environments (as is the case with performance measure number two in Section 2.5.5), since the algorithm will be more effective than a standard algorithm, when the number of function evaluations between changes is reduced. A dynamic optimisation algorithm can thus be improved, not only by reducing the error, but also by making the algorithm reach its lowest error value (before a change occurs in the environment) in fewer function evaluations.

The above argument is the motivation for a new approach to finding solutions earlier which is proposed in this thesis, namely competitive population evaluation (CPE) [du Plessis and Engelbrecht, 2012b]. CPE is an extension to DynDE discussed in Section 3.4.1. The primary goal of the new approach is not to decrease the error value found by DynDE, but rather to make the algorithm reach the lowest error value in fewer function evaluations. The proposed algorithm aims to achieve this by initially allocating all function evaluations, after a change in the environment, to the sub-population that has the current

best solution. Thereafter function evaluations are allocated to other sub-populations.

The mechanism used by CPE to allocate function evaluations is based on the performance of sub-populations. The best-performing sub-population is evolved on its own until its performance drops below that of another sub-population. The new best-performing sub-population then evolves on its own until its performance drops below that of another sub-population. This process is repeated until a change in the environment occurs. Ideally, CPE would locate the global optimum early, while locating local optima later. CPE thus differs from DynDE in that peaks are not located in parallel, but sequentially. The CPE process is detailed in Algorithm 12. Changes in the environment are always followed by the evolution of all sub-populations for two generations. These two generations are necessary in order to calculate the performance value, $\mathcal{P}_k$, for each of the P_k sub-populations. However, after the initial two generations, the CPE process evolves only one sub-population per generation. CPE is thus responds directly to changes in the environment, unlike algorithms like CPSOR (refer to Section 3.3.3.2).

Algorithm 12: Competitive population evaluation

while *termination criterion not met* **do**
  Allow the standard DynDE algorithm to run for two generations;
  repeat
    for $k = 1, \ldots, n_k$ **do**
      Calculate the performance value, $\mathcal{P}_k(t)$
    end
    Select sub-population P_a such that $\mathcal{P}_a(t) = \min_{k=1,\ldots,n_k} \{\mathcal{P}_k(t)\}$;
    Evolve only sub-population P_a using DE for one generation;
    $t = t + 1$;
    Perform exclusion according to Algorithm 7;
    Create Brownian individuals;
  until *a change in the environment occurs*;
end

The performance value, $\mathcal{P}_k$, of sub-population P_k depends on two factors: The current

fitness of the best individual in the sub-population and the error reduction of the best individual during the last evaluation of the sub-population. Let n_k be the number of sub-populations, $\vec{x}_{best,k}$ the best individual in sub-population P_k, and $F(\vec{x}_{best,k}, t)$ the fitness of the best individual in sub-population P_k during iteration t. The performance $\mathcal{P}_k(t)$ of population P_k, after iteration t, is given by:

$$\mathcal{P}_k(t) = (\Delta F(\vec{x}_{best,k}, t) + 1)(R_k(t) + 1) \tag{4.2}$$

where

$$\Delta F(\vec{x}_{best,k}, t) = |F(\vec{x}_{best,k}, t) - F(\vec{x}_{best,k}, t-1)| \tag{4.3}$$

For function maximisation problems, $R_k(t)$ is calculated as:

$$R_k(t) = |F(\vec{x}_{best,k}, t) - \min_{a=1,\ldots,n_k} \{F(\vec{x}_{best,a}, t)\}| \tag{4.4}$$

and for function minimisation problems,

$$R_k(t) = |F(\vec{x}_{best,k}, t) - \max_{a=1,\ldots,n_k} \{F(\vec{x}_{best,a}, t)\}| \tag{4.5}$$

The best performing sub-population is, therefore, the sub-population with the highest product of fitness and improvement. The motivation for the addition of 1 to the first and second terms in equation (4.2) is to prevent a performance value of zero being assigned to a sub-population. Without the addition in the second term, the least fit population will always be assigned a performance value of zero (since the product of the first and second term will be zero) and will never be considered for searching for an optimum. Similarly, without the addition in the first term, a good performing population that does not show any improvement during a specific iteration, is assigned a performance value of zero and will never be considered for evaluation again. The addition of 1 to the first and second terms is thus included to ensure that every sub-population could, potentially, have the highest performance value and subsequently be given function evaluations.

The absolute values of $\Delta F(\vec{x}_{best,k}, t)$ and $R_k(t)$ are taken to ensure that the performance values are always positive. When a population is reinitialised due to exclusion (see Section 3.4.1.2), the fitness of the best individual is likely to be worse than before reinitialisation (since the sub-population now consists of randomly generated individuals).

This results in a large $\Delta F(\vec{x}_{best,k}, t)$ value. The sub-population is consequently assigned a relatively large performance value, making it likely that the sub-population would be allocated fitness evaluations in the near future.

The CPE approach can be clarified by an example. Figure 4.1 plots the error per function evaluation of the best individual in each of three sub-populations (labelled Population 1, Population 2 and Population 3), found during an actual run of the DynDE algorithm on the MPB using Scenario 2 settings. During the period that is depicted, no changes occurred in the environment. Note that one of the sub-populations (Population 3) converged to a global optimum, as is evidenced by its error value approaching zero, while the others likely converged to local optima.

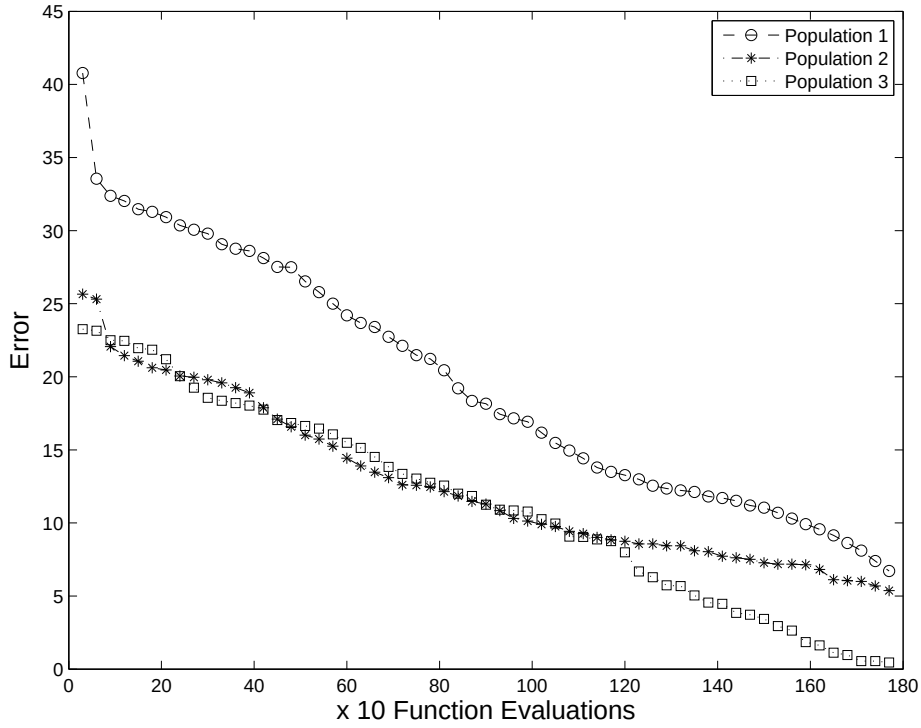

Figure 4.1: Error profile of three DynDE populations in a static environment

The performance of each of the populations was calculated using equation (4.2) on page 111. A plot of the performance of each of the populations is given in Figure 4.2. Population 2 and Population 3 alternated between having the highest performance value, for approximately the first half of the period depicted. After the first half of the depicted

period, Population 3 consistently received the highest performance value, as it converged to a global optimum. Population 3 received, on average, a higher performance value than the other two populations. Population 1, which had the highest overall error, received a relatively low average performance value. Note that it was only after about 1 000 function evaluations that Population 2 and Population 3 received a lower performance value than the initial performance value of Population 1.

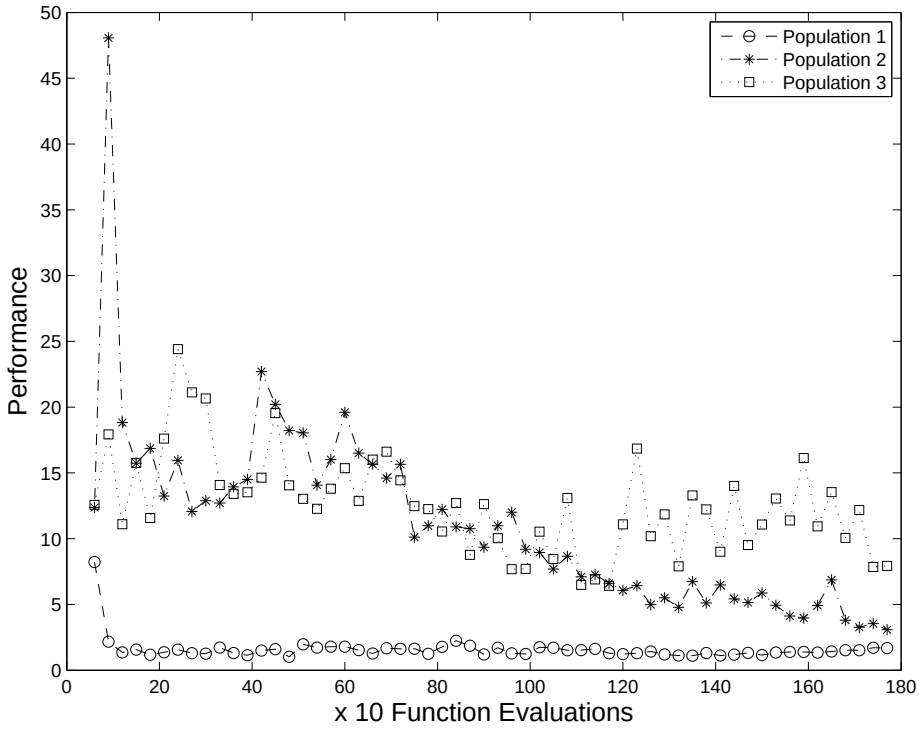

Figure 4.2: Performance, $\mathcal{P}$, of each population

The behaviour of each of the populations, when using the CPE process (refer to Algorithm 12) to selectively evolve sub-populations on their own based on their respective performance values, can now be observed by using the calculated performance values. The plots of the error of each of the populations per function evaluation when using CPE are given in Figure 4.3. Population 2 and Population 3 were alternately evolved for the first few iterations, followed by a period where only Population 2 was evolved. After about 250 function evaluations, Population 3 was evolved (except for a few intermittent iterations around 700 function evaluations) until it converged to the global optimum at about 1 000

function evaluations. Population 2 was evolved during the next period which lasted until about 1 200 function evaluations, while Population 1 was evolved during the last period.

Note that the lowest error was reached after 1 000 function evaluations in Figure 4.3, while this point was only reached after 1 800 function evaluations in Figure 4.1. The performance values of the three populations when using CPE are given in Figure 4.4. Observe that for considerable periods, the performance values of some of the sub-populations remain constant. This occurs because only one sub-population is evolved at a time, during which the performance value of the other populations remains unchanged.

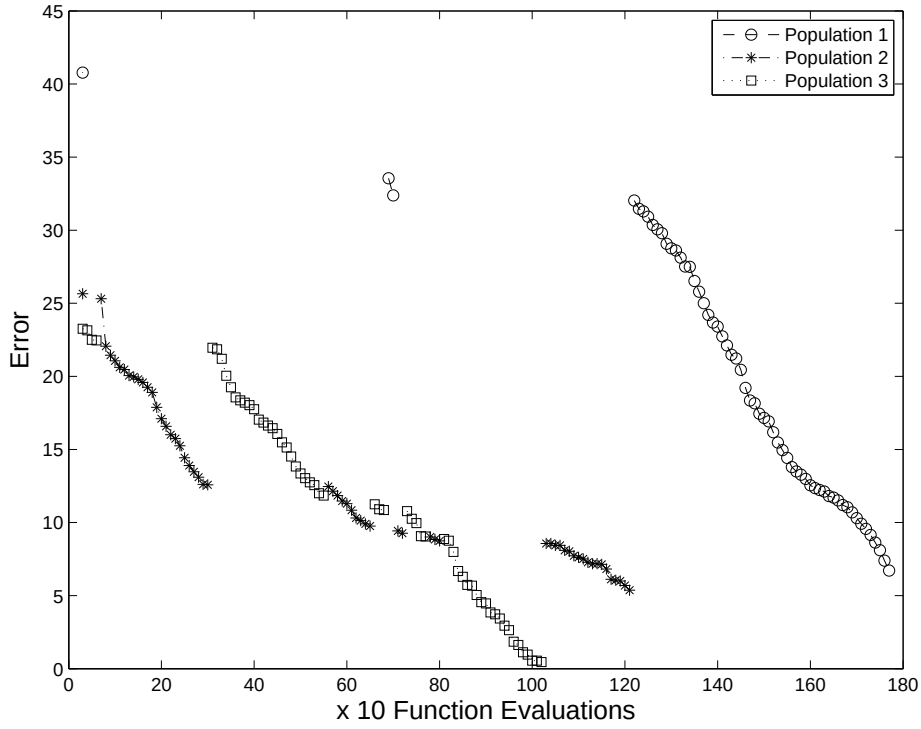

Figure 4.3: Error profile of three populations when using competitive evaluation

More successful sub-populations are clearly allocated more function evaluations earlier in the evolution process. Optima are located sequentially by CPE and in parallel by DynDE. Figure 4.5 depicts the effect that using CPE had on the offline error for the three sub-populations example. The curve of the offline error, when competing sub-populations are used, exhibits a steeper downward gradient than the curve of normal DynDE, due to earlier discovery of the global optimum. Overall, the offline error was reduced by more

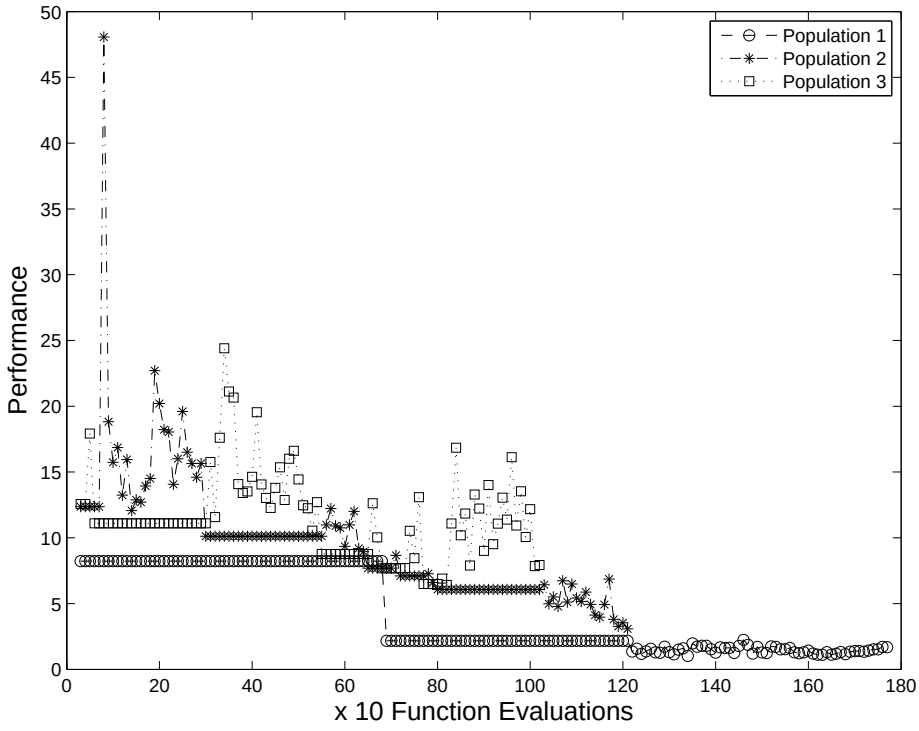

Figure 4.4: Performance, $\mathcal{P}$, of each population when using competitive evaluation

than 30% after 1 800 function evaluations.

The lowest error value is reached sooner by competitively choosing the better performing populations to evolve before other populations, thus reducing the average error. This technique has the added advantage that better performing populations will receive more function evaluations that would otherwise have been wasted on finding the maximum of the sub-optimal peaks. The overall error value should consequently also be reduced.

An advantage of CPE is that it only utilises information that is available in normal DynDE, so that no extra function evaluations are required. A potential disadvantage of the CPE process is that too few function evaluations may be allocated to the weaker sub-populations. Consequently, the weaker sub-populations may not locate local optima which may become global optima after changes in the environment. This is more likely to be a problem in situations where changes in the environment are infrequent, since normal DynDE would have enough function evaluations available to optimise all optima thoroughly. The benefits of locating optima early are also expected to diminish as the

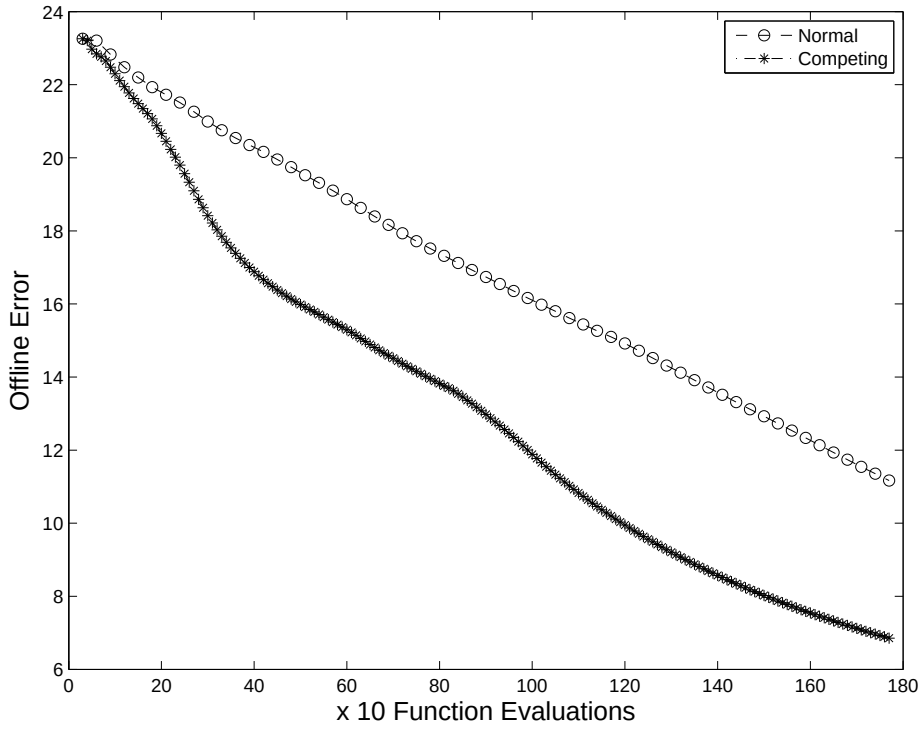

Figure 4.5: Comparison of offline error for normal and competitive evaluation

period between changes in the environment increases, because the offline error would be averaged over a larger number of function evaluations, thus reducing the impact of initial lower errors. CPE is thus expected to be most applicable to rapidly changing dynamic environments.

4.4 Reinitialisation Midpoint Check

The exclusion approach, which was described in Section 3.4.1.2, causes DynDE to reinitialise the weaker sub-population when two sub-populations are located within the exclusion radius of each other. This approach does not take into account the case when two optima are located extremely close to each other, i.e. within the exclusion threshold of one another. One of the sub-populations will be reinitialised in these situations, leaving one of the optima unpopulated. This section proposes that this problem be partially remedied by determining whether the midpoint between the best individuals in each sub-population

constitutes a higher error value than the best individuals of both sub-populations. If this is the case, it implies that a trough exists between the two sub-populations and that neither should be reinitialised (refer to Figure 4.6, scenario A). This approach is referred to in this thesis as the reinitialisation midpoint check (RMC) approach. Algorithm 13 details how exclusion will be performed when RMC is used (assuming function minimisation).

Algorithm 13: RMC Exclusion

```
for k_1 = 1, ..., n_k do
    for k_2 = 1, ..., n_k do
        if ||x_best,k1 − x_best,k2||_2 < r_excl and k_1 ≠ k_2 then
            Let x_mid = (x_best,k1 + x_best,k2)/2;
            if F(x_mid) > F(x_best,k1) and F(x_mid) > F(x_best,k2) then
                if F(x_best,k1) < F(x_best,k2) then
                    Reinitialise population P_k2
                else
                    Reinitialise population P_k1
                end
            end
        end
    end
end
```

It is apparent that RMC does not work in all cases. Scenarios B and C of Figure 4.6 depict situations where multiple optima within the exclusion threshold are not detected by a midpoint check. Scenario C further constitutes an example where no point between the two optima will give a higher error, thus making it impossible to detect two optima by using any number of intermediate point checks.

This approach is similar to, but simpler, than *hill-valley detection* suggested by Ursem [2000] (described in Section 3.3), since only one point is checked between sub-populations. The midpoint check approach provides a method of detecting multiple optima within the exclusion threshold without being computationally expensive or using too many function evaluations, since only one point is evaluated.

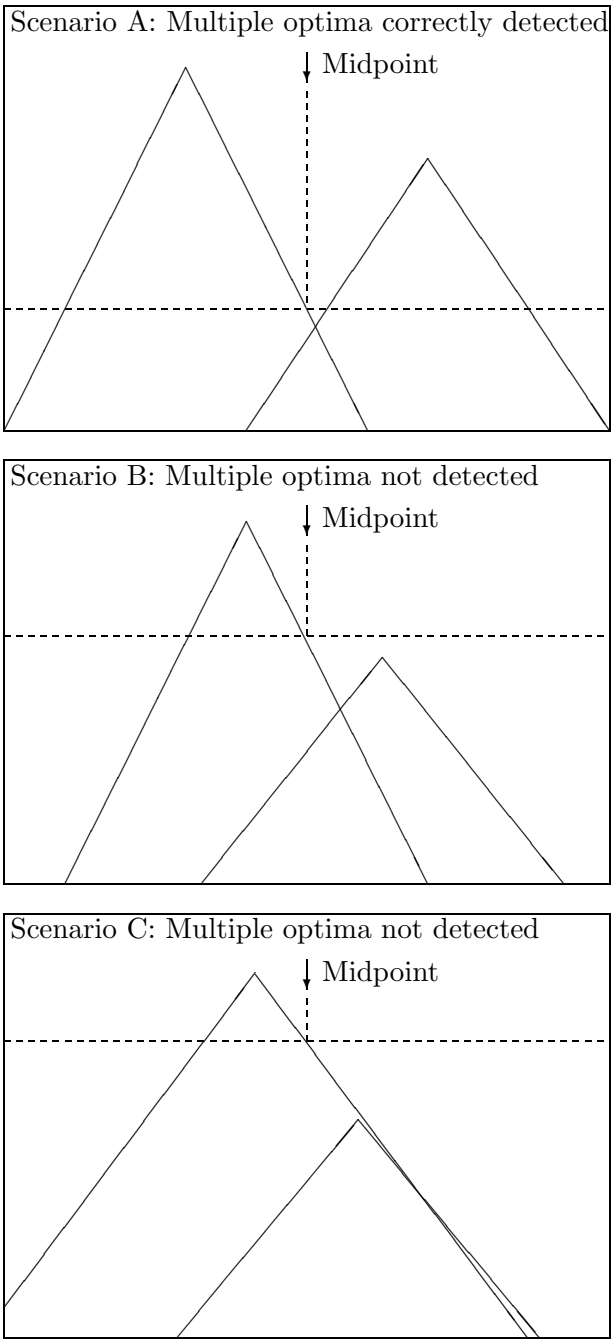

Figure 4.6: Midpoint checking scenarios

4.5 Competing Differential Evolution

The RMC and CPE algorithms are mutually independent and can thus be combined into a single algorithm. This algorithm is referred to as competing differential evolution (CDE). CDE thus consists of the standard DynDE algorithm, with the addition of a midpoint check before reinitialising populations when they are within the exclusion threshold of each other, and the evaluation of populations based on their performance. The pseudo-code for CDE is identical to that of CPE given in Algorithm 12, but with the exclusion step using Algorithm 13.

4.6 Experimental Results

This section describes the empirical analysis of DynDE, CPE, RMC and CDE and discusses the results. Experiments are designed to answer specific research questions given in Section 4.6.1. The general experimental procedure is given in Section 4.6.2. The analysis pertaining to each research question is given in Sections 4.6.3 to 4.6.7 respectively.

4.6.1 Research Questions

The following research questions were identified as pertinent for analysing the algorithms proposed in this chapter:

1. *What are effective settings for sub-population size and number of Brownian individuals?* Mendes and Mohais [2005] recommended using a small sub-population size of six individuals. This can be seen as counterintuitive since larger sub-population sizes should increase diversity. The effect of larger sub-populations is investigated. Various settings for the number of Brownian individuals are also investigated to determine the most effective value.

2. *How do DynDE, RMC, CPE and CDE scale under factors that influence the complexity of a DOP?* Section 2.5.2 described the factors that determine an optimisation algorithm's ability to locate optima in a dynamic environment. The factors that are investigated are: the number of dimensions of the search space, the number of function evaluations between changes in the environment (change period), the severity of

changes, change types, and the function describing the fitness landscape. The scalability study analyses the performance of the algorithms when varying combinations of the previously mentioned factors.

3. *Are RMC, CPE and CDE more effective dynamic optimisation algorithms than DynDE?* The results found when measuring the performance of the algorithms suggested in this chapter on benchmark problems are compared to the results of DynDE.

4. *How does the convergence behaviour of CDE differ from that of DynDE?* The convergence behaviours of DynDE and CDE are compared in terms of diversity, current error and the resulting offline error to assist in explaining the trends observed in the analyses of research questions 1 and 2.

5. *How does the average lowest error, found just before changes in the environment, differ between DynDE and CDE?* CPE aims to locate optima earlier, but is not specifically designed to achieve a lower overall error value before changes occur in the environment. Similarly, RMC is not directly focused on lowering the error, but rather on improving the optima tracking process. This research question is investigated to determine how the performance of the new CDE algorithm differs from that of CDE in terms of the average lowest error values before changes in the environment.

The experimental procedure followed to investigate the research questions is outlined in Section 4.6.2.

4.6.2 Experimental Procedure

The performance of DynDE, CPE, RMC and CDE was evaluated on the MPB and the GDBG. The simplicity of the MPB makes it ideal for investigating the effect of different change severities, while the GDBG provides functionality to investigate different change types and underlying functions. Accordingly, a *standard set* of experiments is defined here. Variations of the Scenario 2 settings on the MPB which are included in the standard set are given in Table 4.1. The settings include several values for change severity and

both the cone and sphere peak functions. The combinations of change severity and peak function results in 12 MPB variations which are included in the standard set.

Table 4.1: MPB Scenario 2 Variations

Setting	**Value**
Number of dimensions (n_d)	5
Number of Peaks	10
Max and Min Peak height	[30,70]
Max and Min Peak width	[1.0,12.0]
Change period (Cp)	5000
Change severity (C_s)	1.0, 5.0, 10.0, 20.0, 40.0, 80.0
Height severity	7.0
Width severity	1.0
Function (F)	Cone, Sphere
Correlation	0.0

The GDBG problems considered here consist of six change types (T_1 to T_6) and six functions (F_1 to F_6). Refer to Section 2.5.4 for details on these functions and change types. The creators of the GDBG suggested that two instances of F_1 be used: one containing 10 peaks and one containing 50 peaks. These instances are denoted by F_{1a} and F_{1b} respectively. The combinations of change types and functions thus yield a total of 42 GDBG environments that are included in the standard set. The GDBG variations are summarised in Table 4.2. The default values for the number of dimensions and change period are in accordance with the settings for the IEEE WCCI-2012 competition on evolutionary computation for dynamic optimisation problems [Li *et al.*, 2011].

Table 4.2: GDBG Variations

Setting	**Value**
Number of dimensions (n_d)	5
Change period (Cp)	50000
Function (F)	F_{1a}, F_{1b}, F_2, F_3, F_4, F_5, F_6
Change type (Ct)	T_1, T_2, T_3, T_4, T_5, T_6

The 54 environments in the standard set thus contain 12 environments from the MPB and 42 environments from the GDBG. The standard set is varied to investigate the effect of different numbers of dimensions and various values for the change period (number of function evaluations between changes in the environment). The variations on the standard are given in Table 4.3.

Table 4.3: Standard Set Variations

Setting	**Values**
Number of dimensions (n_d)	5, 10, 25, 50, 100
Change period (Cp)	100, 500, 1000, 5000, 10000, 25000, 50000, 100000

A stopping criterion of 60 changes in the environment was used for all experiments as suggested in [Li *et al.*, 2008], [Li *et al.*, 2011]. The offline error was used as performance measure for all environments and experiments were repeated 30 times to facilitate drawing statistically valid conclusions from the results. Mann-Whitney U tests were used to test statistical significance when comparing algorithms. Unless otherwise stated, the algorithms used the parameter settings described in Chapter 3.

4.6.3 Research Question 1

What are effective settings for sub-population size and number of Brownian individuals?

Mendes and Mohais [2005] investigated sub-population sizes of 6 and 10 with 2 and 5 Brownian individuals respectively. Experimental results on the MPB showed that the smaller sub-population size was more effective. The experiments conducted by Mendes and Mohais [2005] can be criticized for not being extensive enough for three reasons. Firstly, only two sub-population size values (which are reasonably close to each other) were tested. Secondly, the ratio of Brownian versus normal individuals is not the same in the two tests: the sub-population of size 6 used $33.\dot{3}\%$ of each sub-population as Brownian, while the sub-population of size 10 used 50% of each sub-population as Brownian. The results may thus have been determined by the percentage of Brownian individuals rather than by the sub-population size. Thirdly, the experiments used only the MPB without investigating any variations of the Scenario 2 settings.

This section presents the results of a more extensive investigation to determine the most effective sub-population size and the appropriate proportion of Brownian individuals. The goal of the experimental investigation is to find settings that works well over a broad range of dynamic environments. Four settings for sub-population size were tested: 6, 12, 24, and 48. Six was selected as the smallest sub-population size tested, to prevent the DynDE's DE/best/2/bin scheme from always using the same five individuals to form the mutant vector. Seven values for the percentage of Brownian individuals in each sub-population were investigated: 0%, 16.$\dot{6}$%, 33.$\dot{3}$%, 50%, 66.$\dot{6}$%, 83.$\dot{3}$%, and 100%. These percentages correspond to a sub-population of size six, using zero, one, two, three to six Brownian individuals respectively. The other sub-population sizes that were tested are all multiples of six, so an integer number of Brownian individuals was used in each case.

The four sub-population sizes and the seven percentages of Brownian individuals resulted in 28 combinations of settings being investigated. The standard set of environments (described in Section 4.6.2) was used to test each of the settings. The standard set was also varied for each of the change periods and numbers of dimensions listed in Table 4.3. A total of 648 experiments was thus conducted for each of the 28 settings.

The results for each setting were compared with those of each of the other settings to determine which resulted in the lowest offline error. A Mann-Whitney U test was used in each case to determine whether differences in average offline error were statistically significant at a 95% confidence level. Table 4.4 gives a count for each setting of the number of experiments that yielded a statistically significantly lower offline error than any of the other settings. A maximum count of 17 496 can be found for each setting as 28 settings were each investigated on 648 experiments. Table 4.5 lists a similar count as Table 4.4, but here the counter was only incremented if a particular setting performed better than **all** other settings for a particular experiment. A maximum value of 648 can thus be achieved.

The results presented in Tables 4.4 and 4.5 indicate that using 16.$\dot{6}$% Brownian individuals yields the most effective algorithm. The two instances that scored the highest values in both tables occurred in rows that used 16.$\dot{6}$% Brownian individuals. The conclusion can thus be drawn that using 16.$\dot{6}$% Brownian individuals gives the best performance over a wide range of environments.

Table 4.4: Number of environments in which each setting resulted in better performance than other settings

	Sub-Population Size			
Percentage Brownian	**6**	**12**	**24**	**48**
0%	1585	5172	5396	4251
16.6̇%	14248	14613	12425	8854
33.3̇%	12891	13038	11076	7703
50%	11432	10509	8558	5774
66.6̇%	9656	8440	6432	4275
83.3̇%	7376	6173	4554	2892
100%	6159	5036	3438	2001

Table 4.5: Number of environments in which each setting performed the best

	Sub-Population Size				
Percentage Brownian	**6**	**12**	**24**	**48**	**Total Best**
0%	1	17	14	1	33
16.6̇%	251	56	6	0	313
33.3̇%	1	0	1	0	2
50%	0	0	0	0	0
66.6̇%	0	0	0	0	0
83.3̇%	0	0	0	0	0
100%	15	5	3	0	23
Total Best	268	78	24	1	

The results for the most effective number of sub-populations is less clear-cut. The highest value in Table 4.4 is for a sub-population size of 12 (14 613 out of a possible 17 496), while the highest value in Table 4.5 is for a sub-population size of 6 (251 out of a possible 648). The sub-population size of 12 outperformed more of the other settings per experiment, however the sub-population size of 6 most often performed the best per experiment. Further analysis is thus required to determine whether a sub-population size of 6 or 12 is the most effective.

Table 4.6 gives results comparing the comparative performance of the four sub-population

sizes. The percentage Brownian individuals are kept constant at 16.6̇% for this analysis. Each row of the table gives the number of times (out of a possible 648) that DynDE, using the indicated number of sub-populations, performed better than the sub-population size, shown in the respective columns. Totals are provided for the number of times that each setting performed better (given in the last column) and worse (given in the last row) than other settings.

Table 4.6: Number of environments in which each sub-population size resulted in better performance than other sub-population sizes (row vs column) when using 16.6̇% Brownian individuals

	Sub-Population Size				
Sub-Population Size	**6**	**12**	**24**	**48**	**Total Better**
6	-	340	424	487	1251
12	199	-	476	547	1222
24	150	61	-	529	740
48	98	53	20	-	171
Total Worse	447	454	920	1563	

The sub-population size of six experiments yielded the highest total number of better values (1 251 out of a possible 1 944) and the smallest total number of worse values (447). DynDE, with sub-population size of six, was better than its closest contender (population size of 12) in 340 cases and worse in only 199 cases. These results provide evidence that using a sub-population size of six is the most effective approach. This conclusion is strengthened by summing the number of times that each of the sub-population sizes performed better than all the other three sub-population sizes for each experimental setting. This information is summarised in Table 4.7. The experiments with a sub-population size of six performed the best, considerably more often, than the other sub-population sizes. It can thus be concluded that using a sub-population size of six yields the best results over a broad range of dynamic environments.

A final analysis is given here to confirm the appropriateness of using 16.6̇% Brownian individuals in a sub-population of 6 individuals. Table 4.8 lists the number of experiments in which each of the percentages of Brownian individuals performed the best when a sub-

Table 4.7: Number of environments in which each sub-population size resulted in the best performance when using 16.6̇% Brownian individuals

Sub-Population Size	**6**	**12**	**24**	**48**
Number Best	340	81	23	20

population size of 6 was used. The experiments that used 16.6̇% Brownian individuals outperformed the experiments using other percentages by a wide margin.

Table 4.8: Number of environments in which each percentage of Brownian individuals resulted in the best performance when using a sub-population size of 6

Percentage Brownian	**0%**	**16.6̇%**	**33.3̇%**	**50%**	**66.6̇%**	**83.3̇%**	**100%**
Number Best	18	351	1	0	0	0	53

The 18 cases where using no Brownian individuals yielded better offline errors than the other percentage settings were all in environments with a change period of either 100 or 500. These rapidly changing environments allow DynDE to perform very few generations (about one generation with a change period of 100 and five generations with a change period of 500). DynDE is unable to converge to optima effectively with so few generations, which means that the Brownian individuals are created around inferior individuals and function evaluations are consequently wasted. However, the disadvantage of using Brownian individuals in these rapidly changing environments is minor, because the results were statistically significantly better in only 18 of the 84 rapidly changing environments that were investigated when Brownian individuals were not used.

This research question investigated appropriate settings for the sub-population size and the proportion of Brownian individuals. The results presented in this section indicate that Mendes and Mohais [2005] were correct in recommending a small sub-population size of six, but that, using a single Brownian individual, rather than two, yields the best results over a wide selection of dynamic environments.

4.6.4 Research Question 2

How do DynDE, RMC, CPE and CDE scale under factors that influence the complexity of a DOP?

This section presents the results of a scalability study on DynDE, CPE, RMC and CDE with respect to dimension, change period, change type, change severity, and underlying function. The standard set (refer to Section 4.6.2) of environments was varied over all combinations of change periods and dimensions given in Table 4.3. The standard set contains 54 environments which, multiplied by the five dimensional settings and the eight change period settings, gives a total of 2 160 experiments that were conducted per algorithm.

The rest of this section is structured as follows: Section 4.6.4.1 gives the results of each algorithm on the standard set. Section 4.6.4.2 describes the scalability of the algorithms with respect to change period, Section 4.6.4.3 the scalability with respect to dimension, Section 4.6.4.4 the scalability with respect to change severity, Section 4.6.4.5 the scalability with respect to underlying function, and Section 4.6.4.6 the scalability with respect to change type. The observed trends are summarised in Section 4.6.4.7.

4.6.4.1 Performance on the Standard Set

The large amount of experimental data (a total of 8 640 experiments) makes it inconvenient to present all results in tabular form. Accordingly, the following sections employ graphs to illustrate the observed trends. The offline errors of each of the algorithms on the standard set is given in Table 4.9 to provide context to the graphs given in the following sections. The MPB conical peak function is denoted by "C", the spherical peak function by "S", the change severity by "C_s", the GDBG functions by "F_1" to "F_6", and the GDBG change types by "T_1" to "T_6".

The first observation that can be made from Table 4.9 is that the different settings of the standard set clearly has a large impact on the offine errors produced by each of the algorithms. For example, observe the increase in offline error of the algorithms on the MPB with the conical peak function (denoted by C) as the change severity, C_s, is increased. Also note that the increase in offline error with respect to change severity is

Table 4.9: Offline error of DynDE, CPE, RMC and CDE on the standard set

Settings		DynDE	CPE	p-val	RMC	p-val	CDE	p-val
C	C_s 1.0	1.05 ± 0.12	**0.95 ± 0.15**	0.155	**0.85 ± 0.06**	*0.021*	**0.80 ± 0.21**	*0.000*
	5.0	4.26 ± 0.22	**2.51 ± 0.25**	*0.000*	**4.22 ± 0.25**	0.741	**2.79 ± 0.22**	*0.000*
	10.0	10.87 ± 0.67	**5.53 ± 0.45**	*0.000*	**10.39 ± 0.56**	0.343	**5.60 ± 0.39**	*0.000*
	20.0	20.33 ± 1.08	**11.50 ± 0.71**	*0.000*	**19.04 ± 1.26**	0.075	**11.98 ± 0.79**	*0.000*
	40.0	20.49 ± 1.01	**20.11 ± 1.18**	0.458	20.59 ± 1.46	0.602	**19.47 ± 0.78**	0.109
	80.0	28.53 ± 1.82	**24.36 ± 1.14**	*0.000*	28.54 ± 1.60	0.752	**24.10 ± 0.81**	*0.000*
S	C_s 1.0	1.06 ± 0.17	1.15 ± 0.15	0.300	**0.80 ± 0.10**	*0.017*	1.07 ± 0.21	0.797
	5.0	4.27 ± 0.13	**2.88 ± 0.15**	*0.000*	**4.03 ± 0.08**	*0.009*	**2.58 ± 0.10**	*0.000*
	10.0	18.16 ± 0.23	**10.42 ± 0.21**	*0.000*	**17.79 ± 0.24**	*0.040*	**9.71 ± 0.22**	*0.000*
	20.0	73.74 ± 1.37	**46.01 ± 1.00**	*0.000*	**72.90 ± 2.05**	0.242	**43.71 ± 0.89**	*0.000*
	40.0	166.96 ± 6.38	**152.81 ± 3.75**	*0.000*	**162.76 ± 6.49**	0.350	**147.32 ± 3.87**	*0.000*
	80.0	242.02 ± 10.48	**234.25 ± 6.86**	0.366	**240.73 ± 7.69**	0.947	**220.62 ± 6.11**	*0.005*
F_{1a}	T_1	9.73 ± 0.20	**6.03 ± 0.29**	*0.000*	9.74 ± 0.20	0.982	**5.99 ± 0.22**	*0.000*
	T_2	18.75 ± 0.45	**10.87 ± 0.30**	*0.000*	**18.47 ± 0.37**	0.248	**10.36 ± 0.29**	*0.000*
	T_3	17.35 ± 1.19	**9.84 ± 0.75**	*0.000*	**16.35 ± 0.95**	0.286	**10.56 ± 0.69**	*0.000*
	T_4	20.24 ± 0.33	**13.54 ± 0.32**	*0.000*	**20.13 ± 0.32**	0.513	**13.08 ± 0.27**	*0.000*
	T_5	22.45 ± 0.35	**13.45 ± 0.42**	*0.000*	22.50 ± 0.32	0.832	**13.63 ± 0.48**	*0.000*
	T_6	26.52 ± 0.45	**17.18 ± 0.38**	*0.000*	**26.43 ± 0.37**	0.752	**17.24 ± 0.36**	*0.000*
F_{1b}	T_1	11.56 ± 0.44	**8.89 ± 0.35**	*0.000*	**11.28 ± 0.37**	0.335	**9.35 ± 0.41**	*0.000*
	T_2	16.91 ± 0.36	**11.35 ± 0.32**	*0.000*	**16.63 ± 0.34**	0.358	**11.35 ± 0.28**	*0.000*
	T_3	16.80 ± 0.57	**11.14 ± 0.51**	*0.000*	**16.64 ± 0.61**	0.741	**12.26 ± 0.63**	*0.000*
	T_4	21.64 ± 0.51	**15.34 ± 0.44**	*0.000*	21.91 ± 0.50	0.314	**15.42 ± 0.57**	*0.000*
	T_5	18.84 ± 0.21	**11.93 ± 0.21**	*0.000*	**18.49 ± 0.25**	0.057	**11.78 ± 0.16**	*0.000*
	T_6	25.13 ± 0.37	**15.87 ± 0.24**	*0.000*	**24.74 ± 0.38**	0.159	**16.12 ± 0.32**	*0.000*
F_2	T_1	59.32 ± 1.16	**52.76 ± 1.56**	*0.000*	59.79 ± 1.29	0.476	**50.98 ± 1.97**	*0.000*
	T_2	154.55 ± 1.43	**142.39 ± 3.00**	*0.000*	155.87 ± 2.31	0.382	**141.45 ± 6.47**	*0.000*
	T_3	144.66 ± 2.19	**142.71 ± 3.67**	0.213	147.59 ± 1.99	0.057	**138.18 ± 3.77**	*0.001*
	T_4	109.16 ± 1.99	**87.51 ± 3.17**	*0.000*	109.39 ± 2.08	0.820	**86.27 ± 3.14**	*0.000*
	T_5	182.75 ± 4.08	210.98 ± 3.01	*0.000*	184.87 ± 3.67	0.254	208.00 ± 2.85	*0.000*
	T_6	157.50 ± 2.02	**132.97 ± 2.94**	*0.000*	160.88 ± 2.57	0.173	**135.86 ± 4.14**	*0.000*
F_3	T_1	603.28 ± 33.91	651.14 ± 21.62	*0.037*	629.03 ± 38.33	0.106	**567.47 ± 63.52**	0.843
	T_2	954.22 ± 8.57	**944.15 ± 8.14**	0.177	**949.64 ± 5.98**	0.440	**948.69 ± 8.33**	0.406
	T_3	957.51 ± 6.51	**950.71 ± 7.31**	0.485	**949.20 ± 7.04**	0.230	**947.46 ± 7.91**	0.068
	T_4	823.01 ± 14.58	**813.42 ± 20.65**	0.912	**807.79 ± 24.27**	0.764	**803.66 ± 18.89**	0.116
	T_5	912.61 ± 16.99	928.88 ± 14.77	*0.033*	928.74 ± 15.26	*0.046*	928.02 ± 19.91	0.182
	T_6	977.84 ± 9.30	**975.48 ± 10.82**	0.832	990.33 ± 9.79	0.077	**972.99 ± 10.47**	0.343
F_4	T_1	104.09 ± 2.71	**75.07 ± 3.00**	*0.000*	106.57 ± 2.93	0.572	**74.63 ± 2.00**	*0.000*
	T_2	317.50 ± 4.86	**200.90 ± 5.37**	*0.000*	**314.00 ± 5.21**	0.390	**202.21 ± 5.31**	*0.000*
	T_3	292.79 ± 5.02	**203.19 ± 5.66**	*0.000*	**288.32 ± 4.59**	0.224	**209.94 ± 6.21**	*0.000*
	T_4	210.72 ± 4.30	**141.80 ± 5.39**	*0.000*	214.69 ± 3.45	0.197	**138.78 ± 4.03**	*0.000*
	T_5	354.22 ± 6.60	**315.53 ± 8.49**	*0.000*	359.61 ± 6.88	0.293	**323.35 ± 7.20**	*0.000*
	T_6	304.35 ± 6.22	**207.34 ± 5.38**	*0.000*	**304.12 ± 6.94**	0.686	**208.45 ± 5.25**	*0.000*
F_5	T_1	184.45 ± 4.56	**140.43 ± 2.70**	*0.000*	**179.84 ± 3.90**	0.112	**138.39 ± 2.74**	*0.000*
	T_2	431.23 ± 3.99	**281.12 ± 2.04**	*0.000*	432.09 ± 5.02	0.686	**283.31 ± 2.59**	*0.000*
	T_3	456.55 ± 8.15	**257.72 ± 4.07**	*0.000*	457.50 ± 5.65	0.582	**254.75 ± 4.22**	*0.000*
	T_4	323.49 ± 5.58	**211.48 ± 2.90**	*0.000*	**321.02 ± 7.11**	0.542	**209.22 ± 3.19**	*0.000*
	T_5	660.93 ± 15.61	**309.31 ± 4.75**	*0.000*	**655.21 ± 17.96**	0.495	**304.52 ± 4.42**	*0.000*
	T_6	530.49 ± 13.27	**274.13 ± 4.30**	*0.000*	**524.56 ± 11.99**	0.350	**272.89 ± 4.76**	*0.000*
F_6	T_1	122.00 ± 4.56	**88.10 ± 2.91**	*0.000*	124.65 ± 3.28	0.279	**89.67 ± 3.25**	*0.000*
	T_2	364.30 ± 12.74	**217.14 ± 20.63**	*0.000*	381.20 ± 12.71	0.051	**231.04 ± 10.06**	*0.000*
	T_3	415.76 ± 14.23	**279.81 ± 16.12**	*0.000*	432.63 ± 15.18	0.100	**305.07 ± 22.58**	*0.000*
	T_4	262.26 ± 18.88	**151.61 ± 11.73**	*0.000*	**254.35 ± 14.22**	0.832	**166.38 ± 14.31**	*0.000*
	T_5	498.69 ± 17.72	**383.18 ± 19.69**	*0.000*	525.48 ± 16.39	0.051	**396.75 ± 21.66**	*0.000*
	T_6	430.24 ± 21.31	**277.79 ± 19.30**	*0.000*	444.89 ± 26.90	0.440	**305.50 ± 18.43**	*0.000*

greater for the spherical peak function (denoted by S) than for the conical function. The substantial effect of different settings is also clear on the GDBG experiments. For example, compare the offline errors on function F_{1a} to errors on function F_3.

The standard set does not include variations of the number of dimensions, n_d, or change period, Cp. Experimental results on different settings of n_d and Cp showed that these values also had a considerable influence on offline errors. For example, consider Figure 4.7 which shows the offline error of DynDE on the MPB with the conical peak function and change severity of 1.0 as n_d and Cp are varied. The large effect of these settings is clear.

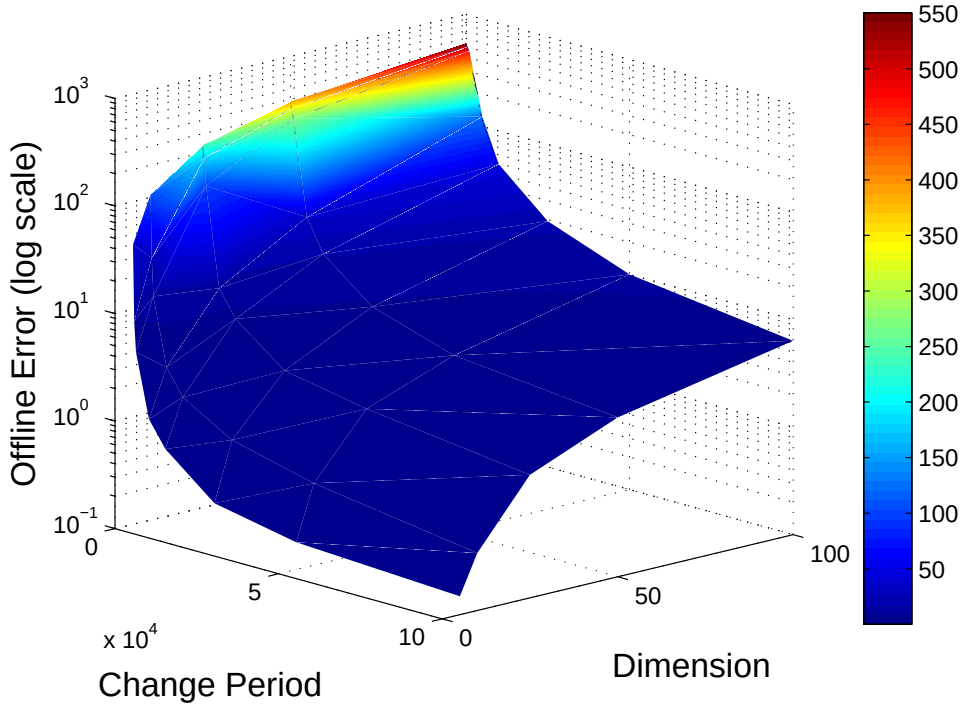

Figure 4.7: Offline error of DynDE on the MPB using the conical peak function with change severity 1.0 for various settings of number of dimensions and change period.

An analysis of the results found that the influence of the various settings is strongly interrelated, for example, a combination of high dimension and low change period results in an extremely high offline error in Figure 4.7. The following sections discuss the trends that were observed for each of the scalability settings, respectively, in the context of the other settings.

4.6.4.2 Trends from Varying the Change Period

Section 2.5.2 pointed out that the frequency of changes (determined by the change period) is one of the defining characteristics of a dynamic environment and argued that the change frequency determines the viability of solving a DOP. Figure 4.7 shows an exponential increase in offline error as the change period is reduced. A similar trend was found for all other experimental environments and algorithms with respect to change period. This trend exists because, as the change period is decreased, the number of available function evaluations between changes in the environment decreases. The algorithms are consequently less likely to locate optima which, in turn, leads to a higher offline error. Conversely, a large change period makes the discovery of optima more likely, which results in a lower offline error. An increase in change period results in diminishing returns, in terms of offline error, as the environment tends towards a static environment. This explains the comparatively low gradient of the curve for large change periods in Figure 4.7.

An analysis of the experimental data found that, in general, DynDE and RMC scaled similarly with respect to change period, but that CPE and CDE exhibited different scaling behaviour. All four algorithms performed similarly in the presence of very frequent changes of 100 function evaluations. This observation is to be expected in the case of CPE since the competition between sub-populations only commences subsequent to the second generation after a change in the environment. A change period of 100 causes a change in the environment before the second generation is completed. Accordingly, the CPE algorithm reduces to DynDE which explains the similar results. The RMC approach is aimed at situations where multiple optima are located within the exclusion radius of each other. RMC aims to prevent the unnecessary reinitialisation of sub-populations when converging to closely located optima. However, in the presence of frequent changes, the algorithm does not have enough time to converge to optima and RMC consequently performs similarly to DynDE.

CPE and CDE were found to be more scalable than DynDE and RMC when the change period was higher than 100, although it was found that the number of dimensions influenced the scalability. The performance of the algorithms on the GDBG function F_{1a}, with change type T_1, was selected as a representative example to illustrate the influence of

the number of dimensions on the scalability trend (refer to Figures 4.8 to 4.11). CPE and CDE scaled better than DynDE and RMC only in the lower change periods (ignoring the change period of 100) in low dimensions, with similar offline errors for the higher change period experiments, as shown in Figure 4.8. The gap between CPE and CDE versus DynDE and RMC widens in the 10 and 25 dimensional cases shown in Figures 4.9 and 4.10 but still yielded similar results for high change periods. However, in 100 dimensions CPE and CDE clearly scale better than DynDE and RMC, even for high change periods.

The trends shown in Figures 4.8 to 4.11 are specific examples, but CPE and CDE tended to scale better than DynDE and RMC in terms of change period on all functions and change types, especially in high dimensions. Only low dimensional experiments yielded situations where the the algorithms performed similarly for high change periods, as shown in Figure 4.9.

4.6.4.3 Trends from Varying the Number of Dimensions

Section 2.5.2 mentioned the number of dimensions as one of the factors that determines the hardness of a fitness landscape. One can generally assume that increasing the number of dimensions increases the hardness of the fitness landscape (refer to page 35) since the size of the search space is increased (there are, however, exceptions to this rule). The experimental results showed that, in general, increasing the number of dimensions resulted in a close-to-linear increase in offline error for all four algorithms. DynDE and RMC showed similar scaling behaviour, while the results of CPE and CDE were similar. Typically, CPE and CDE scaled better in terms of number of dimensions than DynDE and RMC, but it was found that this trend was influenced by the change period.

The performance of the algorithms on the GDBG function F_2 with change type T_4 was selected as a representative example of the scalability of the algorithms in terms of the number of dimensions with respect to change period. These trends are shown in Figures 4.12 to 4.15. The previous section found that all four algorithms performed similarly for low change periods. Figure 4.12 shows similar scaling behaviour over the number of dimensions for DynDE, RMC, CPE and CDE for a change period of 500. As the change period is increased to 5 000 (refer to Figure 4.13) CPE and CDE scales better than DynDE and RMC. A change period of 25 000 function evaluations results in

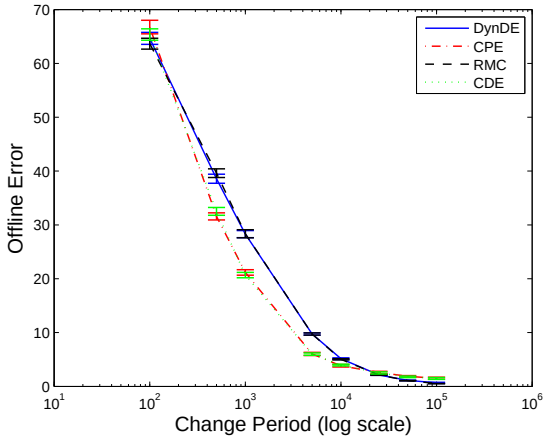

Figure 4.8: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_{1a} and change type T_1 for various settings of change period in 5 dimensions.

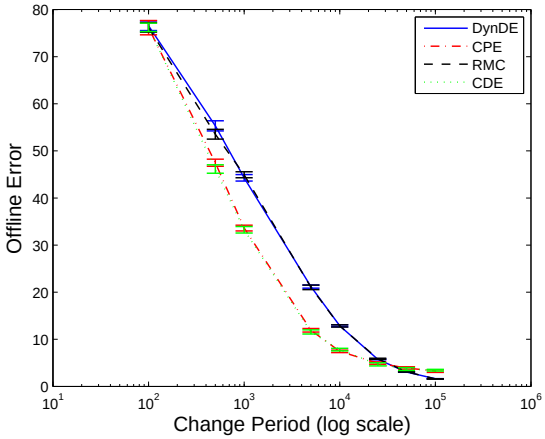

Figure 4.9: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_{1a} and change type T_1 for various settings of change period in 10 dimensions.

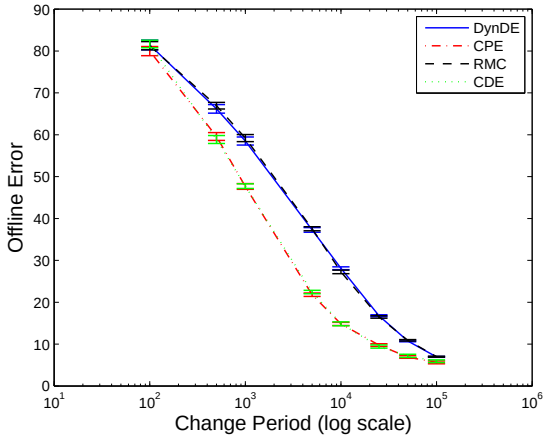

Figure 4.10: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_{1a} and change type T_1 for various settings of change period in 25 dimensions.

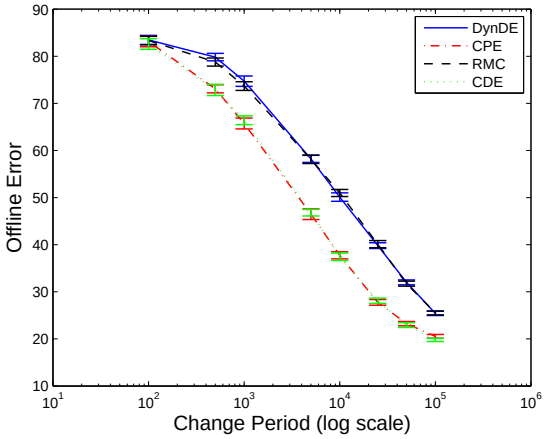

Figure 4.11: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_{1a} and change type T_1 for various settings of change period in 100 dimensions.

similar offline errors for all four algorithms in low dimensions (as shown in Figure 4.14), but with CPE and CDE still scaling better than DynDE and RMC in high dimensions. The diminishing performance of CPE and CDE with respect to DynDE and RMC in low dimensions with an increased change period of 100 000 is apparent in Figure 4.15 where CPE and CDE actually perform worse than DynDE and RMC in low dimensions, although better performance is still observed in high dimensions.

The general trends observed in the examples shown in Figures 4.12 to 4.15 were found in the majority of the benchmark settings that were investigated. CPE and CDE, in general, scale better than DynDE and RMC in terms of number of dimensions. The increase in offine error with the increase in the number of dimensions fits the hypothesis that increasing the number of dimensions makes the DOP harder. The diminished scalability of CPE and CDE in low dimensions with high change periods, in comparison with DynDE and RMC, is explained by the fact that the large change periods allow sufficient function evaluations for all algorithms to locate optima, hence resulting in similar offline errors. As the dimension increases and the problem becomes harder, the more scalable algorithms, CPE and CDE, give lower offline errors.

4.6.4.4 Trends from Varying the Change Severity

The severity of changes in a dynamic environment was identified as a critical factor which determines the ability of an algorithm to solve a DOP (refer to Section 2.5.2). The result of significantly large changes is that the new environment bears no relation to the previous environment. The usefulness of information, gathered about an environment before a change, thus diminishes as the severity of the changes increases. The offline error of an optimisation algorithm is, consequently, expected to increase as the changes become more severe. Experimental results from the experiments on the MPB showed that a considerable increase in offline error for all algorithms does, in fact, result from increasing the change severity.

CPE and CDE, typically, scaled better with respect to change severity than DynDE and RMC. The trends, however, were once again found to be heavily dependent on the change period. Results of the algorithms on the spherical peak function of the MPB in 10 dimensions are used as an example to illustrate the general trend (refer to Figures 4.16 to

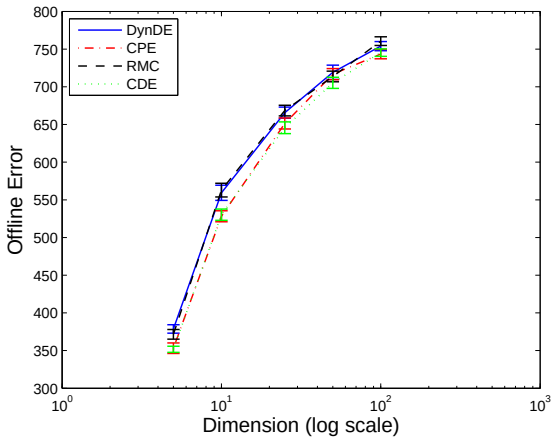

Figure 4.12: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_2 and change type T_4 for various settings of dimension with a change period of 500 function evaluations.

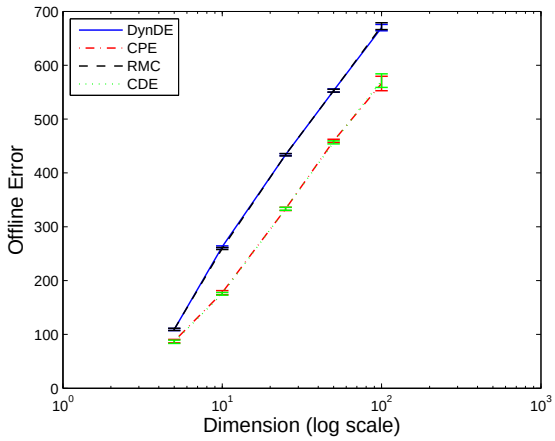

Figure 4.13: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_2 and change type T_4 for various settings of dimension with a change period of 5 000 function evaluations.

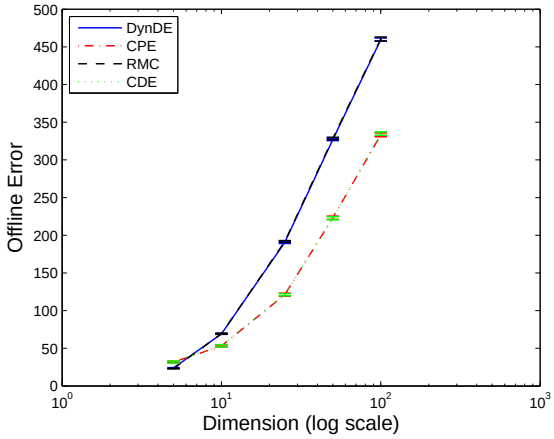

Figure 4.14: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_2 and change type T_4 for various settings of dimension with a change period of 25 000 function evaluations.

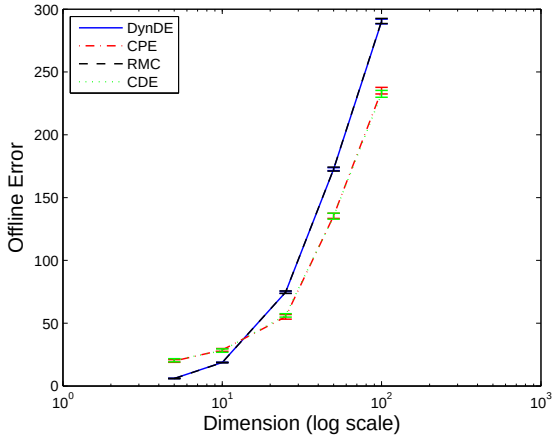

Figure 4.15: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using peak function F_2 and change type T_4 for various settings of dimension with a change period of 100 000 function evaluations.

4.19). Small change periods, as shown in Figure 4.16 resulted in similar scaling behaviour for all four algorithms. A higher change period results in better scaling by CPE and CDE than DynDE and RMC, as shown in Figure 4.17. The gap between the offline errors of CPE and CDE versus DynDE and RMC increases in proportion as the change period is increased to 25 000 in Figure 4.18 and to 100 000 in Figure 4.19.

The superior scalability of CPE and CDE, in terms of change severity, was found to be especially pronounced when using the spherical peak function, but was also present, to a lesser degree, in the conical peak function experiments. The improved scalability of CPE and CDE, with an increasing change period, is in contrast with trends found with respect to scalability in terms of dimension (refer to Section 4.6.4.3) and change period in general (refer to Sections 4.6.4.2). A diminished difference between the scaling behaviour of CPE and CDE versus DynDE and RMC was previously found when increasing the change period.

The superior scaling behaviour exhibited by CPE and CDE is due to large errors resulting from severe changes in the environment, which typically leaves sub-populations located far from the optima. The competing population approach evolves only the best sub-population at any given time and the current error thus decreases in fewer function evaluations than it would when using normal DynDE.

Figure 4.20 shows the offline and current errors of DynDE and CPE on the MPB using the spherical peak function in 10 dimensions with a change period of 5 000, while Figure 4.21 gives the same results with a change period of 100 000. A large change severity value of 80.0 is used. Note that an increase in current error of about 3 000 occurs for both algorithms after changes in the environments, which is roughly equal to the current error at the commencement of the optimisation process. The current error of the CPE algorithm decreases faster than that of DynDE after each change. This allows CPE to achieve a lower current error than DynDE when a low change period is used (note the difference in offline error between DynDE and CPE immediately before changes in Figure 4.20), which lead to a lower offline error.

Large change periods allow both algorithms to attain a current error approaching zero before each change (refer to Figure 4.21), but the current error of CPE reduces faster than that of DynDE. This tendency can be more clearly observed in Figure 4.22, which

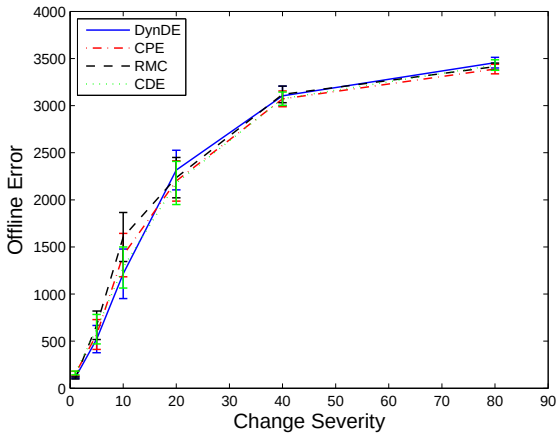

Figure 4.16: Offline errors of DynDE, CPE, RMC, and CDE on the MPB using the spherical peak function in 10 dimensions for various settings of change severity with a change period of 500 function evaluations.

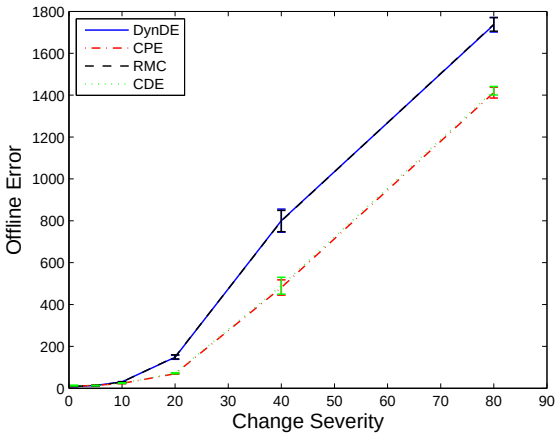

Figure 4.17: Offline errors of DynDE, CPE, RMC, and CDE on the MPB using the spherical peak function in 10 dimensions for various settings of change severity with a change period of 5 000 function evaluations.

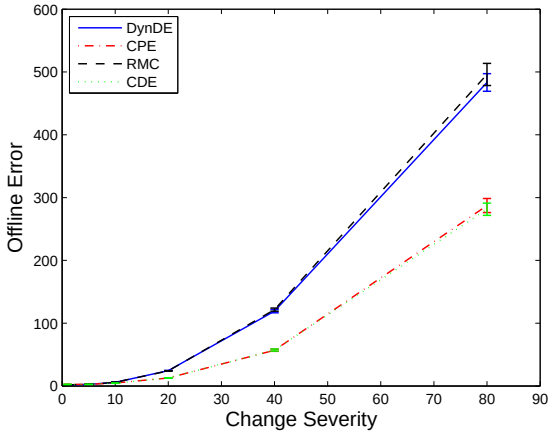

Figure 4.18: Offline errors of DynDE, CPE, RMC, and CDE on the MPB using the spherical peak function in 10 dimensions for various settings of change severity with a change period of 25 000 function evaluations.

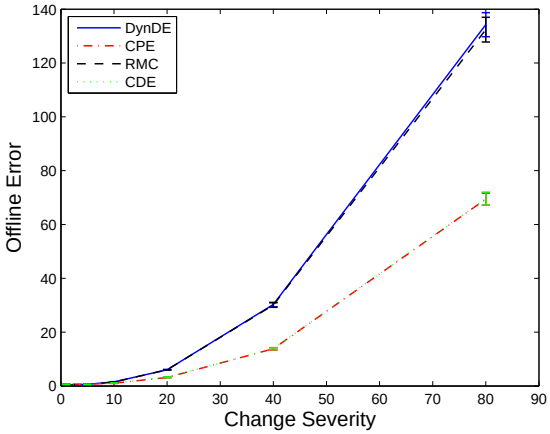

Figure 4.19: Offline errors of DynDE, CPE, RMC, and CDE on the MPB using the spherical peak function in 10 dimensions for various settings of change severity with a change period of 100 000 function evaluations.

shows an enlargement of Figure 4.21 around the first change in the environment. The CPE current error approaches zero in roughly half the function evaluations as does that of DynDE. The faster reduction in error is reflected in the offline error (since it averages over all the best errors found).

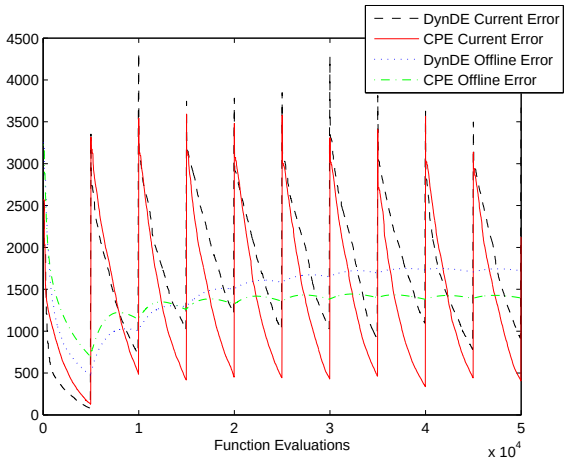

Figure 4.20: Offline and current errors of DynDE and CPE on the MPB spherical function in 10 dimensions, a change period of 5 000, and a change severity of 80.0.

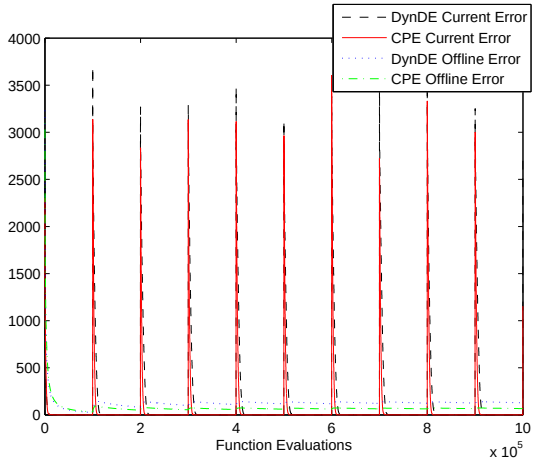

Figure 4.21: Offline and current errors of DynDE and CPE on the MPB spherical function in 10 dimensions, a change period of 100 000, and a change severity of 80.0.

Figure 4.23 shows an enlargement of the current errors of DynDE and CPE in the presence of a change period of 100 000 but with a minor change severity of 1.0. The small change severity of Figure 4.23 (compared to Figure 4.22) results in comparatively small increases in current error after a change in the environment. Both algorithms recover quickly which results in similar behaviour by the two algorithms. The competitive population evaluation approach is thus more useful in the presence of severe changes in the environment.

4.6.4.5 Trends from Various Functions

The underlying function, which determines the topology of the search landscape, was identified in Section 2.5.2 as a factor that influences the intuitive concept of the hardness of the fitness landscape. The underlying function used in a benchmark is consequently expected to influence the performance of a DOP algorithm.

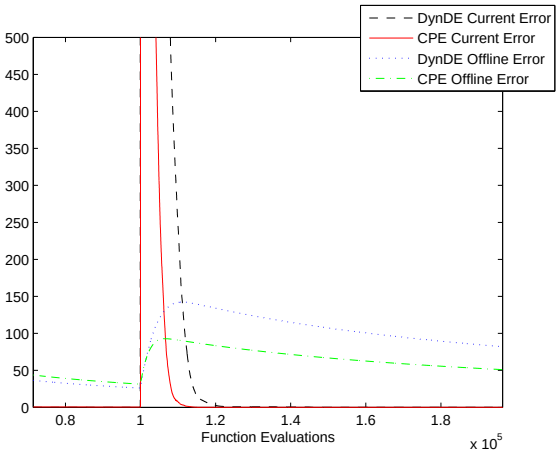

Figure 4.22: Offline and current errors of DynDE and CPE on the MPB spherical function in 10 dimensions, a change period of 100 000, and a change severity of 80.0. The area around the first change is enlarged.

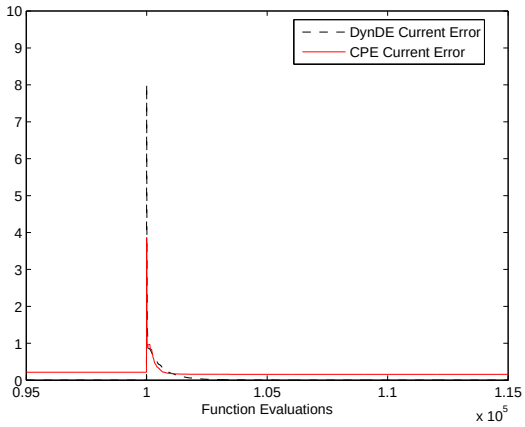

Figure 4.23: Current errors of DynDE and CPE on the MPB spherical function in 10 dimensions, a change period of 100 000, and a change severity of 1.0. The area around the first change is enlarged.

The seven functions of the GDBG listed in Table 4.2 are used in this section to observe the effect of the function on the optimisation process. The number of dimensions was found to have a large impact on the relative performance of DynDE, CPE, RMC and CDE on the seven functions. The performance of the algorithms on environments with a change type T_1 and a change period of 10 000 was selected as a representative example to illustrate the general trends observed over all change types, and the influence of the number of dimensions on the scalability trend (refer to Figures 4.24 to 4.27). Low dimensional experiments typically resulted in the behaviour shown in Figure 4.24. Function F_3, which is a composition of Rastrigin's function (refer to Section 2.5.4.1), yielded a much higher offline error than the other functions. DynDE, CPE, RMC and CDE all performed similarly on F_3, but the performance of DynDE and RMC differed from that of CPE and CDE. Function F_5 shows a more rapid growth in offline error than the other functions, as the number of dimensions is increased (refer to Figure 4.25), while the difference in performance between DynDE and RMC versus CPE and CDE grows.

At high dimensions (refer to Figure 4.26), the offline error on F_5 exceeds that of F_3 while the gap between the performance of DynDE and RMC versus CPE and CDE narrows until their behaviour becomes very similar (refer to Figure 4.27). Note that the

performance of DynDE and RMC versus CPE and CDE still differs for functions F_2, F_4 and F_6 in Figure 4.27.

The general trend that is thus observed is that in low dimensions, function F_3 presents the greatest challenge to the optimisation algorithms, followed by F_5, F_6, F_3, F_2, and finally F_1. High dimensions alter this trend to function F_5 posing the greatest challenge to the algorithms, followed by F_3, F_6, F_4, F_2, and F_1. Function F_3 is comparatively unaffected by increasing the number of dimensions, and causes the most similar scaling behaviour in DynDE, CPE, RMC and CDE. CPE and CDE perform better than DynDE and RMC on all other functions, especially in lower dimensions.

4.6.4.6 Trends from Various Change Types

The six change types of the GDBG listed in Table 4.2 are used in this section to observe the effect of the change type on the optimisation process. The experimental results did not reveal clear, general trends as was the case with the other settings that were varied. The effect of the change type on the optimisation algorithm was found to depend on the underlying function.

Table 4.10 was created by ranking the offline error of DynDE on each change type in ascending order over all experiments for each function. The ranking of each change type was then averaged to give an average ranking per change type, per function. Over all functions, T_1 ranked the highest, which means that, on average, DynDE yielded the lowest offline error on T_1. T_3, T_2, and T_4 received similar rankings, while T_5 and T_6 received lower rankings. T_1 always received the highest ranking while T_6 always ranked the lowest per function, but the rankings per function did differ considerably from the overall ranking in several cases. For example, consider the ranking on function F_3, where T_5 received a relatively high rank and T_2 received a relatively low rank.

Section 4.6.4.2 identified a trend in change period wherein CPE and CDE performed better than DynDE and RMC at low change periods, while DynDE and RMC outperformed CPE and CDE at high change periods. An analysis of the experimental data found that the point where DynDE and RMC start outperforming CPE and CDE depends on the change type for each function. As an example, the offline error per change type of DynDE, CPE, RMC, and CDE on function F_{1a} in 10 dimensions is given in Figures

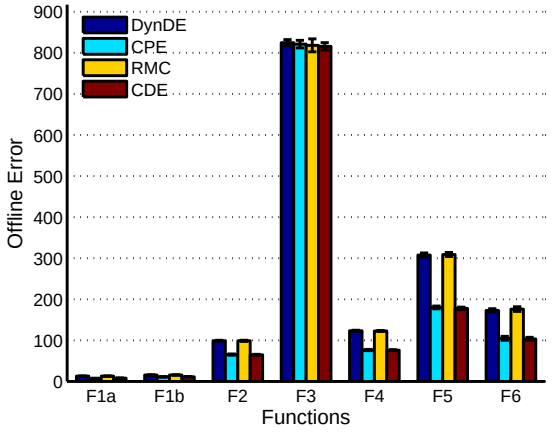

Figure 4.24: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using change type T_1 for various functions in 10 dimensions with a change period of 10 000 function evaluations.

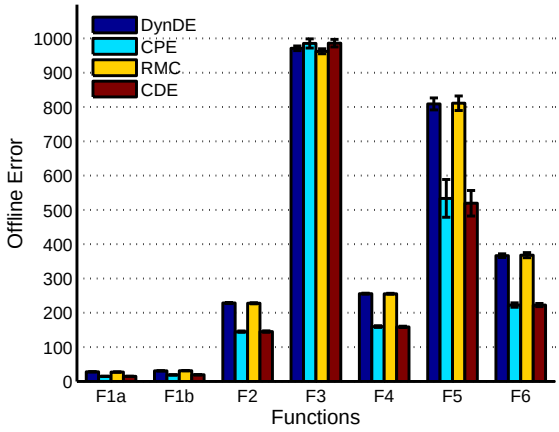

Figure 4.25: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using change type T_1 for various functions in 25 dimensions with a change period of 10 000 function evaluations.

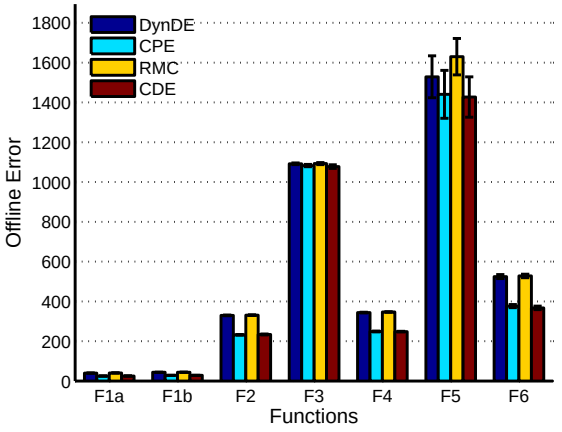

Figure 4.26: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using change type T_1 for various functions in 50 dimensions with a change period of 10 000 function evaluations.

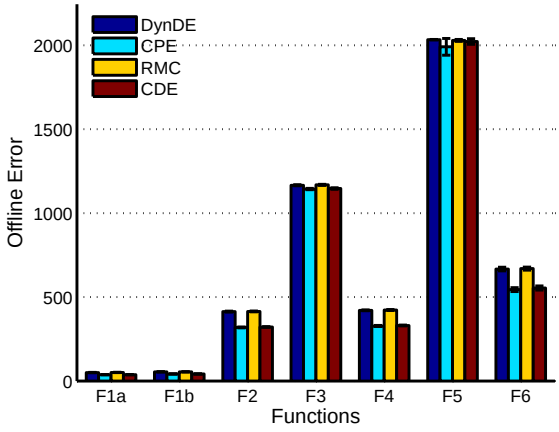

Figure 4.27: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using change type T_1 for various functions in 100 dimensions with a change period of 10 000 function evaluations.

Table 4.10: Average ranking of change type per function

F_{1a}	F_{1b}	F_2	F_3	F_4	F_5	F_6	**All**
T_1 - 1.30	T_1 - 1.80	T_1 - 1.00	T_1 - 1.00	T_1 - 1.00	T_1 - 1.28	T_1 - 1.00	T_1 - 1.20
T_3 - 2.38	T_2 - 2.35	T_4 - 2.95	T_3 - 2.83	T_4 - 2.93	T_4 - 2.80	T_4 - 2.05	T_3 - 3.23
T_2 - 2.58	T_3 - 2.58	T_3 - 3.55	T_5 - 3.38	T_3 - 3.55	T_2 - 3.03	T_2 - 3.23	T_2 - 3.26
T_4 - 4.30	T_5 - 3.60	T_2 - 3.63	T_4 - 3.55	T_2 - 3.80	T_3 - 3.58	T_3 - 4.15	T_4 - 3.36
T_5 - 4.53	T_4 - 4.98	T_5 - 4.73	T_2 - 4.25	T_5 - 4.73	T_5 - 4.60	T_5 - 5.15	T_5 - 4.39
T_6 - 5.93	T_6 - 5.70	T_6 - 5.15	T_6 - 6.00	T_6 - 5.00	T_6 - 5.73	T_6 - 5.43	T_6 - 5.56

4.28 to 4.31.

Very similar offline errors were found for each algorithm in Figure 4.28, which shows the performance with a change period of 100. Figure 4.29, where a change period of 5 000 was used, shows clearly better performance for CPE and CDE than for DynDE and RMC. The diminishing difference in performance between CPE and CDE versus DynDE and RMC is shown in Figure 4.30 (at a change period of 25 000 function evaluations), where T_4 no longer shows clear differences between CPE and CDE versus DynDE and RMC, and the confidence bars of the algorithms on T_3 overlap. Figure 4.31 shows the performance of the algorithms at a change period of 100 000, where DynDE and RMC generally outperformed CPE and CDE. The change type thus has an influence on how early, in terms of the change period setting, the change over point between the performances of the algorithms occurs.

4.6.4.7 Summary for Research Question 2

The trends that emerged from varying the benchmark settings are summarised below for each setting:

Change Period: Increasing the change period resulted in a reduction of offline error. The algorithms have more function evaluations available between changes in the environment which resulted in lower offline errors. CPE and CDE scaled better than DynDE and RMC as changes became more frequent.

Number of Dimensions: Larger numbers of dimensions resulted in larger offline errors. CPE and CDE scaled better than DynDE and RMC as the number of dimensions increased.

Change Severity: More severe changes in the environment resulted in higher offline

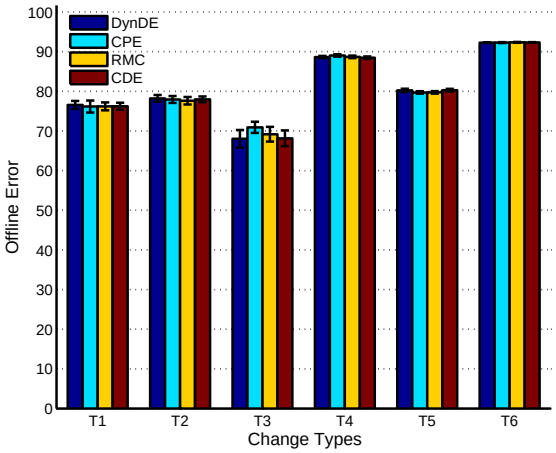

Figure 4.28: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using function F_{1a} for various change types in 10 dimensions with a change period of 100 function evaluations.

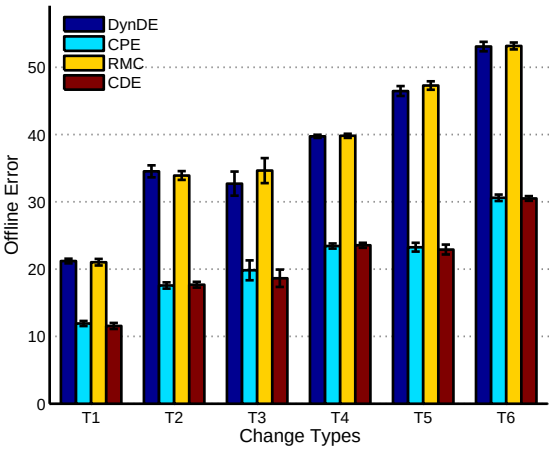

Figure 4.29: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using function F_{1a} for various change types in 10 dimensions with a change period of 5 000 function evaluations.

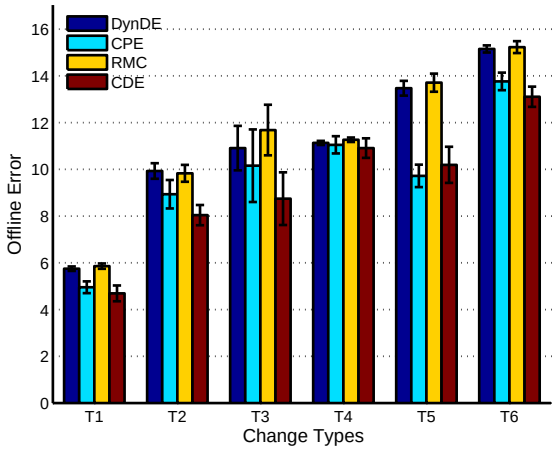

Figure 4.30: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using function F_{1a} for various change types in 10 dimensions with a change period of 25 000 function evaluations.

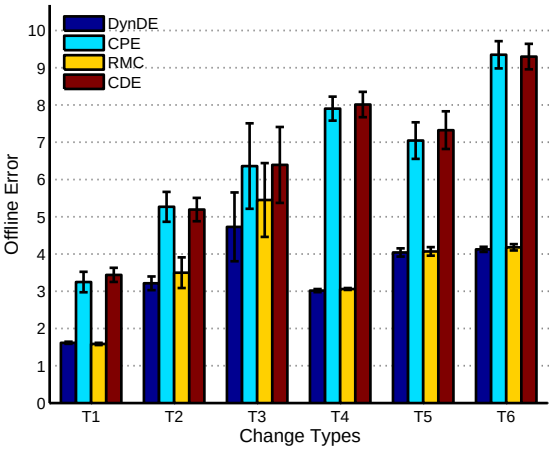

Figure 4.31: Offline errors of DynDE, CPE, RMC, and CDE on the GDBG using function F_{1a} for various change types in 10 dimensions with a change period of 100 000 function evaluations.

errors, as information, gathered before changes, became less relevant. CPE and CDE scaled better than DynDE and RMC as changes became more severe.

Function: The underlying benchmark function was found to have a strong influence on the offline errors of the optimisation algorithms. The underlying function also determines the scalability of each algorithm in terms of dimension. CPE and CDE typically performed better than DynDE and RMC on the various functions.

Change Type: The effect of the change type was found to be related to the function that was being optimised. The change types influence the point where DynDE and RMC starts to outperform CPE and CDE due to an increased change period.

4.6.5 Research Question 3

Are RMC, CPE and CDE more effective dynamic optimisation algorithms than DynDE?

The discussion related to the previous research question identified trends which suggest that CPE and CDE are superior to DynDE and RMC in terms of scalability over the change period, number of dimensions and change severity. The aim of this section is to determine whether RMC, CPE and CDE give significantly lower offline errors than DynDE and are thus more effective algorithms for DOPs.

Table 4.9 gives the offline errors of DynDE, CPE, RMC and CDE on the standard set of experiments which correspond to the benchmark settings in Tables 4.1 and 4.2. The respective offline errors of CPE, RMC and CDE are printed in boldface in shaded cells where the errors are lower than that of DynDE. The outcomes of Mann-Whitney U tests, comparing the results of each algorithm to those of DynDE, are printed in italics if differences in average offline error are statistically significant at a 95% confidence level (i.e. are lower than 0.05). The results in Table 4.9 show that CPE and CDE clearly outperformed DynDE in the majority of the settings of the standard set. RMC, however, yielded a significantly lower offline error in only four cases and was outperformed by DynDE in one case.

The standard set of experiments is not extensive enough to draw meaningful conclusions about the performance of the three algorithms compared to that of DynDE. Experiments were consequently conducted on variations of the standard set, over all com-

binations of settings of change period and number of dimensions given in Table 4.3. The results were analysed by counting the number of times that DynDE was outperformed by each of the algorithms and the number of times that each of the algorithms outperformed DynDE. Only results that were statistically significantly different were considered.

Comparative results for each of the algorithms are summarised in Tables 4.11 to 4.16. Each cell in the tables gives the number of cases in which the relevant algorithm outperformed DynDE (indicated by ↑) and the number of cases in which DynDE outperformed that specific algorithm (indicated by ↓). The results were aggregated by function, change type and change severity for all the values of the change period and number of dimensions and totals given to support the analysis of the data. The value in column **Max** indicates the maximum possible score that can be found in the cells in each row. Note that rows labelled **All**, which gives aggregated results per change period, do not give totals of values in each row as some results are duplicated in the stratifications of the different categories. For example, results for each function are summed over all change types, while the result for each change type is summed over all functions.

The following sections, respectively discuss the performance of CPE, RMC and CDE compared to DynDE. CPE is compared to DynDE in Section 4.6.5.1, RMC is compared to DynDE in Section 4.6.5.2, and CDE is compared to DynDE in Section 4.6.5.3. Concluding remarks on this research question are given in 4.6.5.4.

4.6.5.1 CPE compared to DynDE

Table 4.11 summarises the comparison between the CPE and DynDE results in 5, 10 and 25 dimensions, while Table 4.12 gives the summary for 50 and 100 dimensions, and results summarised over all the dimensions that were investigated. CPE performed significantly better than DynDE in 1 344 experiments out of a total of 2 160 (i.e. 62.2% of all experiments). DynDE performed better than CPE in only 219 experiments (10.1% of all experiments). This clearly indicates that CPE is a more effective algorithm for DOPs than DynDE. The majority of the cases where DynDE performed better than CPE occurred when using a change period 50 000 or higher.

Section 4.3 hypothesised that the competing populations approach could potentially allocate too few function evaluations to weaker sub-populations when changes in the en-

vironment are infrequent, which could lead to inferior performance. This hypothesis has been proved to be correct by the analysis presented in Table 4.11. However, the inferior performance of CPE on high change period problems only occurs in low dimensions (compare the 5 and 10 dimensional results to the 25, 50 and 100 dimensional cases in Tables 4.11 and 4.12). The aggregate analysis over all dimensions, which is given in Table 4.12, shows that, when all dimensions are considered, CPE performed better than DynDE in more cases, even when large change periods were used. These results agree with the trends observed, in terms of change period and number of dimensions, in the scalability study presented in Section 4.6.4.

Experiments, in which a change period of 100 function evaluations were used, resulted in the fewest cases of statistically significant differences between DynDE and CPE. This is because the competing populations process only commences after two generations, following a change in the environment. The two algorithms are identical when a change period of 100 is used, because the population size employed in the experiments results in successive changes in the environment occurring in less than two generations.

The trend that was observed in Section 4.6.4.4 in respect of change severity is confirmed by CPE outperforming DynDE more often when a high change severity is used, in higher than 10 dimensions. CPE outperformed DynDE more often on the MPB's spherical peak function in five dimensions, but in higher dimensions, CPE performed better more often on the conical peak function. The GDBG F_3 function consistently resulted in the fewest number of statistically significantly differing results between CPE and DynDE. Function F_5 resulted in a comparatively large number of cases where CPE outperformed DynDE in low dimensions, but resulted in a comparatively small number of superior results by CPE in high dimensions, thus confirming the trend observed in Section 4.6.4.5.

The analysis described in this section does not give an indication of the magnitude of the improvement of CPE over DynDE. The experimental data was analysed to determine the average percentage improvement (API) in offline error of CPE over DynDE. The API is calculated using:

Table 4.11: CPE vs DynDE performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑*0* ↓*1*	↑1 ↓1	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑3 ↓2**
5	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑10 ↓0**
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑12 ↓0**
20	(2)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑9 ↓0**
40	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑6 ↓1**
80	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑4 ↓1**
C	(6)	↑0 ↓0	**↑1 ↓0**	**↑3 ↓0**	**↑4 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑1 ↓0**	↑*1 ↓2*	**↑16 ↓2**
S	(6)	↑*0 ↓1*	**↑2 ↓1**	**↑3 ↓0**	**↑4 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑28 ↓2**
		GDBG								
F_{1a}	(6)	↑*0 ↓2*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑*0 ↓3*	↑*0 ↓6*	↑*0 ↓6*	**↑24 ↓17**
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	↑*0 ↓5*	↑*0 ↓6*	↑*0 ↓6*	**↑23 ↓17**
F_2	(6)	↑0 ↓0	**↑3 ↓0**	**↑6 ↓0**	**↑4 ↓1**	↑*2 ↓4*	↑*0 ↓6*	↑*0 ↓6*	↑*0 ↓6*	↑*15 ↓23*
F_3	(6)	↑*0 ↓1*	**↑1 ↓0**	**↑4 ↓0**	↑*0 ↓2*	↑*0 ↓1*	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑7 ↓4**
F_4	(6)	↑*0 ↓2*	**↑2 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓1**	↑*0 ↓6*	↑*0 ↓6*	**↑22 ↓15**
F_5	(6)	↑*0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓2**	↑*1 ↓3*	↑*0 ↓6*	**↑29 ↓12**
F_6	(6)	**↑1 ↓0**	**↑3 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑4 ↓2**	↑*1 ↓2*	**↑31 ↓5**
T_1	(7)	↑*0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑4 ↓1**	↑*0 ↓6*	↑*0 ↓6*	↑*1 ↓6*	**↑23 ↓21**
T_2	(7)	↑*0 ↓1*	**↑4 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑3 ↓2**	↑*1 ↓5*	↑*0 ↓5*	**↑26 ↓14**
T_3	(7)	↑0 ↓0	**↑4 ↓0**	**↑7 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑3 ↓1**	↑*1 ↓4*	↑*1 ↓5*	**↑26 ↓11**
T_4	(7)	↑*0 ↓1*	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑*0 ↓4*	↑*0 ↓6*	↑*0 ↓6*	**↑25 ↓17**
T_5	(7)	↑*0 ↓2*	**↑3 ↓0**	**↑5 ↓0**	**↑5 ↓2**	**↑5 ↓1**	↑2 ↓2	↑*2 ↓4*	↑*0 ↓5*	**↑22 ↓16**
T_6	(7)	↑1 ↓1	**↑4 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	↑3 ↓3	↑*1 ↓4*	↑*1 ↓5*	**↑29 ↓14**
All	(54)	↑*1 ↓7*	**↑30 ↓1**	**↑45 ↓0**	**↑42 ↓3**	**↑38 ↓5**	**↑19 ↓18**	↑*11 ↓29*	↑*9 ↓34*	**↑195 ↓97**
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓1	↑1 ↓1	↑0 ↓0	↑1 ↓1	↑*0 ↓1*	↑*0 ↓1*	↑*0 ↓2*	↑*3 ↓7*
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑6 ↓1**
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
20	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
40	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
C	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑4 ↓1**	**↑35 ↓1**
S	(6)	↑0 ↓0	↑*0 ↓1*	**↑2 ↓1**	**↑4 ↓0**	**↑4 ↓2**	**↑4 ↓1**	**↑4 ↓1**	**↑4 ↓1**	**↑22 ↓7**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	↑*0 ↓5*	↑*0 ↓6*	**↑27 ↓11**
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	↑*0 ↓5*	↑*0 ↓6*	**↑28 ↓11**
F_2	(6)	↑*0 ↓1*	**↑2 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓1**	↑3 ↓3	↑*0 ↓4*	↑*0 ↓6*	**↑20 ↓15**
F_3	(6)	↑*0 ↓1*	**↑3 ↓0**	**↑5 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑1 ↓1	**↑11 ↓2**
F_4	(6)	↑0 ↓0	**↑2 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑3 ↓2**	↑*2 ↓4*	↑*0 ↓6*	**↑23 ↓13**
F_5	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓2**	**↑40 ↓2**
F_6	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑39 ↓1**
T_1	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	↑*2 ↓4*	↑*1 ↓6*	**↑33 ↓10**
T_2	(7)	↑0 ↓0	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑4 ↓1**	↑*2 ↓4*	↑*2 ↓4*	**↑33 ↓9**
T_3	(7)	↑*0 ↓1*	**↑3 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑4 ↓0**	↑2 ↓2	↑2 ↓2	↑*2 ↓5*	**↑27 ↓10**
T_4	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑3 ↓2**	↑*1 ↓5*	**↑35 ↓7**
T_5	(7)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓2**	**↑4 ↓2**	↑*2 ↓4*	↑*2 ↓4*	**↑26 ↓12**
T_6	(7)	↑*0 ↓1*	**↑4 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓2**	↑*2 ↓4*	**↑34 ↓7**
All	(54)	↑*0 ↓2*	**↑33 ↓1**	**↑48 ↓1**	**↑46 ↓0**	**↑43 ↓4**	**↑35 ↓6**	**↑22 ↓19**	↑*18 ↓30*	**↑245 ↓63**
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑7 ↓1**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑9 ↓0**
10	(2)	↑*0 ↓1*	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑9 ↓1**
20	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓0**
40	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
C	(6)	↑*0 ↓1*	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓0**	**↑37 ↓1**
S	(6)	↑0 ↓0	↑1 ↓1	**↑4 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑3 ↓0**	**↑4 ↓0**	**↑26 ↓1**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓2**	**↑37 ↓2**
F_{1b}	(6)	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑*1 ↓2*	**↑38 ↓2**
F_2	(6)	↑0 ↓0	**↑5 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑3 ↓0**	**↑34 ↓0**
F_3	(6)	↑0 ↓0	↑0 ↓0	**↑6 ↓0**	**↑5 ↓0**	**↑3 ↓1**	↑*0 ↓2*	↑*0 ↓3*	↑*0 ↓4*	**↑14 ↓10**
F_4	(6)	**↑1 ↓0**	**↑4 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑3 ↓1**	**↑34 ↓1**
F_5	(6)	↑0 ↓0	**↑5 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑38 ↓0**
F_6	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓0**
T_1	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑42 ↓4**
T_2	(7)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓0**	↑2 ↓2	**↑38 ↓2**
T_3	(7)	↑0 ↓0	**↑4 ↓0**	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑3 ↓1**	**↑38 ↓2**
T_4	(7)	**↑1 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑4 ↓3**	**↑43 ↓4**
T_5	(7)	↑0 ↓0	**↑4 ↓0**	**↑4 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑4 ↓0**	**↑4 ↓0**	**↑36 ↓1**
T_6	(7)	**↑1 ↓0**	**↑3 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓2**	**↑38 ↓2**
All	(54)	**↑2 ↓1**	**↑34 ↓1**	**↑45 ↓0**	**↑51 ↓0**	**↑50 ↓1**	**↑47 ↓2**	**↑41 ↓3**	**↑28 ↓9**	**↑298 ↓17**

Table 4.12: CPE vs DynDE performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓0	↑13 ↓0
5	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓0	↑12 ↓0
10	(2)	↑0 ↓0	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓0	↑10 ↓0
20	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑13 ↓0
40	(2)	↑0 ↓0	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑11 ↓0
80	(2)	↑0 ↓0	↑0 ↓0	↑2 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑11 ↓0
C	(6)	↑0 ↓0	↑3 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑3 ↓0	↑36 ↓0
S	(6)	↑0 ↓0	↑1 ↓0	↑4 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑34 ↓0
		GDBG								
F_{1a}	(6)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑40 ↓0
F_{1b}	(6)	↑0 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑40 ↓0
F_2	(6)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓1	↑38 ↓1
F_3	(6)	*↑0 ↓1*	↑1 ↓0	↑4 ↓0	↑6 ↓0	↑5 ↓0	↑0 ↓0	*↑0 ↓3*	*↑0 ↓5*	↑16 ↓9
F_4	(6)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑4 ↓1	↑37 ↓1
F_5	(6)	↑0 ↓0	↑2 ↓0	↑4 ↓0	↑5 ↓0	↑5 ↓0	↑4 ↓0	↑4 ↓0	↑3 ↓1	↑27 ↓1
F_6	(6)	↑0 ↓0	↑3 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑4 ↓0	↑5 ↓0	↑35 ↓0
T_1	(7)	*↑0 ↓1*	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓1	↑5 ↓2	↑37 ↓4
T_2	(7)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑4 ↓0	↑4 ↓3	↑38 ↓3
T_3	(7)	↑0 ↓0	↑3 ↓0	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓1	↑6 ↓1	↑40 ↓2
T_4	(7)	↑0 ↓0	↑4 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓1	↑5 ↓1	↑41 ↓2
T_5	(7)	↑0 ↓0	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑5 ↓0	↑5 ↓0	↑4 ↓1	↑41 ↓1
T_6	(7)	↑0 ↓0	↑3 ↓0	↑5 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓0	↑6 ↓0	↑4 ↓0	↑36 ↓0
All	(54)	*↑0 ↓1*	↑30 ↓0	↑45 ↓0	↑52 ↓0	↑52 ↓0	↑45 ↓0	↑42 ↓3	↑37 ↓8	↑303 ↓12
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑1 ↓1	↑11 ↓2
5	(2)	↑0 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑11 ↓3
10	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑1 ↓1	↑1 ↓1	↑10 ↓3
20	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑1 ↓1	↑11 ↓2
40	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑12 ↓1
80	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑13 ↓0
C	(6)	↑0 ↓0	↑1 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑37 ↓0
S	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑3 ↓3	↑3 ↓3	*↑2 ↓4*	↑31 ↓11
		GDBG								
F_{1a}	(6)	*↑0 ↓1*	↑4 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑39 ↓1
F_{1b}	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑42 ↓0
F_2	(6)	*↑0 ↓1*	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑40 ↓2
F_3	(6)	↑0 ↓0	*↑0 ↓2*	↑4 ↓0	↑5 ↓0	↑6 ↓0	↑4 ↓0	*↑0 ↓2*	*↑0 ↓5*	↑19 ↓9
F_4	(6)	*↑0 ↓1*	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑41 ↓2
F_5	(6)	↑0 ↓0	*↑1 ↓2*	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑4 ↓0	↑20 ↓2
F_6	(6)	*↑0 ↓1*	↑4 ↓0	↑4 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	↑4 ↓1	↑34 ↓3
T_1	(7)	*↑0 ↓1*	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓1	↑6 ↓1	↑39 ↓3
T_2	(7)	↑0 ↓0	↑4 ↓1	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑4 ↓3	↑42 ↓5
T_3	(7)	↑0 ↓0	↑4 ↓1	↑4 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑5 ↓0	↑5 ↓1	↑39 ↓2
T_4	(7)	*↑0 ↓2*	↑5 ↓1	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑6 ↓1	↑45 ↓4
T_5	(7)	↑0 ↓0	↑5 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	↑4 ↓2	↑37 ↓4
T_6	(7)	*↑0 ↓1*	↑3 ↓0	↑4 ↓0	↑5 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓0	↑5 ↓0	↑33 ↓1
All	(54)	*↑0 ↓4*	↑33 ↓4	↑46 ↓0	↑50 ↓0	↑50 ↓1	↑46 ↓3	↑40 ↓6	↑38 ↓12	↑303 ↓30
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	*↑0 ↓1*	↑6 ↓3	↑7 ↓1	↑5 ↓0	↑7 ↓1	↑4 ↓2	↑5 ↓1	↑3 ↓3	↑37 ↓12
5	(10)	↑0 ↓0	↑5 ↓0	↑10 ↓0	↑9 ↓0	↑8 ↓1	↑8 ↓1	↑5 ↓1	↑3 ↓1	↑48 ↓4
10	(10)	*↑0 ↓1*	↑4 ↓0	↑7 ↓0	↑9 ↓0	↑9 ↓1	↑10 ↓0	↑8 ↓1	↑6 ↓1	↑53 ↓4
20	(10)	↑0 ↓0	↑6 ↓0	↑7 ↓0	↑10 ↓0	↑10 ↓0	↑9 ↓1	↑9 ↓0	↑8 ↓1	↑59 ↓2
40	(10)	↑0 ↓0	↑2 ↓0	↑7 ↓0	↑9 ↓0	↑9 ↓0	↑9 ↓0	↑8 ↓1	↑9 ↓1	↑53 ↓2
80	(10)	↑0 ↓0	↑1 ↓0	↑8 ↓0	↑8 ↓0	↑8 ↓0	↑9 ↓0	↑9 ↓0	↑9 ↓1	↑52 ↓1
C	(6)	*↑0 ↓1*	↑14 ↓0	↑27 ↓0	↑27 ↓0	↑27 ↓0	↑26 ↓0	↑23 ↓0	↑17 ↓3	↑161 ↓4
S	(6)	*↑0 ↓1*	↑10 ↓3	↑19 ↓1	↑23 ↓0	↑24 ↓3	↑23 ↓4	↑21 ↓4	↑21 ↓5	↑141 ↓21
		GDBG								
F_{1a}	(30)	*↑0 ↓3*	↑25 ↓0	↑27 ↓0	↑30 ↓0	↑30 ↓0	↑22 ↓3	↑18 ↓11	↑15 ↓14	↑167 ↓31
F_{1b}	(30)	↑1 ↓0	↑29 ↓0	↑30 ↓0	↑30 ↓0	↑29 ↓0	↑22 ↓5	↑18 ↓11	*↑12 ↓14*	↑171 ↓30
F_2	(30)	*↑0 ↓2*	↑20 ↓0	↑26 ↓0	↑28 ↓1	↑24 ↓5	↑21 ↓9	↑15 ↓10	*↑13 ↓14*	↑147 ↓41
F_3	(30)	*↑0 ↓3*	↑5 ↓2	↑23 ↓0	↑17 ↓2	↑15 ↓2	↑4 ↓2	*↑0 ↓8*	*↑3 ↓15*	↑67 ↓34
F_4	(30)	*↑1 ↓3*	↑19 ↓0	↑26 ↓0	↑30 ↓0	↑28 ↓1	↑24 ↓3	↑17 ↓10	*↑12 ↓15*	↑157 ↓32
F_5	(30)	*↑0 ↓1*	↑20 ↓2	↑23 ↓0	↑26 ↓0	↑26 ↓0	↑23 ↓2	↑20 ↓3	↑16 ↓9	↑154 ↓17
F_6	(30)	↑1 ↓1	↑18 ↓0	↑28 ↓0	↑30 ↓0	↑30 ↓0	↑27 ↓1	↑24 ↓3	↑21 ↓4	↑179 ↓9
T_1	(35)	*↑0 ↓3*	↑28 ↓0	↑31 ↓0	↑30 ↓1	↑28 ↓2	↑21 ↓7	↑18 ↓13	↑18 ↓16	↑174 ↓42
T_2	(35)	*↑0 ↓1*	↑24 ↓1	↑32 ↓0	↑33 ↓0	↑33 ↓1	↑26 ↓3	↑17 ↓10	*↑12 ↓17*	↑177 ↓33
T_3	(35)	*↑0 ↓1*	↑18 ↓1	↑29 ↓0	↑33 ↓0	↑30 ↓1	↑24 ↓3	↑19 ↓8	↑17 ↓13	↑170 ↓27
T_4	(35)	*↑1 ↓3*	↑28 ↓1	↑35 ↓0	↑33 ↓0	↑32 ↓0	↑24 ↓4	↑20 ↓10	↑16 ↓16	↑189 ↓34
T_5	(35)	*↑0 ↓2*	↑21 ↓1	↑27 ↓0	↑31 ↓2	↑29 ↓3	↑23 ↓5	↑17 ↓9	↑14 ↓12	↑162 ↓34
T_6	(35)	*↑2 ↓3*	↑17 ↓0	↑29 ↓0	↑31 ↓0	↑30 ↓1	↑25 ↓3	↑21 ↓6	↑15 ↓11	↑170 ↓24
All	(270)	*↑3 ↓15*	↑160 ↓7	↑229 ↓1	↑241 ↓3	↑233 ↓11	↑192 ↓29	↑156 ↓60	↑130 ↓93	↑1344 ↓219

$$API = \frac{\sum_{a=1}^{n_{exp}} 100 \times PI_a}{n_{exp}} \tag{4.6}$$

with

$$PI_a = \begin{cases} \frac{H_{OE,a}(\text{DynDE})}{H_{OE,a}(\text{CDE})} - 1 \text{ if } H_{OE,a}(\text{DynDE}) < H_{OE,a}(\text{CDE}) \\ 1 - \frac{H_{OE,a}(\text{CDE})}{H_{OE,a}(\text{DynDE})} \text{ if } H_{OE,a}(\text{DynDE}) > H_{OE,a}(\text{CDE}) \end{cases} \tag{4.7}$$

where n_{exp} is the total number of experimental environments, and $H_{OE,a}(\text{DynDE})$ and $H_{OE,a}(\text{CDE})$ are the average offline errors of DynDE and CDE on experimental environment a, respectively. API is thus calculated by averaging the percentage improvement or deterioration over all experiments.

The average percentage improvement of CPE over DynDE over all experiments was found to be 10.88%. The APIs per dimension were found to be 3.12%, 10.57%, 15.67%, 13.95% and 11.05% for 5, 10, 25, 50 and 100 dimensions respectively. Larger improvements in offline error were thus found in higher dimensions. The APIs per change period were found to be -0.29%, 4.33%, 11.43%, 22.72%, 22.97%, 17.02%, 8.54% and 0.3% for change periods of 100, 500, 1 000, 5 000, 10 000, 25 000, 50 000 and 100 000 function evaluations respectively. The maximum improvements were thus found for change periods of 5 000 and 10 000, while small improvements were found for large change periods and a deterioration in offline error was found for the change periods of 100 function evaluations.

4.6.5.2 RMC compared to DynDE

The scalability study presented in Section 4.6.4 did not reveal any obviously different trends between DynDE and RMC. The analysis of the number of times that DynDE and RMC outperformed each other, given in Tables 4.13 and 4.14, shows that the average offline errors of DynDE and RMC differed statistically significantly in only 152 of the 2 160 experiments.

RMC outperformed DynDE in 102 cases while DynDE outperformed RMC in 50 cases. This indicates that there is very little benefit in using RMC. RMC did, however, outperform DynDE in 49 of the five dimensional experiments (i.e. 11.3% of the five dimensional

experiments), while it was only outperformed by DynDE in seven cases. This shows that while, in general, the RMC approach has a very small impact, it is still beneficial in low dimensional problems. The underlying function had a clear impact on when RMC proved useful. For example, on the spherical peak function, RMC outperformed DynDE in 15 (i.e. 31.2%) of the cases but was inferior to DynDE in only one case.

An experiment was conducted to investigate why RMC delivered improvements only in low dimensions, and why RMC is more effective when using the spherical peak function. The experiment used 100 000 random moving peak function fitness landscapes which were created for each of the numbers of dimensions from 2 to 40. For each case, the number of pairs of peaks, located within an Euclidean distance of less than the exclusion threshold of each other, was counted. For each pair of these peaks, a midpoint check was performed to determine whether the RMC algorithm would have detected multiple peaks correctly. The results of these experiments are depicted in Figures 4.32 and 4.33 for simulations on the conical and spherical peak functions respectively. Observe that the number of peaks, located within the exclusion threshold of each other, drops sharply from over 300 000 to zero between 2 and 16 dimensions. This explains why the RMC approach is only effective in low dimensional cases. The situations where RMC would have been useful did not occur in high dimensional problems. The midpoint check approach was more effective at identifying pairs of peaks on the spherical function than on the conical function in low dimensions (i.e. more cases as depicted in Figure 4.6, scenarios B and C, occur with the conical peak function). This explains why RMC yielded larger improvements in offline error on the spherical peak function than on the conical peak function.

The average percentage improvement of RMC over DynDE, over all experiments, was found to be 0.2%. The APIs per dimension were found to be 1.42%, -0.445%, 0.06%, -0.19% and 0.13% for 5, 10, 25, 50 and 100 dimensions respectively. The largest improvement was thus found in 5 dimensions. The APIs per change period were found to be -0.36%, 0.20%, 0.38%, 0.20%, 0.37%, 0.17%, 0.31% and 0.29% for change periods of 100, 500, 1 000, 5 000, 10 000, 25 000, 50 000 and 100 000 function evaluations respectively. The general improvement of RMC over CDE is thus minor and localised to 5 dimensions. Although RMC yielded low improvements on average, specific cases where large improvements were found to exist; for example, the experiment using the MPB's spherical peak function in five

Table 4.13: RMC vs DynDE performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑7 ↓0**
5	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑5 ↓0**
10	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑1 ↓1	**↑3 ↓1**
20	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑3 ↓0**
40	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
80	(2)	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓1**
C	(6)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑1 ↓1	**↑5 ↓1**
S	(6)	*↑0 ↓1*	↑0 ↓0	**↑1 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑5 ↓0**	**↑15 ↓1**
		GDBG								
F_{1a}	(6)	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑1 ↓0**	**↑4 ↓0**	**↑11 ↓0**
F_{1b}	(6)	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑0 ↓0	**↑9 ↓0**
F_2	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**
F_3	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑1 ↓1
F_4	(6)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
F_5	(6)	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	↑0 ↓0	**↑2 ↓0**	**↑5 ↓1**
F_6	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	*↑0 ↓3*
T_1	(7)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑6 ↓0**
T_2	(7)	*↑0 ↓1*	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	*↑0 ↓1*	**↑2 ↓0**	**↑6 ↓2**
T_3	(7)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑1 ↓1	**↑3 ↓1**
T_4	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑2 ↓0**
T_5	(7)	**↑3 ↓0**	**↑2 ↓0**	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑10 ↓1**
T_6	(7)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓1**
All	(54)	**↑3 ↓2**	**↑4 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑6 ↓1**	**↑7 ↓0**	**↑6 ↓1**	**↑13 ↓2**	**↑49 ↓7**
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
5	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
10	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*
20	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
40	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
C	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	**↑2 ↓1**
S	(6)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	*↑0 ↓1*	*↑1 ↓3*
F_{1b}	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
F_2	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	↑0 ↓0	↑1 ↓1	↑0 ↓0	↑0 ↓0	**↑4 ↓1**
F_3	(6)	↑0 ↓0	*↑0 ↓1*	*↑0 ↓2*	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	*↑0 ↓4*
F_4	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*
F_5	(6)	↑0 ↓0	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑3 ↓1**
F_6	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
T_1	(7)	↑0 ↓0	↑1 ↓1	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑1 ↓1
T_2	(7)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
T_3	(7)	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑1 ↓1
T_4	(7)	↑0 ↓0	**↑1 ↓0**	*↑0 ↓1*	**↑1 ↓0**	↑0 ↓0	↑1 ↓1	*↑0 ↓1*	*↑0 ↓1*	*↑3 ↓4*
T_5	(7)	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑2 ↓1**
T_6	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓3*	*↑0 ↓1*	↑0 ↓0	*↑0 ↓4*
All	(54)	**↑1 ↓0**	**↑2 ↓1**	*↑2 ↓3*	**↑3 ↓0**	↑0 ↓0	*↑2 ↓4*	*↑1 ↓3*	*↑0 ↓1*	*↑11 ↓12*
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
5	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
10	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
20	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑2 ↓0**
40	(2)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
80	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
C	(6)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
S	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑1 ↓1	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑3 ↓1**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓0**
F_{1b}	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
F_2	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑2 ↓0**
F_3	(6)	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	*↑1 ↓3*
F_4	(6)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑1 ↓1
F_5	(6)	↑1 ↓1	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓1**
F_6	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓2*	↑0 ↓0	↑0 ↓0	↑2 ↓2
T_1	(7)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓0**
T_2	(7)	*↑0 ↓1*	**↑2 ↓0**	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑3 ↓1**
T_3	(7)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓2*	**↑1 ↓0**	↑0 ↓0	**↑3 ↓2**
T_4	(7)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑1 ↓1
T_5	(7)	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓2*	↑0 ↓0	↑0 ↓0	*↑0 ↓3*
T_6	(7)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓0**
All	(54)	↑2 ↓2	**↑3 ↓1**	**↑2 ↓0**	**↑3 ↓0**	**↑2 ↓1**	*↑0 ↓4*	**↑2 ↓1**	↑0 ↓0	**↑14 ↓9**

Table 4.14: RMC vs DynDE performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
5	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
10	(2)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
20	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*
40	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
80	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
C	(6)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓3*
S	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑1 ↓1
		GDBG								
F_{1a}	(6)	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	*↑0 ↓4*
F_{1b}	(6)	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑4 ↓0**
F_2	(6)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓1**
F_3	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓1*	**↑2 ↓1**
F_4	(6)	↑0 ↓0	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	*↑2 ↓3*
F_5	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑1 ↓1	↑0 ↓0	↑1 ↓1
F_6	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
T_1	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	*↑0 ↓3*
T_2	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓3*
T_3	(7)	↑1 ↓1	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑3 ↓2**
T_4	(7)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	*↑0 ↓2*	↑0 ↓0	**↑4 ↓2**
T_5	(7)	↑0 ↓0	↑1 ↓1	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑3 ↓1**
T_6	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
All	(54)	↑1 ↓1	↑2 ↓2	↑1 ↓1	↑2 ↓2	*↑1 ↓3*	**↑3 ↓0**	*↑1 ↓3*	*↑1 ↓3*	*↑12 ↓15*
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓1*
5	(2)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
10	(2)	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	*↑0 ↓1*	↑2 ↓2
20	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
40	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
80	(2)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0
C	(6)	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑1 ↓2*
S	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	*↑0 ↓1*	**↑2 ↓1**
		GDBG								
F_{1a}	(6)	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑2 ↓1**
F_{1b}	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑5 ↓0**
F_2	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑2 ↓0**
F_3	(6)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
F_4	(6)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
F_5	(6)	↑0 ↓0	*↑0 ↓1*	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑1 ↓1
F_6	(6)	*↑0 ↓1*	*↑0 ↓1*	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑1 ↓2*
T_1	(7)	↑0 ↓0	↑1 ↓1	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓1**
T_2	(7)	↑0 ↓0	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓1**
T_3	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**
T_4	(7)	↑1 ↓1	*↑0 ↓1*	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑3 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑2 ↓2
T_5	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑5 ↓0**
T_6	(7)	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑1 ↓0**
All	(54)	↑1 ↓1	*↑2 ↓4*	**↑2 ↓1**	↑0 ↓0	**↑2 ↓0**	**↑5 ↓0**	**↑2 ↓0**	**↑2 ↓1**	**↑16 ↓7**
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	**↑1 ↓0**	*↑0 ↓1*	**↑2 ↓0**	**↑3 ↓0**	**↑2 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑10 ↓1**
5	(10)	↑0 ↓0	**↑1 ↓0**	↑0 ↓0	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑6 ↓1**
10	(10)	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	*↑0 ↓1*	*↑1 ↓2*	↑5 ↓5
20	(10)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓1**	**↑5 ↓1**
40	(10)	↑0 ↓0	*↑0 ↓1*	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑2 ↓1**
80	(10)	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑1 ↓2*	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	*↑2 ↓3*
C	(6)	↑0 ↓0	*↑0 ↓3*	**↑3 ↓1**	**↑2 ↓1**	**↑2 ↓1**	↑0 ↓0	*↑0 ↓1*	↑1 ↓1	↑8 ↓8
S	(6)	↑1 ↓1	**↑1 ↓0**	**↑2 ↓0**	**↑4 ↓0**	**↑4 ↓1**	**↑2 ↓0**	**↑3 ↓0**	**↑5 ↓2**	**↑22 ↓4**
		GDBG								
F_{1a}	(30)	**↑2 ↓1**	**↑4 ↓2**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑3 ↓1**	*↑1 ↓2*	**↑4 ↓2**	**↑16 ↓8**
F_{1b}	(30)	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑0 ↓0	**↑3 ↓0**	**↑4 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑19 ↓0**
F_2	(30)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑3 ↓0**	*↑0 ↓1*	**↑2 ↓1**	**↑3 ↓0**	**↑1 ↓0**	**↑12 ↓2**
F_3	(30)	*↑0 ↓1*	**↑2 ↓1**	*↑1 ↓2*	↑1 ↓1	↑0 ↓0	*↑1 ↓2*	*↑0 ↓1*	*↑0 ↓1*	*↑5 ↓9*
F_4	(30)	**↑2 ↓0**	**↑1 ↓0**	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	**↑2 ↓1**	*↑0 ↓2*	↑0 ↓0	↑5 ↓5
F_5	(30)	*↑1 ↓2*	↑1 ↓1	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	**↑3 ↓0**	**↑2 ↓1**	**↑2 ↓0**	**↑12 ↓5**
F_6	(30)	*↑0 ↓1*	↑1 ↓1	**↑1 ↓0**	↑1 ↓1	*↑0 ↓1*	*↑0 ↓3*	*↑0 ↓1*	*↑0 ↓1*	*↑3 ↓9*
T_1	(35)	↑0 ↓0	**↑4 ↓2**	**↑1 ↓0**	↑0 ↓0	↑1 ↓1	**↑1 ↓0**	**↑2 ↓1**	**↑2 ↓1**	**↑11 ↓5**
T_2	(35)	*↑0 ↓2*	**↑3 ↓1**	**↑2 ↓0**	↑1 ↓1	**↑2 ↓1**	**↑1 ↓0**	*↑0 ↓1*	**↑3 ↓1**	**↑12 ↓7**
T_3	(35)	**↑2 ↓1**	↑0 ↓0	*↑1 ↓2*	**↑1 ↓0**	↑0 ↓0	↑2 ↓2	**↑3 ↓0**	**↑2 ↓1**	**↑11 ↓6**
T_4	(35)	↑1 ↓1	**↑2 ↓1**	**↑3 ↓1**	**↑1 ↓0**	**↑1 ↓0**	**↑4 ↓1**	*↑0 ↓4*	*↑0 ↓1*	**↑12 ↓9**
T_5	(35)	**↑3 ↓1**	**↑3 ↓1**	*↑0 ↓1*	**↑2 ↓1**	**↑1 ↓0**	**↑5 ↓2**	**↑4 ↓0**	**↑2 ↓0**	**↑20 ↓6**
T_6	(35)	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	*↑0 ↓1*	*↑2 ↓3*	*↑0 ↓1*	**↑1 ↓0**	**↑6 ↓5**
All	(270)	**↑8 ↓6**	**↑13 ↓8**	**↑13 ↓5**	**↑12 ↓3**	**↑11 ↓5**	**↑17 ↓8**	**↑12 ↓8**	**↑16 ↓7**	**↑102 ↓50**

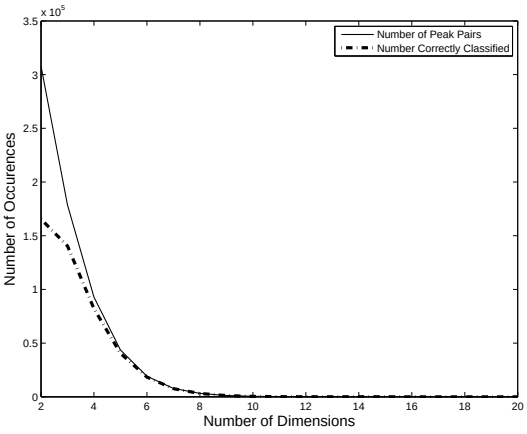

Figure 4.32: Number of peak pairs that fall within the exclusion threshold and number of correct classifications by RMC per dimension on the conical peak function.

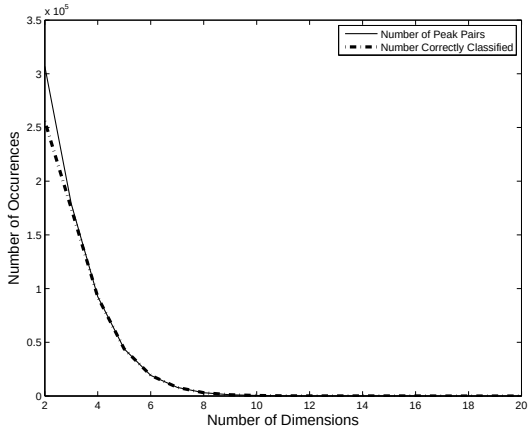

Figure 4.33: Number of peak pairs that fall within the exclusion threshold and number of correct classifications by RMC per dimension on the spherical peak function.

dimensions with a change period of 50 000 yielded a 54.61% improvement over DynDE.

The benefits of using RMC are less pronounced and wide-ranging than that of CPE. However, RMC does constitute an improvement over DynDE in a specific sub-set of dynamic environments.

4.6.5.3 CDE compared to DynDE

The analysis of the CDE results, in comparison with DynDE, is given in Tables 4.15 and 4.16. CDE performed significantly better than DynDE in 1 351 experiments and worse than DynDE in 208 cases. CDE was thus better than DynDE in 62.5% of all experiments and DynDE was better than CDE in only 9.6% of the experiments. This performance is considerably better than that of RMC, and slightly better than that of CPE. The most noticeable difference between CPE and CDE is in five dimensional experiments, in which the number of cases where CDE is better than DynDE is 17 more than those of CPE, and the number of cases where CDE is worse than DynDE is 10 less than CPE. The higher dimensional experiments yielded roughly equivalent performance in comparison to DynDE for CPE and CDE. RMC's positive contribution was shown to be localised to low dimensions, and the benefits of incorporating RMC into CPE are also limited to the five dimensional experiments.

CPE was shown to be more effective when using a low change period than when using a high change period. This trend is continued in CDE, where 86% of the cases where DynDE performed the best, occurred when a change period of 25 000 or higher was used.

The average percentage improvement of CDE over DynDE over all experiments was found to be 11.09%. The APIs per dimension were found to be 4.30%, 10.69%, 15.51%, 13.96% and 10.97% for 5, 10, 25, 50 and 100 dimensions respectively. Larger improvements in offline error were thus found in higher dimensions, as was the case with CPE. The APIs per change period, were found to be -0.22%, 4.56%, 11.51%, 22.66%, 22.68%, 17.45%, 9.02% and 1.04% for change periods of 100, 500, 1 000, 5 000, 10 000, 25 000, 50 000 and 100 000 function evaluations respectively. The maximum improvements were thus found for change periods of 5 000 and 10 000, while small improvements were found for large change periods and a deterioration in offline error was found for the change period of 100.

4.6.5.4 Summary for Research Question 3

CPE, RMC and CDE were all found to outperform DynDE more often than being outperformed by DynDE. RMC differed from DynDE in only a small number of instances, but was shown to be effective on isolated functions. CPE generally outperformed DynDE, with cases where it was inferior to DynDE being concentrated in the low dimensions with a high change period.

CDE was better than DynDE in more cases than both CPE and RMC. The scalability study conducted in Section 4.6.4 found that CDE scaled very similar to CPE, and is especially superior to DynDE in high dimensional, low change period and severely changing environments. Section 2.5.2 argued that the set of feasible DOPs for any particular function is bounded by values of the change severity and change period. CDE thus increases the set of feasible DOP by being more effective on high dimensional environments in which changes occur frequently.

An analysis was performed to determine whether CDE results in statistically significant improvements over its sub-components, CPE and RMC. Full results of these comparisons are given in Appendix B. CDE performed better than CPE in 86 of the 2160 experiments and performed worse in 47. The majority of the cases where CDE outperformed CPE (45 cases) was in five dimensional experiments due to the influence of the RMC component.

Table 4.15: CDE vs DynDE performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑7 ↓0**
5	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
10	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑11 ↓0**
20	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑10 ↓0**
40	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑1 ↓1	↑1 ↓1	**↑6 ↓2**
80	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑7 ↓1**
C	(6)	↑0 ↓0	**↑1 ↓0**	**↑3 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑4 ↓0**	**↑3 ↓1**	*↑1 ↓2*	**↑21 ↓3**
S	(6)	↑0 ↓0	↑0 ↓0	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑32 ↓0**
		GDBG								
F_{1a}	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓3*	*↑0 ↓6*	*↑0 ↓6*	**↑24 ↓16**
F_{1b}	(6)	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	*↑0 ↓4*	*↑0 ↓6*	*↑0 ↓6*	**↑24 ↓16**
F_2	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑5 ↓1**	↑3 ↓3	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑18 ↓22*
F_3	(6)	↑0 ↓0	**↑3 ↓0**	**↑4 ↓0**	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓1**
F_4	(6)	↑0 ↓0	**↑2 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	*↑0 ↓6*	*↑0 ↓6*	**↑24 ↓13**
F_5	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓2**	*↑1 ↓4*	*↑0 ↓5*	**↑29 ↓11**
F_6	(6)	↑0 ↓0	**↑3 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑4 ↓2**	*↑0 ↓2*	**↑29 ↓5**
T_1	(7)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	*↑0 ↓6*	*↑1 ↓6*	*↑1 ↓6*	**↑24 ↓19**
T_2	(7)	*↑0 ↓1*	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓1**	*↑1 ↓5*	*↑0 ↓5*	**↑28 ↓12**
T_3	(7)	↑0 ↓0	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑3 ↓2**	*↑1 ↓4*	*↑0 ↓5*	**↑27 ↓12**
T_4	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓4*	*↑1 ↓6*	*↑0 ↓6*	**↑26 ↓16**
T_5	(7)	**↑1 ↓0**	**↑3 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑5 ↓1**	**↑3 ↓2**	*↑2 ↓4*	*↑0 ↓4*	**↑25 ↓12**
T_6	(7)	↑0 ↓0	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑3 ↓3	*↑1 ↓5*	*↑1 ↓5*	**↑29 ↓13**
All	(54)	↑1 ↓1	**↑31 ↓0**	**↑49 ↓0**	**↑45 ↓1**	**↑42 ↓3**	**↑21 ↓18**	*↑15 ↓31*	*↑8 ↓33*	**↑212 ↓87**
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓1	↑1 ↓1	↑1 ↓1	*↑0 ↓1*	*↑0 ↓1*	↑0 ↓0	*↑0 ↓2*	*↑3 ↓7*
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑7 ↓0**
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
20	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓0**
40	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓0**
C	(6)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑33 ↓1**
S	(6)	↑0 ↓0	*↑0 ↓1*	**↑3 ↓1**	**↑4 ↓1**	**↑4 ↓1**	**↑4 ↓1**	**↑4 ↓0**	**↑4 ↓1**	**↑23 ↓6**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	*↑0 ↓4*	*↑0 ↓6*	**↑29 ↓10**
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓0**	*↑0 ↓5*	*↑0 ↓5*	**↑27 ↓10**
F_2	(6)	↑1 ↓1	**↑3 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑3 ↓1**	↑3 ↓3	*↑0 ↓4*	*↑0 ↓6*	**↑21 ↓15**
F_3	(6)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑3 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	↑1 ↓1	**↑12 ↓1**
F_4	(6)	↑0 ↓0	**↑2 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑3 ↓2**	*↑2 ↓4*	*↑0 ↓6*	**↑22 ↓13**
F_5	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓2**	**↑40 ↓2**
F_6	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑40 ↓1**
T_1	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	*↑2 ↓4*	*↑0 ↓6*	**↑32 ↓10**
T_2	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑4 ↓0**	**↑4 ↓1**	*↑2 ↓4*	*↑2 ↓4*	**↑32 ↓9**
T_3	(7)	*↑0 ↓1*	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑5 ↓0**	**↑3 ↓2**	*↑2 ↓2*	*↑2 ↓3*	**↑31 ↓8**
T_4	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑3 ↓2**	*↑0 ↓5*	**↑34 ↓7**
T_5	(7)	**↑1 ↓0**	**↑4 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑4 ↓2**	**↑4 ↓2**	*↑2 ↓3*	*↑3 ↓4*	**↑28 ↓11**
T_6	(7)	↑0 ↓0	**↑4 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓2**	*↑2 ↓5*	**↑34 ↓7**
All	(54)	↑1 ↓1	**↑35 ↓1**	**↑47 ↓1**	**↑49 ↓1**	**↑40 ↓3**	**↑35 ↓6**	**↑23 ↓17**	*↑17 ↓29*	**↑247 ↓59**
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑5 ↓0**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓0**
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑8 ↓0**
20	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓0**
40	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑13 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓0**
C	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑37 ↓0**
S	(6)	↑0 ↓0	**↑2 ↓0**	**↑4 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑26 ↓0**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑2 ↓2	**↑37 ↓2**
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	*↑1 ↓2*	**↑36 ↓2**
F_2	(6)	↑0 ↓0	**↑4 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑3 ↓1**	**↑33 ↓1**
F_3	(6)	↑0 ↓0	↑0 ↓0	**↑6 ↓0**	**↑5 ↓0**	**↑2 ↓1**	*↑0 ↓1*	*↑0 ↓5*	*↑0 ↓4*	**↑13 ↓11**
F_4	(6)	↑1 ↓1	**↑3 ↓0**	**↑2 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑3 ↓1**	**↑31 ↓2**
F_5	(6)	*↑0 ↓1*	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑39 ↓1**
F_6	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑39 ↓0**
T_1	(7)	*↑0 ↓2*	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑42 ↓6**
T_2	(7)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓1**	*↑2 ↓3*	**↑37 ↓4**
T_3	(7)	↑0 ↓0	**↑3 ↓0**	**↑4 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑2 ↓1**	**↑35 ↓2**
T_4	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑4 ↓3**	**↑42 ↓4**
T_5	(7)	↑0 ↓0	**↑4 ↓0**	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓1**	**↑4 ↓0**	**↑35 ↓1**
T_6	(7)	**↑1 ↓0**	**↑3 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑3 ↓2**	**↑37 ↓2**
All	(54)	*↑1 ↓2*	**↑33 ↓0**	**↑45 ↓0**	**↑50 ↓0**	**↑47 ↓1**	**↑47 ↓1**	**↑39 ↓5**	**↑29 ↓10**	**↑291 ↓19**

Table 4.16: CDE vs DynDE performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑12 ↓0**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑12 ↓0**
10	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑13 ↓0**
20	(2)	*↑0 ↓1*	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓1**
40	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓0**
80	(2)	**↑1 ↓0**	↑1 ↓1	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓1**
C	(6)	**↑1 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑3 ↓0**	**↑37 ↓0**
S	(6)	*↑0 ↓1*	**↑2 ↓1**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑37 ↓2**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑4 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑38 ↓0**
F_{1b}	(6)	*↑0 ↓1*	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑40 ↓1**
F_2	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑39 ↓2**
F_3	(6)	*↑0 ↓1*	**↑1 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓0**	↑0 ↓0	*↑0 ↓4*	*↑0 ↓4*	**↑16 ↓9**
F_4	(6)	↑0 ↓0	**↑5 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑4 ↓1**	**↑35 ↓1**
F_5	(6)	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑24 ↓0**
F_6	(6)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑35 ↓0**
T_1	(7)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑5 ↓1**	**↑38 ↓2**
T_2	(7)	**↑1 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑4 ↓3**	**↑38 ↓3**
T_3	(7)	↑0 ↓0	**↑2 ↓0**	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑5 ↓1**	**↑38 ↓2**
T_4	(7)	*↑0 ↓1*	**↑5 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑5 ↓1**	**↑41 ↓3**
T_5	(7)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑4 ↓0**	**↑38 ↓2**
T_6	(7)	*↑0 ↓1*	**↑3 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑34 ↓1**
All	(54)	*↑2 ↓4*	**↑31 ↓1**	**↑43 ↓0**	**↑52 ↓0**	**↑51 ↓0**	**↑46 ↓0**	**↑40 ↓4**	**↑36 ↓6**	**↑301 ↓15**
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	**↑10 ↓5**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	**↑2 ↓0**	↑1 ↓1	*↑0 ↓1*	**↑9 ↓3**
10	(2)	*↑0 ↓1*	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑13 ↓1**
20	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	**↑2 ↓0**	**↑10 ↓3**
40	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	**↑2 ↓0**	**↑12 ↓1**
80	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑13 ↓0**
C	(6)	↑0 ↓0	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑36 ↓0**
S	(6)	↑1 ↓1	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓1**	↑3 ↓3	**↑4 ↓2**	*↑2 ↓4*	**↑4 ↓2**	**↑31 ↓13**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓0**
F_{1b}	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓0**
F_2	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑39 ↓1**
F_3	(6)	↑0 ↓0	*↑0 ↓2*	**↑4 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓0**	*↑0 ↓1*	*↑0 ↓5*	**↑19 ↓8**
F_4	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑41 ↓2**
F_5	(6)	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑3 ↓0**	**↑4 ↓0**	**↑20 ↓0**
F_6	(6)	*↑0 ↓1*	**↑5 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑4 ↓2**	**↑34 ↓4**
T_1	(7)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓1**	**↑5 ↓1**	**↑39 ↓2**
T_2	(7)	↑0 ↓0	**↑6 ↓1**	**↑7 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓3**	**↑44 ↓4**
T_3	(7)	↑0 ↓0	**↑3 ↓1**	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓2**	**↑38 ↓3**
T_4	(7)	*↑0 ↓2*	**↑4 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑43 ↓3**
T_5	(7)	**↑1 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑5 ↓2**	**↑37 ↓3**
T_6	(7)	↑0 ↓0	**↑2 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑32 ↓0**
All	(54)	*↑2 ↓3*	**↑32 ↓2**	**↑45 ↓0**	**↑50 ↓1**	**↑47 ↓3**	**↑46 ↓2**	**↑39 ↓6**	**↑39 ↓11**	**↑300 ↓28**
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	**↑1 ↓0**	**↑7 ↓1**	**↑8 ↓1**	**↑6 ↓2**	**↑6 ↓2**	**↑5 ↓2**	**↑2 ↓1**	*↑2 ↓3*	**↑37 ↓12**
5	(10)	↑0 ↓0	**↑4 ↓0**	**↑10 ↓0**	**↑8 ↓0**	**↑7 ↓1**	**↑9 ↓0**	**↑8 ↓1**	**↑5 ↓1**	**↑51 ↓3**
10	(10)	*↑0 ↓1*	**↑5 ↓0**	**↑8 ↓0**	**↑9 ↓0**	**↑9 ↓0**	**↑10 ↓0**	**↑9 ↓0**	**↑7 ↓0**	**↑57 ↓1**
20	(10)	*↑0 ↓1*	**↑3 ↓0**	**↑8 ↓0**	**↑10 ↓0**	**↑9 ↓1**	**↑9 ↓1**	**↑9 ↓1**	**↑9 ↓0**	**↑57 ↓4**
40	(10)	↑0 ↓0	**↑2 ↓0**	**↑8 ↓0**	**↑9 ↓0**	**↑9 ↓0**	**↑9 ↓0**	**↑8 ↓2**	**↑9 ↓1**	**↑54 ↓3**
80	(10)	**↑1 ↓0**	**↑2 ↓1**	**↑8 ↓0**	**↑10 ↓0**	**↑9 ↓0**	**↑9 ↓0**	**↑9 ↓0**	**↑9 ↓1**	**↑57 ↓2**
C	(6)	**↑1 ↓0**	**↑13 ↓0**	**↑26 ↓0**	**↑29 ↓0**	**↑27 ↓0**	**↑27 ↓0**	**↑24 ↓1**	**↑17 ↓3**	**↑164 ↓4**
S	(6)	*↑1 ↓2*	**↑10 ↓2**	**↑24 ↓1**	**↑23 ↓2**	**↑22 ↓4**	**↑24 ↓3**	**↑21 ↓4**	**↑24 ↓3**	**↑149 ↓21**
		GDBG								
F_{1a}	(30)	*↑0 ↓1*	**↑26 ↓0**	**↑28 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑23 ↓3**	**↑18 ↓10**	*↑13 ↓14*	**↑168 ↓28**
F_{1b}	(30)	↑1 ↓1	**↑27 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑29 ↓0**	**↑21 ↓4**	**↑17 ↓11**	*↑12 ↓13*	**↑167 ↓29**
F_2	(30)	*↑1 ↓2*	**↑21 ↓0**	**↑26 ↓0**	**↑29 ↓1**	**↑24 ↓4**	**↑21 ↓9**	**↑15 ↓10**	*↑13 ↓15*	**↑150 ↓41**
F_3	(30)	*↑0 ↓1*	**↑7 ↓2**	**↑24 ↓0**	**↑20 ↓0**	**↑11 ↓1**	**↑4 ↓2**	*↑2 ↓10*	*↑3 ↓14*	**↑71 ↓30**
F_4	(30)	*↑1 ↓2*	**↑18 ↓0**	**↑23 ↓0**	**↑30 ↓0**	**↑28 ↓1**	**↑25 ↓3**	**↑16 ↓10**	*↑12 ↓15*	**↑153 ↓31**
F_5	(30)	**↑2 ↓1**	**↑19 ↓0**	**↑21 ↓0**	**↑25 ↓0**	**↑26 ↓0**	**↑24 ↓2**	**↑19 ↓4**	**↑16 ↓7**	**↑152 ↓14**
F_6	(30)	*↑0 ↓1*	**↑21 ↓0**	**↑27 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑26 ↓1**	**↑24 ↓3**	**↑19 ↓5**	**↑177 ↓10**
T_1	(35)	*↑0 ↓2*	**↑28 ↓0**	**↑32 ↓0**	**↑30 ↓0**	**↑28 ↓2**	**↑22 ↓7**	**↑19 ↓13**	**↑16 ↓15**	**↑175 ↓39**
T_2	(35)	↑1 ↓1	**↑26 ↓1**	**↑32 ↓0**	**↑34 ↓0**	**↑31 ↓0**	**↑26 ↓2**	**↑17 ↓10**	*↑12 ↓18*	**↑179 ↓32**
T_3	(35)	*↑0 ↓1*	**↑18 ↓1**	**↑28 ↓0**	**↑34 ↓0**	**↑31 ↓1**	**↑24 ↓4**	**↑20 ↓8**	**↑14 ↓12**	**↑169 ↓27**
T_4	(35)	*↑0 ↓3*	**↑28 ↓0**	**↑33 ↓0**	**↑34 ↓0**	**↑32 ↓0**	**↑23 ↓4**	**↑21 ↓10**	*↑15 ↓16*	**↑186 ↓33**
T_5	(35)	**↑3 ↓1**	**↑22 ↓0**	**↑26 ↓0**	**↑31 ↓1**	**↑27 ↓3**	**↑23 ↓4**	**↑15 ↓10**	**↑16 ↓10**	**↑163 ↓29**
T_6	(35)	↑1 ↓1	**↑17 ↓0**	**↑28 ↓0**	**↑31 ↓0**	**↑29 ↓0**	**↑26 ↓3**	**↑19 ↓7**	**↑15 ↓12**	**↑166 ↓23**
All	(270)	*↑7 ↓11*	**↑162 ↓4**	**↑229 ↓1**	**↑246 ↓3**	**↑227 ↓10**	**↑195 ↓27**	**↑156 ↓63**	**↑129 ↓89**	**↑1351 ↓208**

The results indicate that CDE is thus a slightly better algorithm than CPE. CDE performed better than RMC in 1 342 cases and performed worse in only 209 experiments. CDE is thus clearly a better algorithm than RMC.

4.6.6 Research Question 4

How does the convergence behaviour of CDE differ from that of DynDE?

The analysis under the previous research question found that CDE is a more effective DOP algorithm than DynDE. This research question investigates the difference between DynDE and CDE in terms of diversity and current error with the aim of explaining trends observed in the previous sections.

Section 3.4.1.5 showed that one of the reasons why DynDE is more effective than normal DE on DOPs is that DynDE maintains higher diversity during the optimisation process. The average diversity (calculated using equation (2.7)) of CDE was measured on the conical peak function of the MPB in five dimensions with a change period of 5 000 function evaluations. The optimisation process was allowed to continue for 500 000 function evaluations and was repeated 30 times. The average diversity per generation over all repeats was found to be 0.2624 for CDE, compared to the values of 0.2661 for DynDE, and 0.0017 for DE. The diversity of CDE is thus virtually identical to that of DynDE.

The diversity measure given in equation (2.7) gives the diversity of all individuals used by the algorithm. This measure will appear artificially high because sub-populations converge to optima that are uniformly distributed around the fitness landscape. The measure also gives no information regarding the diversity within sub-populations. Equation (2.7) can be adapted to give the average diversity per sub-population, D_{AP}, as shown in equation (4.8).

$$D_{AP} = \frac{\sum_{i=1}^{n_k} \frac{\sum_{i=1}^{n_{I,k}} \|\vec{d} - \vec{x}_{i,k}\|_2}{n_{I,k}}}{n_k L} \tag{4.8}$$

D_{AP} can be used to study the diversity within sub-populations. Figure 4.34 gives the offline error, current error, diversity and average diversity per population for DynDE and CDE on the conical peak function of the MPB in five dimensions with a change period of 5 000 function evaluations. The first 10 changes in the environment are depicted.

The figure shows that, while DynDE and CDE had very similar normal diversity profiles, the average diversity per sub-population differed drastically between DynDE and CDE. DynDE's value for D_{AP} decreased gradually over many function evaluations. This means that DynDE's sub-populations slowly converged to optima in the fitness landscape.

CDE's value for D_{AP}, in contrast to DynDE, rapidly decreased to a relatively low value. CDE thus converged to optima faster than DynDE, due to the competitive population evaluation approach which allows sub-populations to evolve in sequence and thus discover optima faster. The RMC component of CDE also reduced the average sub-population diversity as it prevents the unnecessary reinitialisation of sub-populations which would have dispersed individuals randomly over the search space. Figure 4.34 shows that CDE's current error reduced faster than that of DynDE after changes in the environment. This is due to CDE's sub-populations converging to optima earlier than DynDE's sub-populations.

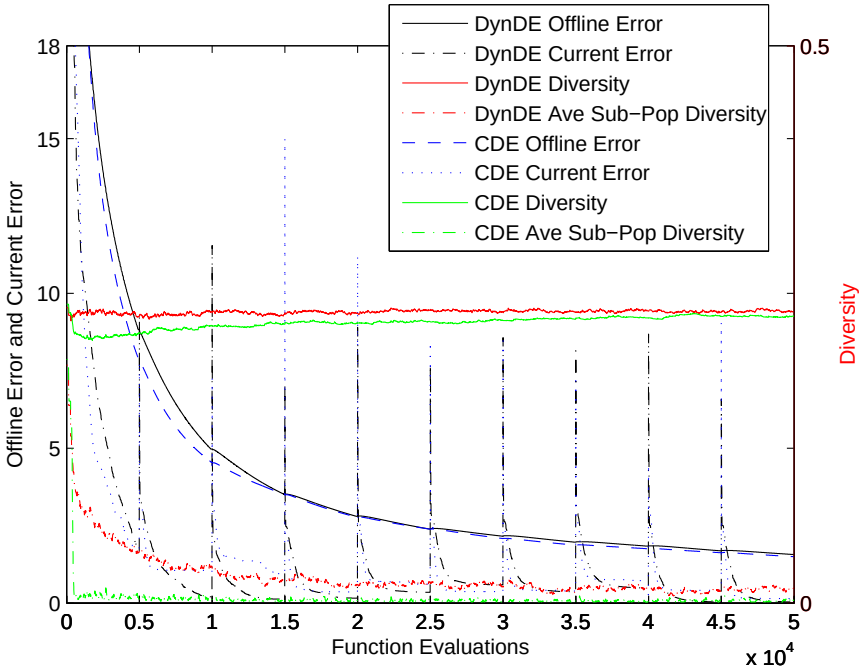

Figure 4.34: Diversity, current error, offline error and average sub-population diversity of DynDE and CDE on the MPB, Scenario 2

The previous sections found that CDE is more effective than DynDE, especially when a low change period and high dimensions are present in the environment. The convergence behaviour of DynDE and CDE is investigated here on environments with low change

period - low dimensions, low change period - high dimensions, high change period - low dimensions, and high change period - high dimensions.

Figure 4.35 gives the offline error, current error, diversity and average diversity per population for DynDE and CDE on the conical peak function of the MPB in five dimensions with a change period of 1 000 function evaluations. The figure confirms that the average diversity per sub-population of CDE droped noticeably faster than that of DynDE. This resulted in faster reductions in CDE current error after changes in the environment, and current errors that were consistently lower than those of DynDE. CDE thus not only found optima faster, but also achieved lower current errors than DynDE in the presence of frequent changes.

Figure 4.36 gives the offline error, current error, diversity and average diversity per population for DynDE and CDE on the conical peak function of the MPB in 100 dimensions with a change period of 1 000 function evaluations. The average diversity per sub-population of CDE was once again lower than that of DynDE. The high number of dimensions resulted in considerably larger errors than were found in the five dimensional cases. The magnitude of the difference between the current errors of DynDE and CDE was greater than what it was on the low dimensional experiment (compare Figure 4.36 to Figure 4.35). The allocation of function evaluations based on performance was thus especially beneficial in high dimensions, which explains why CDE generally achieved better offline errors than DynDE in high dimensional DOPs.

The offline error, current error, diversity and average diversity per population for DynDE and CDE on the conical peak function of the MPB in five dimensions with a change period of 100 000 function evaluations, is given in Figure 4.37. The average diversity per sub-population of CDE reduced faster than that of DynDE. However, the large number of function evaluations between changes allowed DynDE's sub-populations more function evaluations to converge. The result was that the average diversity per sub-population of DynDE became much closer to that of CDE than what it was in Figure 4.35. The high change period allowed both algorithms enough function evaluations to achieve low offline errors.

Figure 4.38 gives an enlargement of the area around the third change in the environment. CDE's current error decreased slightly faster than that of DynDE after the change,

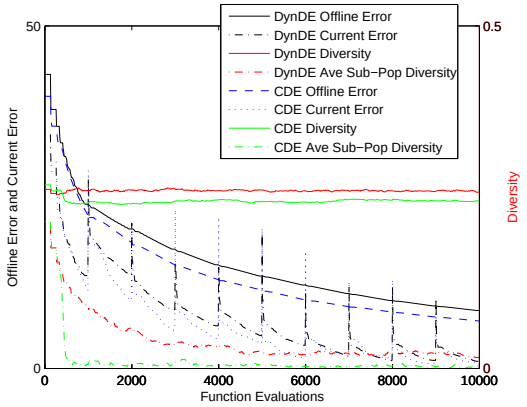

Figure 4.35: DynDE and CDE on the MPB with a conical peak function in five dimensions using a change period of 1 000

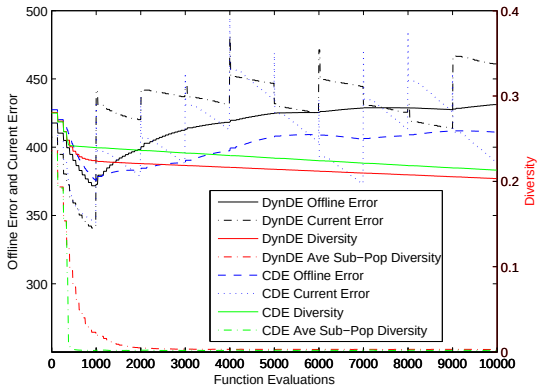

Figure 4.36: DynDE and CDE on the MPB with a conical peak function in 100 dimensions using a change period of 1 000

but DynDE's current error eventually reached a lower value than that of CDE. The large number of function evaluations between changes made the initial fast decrease in current error of CDE negligible, and DynDE thus yielded a lower offline error on this environment.

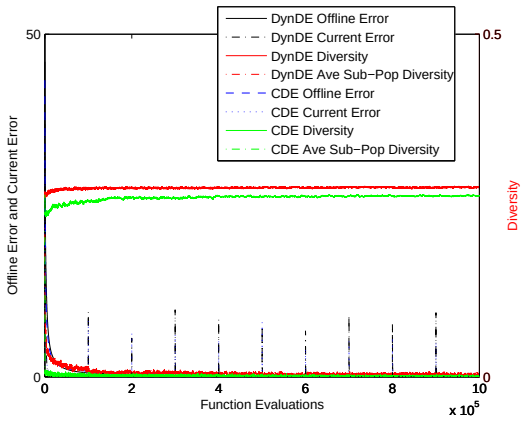

Figure 4.37: DynDE and CDE on the MPB with a conical peak function in five dimensions using a change period of 100 000

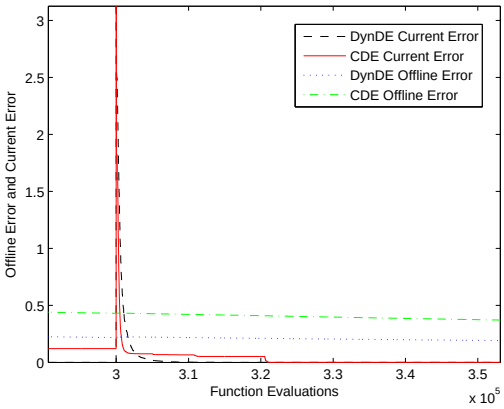

Figure 4.38: DynDE and CDE on the MPB with a conical peak function in five dimensions using a change period of 100 000, enlarged

Figure 4.39 gives the offline error, current error, diversity and average diversity per population for DynDE and CDE on the conical peak function of the MPB in 100 dimensions with a change period of 100 000 function evaluations. The average diversity per sub-

population of DynDE and CDE was, once again, very similar due to the large number of function evaluations available between changes in the environment. Note that the normal diversity profile of both algorithms initially decreased as the sub-populations frequently converged to the same optima. The normal diversity and the average sub-population diversity then increased at about 60 000 function evaluations as sub-populations were reinitialised due to exclusion. Note that much lower current and offline errors were achieved by both algorithms than were found when using a low change period in conjunction with a high number of dimensions (compare Figure 4.39 to Figure 4.36).

Figure 4.40 gives an enlargement of the area around the second change in the environment. DynDE's current error typically took a large number of function evaluations to reach its lowest value, while CDE's current error decreased faster after a change in the environment. CDE consequently yielded lower offline errors on high dimensional problems with a high change period.

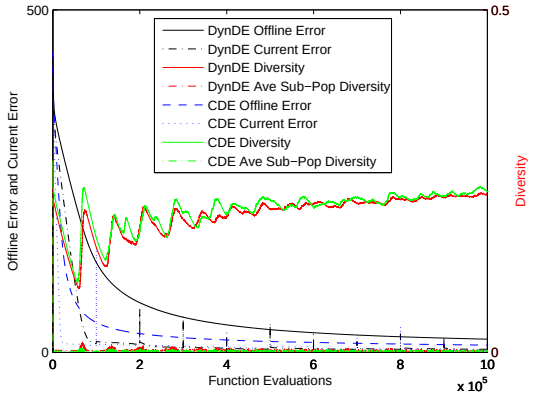

Figure 4.39: DynDE and CDE on the MPB with a conical peak function in 100 dimensions using a change period of 100 000

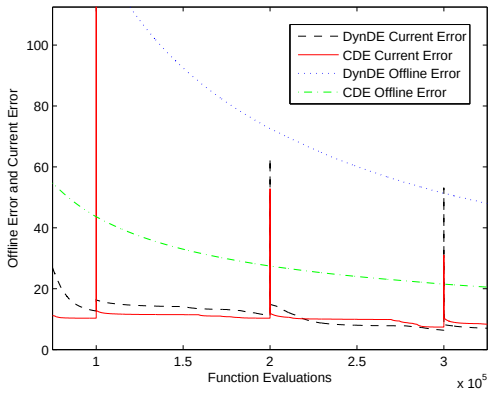

Figure 4.40: DynDE and CDE on the MPB with a conical peak function in 100 dimensions using a change period of 100 000, enlarged

4.6.7 Research Question 5

How does the average lowest error found just before changes in the environment differ between DynDE and CDE?

The components used to form CDE have been shown to produce a lower offline error than normal DynDE. However, CDE could, potentially negatively, affect the lowest error

found immediately before changes in the environment occur. A reduction in offline error does not necessarily imply that the lowest error, found before a change in the environment occurs, is also reduced. For example, the CPE component could allocate too many function evaluations to a sub-population that is positioned on a sub-optimal peak, to the detriment of the sub-population that is positioned on the global optimum. This research question investigates the possibility that CDE merely exploits the offline error performance measure and is not effective when the errors immediately before changes are considered.

The second performance measurement discussed in Section 2.5.5, H_{AEBI}, calculates the average error of the best individuals found just before changes in the environment. This performance measure was not used as the main performance measure in this study, because it only considers the performance immediately before changes in the environment, and ignores the performance during periods between changes. This performance measure is, however, ideal for answering this research question.

The same set of experiments, used to compare the performance of DynDE and CDE in Section 4.6.5, was repeated using the H_{AEBI} performance measure. A total of 2 160 experiments were thus conducted for each algorithm. The same analysis that was performed in Section 4.6.5, i.e. a count of the number of instances that each algorithm was statistically significantly better than the other, was performed. The results of this analysis are given in Tables 4.17 and 4.18.

CDE performed significantly better than DynDE in just under half of the experiments (a total of 1 068 cases). DynDE performed better than CDE in 470 cases. The general trend that can be observed is that DynDE performed better than CDE when a high change period was used. The analysis of the five dimensional experiments shows that CDE performed better, more often, than DynDE only in the experiments that used a change period of 500 or 1 000. As the number of dimensions was increased, CDE performed comparatively better then DynDE. CDE performed better more often than DynDE in 100 dimensions for all settings of change period except in the experiments that used a change period of 100 000.

These results are consistent with the comparisons done in the previous section in terms of the current errors of CDE and DynDE. The low dimensional - low change period experiment, shown in Figure 4.35, found that CDE's current error was typically lower

than that of DynDE just before changes occur in the environment. H_{AEBI} only considers the error just before a change, and CDE consequently performed better than DynDE in terms of H_{AEBI} for this case. The low dimensional - high change period experiment, shown in Figure 4.37, found that CDE, typically, had a higher current error than DynDE immediately before a change. The H_{AEBI} performance measure thus rates CDE lower than DynDE when low dimensions are used in conjunction with a high change period.

The analysis presented in this section suggests that CDE is inferior to DynDE in terms of the H_{AEBI} performance measure in the presence of large change periods. The competitive evaluation approach is aimed at discovering optima early and is consequently not beneficial in problems where only the error immediately before infrequent changes is important. However, CDE did perform better than DynDE more than twice as often as DynDE performed better than CDE, over all experiments. The cases where CDE was better than DynDE where mostly in low change period - high dimensional problems. The algorithms presented in this chapter thus improved the offline error of DynDE without completely deteriorating the error immediately before changes in the environment.

4.7 Comparison to Other Approaches

This chapter showed that using RMC and CPE yields better results than DynDE alone. This section compares the combined CDE algorithm to other algorithms in the literature. Section 4.7.1 compares the performance of CDE to that of *jDE* (refer to Section 3.4.2). Section 4.7 compares CDE to the published results from other algorithms.

4.7.1 CDE Compared to jDE

jDE, developed by Brest *et al.* [2009], was the winning algorithm of the CEC2009 competition on dynamic optimisation, and is currently one of the state-of-the-art algorithms for optimisation in dynamic environments. *jDE* was selected for a detailed comparison to CDE because of its good performance, and because it is also based on DE.

The *jDE* algorithm was implemented by the author in order to compare its performance to the algorithms presented in this thesis. The *jDE* algorithm was executed on the standard set of experiments for all the combinations of change period and number

Table 4.17: CDE vs DynDE H_{AEBI} performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓1	*↑0 ↓1*	↑0 ↓0	↑1 ↓1	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑6 ↓3**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑8 ↓1**
10	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑1 ↓1	*↑0 ↓2*	↑1 ↓1	**↑6 ↓4**
20	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	*↑0 ↓1*	*↑0 ↓1*	↑1 ↓1	↑1 ↓1	**↑7 ↓4**
40	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	*↑0 ↓1*	↑1 ↓1	*↑0 ↓1*	**↑6 ↓4**
80	(2)	*↑0 ↓1*	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	*↑0 ↓2*	*↑0 ↓1*	*↑0 ↓1*	↑5 ↓5
C	(6)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑3 ↓1**	*↑0 ↓3*	*↑0 ↓4*	*↑1 ↓4*	*↑0 ↓4*	*↑12 ↓16*
S	(6)	*↑0 ↓1*	*↑0 ↓1*	**↑5 ↓1**	**↑5 ↓0**	**↑5 ↓0**	**↑3 ↓1**	**↑4 ↓1**	**↑4 ↓0**	**↑26 ↓5**
		GDBG								
F_{1a}	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓4*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑12 ↓29*
F_{1b}	(6)	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑13 ↓30*
F_2	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑11 ↓30*
F_3	(6)	↑0 ↓0	**↑5 ↓0**	**↑3 ↓0**	*↑0 ↓1*	↑0 ↓0	*↑0 ↓1*	**↑2 ↓0**	**↑2 ↓0**	**↑12 ↓2**
F_4	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓1**	*↑0 ↓5*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑16 ↓24*
F_5	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓2**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑16 ↓27*
F_6	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑2 ↓1**	*↑0 ↓1*	*↑0 ↓2*	*↑1 ↓3*	*↑0 ↓3*	**↑15 ↓10**
T_1	(7)	↑0 ↓0	**↑7 ↓0**	**↑6 ↓0**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑1 ↓6*	*↑1 ↓6*	*↑15 ↓30*
T_2	(7)	*↑0 ↓1*	**↑7 ↓0**	**↑7 ↓0**	**↑3 ↓2**	*↑0 ↓5*	*↑0 ↓5*	*↑0 ↓6*	*↑0 ↓6*	*↑17 ↓25*
T_3	(7)	*↑0 ↓1*	**↑7 ↓0**	**↑7 ↓0**	↑2 ↓2	*↑0 ↓5*	*↑0 ↓6*	*↑0 ↓5*	*↑0 ↓5*	*↑16 ↓24*
T_4	(7)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	*↑1 ↓4*	*↑0 ↓5*	*↑0 ↓6*	*↑1 ↓6*	*↑0 ↓6*	*↑14 ↓27*
T_5	(7)	**↑1 ↓0**	**↑5 ↓0**	**↑6 ↓0**	*↑2 ↓4*	*↑0 ↓4*	*↑0 ↓5*	*↑1 ↓5*	*↑0 ↓5*	*↑15 ↓23*
T_6	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	↑3 ↓3	*↑0 ↓5*	*↑0 ↓5*	*↑0 ↓5*	*↑1 ↓5*	*↑18 ↓23*
All	(54)	*↑1 ↓3*	**↑42 ↓1**	**↑49 ↓1**	*↑19 ↓22*	*↑5 ↓33*	*↑3 ↓38*	*↑8 ↓38*	*↑6 ↓37*	*↑133 ↓173*
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑1 ↓1	↑1 ↓1	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓2*	*↑0 ↓2*	*↑2 ↓7*
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	*↑0 ↓2*	*↑0 ↓2*	*↑3 ↓4*
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	**↑3 ↓2**
20	(2)	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	↑0 ↓0	*↑0 ↓1*	↑1 ↓1	*↑0 ↓1*	**↑4 ↓3**
40	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	*↑0 ↓1*	*↑0 ↓1*	*↑0 ↓1*	**↑6 ↓3**
80	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑0 ↓0	*↑0 ↓2*	*↑0 ↓1*	**↑6 ↓3**
C	(6)	↑0 ↓0	**↑3 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑2 ↓0**	*↑0 ↓3*	*↑1 ↓3*	*↑0 ↓2*	**↑16 ↓8**
S	(6)	↑0 ↓0	*↑0 ↓1*	**↑3 ↓1**	**↑3 ↓0**	**↑2 ↓0**	↑0 ↓0	*↑0 ↓6*	*↑0 ↓6*	*↑8 ↓14*
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓4*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑18 ↓22*
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑0 ↓5*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑18 ↓23*
F_2	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓2**	*↑2 ↓3*	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑17 ↓24*
F_3	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	*↑0 ↓1*	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑1 ↓0**	**↑12 ↓1**
F_4	(6)	↑0 ↓0	**↑3 ↓0**	**↑6 ↓0**	**↑3 ↓1**	**↑3 ↓2**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	*↑15 ↓21*
F_5	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑1 ↓3*	*↑0 ↓4*	*↑0 ↓5*	**↑25 ↓12**
F_6	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑4 ↓2**	**↑5 ↓1**	**↑37 ↓4**
T_1	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑3 ↓2**	*↑0 ↓6*	*↑0 ↓6*	*↑0 ↓6*	**↑23 ↓20**
T_2	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑4 ↓2**	*↑2 ↓3*	*↑1 ↓5*	*↑1 ↓5*	*↑1 ↓5*	**↑22 ↓20**
T_3	(7)	*↑0 ↓1*	**↑6 ↓0**	**↑7 ↓0**	**↑4 ↓0**	↑2 ↓2	*↑1 ↓4*	*↑1 ↓5*	*↑1 ↓5*	**↑22 ↓17**
T_4	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓2**	*↑0 ↓5*	*↑0 ↓6*	*↑1 ↓5*	**↑25 ↓18**
T_5	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑4 ↓2**	*↑2 ↓3*	*↑2 ↓4*	*↑1 ↓4*	*↑2 ↓5*	**↑24 ↓18**
T_6	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑4 ↓2**	*↑1 ↓4*	*↑1 ↓4*	*↑1 ↓4*	**↑26 ↓14**
All	(54)	*↑0 ↓1*	**↑41 ↓1**	**↑51 ↓1**	**↑37 ↓4**	**↑21 ↓14**	*↑5 ↓31*	*↑5 ↓39*	*↑6 ↓38*	**↑166 ↓129**
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑7 ↓0**
5	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑8 ↓0**
10	(2)	↑0 ↓0	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑2 ↓0**	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	**↑5 ↓2**
20	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑0 ↓0	↑0 ↓0	↑0 ↓0	**↑8 ↓0**
40	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	↑0 ↓0	**↑8 ↓0**
80	(2)	↑0 ↓0	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑8 ↓0**
C	(6)	↑0 ↓0	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑0 ↓0	*↑0 ↓1*	*↑0 ↓1*	**↑22 ↓2**
S	(6)	↑0 ↓0	**↑2 ↓0**	**↑4 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓0**	↑0 ↓0	**↑1 ↓0**	**↑22 ↓0**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑2 ↓2	*↑0 ↓3*	*↑0 ↓5*	**↑26 ↓10**
F_{1b}	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	*↑1 ↓3*	*↑0 ↓6*	*↑0 ↓6*	**↑25 ↓15**
F_2	(6)	*↑0 ↓1*	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑3 ↓2**	*↑0 ↓4*	**↑28 ↓7**
F_3	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑2 ↓1**	*↑0 ↓4*	*↑0 ↓5*	*↑0 ↓5*	*↑0 ↓4*	*↑13 ↓19*
F_4	(6)	*↑1 ↓2*	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑3 ↓1**	**↑3 ↓0**	*↑0 ↓3*	**↑29 ↓6**
F_5	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑2 ↓0**	*↑0 ↓1*	**↑31 ↓2**
F_6	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑41 ↓0**
T_1	(7)	*↑0 ↓2*	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑3 ↓2**	*↑1 ↓5*	**↑35 ↓12**
T_2	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	*↑2 ↓3*	*↑1 ↓4*	*↑1 ↓6*	**↑30 ↓13**
T_3	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑3 ↓1**	↑2 ↓2	*↑1 ↓2*	**↑32 ↓6**
T_4	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	↑3 ↓3	↑3 ↓3	*↑1 ↓5*	**↑33 ↓12**
T_5	(7)	*↑0 ↓2*	**↑7 ↓0**	**↑7 ↓0**	**↑5 ↓0**	**↑4 ↓0**	**↑4 ↓1**	*↑1 ↓3*	*↑1 ↓2*	**↑29 ↓8**
T_6	(7)	**↑1 ↓0**	**↑5 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑4 ↓2**	**↑3 ↓2**	*↑1 ↓3*	**↑34 ↓8**
All	(54)	*↑1 ↓4*	**↑45 ↓0**	**↑52 ↓0**	**↑48 ↓1**	**↑46 ↓4**	**↑25 ↓11**	*↑13 ↓17*	*↑7 ↓24*	**↑237 ↓61**

Table 4.18: CDE vs DynDE H_{AEBI} performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓1	↑0 ↓2	↑10 ↓3
5	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓1	↑0 ↓1	↑9 ↓2
10	(2)	↑0 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓1	↑0 ↓2	↑10 ↓3
20	(2)	↑0 ↓1	↑0 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓1	↑0 ↓2	↑8 ↓4
40	(2)	↑0 ↓0	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓0	↑0 ↓1	↑7 ↓1
80	(2)	↑0 ↓0	↑1 ↓1	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓0	↑0 ↓1	↑10 ↓2
C	(6)	↑0 ↓0	↑4 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑1 ↓4	↑0 ↓4	↑29 ↓8
S	(6)	↑0 ↓1	↑2 ↓1	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑0 ↓0	↑0 ↓5	↑25 ↓7
		GDBG								
F_{1a}	(6)	↑0 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑2 ↓2	↑1 ↓3	↑31 ↓5
F_{1b}	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑2 ↓4	↑1 ↓3	↑32 ↓7
F_2	(6)	↑0 ↓1	↑6 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑5 ↓1	↑3 ↓1	↑36 ↓4
F_3	(6)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑0 ↓0	↑0 ↓6	↑0 ↓5	↑0 ↓5	↑16 ↓16
F_4	(6)	↑0 ↓0	↑6 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑4 ↓1	↑3 ↓1	↑35 ↓3
F_5	(6)	↑1 ↓0	↑2 ↓0	↑3 ↓0	↑4 ↓0	↑5 ↓0	↑4 ↓1	↑4 ↓0	↑2 ↓1	↑25 ↓2
F_6	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓0	↑4 ↓0	↑5 ↓0	↑37 ↓0
T_1	(7)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓2	↑5 ↓1	↑4 ↓2	↑37 ↓5
T_2	(7)	↑1 ↓0	↑7 ↓0	↑5 ↓0	↑7 ↓0	↑6 ↓0	↑4 ↓3	↑2 ↓4	↑2 ↓5	↑34 ↓12
T_3	(7)	↑0 ↓0	↑5 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑3 ↓1	↑3 ↓2	↑2 ↓1	↑33 ↓4
T_4	(7)	↑0 ↓0	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓1	↑3 ↓3	↑3 ↓3	↑37 ↓7
T_5	(7)	↑0 ↓1	↑7 ↓0	↑6 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓1	↑4 ↓1	↑2 ↓1	↑37 ↓4
T_6	(7)	↑0 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓1	↑4 ↓2	↑2 ↓2	↑34 ↓5
All	(54)	↑1 ↓2	↑42 ↓1	↑47 ↓0	↑52 ↓0	↑47 ↓0	↑40 ↓9	↑22 ↓17	↑15 ↓23	↑266 ↓52
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑0 ↓2	↑9 ↓6
5	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑0 ↓1	↑0 ↓2	↑8 ↓4
10	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑0 ↓1	↑0 ↓2	↑9 ↓3
20	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑1 ↓1	↑0 ↓1	↑0 ↓2	↑7 ↓5
40	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑0 ↓2	↑10 ↓3
80	(2)	↑0 ↓0	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑1 ↓1	↑2 ↓0	↑0 ↓2	↑10 ↓3
C	(6)	↑0 ↓0	↑1 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑3 ↓0	↑0 ↓6	↑28 ↓6
S	(6)	↑1 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑3 ↓3	↑3 ↓3	↑1 ↓5	↑0 ↓6	↑25 ↓18
		GDBG								
F_{1a}	(6)	↑0 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓0	↑2 ↓2	↑35 ↓2
F_{1b}	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑3 ↓0	↑1 ↓3	↑34 ↓3
F_2	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	↑2 ↓1	↑36 ↓2
F_3	(6)	↑0 ↓0	↑2 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑0 ↓2	↑0 ↓6	↑0 ↓6	↑19 ↓14
F_4	(6)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑2 ↓2	↑37 ↓3
F_5	(6)	↑1 ↓0	↑1 ↓0	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑3 ↓0	↑4 ↓0	↑21 ↓0
F_6	(6)	↑0 ↓1	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑2 ↓2	↑2 ↓2	↑4 ↓2	↑31 ↓7
T_1	(7)	↑0 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑5 ↓1	↑4 ↓1	↑38 ↓2
T_2	(7)	↑0 ↓0	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑5 ↓0	↑2 ↓3	↑2 ↓4	↑36 ↓7
T_3	(7)	↑0 ↓0	↑4 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑5 ↓2	↑4 ↓2	↑1 ↓3	↑35 ↓7
T_4	(7)	↑0 ↓1	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓0	↑4 ↓1	↑4 ↓3	↑41 ↓5
T_5	(7)	↑1 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓2	↑3 ↓2	↑3 ↓2	↑34 ↓6
T_6	(7)	↑0 ↓0	↑4 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	↑4 ↓0	↑3 ↓1	↑1 ↓3	↑29 ↓4
All	(54)	↑2 ↓1	↑38 ↓0	↑51 ↓0	↑50 ↓1	↑47 ↓3	↑38 ↓7	↑25 ↓15	↑15 ↓28	↑266 ↓55
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	↑1 ↓0	↑7 ↓2	↑6 ↓2	↑5 ↓1	↑6 ↓2	↑5 ↓2	↑3 ↓4	↑1 ↓6	↑34 ↓19
5	(10)	↑0 ↓0	↑5 ↓0	↑10 ↓0	↑7 ↓1	↑6 ↓1	↑6 ↓0	↑1 ↓4	↑1 ↓5	↑36 ↓11
10	(10)	↑0 ↓0	↑5 ↓0	↑8 ↓0	↑7 ↓0	↑7 ↓0	↑5 ↓1	↑0 ↓6	↑1 ↓7	↑33 ↓14
20	(10)	↑0 ↓1	↑4 ↓0	↑9 ↓0	↑10 ↓0	↑5 ↓2	↑3 ↓3	↑2 ↓4	↑1 ↓6	↑34 ↓16
40	(10)	↑0 ↓0	↑2 ↓0	↑9 ↓0	↑10 ↓0	↑9 ↓1	↑5 ↓2	↑2 ↓3	↑0 ↓5	↑37 ↓11
80	(10)	↑0 ↓1	↑2 ↓1	↑10 ↓0	↑10 ↓0	↑9 ↓0	↑4 ↓3	↑3 ↓3	↑1 ↓5	↑39 ↓13
C	(6)	↑0 ↓0	↑15 ↓0	↑29 ↓0	↑25 ↓1	↑20 ↓3	↑12 ↓7	↑6 ↓12	↑0 ↓17	↑107 ↓40
S	(6)	↑1 ↓2	↑10 ↓3	↑23 ↓2	↑24 ↓1	↑22 ↓3	↑16 ↓4	↑5 ↓12	↑5 ↓17	↑106 ↓44
		GDBG								
F_{1a}	(30)	↑0 ↓1	↑28 ↓0	↑30 ↓0	↑24 ↓4	↑18 ↓10	↑13 ↓14	↑6 ↓17	↑3 ↓22	↑122 ↓68
F_{1b}	(30)	↑1 ↓0	↑30 ↓0	↑30 ↓0	↑24 ↓6	↑18 ↓11	↑12 ↓15	↑5 ↓22	↑2 ↓24	↑122 ↓78
F_2	(30)	↑0 ↓3	↑28 ↓0	↑29 ↓0	↑20 ↓8	↑19 ↓9	↑15 ↓13	↑12 ↓16	↑5 ↓18	↑128 ↓67
F_3	(30)	↑0 ↓0	↑22 ↓0	↑26 ↓0	↑14 ↓3	↑5 ↓4	↑0 ↓14	↑2 ↓16	↑3 ↓15	↑72 ↓52
F_4	(30)	↑1 ↓2	↑25 ↓0	↑29 ↓0	↑26 ↓2	↑20 ↓7	↑14 ↓14	↑12 ↓14	↑5 ↓18	↑132 ↓57
F_5	(30)	↑2 ↓2	↑21 ↓0	↑24 ↓0	↑23 ↓2	↑20 ↓6	↑13 ↓10	↑9 ↓10	↑6 ↓13	↑118 ↓43
F_6	(30)	↑0 ↓1	↑29 ↓0	↑30 ↓0	↑26 ↓1	↑24 ↓1	↑16 ↓5	↑16 ↓7	↑20 ↓6	↑161 ↓21
T_1	(35)	↑0 ↓2	↑33 ↓0	↑32 ↓0	↑24 ↓7	↑20 ↓9	↑15 ↓15	↑14 ↓16	↑10 ↓20	↑148 ↓69
T_2	(35)	↑1 ↓1	↑32 ↓0	↑33 ↓0	↑28 ↓4	↑21 ↓8	↑12 ↓16	↑6 ↓22	↑6 ↓26	↑139 ↓77
T_3	(35)	↑0 ↓2	↑29 ↓0	↑35 ↓0	↑26 ↓2	↑21 ↓8	↑12 ↓14	↑10 ↓16	↑5 ↓16	↑138 ↓58
T_4	(35)	↑0 ↓1	↑32 ↓0	↑34 ↓0	↑27 ↓4	↑23 ↓8	↑14 ↓15	↑11 ↓19	↑9 ↓22	↑150 ↓69
T_5	(35)	↑2 ↓3	↑30 ↓0	↑32 ↓0	↑24 ↓6	↑18 ↓7	↑15 ↓13	↑10 ↓15	↑8 ↓15	↑139 ↓59
T_6	(35)	↑1 ↓0	↑27 ↓0	↑32 ↓0	↑28 ↓3	↑21 ↓8	↑15 ↓12	↑11 ↓14	↑6 ↓17	↑141 ↓54
All	(270)	↑5 ↓11	↑208 ↓3	↑250 ↓2	↑206 ↓28	↑166 ↓54	↑111 ↓96	↑73 ↓126	↑49 ↓150	↑1068 ↓470

of dimensions listed in Table 4.3. This is the same set of experiments used to compare CPE, RMC and CDE to DynDE. *jDE* was thus tested on 2 160 different dynamic environments. The offline errors resulting from the *jDE* experiments were analysed with respect to the offline errors produced by CDE. The number of times that each algorithm was statistically significantly better than the other (according to Mann-Whitney U tests), was counted. The analysis is summarised in Tables 4.19 and 4.20.

CDE performed significantly better than *jDE* in 1 427 cases and worse in 472 cases. The analysis over all dimensions (refer to Table 4.20) showed that *jDE* performed better than CDE more often when a change period of 100 function evaluations was used. At this change period CDE reduced to DynDE as the competitive population evaluation component only functions after the second generation. *jDE* also performed better than CDE on the GDBG function F_3. The scalability study presented in Section 4.6.4 found that CDE and DynDE performed similarly on function F_3, and always resulted in a large offline error. A possible explanation for the comparatively good performance of *jDE* on function F_3 is that F_3 has a large number of local optima (refer to Figure 2.6). The ageing component of *jDE*, which continuously reinitialises individuals, may be useful on this function to reduce the likelihood of convergence to local optima.

jDE performed better more often than CDE in low dimensions when a change period of 100 000 was used. This result is to be expected as previous sections found that CDE's performance deteriorated with respect to DynDE's when the change period was increased (refer to Section 4.6.5.3). *jDE* also performed better than CDE on the MPB functions when a change period of 500 was used as insufficient function evaluations are available to the competitive population evaluation to start locating optima early.

The *jDE* algorithm maintains a history archive of previously found good solutions. The purpose of this archive is to make the algorithms more effective on environments which are cyclic (refer to the definition in Section 2.5.2). *jDE* did not, however, perform better than CDE more often on the two recurrent change types of the GDBG (T_5 and T_6, described in Section 2.5.4.2). This result is unexpected, as CDE does not make use of a history archive.

The analysis in this section shows that, although *jDE* performs better than CDE in isolated cases, CDE generally performs better than *jDE*.

Table 4.19: CDE vs *jDE* performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓2	**↑2 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑14 ↓2**
5	(2)	↑0 ↓1	↑0 ↓1	**↑2 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑12 ↓2**
10	(2)	↑0 ↓1	↑0 ↓2	↑1 ↓1	**↑2 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑11 ↓4**
20	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓1	**↑2 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑10 ↓5**
40	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓2	**↑2 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑6 ↓10
80	(2)	↑0 ↓1	↑0 ↓2	↑0 ↓2	**↑2 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑6 ↓9
C	(6)	↑0 ↓3	↑1 ↓4	**↑3 ↓2**	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑34 ↓9**
S	(6)	↑0 ↓6	↑1 ↓5	↑2 ↓4	↑6 ↓0	↑4 ↓2	↑4 ↓2	↑4 ↓2	↑4 ↓2	**↑25 ↓23**
		GDBG								
F_{1a}	(6)	↑1 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑2 ↓1	↑2 ↓4	**↑35 ↓5**
F_{1b}	(6)	↑2 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	↑2 ↓3	↑1 ↓4	**↑33 ↓9**
F_2	(6)	↑1 ↓1	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	↑0 ↓6	↑0 ↓6	**↑28 ↓14**
F_3	(6)	↑1 ↓0	↑1 ↓4	↑4 ↓0	↑5 ↓0	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑11 ↓28
F_4	(6)	↑1 ↓1	↑2 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑1 ↓3	**↑34 ↓5**
F_5	(6)	↑2 ↓2	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑1 ↓3	**↑39 ↓5**
F_6	(6)	↑1 ↓0	↑4 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓0	↑1 ↓4	↑0 ↓6	**↑28 ↓10**
T_1	(7)	↑4 ↓0	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑4 ↓2	↑2 ↓3	↑0 ↓7	**↑37 ↓13**
T_2	(7)	↑1 ↓0	↑5 ↓1	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑6 ↓1	↑5 ↓2	↑1 ↓3	**↑38 ↓8**
T_3	(7)	↑0 ↓1	↑5 ↓1	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑6 ↓1	↑4 ↓3	↑2 ↓3	**↑37 ↓10**
T_4	(7)	↑4 ↓0	↑4 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓1	↑5 ↓1	↑2 ↓5	↑0 ↓7	**↑33 ↓15**
T_5	(7)	↑0 ↓3	↑3 ↓2	↑6 ↓0	↑7 ↓0	↑6 ↓1	↑4 ↓2	↑2 ↓3	↑2 ↓5	**↑30 ↓16**
T_6	(7)	↑0 ↓1	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑5 ↓1	↑2 ↓4	↑0 ↓7	**↑33 ↓14**
All	(54)	↑9 ↓14	**↑32 ↓14**	**↑45 ↓6**	**↑53 ↓0**	**↑46 ↓8**	**↑40 ↓10**	**↑27 ↓22**	↑15 ↓34	**↑267 ↓108**
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓2	↑0 ↓1	↑1 ↓1	**↑1 ↓0**	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑9 ↓4**
5	(2)	↑0 ↓2	↑0 ↓1	**↑1 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑11 ↓3**
10	(2)	↑0 ↓2	↑0 ↓2	↑1 ↓1	**↑2 ↓0**	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑11 ↓5**
20	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑10 ↓6**
40	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑1 ↓1	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑9 ↓7**
80	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑1 ↓1	**↑1 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑5 ↓10
C	(6)	↑0 ↓6	↑0 ↓4	↑3 ↓3	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑33 ↓13**
S	(6)	↑0 ↓6	↑0 ↓6	↑0 ↓5	**↑3 ↓2**	↑4 ↓0	↑5 ↓1	↑5 ↓1	↑5 ↓1	↑22 ↓22
		GDBG								
F_{1a}	(6)	↑2 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑2 ↓3	**↑40 ↓3**
F_{1b}	(6)	↑3 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓2	↑2 ↓3	**↑39 ↓5**
F_2	(6)	↑2 ↓1	↑3 ↓1	↑5 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	**↑39 ↓4**
F_3	(6)	↑2 ↓0	↑0 ↓5	↑0 ↓1	↑6 ↓0	↑4 ↓0	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑12 ↓24
F_4	(6)	↑2 ↓1	↑2 ↓2	↑5 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑39 ↓4**
F_5	(6)	↑3 ↓2	↑4 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑43 ↓3**
F_6	(6)	↑2 ↓0	↑2 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑3 ↓2	↑3 ↓3	**↑34 ↓5**
T_1	(7)	↑2 ↓0	↑6 ↓1	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑5 ↓2	↑3 ↓4	**↑42 ↓8**
T_2	(7)	↑2 ↓3	↑3 ↓1	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑6 ↓1	↑6 ↓1	**↑43 ↓7**
T_3	(7)	↑0 ↓1	↑2 ↓2	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑5 ↓2	↑5 ↓2	**↑38 ↓8**
T_4	(7)	↑5 ↓0	↑6 ↓1	↑6 ↓1	↑7 ↓0	↑6 ↓0	↑6 ↓1	↑5 ↓2	↑4 ↓3	**↑45 ↓8**
T_5	(7)	↑2 ↓0	↑2 ↓4	↑4 ↓2	↑7 ↓0	↑6 ↓0	↑6 ↓1	↑5 ↓2	↑3 ↓4	**↑35 ↓13**
T_6	(7)	↑5 ↓0	↑4 ↓0	↑6 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑5 ↓1	↑3 ↓2	**↑43 ↓4**
All	(54)	↑16 ↓16	**↑23 ↓19**	**↑37 ↓11**	**↑51 ↓2**	**↑50 ↓0**	**↑47 ↓7**	**↑42 ↓11**	**↑35 ↓17**	**↑301 ↓83**
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓2	↑0 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑6 ↓9
5	(2)	↑0 ↓1	↑0 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	↑6 ↓6
10	(2)	↑0 ↓2	↑0 ↓2	↑1 ↓1	**↑1 ↓0**	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑9 ↓5**
20	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓1	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑10 ↓5**
40	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓1	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑10 ↓5**
80	(2)	↑0 ↓2	↑0 ↓2	↑0 ↓1	↑1 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	↑2 ↓0	**↑9 ↓5**
C	(6)	↑0 ↓5	↑0 ↓4	↑3 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑33 ↓9**
S	(6)	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑2 ↓2	**↑3 ↓2**	↑4 ↓2	↑4 ↓1	↑4 ↓1	↑17 ↓26
		GDBG								
F_{1a}	(6)	↑3 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓0	**↑43 ↓0**
F_{1b}	(6)	↑4 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑4 ↓1	**↑44 ↓1**
F_2	(6)	↑3 ↓1	↑3 ↓1	↑4 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑40 ↓3**
F_3	(6)	↑2 ↓2	↑0 ↓5	↑0 ↓5	**↑3 ↓0**	↑4 ↓0	↑2 ↓2	↑0 ↓6	↑0 ↓6	↑11 ↓26
F_4	(6)	↑2 ↓0	↑3 ↓1	↑4 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	**↑39 ↓2**
F_5	(6)	↑2 ↓0	↑3 ↓1	↑3 ↓0	↑5 ↓0	↑5 ↓0	↑6 ↓0	↑5 ↓0	↑4 ↓1	**↑33 ↓2**
F_6	(6)	↑3 ↓2	↑2 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑3 ↓1	**↑36 ↓4**
T_1	(7)	↑0 ↓0	↑6 ↓1	↑6 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓1	↑6 ↓1	↑5 ↓1	**↑41 ↓5**
T_2	(7)	↑1 ↓3	↑5 ↓1	↑6 ↓1	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑6 ↓1	**↑45 ↓7**
T_3	(7)	↑0 ↓2	↑1 ↓1	↑2 ↓1	↑7 ↓0	↑7 ↓0	↑7 ↓0	↑6 ↓1	↑5 ↓1	**↑35 ↓6**
T_4	(7)	↑7 ↓0	↑6 ↓1	↑6 ↓1	↑6 ↓0	↑6 ↓0	↑6 ↓1	↑6 ↓1	↑4 ↓2	**↑47 ↓6**
T_5	(7)	↑5 ↓0	↑2 ↓4	↑3 ↓3	↑6 ↓0	↑7 ↓0	↑6 ↓0	↑5 ↓2	↑4 ↓2	**↑38 ↓11**
T_6	(7)	↑6 ↓0	↑2 ↓0	↑5 ↓0	↑6 ↓0	↑6 ↓0	↑6 ↓0	↑5 ↓1	↑4 ↓2	**↑40 ↓3**
All	(54)	**↑19 ↓16**	**↑22 ↓18**	**↑31 ↓13**	**↑46 ↓2**	**↑48 ↓2**	**↑48 ↓4**	**↑44 ↓8**	**↑38 ↓10**	**↑296 ↓73**

Table 4.20: CDE vs *jDE* performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑*0* ↓*2*	↑*0* ↓*1*	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑*6* ↓*9*
5	(2)	↑*0* ↓*2*	↑*0* ↓*1*	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑1 ↓1	↑*6* ↓*9*
10	(2)	↑*0* ↓*2*	↑*0* ↓*1*	↑1 ↓1	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑6 ↓4**
20	(2)	↑*0* ↓*2*	↑*0* ↓*2*	↑1 ↓1	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓5**
40	(2)	↑*0* ↓*2*	↑*0* ↓*2*	↑*0* ↓*1*	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑10 ↓5**
80	(2)	↑*0* ↓*1*	↑*0* ↓*1*	↑1 ↓1	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓3**
C	(6)	↑*0* ↓*5*	↑*0* ↓*2*	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑35 ↓7**
S	(6)	↑*0* ↓*6*	↑*0* ↓*6*	↑*0* ↓*6*	**↑3 ↓2**	**↑3 ↓2**	**↑3 ↓2**	**↑3 ↓2**	**↑3 ↓2**	↑*15* ↓*28*
		GDBG								
F_{1a}	(6)	**↑2 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑42 ↓0**
F_{1b}	(6)	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑47 ↓0**
F_2	(6)	↑*2* ↓*3*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑44 ↓3**
F_3	(6)	↑*2* ↓*3*	↑*0* ↓*6*	↑*0* ↓*5*	↑*1* ↓*3*	**↑3 ↓1**	**↑3 ↓2**	↑*0* ↓*5*	↑*0* ↓*6*	↑*9* ↓*31*
F_4	(6)	↑3 ↓3	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑44 ↓3**
F_5	(6)	↑1 ↓1	↑2 ↓2	↑*2* ↓*3*	↑*2* ↓*3*	**↑2 ↓1**	**↑4 ↓1**	**↑5 ↓1**	**↑3 ↓1**	**↑21 ↓13**
F_6	(6)	↑*2* ↓*3*	↑*1* ↓*4*	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑3 ↓1**	**↑34 ↓8**
T_1	(7)	↑*1* ↓*5*	**↑6 ↓1**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑42 ↓11**
T_2	(7)	↑*1* ↓*4*	**↑5 ↓2**	**↑6 ↓1**	**↑6 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑44 ↓9**
T_3	(7)	↑*1* ↓*4*	**↑3 ↓2**	**↑4 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑5 ↓1**	**↑38 ↓10**
T_4	(7)	**↑5 ↓0**	**↑3 ↓2**	**↑5 ↓2**	**↑5 ↓1**	**↑5 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑40 ↓8**
T_5	(7)	**↑3 ↓0**	**↑4 ↓3**	**↑5 ↓2**	**↑5 ↓2**	**↑5 ↓1**	**↑5 ↓0**	**↑6 ↓1**	**↑4 ↓2**	**↑37 ↓11**
T_6	(7)	**↑6 ↓0**	**↑4 ↓2**	**↑4 ↓1**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑4 ↓2**	**↑40 ↓9**
All	(54)	↑*17* ↓*24*	**↑25 ↓20**	**↑34 ↓14**	**↑42 ↓8**	**↑44 ↓4**	**↑46 ↓5**	**↑44 ↓8**	**↑39 ↓10**	**↑291 ↓93**
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	↑*0* ↓*2*	↑*0* ↓*1*	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑1 ↓0**	**↑6 ↓3**
5	(2)	↑*0* ↓*2*	↑*0* ↓*2*	↑1 ↓1	**↑1 ↓0**	↑1 ↓1	↑1 ↓1	**↑1 ↓0**	↑1 ↓1	↑*6* ↓*8*
10	(2)	↑*0* ↓*2*	↑*0* ↓*1*	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	↑1 ↓1	**↑10 ↓5**
20	(2)	↑*0* ↓*2*	↑*0* ↓*1*	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	**↑2 ↓0**	**↑11 ↓4**
40	(2)	↑*0* ↓*2*	↑*0* ↓*2*	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓4**
80	(2)	↑*0* ↓*2*	↑*0* ↓*2*	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑11 ↓4**
C	(6)	↑*0* ↓*6*	↑*0* ↓*3*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑36 ↓9**
S	(6)	↑*0* ↓*6*	↑*0* ↓*6*	**↑2 ↓1**	**↑4 ↓0**	**↑4 ↓1**	**↑4 ↓1**	↑2 ↓2	**↑3 ↓2**	↑19 ↓19
		GDBG								
F_{1a}	(6)	**↑3 ↓1**	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑43 ↓1**
F_{1b}	(6)	**↑3 ↓1**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑43 ↓1**
F_2	(6)	↑*2* ↓*3*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑44 ↓3**
F_3	(6)	↑*2* ↓*3*	↑*0* ↓*6*	↑*0* ↓*6*	↑*0* ↓*4*	**↑3 ↓1**	**↑5 ↓0**	↑*0* ↓*3*	↑*0* ↓*6*	↑*10* ↓*29*
F_4	(6)	↑*2* ↓*3*	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑44 ↓3**
F_5	(6)	↑*0* ↓*2*	↑*0* ↓*6*	↑*0* ↓*4*	↑*0* ↓*6*	↑*0* ↓*5*	↑*0* ↓*5*	↑*0* ↓*5*	↑*1* ↓*5*	↑*1* ↓*38*
F_6	(6)	↑*2* ↓*3*	↑*0* ↓*4*	**↑3 ↓2**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	↑3 ↓3	**↑32 ↓12**
T_1	(7)	↑*0* ↓*5*	**↑4 ↓2**	**↑5 ↓2**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓2**	**↑5 ↓2**	**↑36 ↓16**
T_2	(7)	↑*2* ↓*4*	**↑4 ↓3**	**↑5 ↓1**	**↑5 ↓2**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓1**	**↑39 ↓11**
T_3	(7)	↑*0* ↓*5*	↑*2* ↓*3*	**↑3 ↓2**	**↑5 ↓2**	**↑6 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑4 ↓3**	**↑31 ↓18**
T_4	(7)	**↑4 ↓2**	**↑4 ↓2**	**↑5 ↓2**	**↑5 ↓2**	**↑5 ↓1**	**↑6 ↓1**	**↑5 ↓2**	**↑5 ↓2**	**↑39 ↓14**
T_5	(7)	**↑2 ↓0**	**↑4 ↓3**	**↑4 ↓2**	**↑5 ↓2**	**↑5 ↓2**	**↑5 ↓1**	**↑5 ↓2**	**↑4 ↓3**	**↑34 ↓15**
T_6	(7)	**↑6 ↓0**	↑3 ↓3	**↑4 ↓3**	**↑5 ↓1**	**↑5 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑4 ↓3**	**↑38 ↓13**
All	(54)	↑*14* ↓*28*	↑*21* ↓*25*	**↑34 ↓13**	**↑40 ↓10**	**↑43 ↓7**	**↑45 ↓6**	**↑38 ↓10**	**↑37 ↓16**	**↑272 ↓115**
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	↑*0* ↓*10*	↑*2* ↓*4*	**↑6 ↓3**	**↑6 ↓2**	**↑6 ↓2**	**↑7 ↓2**	**↑7 ↓2**	**↑7 ↓2**	**↑41 ↓27**
5	(10)	↑*0* ↓*8*	↑*0* ↓*6*	**↑6 ↓3**	**↑7 ↓2**	**↑7 ↓3**	**↑7 ↓3**	**↑7 ↓1**	**↑7 ↓2**	**↑41 ↓28**
10	(10)	↑*0* ↓*9*	↑*0* ↓*8*	**↑6 ↓4**	**↑8 ↓0**	**↑8 ↓0**	**↑9 ↓0**	**↑8 ↓1**	**↑8 ↓1**	**↑47 ↓23**
20	(10)	↑*0* ↓*10*	↑*0* ↓*9*	↑*3* ↓*5*	**↑10 ↓0**	**↑10 ↓0**	**↑10 ↓0**	**↑9 ↓1**	**↑10 ↓0**	**↑52 ↓25**
40	(10)	↑*0* ↓*10*	↑*0* ↓*10*	↑*1* ↓*6*	**↑9 ↓1**	**↑9 ↓1**	**↑9 ↓1**	**↑9 ↓1**	**↑9 ↓1**	**↑46 ↓31**
80	(10)	↑*0* ↓*8*	↑*0* ↓*9*	↑*2* ↓*6*	**↑8 ↓1**	**↑8 ↓1**	**↑8 ↓2**	**↑8 ↓2**	**↑8 ↓2**	**↑42 ↓31**
C	(6)	↑*0* ↓*25*	↑*1* ↓*17*	**↑20 ↓5**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑171 ↓47**
S	(6)	↑*0* ↓*30*	↑*1* ↓*29*	↑*4* ↓*22*	**↑18 ↓6**	**↑18 ↓7**	**↑20 ↓8**	**↑18 ↓8**	**↑19 ↓8**	↑*98* ↓*118*
		GDBG								
F_{1a}	(30)	**↑11 ↓1**	**↑27 ↓0**	**↑28 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑26 ↓1**	**↑21 ↓7**	**↑203 ↓9**
F_{1b}	(30)	**↑17 ↓2**	**↑28 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑28 ↓1**	**↑24 ↓5**	**↑19 ↓8**	**↑206 ↓16**
F_2	(30)	**↑10 ↓9**	**↑23 ↓2**	**↑27 ↓2**	**↑30 ↓0**	**↑30 ↓0**	**↑28 ↓1**	**↑24 ↓6**	**↑23 ↓7**	**↑195 ↓27**
F_3	(30)	**↑9 ↓8**	↑*1* ↓*26*	↑*4* ↓*17*	**↑15 ↓7**	**↑14 ↓8**	↑*10* ↓*16*	↑*0* ↓*26*	↑*0* ↓*30*	↑*53* ↓*138*
F_4	(30)	**↑10 ↓8**	**↑18 ↓4**	**↑27 ↓2**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑30 ↓0**	**↑25 ↓3**	**↑200 ↓17**
F_5	(30)	**↑8 ↓7**	**↑15 ↓10**	**↑17 ↓7**	**↑19 ↓9**	**↑19 ↓6**	**↑22 ↓6**	**↑22 ↓6**	**↑15 ↓10**	**↑137 ↓61**
F_6	(30)	**↑10 ↓8**	**↑9 ↓8**	**↑24 ↓2**	**↑30 ↓0**	**↑30 ↓0**	**↑28 ↓0**	**↑21 ↓7**	↑*12* ↓*14*	**↑164 ↓39**
T_1	(35)	↑7 ↓*10*	**↑29 ↓5**	**↑29 ↓4**	**↑31 ↓2**	**↑31 ↓2**	**↑28 ↓6**	**↑24 ↓9**	**↑19 ↓15**	**↑198 ↓53**
T_2	(35)	↑7 ↓*14*	**↑22 ↓8**	**↑30 ↓3**	**↑32 ↓2**	**↑33 ↓1**	**↑32 ↓2**	**↑28 ↓5**	**↑25 ↓7**	**↑209 ↓42**
T_3	(35)	↑*1* ↓*13*	**↑13 ↓9**	**↑22 ↓4**	**↑32 ↓3**	**↑32 ↓2**	**↑32 ↓3**	**↑26 ↓8**	**↑21 ↓10**	**↑179 ↓52**
T_4	(35)	**↑25 ↓2**	**↑23 ↓7**	**↑28 ↓6**	**↑29 ↓3**	**↑28 ↓2**	**↑29 ↓5**	**↑24 ↓11**	**↑18 ↓15**	**↑204 ↓51**
T_5	(35)	**↑12 ↓3**	↑*15* ↓*16*	**↑22 ↓9**	**↑30 ↓4**	**↑29 ↓4**	**↑26 ↓4**	**↑23 ↓10**	**↑17 ↓16**	**↑174 ↓66**
T_6	(35)	**↑23 ↓1**	**↑19 ↓5**	**↑26 ↓4**	**↑30 ↓2**	**↑30 ↓3**	**↑29 ↓4**	**↑22 ↓8**	↑*15* ↓*16*	**↑194 ↓43**
All	(270)	↑*75* ↓*98*	**↑123 ↓96**	**↑181 ↓57**	**↑232 ↓22**	**↑231 ↓21**	**↑226 ↓32**	**↑195 ↓59**	**↑164 ↓87**	**↑1427 ↓472**

4.7.2 CDE Compared to Other Algorithms

This section compares the performance CDE to the results of other algorithms published in the literature. Algorithms are generally required to detect changes in the environment, which makes a direct comparison with the results presented in this chapter inappropriate, as CDE used an automatic detection strategy which allowed the algorithm to check for changes in the environment without using function evaluations. A fair comparison to the published results of other algorithms necessitates the incorporation of a detection strategy into CDE.

The algorithms used for comparison in this section all use one of two detection strategies. The first strategy is to re-evaluate the performance of the best individual that has been found since the last change in the environment. A change is detected if the fitness of the individual changed since it was last evaluated. The second detection strategy, commonly used, is to re-evaluate the fitness of the best individual in each of the sub-populations. A change is detected if the fitness of any the best individuals has changed since it was last evaluated. The first strategy has the advantage of using only one function evaluation per generation to detect changes, but the disadvantage that sampling only a single point in the fitness landscape could lead to the strategy's not detecting a change when it occurs. The second strategy has the advantage of sampling multiple points in the fitness landscape, but unfortunately uses more function evaluations to detect the changes than the first strategy does.

Any two algorithms should ideally use the same change detection strategy when being compared. However, using the same detection strategy does not guarantee that the algorithms will use the same number of function evaluations to detect changes, or that changes will be detected equally effectively. The functioning of each algorithm influences the performance of a change detection strategy. For example, the number of sub-populations and the sub-population size used by an algorithm determines the number of function evaluations between generations, and consequently how frequently the change detection strategy is performed. Algorithms that employ a large sub-population size or use a large number of sub-populations, perform change detections less frequently than algorithms that uses a small number of sub-populations or a small sub-population size. The problem

is further complicated by the fact that not all algorithms maintain a constant number of sub-populations during the optimisation process. Furthermore, CDE only evolves one sub-population per generation, which leads to fewer function evaluations between executions of the change detection strategy.

This section presents CDE results, using several change detection strategies on variations of the MPB Scenario 2 which are commonly used by researchers to evaluate their algorithms. The results of each of the algorithms used for comparison can then be compared to CDE's results using the closest matching detection strategy. Four detection strategies were selected for incorporation into CDE. The first, Det_{best}, re-evaluates the best individual over all sub-populations, after each generation. Det_{best} thus uses a single function evaluation per generation. The second change detection strategy, Det_{local}, re-evaluates the best individual in each sub-population after each generation. Det_{local} thus uses n_k function evaluations per generation, where n_k is the number of sub-populations. The Det_{best} and Det_{local} change detection strategies function differently on CDE than on standard multi-population algorithms (like DynDE) where all sub-populations are evolved per generation. The third and fourth change detection strategies are variations of Det_{best} and Det_{local} where the detection is not performed in each generation, but once every n_k generations. These strategies are denoted by Det_{n_k-best} and $Det_{n_k-local}$. Detecting changes once every n_k generations ensures that the number of function evaluations used by Det_{n_k-best} and $Det_{n_k-local}$ in CDE would, respectively, be similar to the number of function evaluations used by Det_{best} and Det_{local} in standard multi-population algorithms.

The offline errors and 95% confidence intervals of CDE, using the four change detection strategies on variations of the MPB Scenario 2, are given in Table 4.21. Outcomes of Mann-Whitney U tests comparing the results of CDE using each of the detection strategies and CDE using the automatic detection strategy are included in Table 4.21. Results that are statistically significantly worse than when the automatic detection strategy is used are printed in italics. The results show that Det_{best} did not statistically significantly affect the performance of CDE. The Det_{local} detection strategy resulted in inferior performance by CDE in almost all cases. This bad performance when using Det_{local} is due to too many function evaluations being wasted to detect changes. The CDE algorithm uses 10 sub-populations. During the period when only one sub-population is evolved per generation,

the algorithm, on average, uses more than half of the function evaluations per generation to detect changes using Det_{local}. The benefits of detecting changes correctly are thus outweighed by the cost in function evaluations. The performance of CDE deteriorated significantly in only two cases when using Det_{n_k-best} and four cases when using $Det_{n_k-local}$. Changes can thus be effectively detected without a large deterioration in performance on the MBP, if an appropriate detection strategy is used.

Table 4.21: CDE using various detection strategies

Settings	Automatic	Det_{best}	p-val	Det_{local}	p-val	Det_{n_k-best}	p-val	$Det_{n_k-local}$	p-val
C_s 1	0.79 ± 0.13	0.79 ± 0.15	0.686	0.9 ± 0.09	0.061	0.71 ± 0.12	0.415	0.71 ± 0.1	0.467
C_s 2	1.1 ± 0.08	1.21 ± 0.13	0.39	*1.62 ± 0.16*	0.000	*1.29 ± 0.12*	0.026	1.11 ± 0.1	0.947
C_s 4	2.17 ± 0.12	2.17 ± 0.13	0.901	*2.96 ± 0.22*	0.000	2.19 ± 0.15	0.901	2.41 ± 0.19	0.075
C_s 6	3.22 ± 0.19	3.46 ± 0.25	0.138	*4.61 ± 0.26*	0.000	*3.45 ± 0.15*	0.034	*3.72 ± 0.21*	0.001
Cp 500	4.26 ± 0.12	4.19 ± 0.12	0.293	*5.47 ± 0.28*	0.000	4.36 ± 0.17	0.44	*4.52 ± 0.18*	0.046
Cp 1000	2.43 ± 0.11	2.36 ± 0.09	0.504	*3.05 ± 0.12*	0.000	2.55 ± 0.11	0.146	*2.60 ± 0.12*	0.036
Cp 2500	1.16 ± 0.1	1.16 ± 0.1	0.82	*1.49 ± 0.11*	0.000	1.13 ± 0.12	0.307	1.13 ± 0.09	0.592
Cp 10000	0.52 ± 0.15	0.45 ± 0.11	0.467	0.58 ± 0.14	0.127	0.49 ± 0.11	0.994	0.43 ± 0.08	0.592
n_d 10	1.9 ± 0.24	2.03 ± 0.24	0.382	*2.66 ± 0.29*	0.000	2.02 ± 0.3	0.476	2.09 ± 0.32	0.532
n_d 50	12.4 ± 1.13	12.77 ± 0.8	0.134	*18.80 ± 1.24*	0.000	11.43 ± 0.67	0.523	12.74 ± 0.89	0.219
n_d 100	25.81 ± 1.71	26.27 ± 1.91	0.901	*40.09 ± 3.95*	0.000	24.42 ± 1.57	0.314	26.51 ± 2.2	0.843

Table 4.22 lists the published results for variations of the MBP Scenario 2 for 12 of the most seminal and modern algorithms aimed at the MPB. An overview of these algorithms was given in Section 3.3. The 95% confidence intervals were calculated from the reported standard errors or standard deviations in cases where the confidence interval was not reported. Results that are better than the corresponding CDE results are printed in italics while results that are worse than CDE's are printed in boldface in shaded cells. Results with overlapping confidence intervals are considered to be similar and are consequently printed in the normal font.

Table 4.22: Reported offline errors of various algorithms

	MMEO	HJEO	LSEO	CESO	ESCA	MPSO
C_s 1	0.66 ± 0.39	*0.25 ± 0.20*	*0.25 ± 0.16*	**1.38 ± 0.04**	**1.53 ± 0.02**	**1.75 ± 0.12**
C_s 2	0.86 ± 0.41	*0.52 ± 0.27*	*0.47 ± 0.24*	**1.78 ± 0.04**	**1.57 ± 0.02**	**2.4 ± 0.12**
C_s 4	*0.97 ± 0.41*	*0.64 ± 0.31*	*0.53 ± 0.25*	2.23 ± 0.10	*1.72 ± 0.06*	**3.59 ± 0.20**
C_s 6	*1.09 ± 0.43*	*0.9 ± 0.33*	*0.77 ± 0.47*	*2.74 ± 0.20*	*1.79 ± 0.06*	**4.79 ± 0.20**
n_d 10	2.44 ± 1.51	2.17 ± 1.57	2.25 ± 1.67	**2.51 ± 0.08**	N/A	**4.17 ± 0.29**
n_d 50	**206.3 ± 69.97**	*5.79 ± 2.74*	*6.22 ± 3.14*	*6.81 ± 0.14*	N/A	N/A
n_d 100	**480.5 ± 137.39**	16.5 ± 10.86	17.8 ± 13.52	24.6 ± 0.49	N/A	N/A
	CPSO	**MPSOD**	**HMSO**	**MSO**	**Cellular DE**	**Cellular PSO**
C_s 1	**1.06 ± 0.07**	**1.06 ± 0.03**	**1.42 ± 0.04**	**1.51 ± 0.04**	**1.64 ± 0.03**	**1.14 ± 0.13**
C_s 2	1.17 ± 0.06	**1.51 ± 0.04**	N/A	N/A	N/A	N/A
C_s 4	*1.38 ± 0.08*	N/A	N/A	N/A	N/A	N/A
C_s 6	*1.53 ± 0.08*	N/A	N/A	N/A	N/A	N/A
Cp 500	N/A	N/A	**7.56 ± 0.27**	**5.95 ± 0.09**	N/A	*1.44 ± 0.13*
Cp 1000	N/A	**3.58 ± 0.05**	**4.61 ± 0.07**	**3.94 ± 0.08**	**3.98 ± 0.03**	*1.33 ± 0.11*
Cp 2500	N/A	N/A	**2.39 ± 0.16**	N/A	**2.42 ± 0.02**	1.08 ± 0.09
Cp 10000	0.63 ± N/A	N/A	**0.94 ± 0.09**	**0.97 ± 0.04**	N/A	**1.1 ± 0.18**

The comparison of CDE with each of the tabulated algorithms are briefly discussed here.

MMEO [Moser and Hendtlass, 2007] MMEO is an algorithm based on extremal optimisation and searches for optima in parallel. The algorithm detects changes by re-evaluating all the best solutions in the search space and is thus comparable to the Det_{local} and $Det_{n_k-local}$ detection strategies. MMEO yielded a lower offline error than CDE on experiments with change severities 1 and 2, but confidence intervals of the two algorithms overlap and MMEO, accordingly, cannot conclusively be considered superior to CDE. Change severities of 4 and 6 yielded MMEO results that are clearly superior to CDE's results. Overlapping confidence intervals were found in 10 dimensions, but CDE clearly performed better than MMEO in 50 and 100 dimensions.

HJEO [Moser, 2007] HJEO is an extension of MMEO which incorporates a Hooke-Jeeves local search. HJEO performed better than CDE on all reported cases, except 10 and 100 dimensions where the confidence intervals overlapped.

LSEO [Moser and Chiong, 2010] The local search component of MMEO was further improved in the LSEO algorithm. LSEO performed better than CDE on all reported cases, except in 10 and 100 dimensions, where the confidence intervals overlapped.

CESO [Lung and Dumitrescu, 2007] CESO uses a single DE population and a single PSO population to solve DOPs. Changes are detected by re-evaluating the best individual in the DE population and is thus comparable to the Det_{best} detection strategy. CDE performed better than CESO on experiments with a change severity of 1 and 2. Overlapping confidence intervals were found when using a change severity of 4, but CESO performed better than CDE when using a change severity of 6. CDE performed better than CESO in 10 dimensions but worse in 50 dimensions. Overlapping confidence intervals were found in 100 dimensions.

ESCA [Lung and Dumitrescu, 2010] ESCA is an adaptation of CESO which employs two DE populations in combination with the PSO population. CDE performed

better than ESCA on change severities of 1 and 2, but worse on change severities of 4 and 6.

MPSO [Blackwell and Branke, 2006] MPSO is a multi-swarm algorithm based on PSO. MPSO uses the Det_{local} detection strategy, and is consequently comparable to CDE using the $Det_{n_k-local}$ detection strategy. CDE performed better than MPSO in all reported cases. Overlapping confidence intervals were found on the change severity of 6 experiment when using Det_{local} detection strategy.

CPSO [Yang and Li, 2010] CPSO clusters particles of an PSO algorithm into sub-swarms to track optima in a dynamic environment. CPSO detects changes using the Det_{best} detection strategy and is thus roughly comparable to CDE using the Det_{n_k-best} strategy. CPE performed better than CPSO in experiments with a change severity of 1, but overlapping confidence intervals were found with a change severity of 2. CPSO performed better than CPE with change severities of 4 and 6. CPE performed better than CPSO when a change period of 10 000 was used, but as Yang and Li [2010] did not report the confidence interval for this experiment, CPE's superiority cannot be confirmed for this experiment.

MPSOD [Novoa-Hernández *et al.*, 2011] MPSOD is an extension of MPSO. The authors do not explicitly state which change detection strategy is used, but it is assumed that MPSOD employs Det_{local} which is used in MPSO. MPSOD is consequently compared to CDE using the $Det_{n_k-local}$ detection strategy. CDE performed better than MPSOD in all reported cases.

HMSO [Kamosi *et al.*, 2010a] HMSO is an extension of MPSO which hibernates unproductive sub-swarms. HMSO uses the Det_{best} change detection strategy, and, as not all sub-populations are evolved per generation, it can be compared to CDE using Det_{best}. CDE performed better than HMSO in all reported cases.

MSO [Kamosi *et al.*, 2010b] MSO is a PSO-based algorithm that utilises a dynamic number of swarms to locate optima in a dynamic environment. Kamosi *et al.* [2010b] do not specify the change detection strategy that is used in MSO, but it is assumed

that the Det_{best} strategy that was used in HMSO (which was developed by the same authors) is also used in MSO. CDE performed better than MSO in all reported cases.

Cellular DE [Noroozi *et al.*, 2011] Cellular DE creates sub-populations by dividing the search space into equally sized cells. Cellular DE employs the Det_{local} change detecting strategy and is thus comparable to CDE using the $Det_{n_k-local}$ strategy. CDE performed better than Cellular DE in all reported cases.

Cellular PSO [Hashemi and Meybodi, 2009a] Cellular PSO creates sub-swarms by dividing the search space into equally sized cells. Cellular PSO uses the Det_{local} change detecting strategy and is thus comparable to CDE using the $Det_{n_k-local}$ strategy. Cellular PSO performed better than CDE when using change periods of 500 and 1 000, but overlapping confidence intervals were found when using a change period of 2 500. CDE performed better than Cellular PSO in when using a change period of 5 000 and 10 000.

This section compared CDE results with the reported results of other algorithms. Moser and Chiong [2010] currently report the lowest offline errors in the literature on the MPB with the LSEO algorithm. The offline errors of LSEO and HJEO are considerably lower than the reported results of any other algorithm, and CDE is no exception. CDE does, however, outperform each of the other algorithms to which it was compared on at least one of the tested environments. The results presented in this section indicate that CDE compares favourably with the state-of-the-art approaches in the literature.

4.8 General Applicability

DynDE [Mendes and Mohais, 2005] was used as base algorithm for the CPE and RMC adaptations in this chapter. However, while DynDE is well suited for adaptation, the novel adaptations presented here can, in theory, be incorporated into any multi-population optimisation algorithm. This section investigates the general applicability of the competitive population evaluation and reinitialisation midpoint check approaches by incorporating these components into *jDE* of Brest *et al.* [2009].

jDE was adapted to incorporate the competitive population evaluation component of CDE to allocate function evaluations to sub-populations based on performance. The exclusion component of *jDE* was replaced with the reinitialisation midpoint check component and exclusion threshold of CDE. The self-adaptive control parameters, aging, local exclusion and custom reinitialisation components of *jDE* were left intact. The combination of *jDE* and CDE is referred to as CjDE.

CjDE was evaluated on the standard set for all combinations of dimension and change period listed in Table 4.3. The results were compared to those of *jDE* and the number of times that each algorithm performed statistically significantly better than the other were counted. The results of this analysis are given in Tables 4.23 and 4.24. The new components improved the *jDE* algorithm in 1 382 cases and deteriorated *jDE*'s performance in only 346 cases. The benefit of the new components are clear at lower change periods, but *jDE* did outperform CjDE at change periods of over 25 000 in low dimensions. This confirms the trends observed earlier in this chapter that the competitive population evaluation approach is only useful in problems with a low change period, and is more useful at high dimensions.

The results of the CDE algorithm were compared to CjDE on the same experimental set as described above. An analysis of the results showed that CDE performed better than CjDE in 1 273 experiments and worse in 631 experiments (full results of this analysis are given in Appendix B). CDE outperformed CjDE in the majority of experiments, but considering that *jDE* outperformed CDE in only 472 experiments and was inferior to CDE in 1 427 cases, it can be concluded that CjDE performed better with respect to CDE than *jDE*.

CPE and RMC outperformed *jDE* in the majority of environments that were investigated. These results prove that the approaches suggested in this chapter can be successfully incorporated into other multi-population algorithms aimed at DOPs.

4.9 Conclusions

This chapter evaluated DynDE on a wide range of environments. An investigation into the appropriate sub-population size and number of Brownian individuals confirmed the

Table 4.23: CjDE vs *jDE* performance analysis - Part 1

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**5 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓1	↑**2** ↓**0**	↑**1** ↓**0**	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑3 ↓11
5	(2)	↑0 ↓0	↑**1** ↓**0**	↑**1** ↓**0**	↑0 ↓1	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑2 ↓8
10	(2)	↑**1** ↓**0**	↑**1** ↓**0**	↑0 ↓1	↑1 ↓1	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑3 ↓9
20	(2)	↑0 ↓0	↑**1** ↓**0**	↑1 ↓1	↑**2** ↓**0**	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑4 ↓8
40	(2)	↑**1** ↓**0**	↑0 ↓1	↑0 ↓1	↑**2** ↓**0**	↑**1** ↓**0**	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑4 ↓8
80	(2)	↑0 ↓0	↑0 ↓1	↑0 ↓1	↑**2** ↓**0**	↑**2** ↓**0**	↑0 ↓0	↑0 ↓2	↑0 ↓2	↑4 ↓6
C	(6)	↑**1** ↓**0**	↑**4** ↓**0**	↑**3** ↓**0**	↑**4** ↓**1**	↑1 ↓2	↑0 ↓5	↑0 ↓6	↑0 ↓6	↑13 ↓20
S	(6)	↑1 ↓1	↑1 ↓2	↑0 ↓4	↑3 ↓3	↑2 ↓3	↑0 ↓5	↑0 ↓6	↑0 ↓6	↑7 ↓30
		GDBG								
F_{1a}	(6)	↑0 ↓1	↑**4** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓3	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑17 ↓22
F_{1b}	(6)	↑**2** ↓**0**	↑**5** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑1 ↓3	↑0 ↓6	↑0 ↓6	↑0 ↓6	↑19 ↓21
F_2	(6)	↑0 ↓1	↑**5** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑2 ↓3	↑0 ↓6	↑0 ↓6	↑**23** ↓**16**
F_3	(6)	↑**3** ↓**0**	↑**1** ↓**0**	↑**2** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**2** ↓**1**	↑0 ↓5	↑0 ↓6	↑**20** ↓**12**
F_4	(6)	↑0 ↓0	↑**4** ↓**1**	↑**3** ↓**1**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓1	↑0 ↓6	↑0 ↓6	↑**20** ↓**15**
F_5	(6)	↑**2** ↓**1**	↑**3** ↓**2**	↑**3** ↓**1**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**1**	↑**3** ↓**2**	↑0 ↓6	↑**28** ↓**13**
F_6	(6)	↑**2** ↓**1**	↑**3** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓1	↑0 ↓6	↑0 ↓6	↑**23** ↓**14**
T_1	(7)	↑**1** ↓**0**	↑0 ↓2	↑**2** ↓**1**	↑**7** ↓**0**	↑**4** ↓**2**	↑0 ↓6	↑0 ↓7	↑0 ↓7	↑14 ↓25
T_2	(7)	↑**2** ↓**0**	↑**5** ↓**1**	↑**5** ↓**1**	↑**7** ↓**0**	↑**5** ↓**0**	↑2 ↓3	↑1 ↓6	↑0 ↓7	↑**27** ↓**18**
T_3	(7)	↑**2** ↓**0**	↑**3** ↓**0**	↑**4** ↓**0**	↑**7** ↓**0**	↑**5** ↓**0**	↑2 ↓2	↑1 ↓6	↑0 ↓7	↑**24** ↓**15**
T_4	(7)	↑1 ↓1	↑**7** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**5** ↓**2**	↑3 ↓3	↑0 ↓7	↑0 ↓7	↑**30** ↓**20**
T_5	(7)	↑0 ↓3	↑**5** ↓**0**	↑**6** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**3** ↓**2**	↑1 ↓5	↑0 ↓7	↑**29** ↓**17**
T_6	(7)	↑**3** ↓**0**	↑**5** ↓**0**	↑**5** ↓**0**	↑**7** ↓**0**	↑**5** ↓**2**	↑1 ↓3	↑0 ↓6	↑0 ↓7	↑**26** ↓**18**
All	(54)	↑**11** ↓**5**	↑**30** ↓**5**	↑**32** ↓**6**	↑**49** ↓**4**	↑**34** ↓**11**	↑11 ↓29	↑3 ↓49	↑0 ↓54	↑**170** ↓**163**
Set.	**Max**	**10 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑**2** ↓**0**	↑**1** ↓**0**	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑3 ↓10
5	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑1 ↓1	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑5 ↓8
10	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑5 ↓7
20	(2)	↑**1** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑**2** ↓**0**	↑0 ↓2	↑0 ↓2	↑0 ↓2	↑**8** ↓**6**
40	(2)	↑0 ↓1	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑**2** ↓**0**	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑**7** ↓**6**
80	(2)	↑0 ↓0	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑0 ↓0	↑0 ↓2	↑0 ↓2	↑**6** ↓**4**
C	(6)	↑1 ↓1	↑**5** ↓**0**	↑**5** ↓**0**	↑**5** ↓**1**	↑**3** ↓**1**	↑0 ↓4	↑0 ↓6	↑0 ↓6	↑19 ↓19
S	(6)	↑0 ↓0	↑**5** ↓**0**	↑**6** ↓**0**	↑1 ↓2	↑3 ↓3	↑0 ↓5	↑0 ↓6	↑0 ↓6	↑15 ↓22
		GDBG								
F_{1a}	(6)	↑**1** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓4	↑0 ↓4	↑0 ↓6	↑**25** ↓**14**
F_{1b}	(6)	↑**1** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓4	↑0 ↓5	↑0 ↓6	↑**26** ↓**15**
F_2	(6)	↑1 ↓1	↑**3** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑1 ↓1	↑0 ↓6	↑**28** ↓**8**
F_3	(6)	↑0 ↓0	↑**4** ↓**0**	↑**2** ↓**1**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**3** ↓**0**	↑**33** ↓**1**
F_4	(6)	↑**2** ↓**1**	↑**3** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**1** ↓**0**	↑0 ↓4	↑0 ↓6	↑**24** ↓**11**
F_5	(6)	↑**1** ↓**0**	↑**6** ↓**0**	↑**4** ↓**2**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**4** ↓**1**	↑**38** ↓**3**
F_6	(6)	↑**1** ↓**0**	↑**3** ↓**1**	↑**3** ↓**1**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑1 ↓3	↑**31** ↓**5**
T_1	(7)	↑0 ↓0	↑**7** ↓**0**	↑**5** ↓**1**	↑**7** ↓**0**	↑**7** ↓**0**	↑**4** ↓**2**	↑2 ↓4	↑1 ↓5	↑**33** ↓**12**
T_2	(7)	↑2 ↓2	↑**4** ↓**0**	↑**4** ↓**2**	↑**7** ↓**0**	↑**7** ↓**0**	↑**4** ↓**2**	↑2 ↓2	↑1 ↓5	↑**31** ↓**13**
T_3	(7)	↑0 ↓0	↑**3** ↓**1**	↑**4** ↓**1**	↑**7** ↓**0**	↑**7** ↓**0**	↑**4** ↓**0**	↑**3** ↓**2**	↑1 ↓5	↑**29** ↓**9**
T_4	(7)	↑**2** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**5** ↓**2**	↑**3** ↓**2**	↑1 ↓4	↑**39** ↓**8**
T_5	(7)	↑0 ↓0	↑**4** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**3** ↓**1**	↑2 ↓5	↑**36** ↓**6**
T_6	(7)	↑**3** ↓**0**	↑**5** ↓**0**	↑**5** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**4** ↓**2**	↑**4** ↓**3**	↑2 ↓4	↑**37** ↓**9**
All	(54)	↑**8** ↓**3**	↑**40** ↓**1**	↑**43** ↓**4**	↑**48** ↓**3**	↑**48** ↓**4**	↑**27** ↓**17**	↑17 ↓26	↑8 ↓40	↑**239** ↓**98**
Set.	**Max**	**25 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑1 ↓1	↑0 ↓1	↑0 ↓2	↑0 ↓2	↑6 ↓6
5	(2)	↑1 ↓1	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑1 ↓1	↑0 ↓2	↑0 ↓2	↑**10** ↓**6**
10	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑0 ↓1	↑0 ↓2	↑**10** ↓**3**
20	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑0 ↓2	↑**11** ↓**2**
40	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑1 ↓1	↑**13** ↓**1**
80	(2)	↑0 ↓0	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**2** ↓**0**	↑**1** ↓**0**	↑**13** ↓**0**
C	(6)	↑**1** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**5** ↓**1**	↑**4** ↓**2**	↑2 ↓3	↑0 ↓5	↑**29** ↓**11**
S	(6)	↑0 ↓1	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**3** ↓**2**	↑2 ↓4	↑**34** ↓**7**
		GDBG								
F_{1a}	(6)	↑0 ↓1	↑**5** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**4** ↓**2**	↑1 ↓4	↑**32** ↓**7**
F_{1b}	(6)	↑0 ↓0	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**3** ↓**2**	↑1 ↓4	↑**34** ↓**6**
F_2	(6)	↑0 ↓0	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**5** ↓**0**	↑**39** ↓**0**
F_3	(6)	↑0 ↓0	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**42** ↓**0**
F_4	(6)	↑0 ↓0	↑**5** ↓**0**	↑**5** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**39** ↓**0**
F_5	(6)	↑0 ↓0	↑**6** ↓**0**	↑**5** ↓**0**	↑0 ↓4	↑**2** ↓**1**	↑**6** ↓**0**	↑**6** ↓**0**	↑**5** ↓**0**	↑**30** ↓**5**
F_6	(6)	↑0 ↓0	↑**5** ↓**0**	↑**4** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**39** ↓**0**
T_1	(7)	↑0 ↓0	↑**7** ↓**0**	↑**6** ↓**0**	↑**6** ↓**1**	↑**7** ↓**0**	↑**7** ↓**0**	↑**5** ↓**2**	↑**5** ↓**2**	↑**43** ↓**5**
T_2	(7)	↑0 ↓1	↑**7** ↓**0**	↑**6** ↓**0**	↑**6** ↓**1**	↑**6** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**3** ↓**2**	↑**41** ↓**4**
T_3	(7)	↑0 ↓0	↑**4** ↓**0**	↑**4** ↓**0**	↑**6** ↓**1**	↑**6** ↓**1**	↑**7** ↓**0**	↑**7** ↓**0**	↑**5** ↓**0**	↑**39** ↓**2**
T_4	(7)	↑0 ↓0	↑**7** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**5** ↓**2**	↑**5** ↓**2**	↑**43** ↓**4**
T_5	(7)	↑0 ↓0	↑**7** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**6** ↓**0**	↑**7** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**46** ↓**0**
T_6	(7)	↑0 ↓0	↑**6** ↓**0**	↑**7** ↓**0**	↑**6** ↓**1**	↑**6** ↓**0**	↑**7** ↓**0**	↑**6** ↓**0**	↑**5** ↓**2**	↑**43** ↓**3**
All	(54)	↑1 ↓2	↑**50** ↓**0**	↑**49** ↓**0**	↑**47** ↓**4**	↑**49** ↓**2**	↑**50** ↓**2**	↑**41** ↓**9**	↑**31** ↓**17**	↑**318** ↓**36**

Table 4.24: CjDE vs *jDE* performance analysis - Part 2

Cp		100	500	1000	5000	10000	25000	50000	100000	Total
Set.	**Max**	**50 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑1 ↓1	↑1 ↓1	↑1 ↓1	**↑10 ↓3**
5	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑12 ↓1**
10	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	↑1 ↓1	**↑13 ↓1**
20	(2)	↑0 ↓0	**↑1 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑11 ↓0**
40	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑13 ↓0**
80	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓0**
C	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑4 ↓1**	*↑2 ↓3*	**↑34 ↓5**
S	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑39 ↓0**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑4 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓1**	**↑37 ↓1**
F_{1b}	(6)	↑0 ↓0	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑37 ↓1**
F_2	(6)	↑0 ↓0	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑41 ↓0**
F_3	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓0**
F_4	(6)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑41 ↓0**
F_5	(6)	**↑1 ↓0**	**↑4 ↓0**	**↑5 ↓0**	*↑0 ↓4*	*↑0 ↓6*	*↑1 ↓3*	↑2 ↓2	**↑4 ↓1**	**↑17 ↓16**
F_6	(6)	*↑0 ↓1*	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓1**
T_1	(7)	*↑0 ↓1*	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑39 ↓3**
T_2	(7)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑7 ↓0**	**↑7 ↓0**	**↑7 ↓0**	**↑45 ↓2**
T_3	(7)	↑0 ↓0	**↑5 ↓0**	**↑5 ↓0**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑7 ↓0**	**↑40 ↓3**
T_4	(7)	**↑1 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓0**	**↑6 ↓1**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓2**	**↑43 ↓3**
T_5	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑44 ↓5**
T_6	(7)	↑0 ↓0	**↑6 ↓0**	**↑7 ↓0**	**↑5 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑42 ↓3**
All	(54)	↑1 ↓1	**↑47 ↓0**	**↑49 ↓0**	**↑43 ↓4**	**↑47 ↓6**	**↑48 ↓4**	**↑47 ↓3**	**↑44 ↓6**	**↑326 ↓24**
Set.	**Max**	**100 Dimensions**								
		MPB								
C_s 1	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	↑1 ↓1	**↑12 ↓1**
5	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑13 ↓0**
10	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓0**
20	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑14 ↓0**
40	(2)	↑0 ↓0	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑13 ↓0**
80	(2)	**↑1 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑2 ↓0**	**↑15 ↓0**
C	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑4 ↓1**	**↑39 ↓1**
S	(6)	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑42 ↓0**
		GDBG								
F_{1a}	(6)	↑0 ↓0	**↑5 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑39 ↓0**
F_{1b}	(6)	↑0 ↓0	**↑3 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑5 ↓0**	**↑35 ↓0**
F_2	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑42 ↓0**
F_3	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑41 ↓0**
F_4	(6)	**↑1 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑43 ↓0**
F_5	(6)	↑0 ↓0	**↑4 ↓0**	**↑3 ↓0**	*↑1 ↓4*	*↑0 ↓6*	*↑0 ↓5*	*↑0 ↓4*	*↑0 ↓5*	*↑8 ↓24*
F_6	(6)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑40 ↓0**
T_1	(7)	↑0 ↓0	**↑6 ↓0**	**↑6 ↓0**	**↑5 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑41 ↓4**
T_2	(7)	↑0 ↓0	**↑5 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑6 ↓0**	**↑6 ↓0**	**↑42 ↓2**
T_3	(7)	**↑1 ↓0**	**↑6 ↓0**	**↑4 ↓0**	**↑5 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓0**	**↑6 ↓1**	**↑40 ↓4**
T_4	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑5 ↓1**	**↑5 ↓1**	**↑6 ↓1**	**↑5 ↓1**	**↑4 ↓1**	**↑39 ↓5**
T_5	(7)	↑0 ↓0	**↑7 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑44 ↓5**
T_6	(7)	↑0 ↓0	**↑5 ↓0**	**↑6 ↓0**	**↑7 ↓0**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑6 ↓1**	**↑42 ↓4**
All	(54)	**↑2 ↓0**	**↑48 ↓0**	**↑49 ↓0**	**↑46 ↓4**	**↑46 ↓6**	**↑48 ↓5**	**↑46 ↓4**	**↑44 ↓6**	**↑329 ↓25**
Set.	**Max**	**All Dimensions**								
		MPB								
C_s 1	(10)	*↑0 ↓1*	**↑10 ↓0**	**↑8 ↓0**	**↑5 ↓4**	*↑4 ↓5*	*↑3 ↓6*	*↑2 ↓7*	*↑2 ↓8*	**↑34 ↓31**
5	(10)	↑1 ↓1	**↑9 ↓0**	**↑9 ↓0**	**↑7 ↓2**	**↑6 ↓2**	↑5 ↓5	*↑3 ↓6*	*↑2 ↓7*	**↑42 ↓23**
10	(10)	**↑1 ↓0**	**↑9 ↓0**	**↑8 ↓1**	**↑8 ↓1**	**↑6 ↓2**	**↑6 ↓4**	*↑4 ↓5*	*↑3 ↓7*	**↑45 ↓20**
20	(10)	**↑1 ↓0**	**↑8 ↓0**	**↑9 ↓1**	**↑8 ↓0**	**↑8 ↓1**	**↑6 ↓4**	**↑5 ↓4**	*↑3 ↓6*	**↑48 ↓16**
40	(10)	↑1 ↓1	**↑8 ↓1**	**↑8 ↓1**	**↑8 ↓0**	**↑8 ↓0**	**↑6 ↓3**	**↑6 ↓4**	↑5 ↓5	**↑50 ↓15**
80	(10)	**↑1 ↓0**	**↑6 ↓1**	**↑8 ↓1**	**↑10 ↓0**	**↑10 ↓0**	**↑6 ↓0**	**↑6 ↓4**	**↑5 ↓4**	**↑52 ↓10**
C	(6)	**↑3 ↓1**	**↑27 ↓0**	**↑26 ↓0**	**↑26 ↓2**	**↑20 ↓4**	**↑15 ↓12**	*↑11 ↓16*	*↑6 ↓21*	**↑134 ↓56**
S	(6)	↑2 ↓2	**↑23 ↓2**	**↑24 ↓4**	**↑20 ↓5**	**↑22 ↓6**	**↑17 ↓10**	**↑15 ↓14**	*↑14 ↓16*	**↑137 ↓59**
		GDBG								
F_{1a}	(30)	*↑1 ↓2*	**↑23 ↓0**	**↑27 ↓0**	**↑30 ↓0**	**↑25 ↓3**	**↑18 ↓10**	**↑16 ↓12**	*↑10 ↓17*	**↑150 ↓44**
F_{1b}	(30)	**↑3 ↓0**	**↑25 ↓0**	**↑27 ↓0**	**↑30 ↓0**	**↑24 ↓3**	**↑19 ↓10**	↑13 ↓13	*↑10 ↓17*	**↑151 ↓43**
F_2	(30)	*↑1 ↓2*	**↑25 ↓0**	**↑27 ↓0**	**↑30 ↓0**	**↑29 ↓0**	**↑26 ↓3**	**↑18 ↓7**	**↑17 ↓12**	**↑173 ↓24**
F_3	(30)	**↑3 ↓0**	**↑23 ↓0**	**↑22 ↓1**	**↑27 ↓0**	**↑30 ↓0**	**↑26 ↓1**	**↑24 ↓5**	**↑21 ↓6**	**↑176 ↓13**
F_4	(30)	**↑3 ↓1**	**↑23 ↓1**	**↑26 ↓1**	**↑30 ↓0**	**↑30 ↓0**	**↑20 ↓1**	**↑18 ↓10**	**↑17 ↓12**	**↑167 ↓26**
F_5	(30)	**↑4 ↓1**	**↑23 ↓2**	**↑20 ↓3**	**↑13 ↓12**	**↑14 ↓13**	**↑18 ↓9**	**↑16 ↓8**	↑13 ↓13	**↑121 ↓61**
F_6	(30)	**↑3 ↓2**	**↑23 ↓1**	**↑23 ↓1**	**↑27 ↓0**	**↑30 ↓0**	**↑25 ↓1**	**↑23 ↓6**	**↑19 ↓9**	**↑173 ↓20**
T_1	(35)	↑1 ↓1	**↑26 ↓2**	**↑24 ↓2**	**↑30 ↓2**	**↑30 ↓4**	**↑23 ↓9**	**↑19 ↓14**	**↑17 ↓15**	**↑170 ↓49**
T_2	(35)	**↑4 ↓3**	**↑27 ↓1**	**↑28 ↓3**	**↑32 ↓3**	**↑30 ↓2**	**↑26 ↓5**	**↑22 ↓8**	**↑17 ↓14**	**↑186 ↓39**
T_3	(35)	**↑3 ↓0**	**↑21 ↓1**	**↑21 ↓1**	**↑30 ↓3**	**↑30 ↓3**	**↑25 ↓4**	**↑23 ↓8**	**↑19 ↓13**	**↑172 ↓33**
T_4	(35)	**↑4 ↓1**	**↑34 ↓0**	**↑35 ↓0**	**↑31 ↓1**	**↑30 ↓4**	**↑26 ↓6**	**↑19 ↓12**	*↑15 ↓16*	**↑194 ↓40**
T_5	(35)	*↑0 ↓3*	**↑30 ↓0**	**↑34 ↓0**	**↑32 ↓2**	**↑32 ↓2**	**↑28 ↓4**	**↑23 ↓8**	**↑20 ↓14**	**↑199 ↓33**
T_6	(35)	**↑6 ↓0**	**↑27 ↓0**	**↑30 ↓0**	**↑32 ↓1**	**↑30 ↓4**	**↑24 ↓7**	**↑22 ↓11**	**↑19 ↓14**	**↑190 ↓37**
All	(270)	**↑23 ↓11**	**↑215 ↓6**	**↑222 ↓10**	**↑233 ↓19**	**↑224 ↓29**	**↑184 ↓57**	**↑154 ↓91**	**↑127 ↓123**	**↑1382 ↓346**

result of Mendes and Mohais [2005] that using a small sub-population size of six is an effective parameter setting, but concluded that only one Brownian individual should be used, rather than two.

Three DynDE-based algorithms were introduced and evaluated in this chapter. The first algorithm, competitive population evaluation (CPE), allows sub-populations to compete for function evaluations based on performance. The second algorithm, reinitialisation midpoint check (RMC), is aimed at improving the exclusion process of DynDE. The third algorithm, competitive differential evolution (CDE) is a combination of CPE and RMC.

A scalability study was conducted on DynDE, CPE, RMC and CDE to determine how the algorithms scale with respect to the number of dimensions, change period, severity of changes, change types, and underlying function in dynamic environments. The study found that DynDE and RMC exhibit similar scaling behaviour, while CPE and CDE exhibit similar scaling behaviour. An increase in the number of dimensions resulted in an increase in offline error for all the algorithms, with CPE and CDE yielding lower offline errors as the number of dimensions is increased.

CPE and CDE scaled better than DynDE and RMC as the period between changes in the environment is decreased, with the exception of extremely low change periods where the algorithms performed similarly. The offline errors of all the algorithms were found to decrease as the change period is increased, since more function evaluations are available between changes. At high change periods DynDE and RMC were found to perform better than CPE and CDE.

An increase in change severity resulted in lower offline errors for all algorithms. CPE and CDE were found to scale better than DynDE and RMC as the severity of changes in the environment increases. Experimental results showed that the improved performance of CPE and CDE vs DynDE and RMC on high change severity problems is due to CPE and CDE locating optima faster, after changes in the environment.

The underlying function used in the benchmark was shown to have a strong influence on the performance of the optimisation algorithm. The underlying function generally determines the magnitude of the difference in offline error of CPE and CDE vs DynDE and RMC. The influence of the type of changes in the environment was found to be strongly linked to the underlying function. Generally, the change type influences the point at which

DynDE and RMC start outperforming CPE and CDE due to an increased change period.

An analysis was conducted to determine whether CPE, RMC and CDE yielded statistically significantly better results than DynDE. CPE was found to perform significantly better than DynDE in the vast majority of experiments. The improvements were found to be especially pronounced in the lower change period experiments. RMC was found to perform better than DynDE in a small number of experiments localised in the low dimensional environments. CDE performed significantly better than DynDE in the majority of experiments and was shown to perform better than both RMC and CPE.

Experimental evidence was presented to show that CDE yielded a lower offline error than DynDE, because CDE's error values decreased faster after a change in the environment than those of DynDE. The diversity of the individuals in the search space differed very little between CDE and DynDE, but the average diversity per sub-population of CDE decreased faster than that of DynDE.

CDE and DynDE were compared in terms of the average error immediately before changes in the environment to investigate the possibility that CDE merely exploits the offline error performance measure. An analysis of the experimental results showed that, while improvements in terms of the average error immediately before changes were less pronounced than when using the offline error, CDE still performed significantly better than DynDE in the majority of the experiments.

An extensive comparison to *jDE*, a state-of-the-art optimisation algorithm for DOPs, found that CDE performed significantly better than *jDE* in the majority of investigated environments, especially at lower change periods. A comparison of CDE to the reported results of other algorithms found in the literature showed that CDE compares favourably with state-of-the-art algorithms.

The general applicability of the approaches suggested in this chapter was investigated by incorporating the novel components of CDE into the *jDE* algorithm. The experimental results proved that *jDE* was improved by the incorporation of the new approaches, but that the resultant algorithm was still inferior to CDE.

The evaluations in this chapter did not focus on the effect of the number of optima in the environment. The following chapter focuses on this important consideration.