Adapting Weighted Aggregation for Multiobjective Evolution Strategies

Yaochu Jin, Tatsuya Okabe, and Bernhard Sendhoff

Future Technology Research
Honda R&D Europe (D) GmbH
63073 Offenbach/Main, Germany
yaochu.jin@hre-ftr.f.rd.honda.co.jp

Abstract. The conventional weighted aggregation method is extended to realize multi-objective optimization. The basic idea is that by systematically changing the weights during evolution will lead the population to the Pareto front. Two possible methods are investigated. One method is to assign a uniformly distributed random weight to each individual in the population in each generation. The other method is to change the weight periodically with the process of the evolution. We found in both cases that the population is able to approach the Pareto front, although it will not keep all the found Pareto solutions in the population. Therefore, an archive of non-dominated solutions is maintained. Case studies are carried out on some of the test functions used in [1] and [2]. Simulation results show that the proposed approaches are simple and effective.

1 Introduction

A large number of evolutionary multiobjective algorithms (EMOA) have been proposed [3, 4]. So far, there are three main approaches to evolutionary multi-objective optimization, namely, aggregation approaches, population-based non-Pareto approaches and Pareto-based approaches [4]. In the recent years, the Pareto-based approaches have gained increasing attention in the evolutionary computation community and several successful algorithms have been proposed [5].

Despite their weaknesses, the aggregation approaches are very easy to implement and computationally efficient. Usually, aggregation approaches can provide only one Pareto solution if the weights are fixed using problem-specific prior knowledge. However, it is also possible to find more than one Pareto solution using this method by changing the weights during optimization. In [6], the weights of the different objectives are encoded in the chromosome to obtain more than one Pareto solutions. Phenotypic fitness sharing is used to keep the diversity of the weight combinations and mating restrictions are required so that the algorithm can work properly.

Most of the EMOAs are based on Genetic Algorithms and relatively little attention has been paid to evolution strategies. Some exceptions are [2, 7–9]. In [7], average ranking is used to dictate the deletion of a fraction of the population. A predator-prey-model is proposed in [9]. A selection method that is similar to the VEGA approach [10] is adopted in [8]. An algorithm called Pareto Archived Evolution Strategy (PAES) is suggested in [2], in which a non-Pareto approach together with an archive of the found Pareto solutions are used.

This paper investigates two methods using the aggregation-based approach. To approximate the Pareto front instead of a certain Pareto solution, the weight for each objective should be changed systematically. One method is to distribute the weights uniformly among the individuals in the population. The other method is to periodically change the weights with the process of the evolution. Although these methods seem to be very simple, we will show that they work effectively for two objective optimization problems. Simulations are carried out on different test functions studied in [1, 2]. Different evolution strategies, including the standard evolution trategy [11], the Evolution Strategy with Rotation Matrix Adaptation [11] and the Evolution Strategy with Covariance Matrix Adaptation [12, 13] are employed.

What is quite surprising from our simulation results is that our algorithms work well even for problems with a concave Pareto front (see Section 4 for details), which is usually thought to be not obtainable by aggregation based methods [4]. Our preliminary explanation is that if the search algorithm goes through the concave region of the Pareto front (which is locally near-optimal when the objectives are aggregated into one single objective function) and if the near optimal solutions are archived, then the Pareto solutions within the concave region can also be found using the aggregation method. Nevertheless, further studies on this issue are desirable.

2 The Aggregation Based Multiobjective Algorithms

2.1 The Evolution Strategies

In the standard Evolution Strategy (ES), the mutation of the objective parameters is carried out by adding an $N(0, \sigma_i^2)$ distributed random number. The step sizes σ_i are also encoded in the genotype and subject to mutations. A standard Evolution Strategy can be described as follows:

$$\mathbf{x}(t) = \mathbf{x}(t-1) + \tilde{\mathbf{z}} \tag{1}$$

$$\sigma_i(t) = \sigma_i(t-1)\exp(\tau' z)\exp(\tau z_i); i = 1, ..., n \tag{2}$$

where $\mathbf{x}$ is an n-dimensional parameter vector to be optimized, $\tilde{\mathbf{z}}$ is an n-dimensional random number vector with $\tilde{\mathbf{z}} \sim N(\mathbf{0}, \boldsymbol{\sigma}(t)^2)$, z and z_i are normally distributed random numbers with $z, z_i \sim N(0, 1)$. Parameters τ, τ' and σ_i are the strategy parameters, where σ_i is mutated as in equation(2) and τ, τ' are constants as follows:

$$\tau = \left(\sqrt{2\sqrt{n}}\right)^{-1} ; \tau' = \left(\sqrt{2n}\right)^{-1} \tag{3}$$

There are several extensions to the above standard ES. In our simulations, an ES with Rotation Matrix Adaptation and an ES with Covariance Matrix Adaptation as well as the standard ES are used to investigate the effectiveness of the proposed multiobjective algorithms using different search strategies. For the detailed description of the evolution strategies, please refer to [11, 12] respectively.

Two main different selection schemes are used in evolution strategies. Suppose there are μ and λ individuals in the parent and offspring population, usually $\mu \leq \lambda$. One method is to select the μ parent individuals only from the λ offspring, which is usually noted as (μ,λ)-ES. If the μ parent individuals are selected from a combination of the μ parent individuals and the λ offspring individuals, the algorithm is noted as ($\mu + \lambda$)-ES. In our study, the (μ, λ)-ES is adopted.

2.2 Random Distribution of Weights within a Population

For the sake of clarity, we consider the two objective problems in the current discussion; the extension to problems with more than two objectives is straightforward. For a conventional aggregation method, the

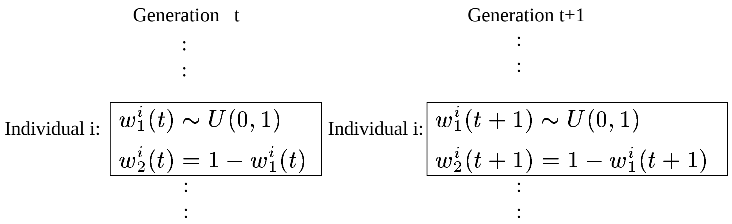


Fig. 1. Determination of the weights: Random distribution.

fitness function is the weighted sum of the two different objectives f_1 and f_2:

$$Fitness = w_1 f_1 + w_2 f_2, \tag{4}$$

where w_1 and w_2 are two constants determined using a prior knowledge about the problem. It is clear that by using a pair of fixed weights, only one Pareto solution can be obtained.

Imagine that we run the algorithm so many times that every weight combination has been used. In this way, we can obtain all Pareto solutions that the Pareto front consists of. Notice, that it has been argued that the Pareto solutions locating in the concave region of the Pareto front cannot be obtained by aggregation methods. However, in the experiments in Section 4, we found that our algorithms are successful in obtaining a very complete concave Pareto front for low dimensional problems (e.g. $n = 2$), and a quite complete concave Pareto front with a dimension as high as 10.

Of course, it is unpractical, if not impossible to run the evolutionary algorithm so many times to exhaust all the weight combinations. Since we are using evolutionary optimization, it is natural to take advantage of the population for this purpose. If the different weight combinations can be distributed among the individuals, the population may be able to approach the Pareto front during the process of evolution. Suppose we use the $(\mu,\ \lambda)$-ES, then the weight combinations can be distributed uniformly among the λ individuals in the offspring population. Let

$$w_1^i(t) = random(\lambda)/\lambda, \tag{5}$$

$$w_2^i(t) = 1.0 - w_1^i(t), \tag{6}$$

where $i = 1, 2, ..., \lambda$ and t is the index for generation number. The function $random(\lambda)$ generates a uniformly distributed random number between 0 and λ. In this way, we can get a uniformly distributed random weight combination $(w_1^i,\ w_2^i)$ among the individuals, where $0 \leq w_1^i, w_2^i \leq 1$ and $w_1^i + w_2^i = 1$, refer also to Fig.1, in which $U(0,1)$ denotes a uniform distribution. Notice that the weight combinations are regenerated in every generation.

2.3 Generation-Based Periodical Variation of the Weights

The idea of a uniformly distributed weight combination can straightforwardly be extended to a generation based approach. However, if we still use a random weight combination, convergence of the evolutionary algorithm will be in question. Therefore, instead of using a randomly distributed weight combination, we use a weight combination that is changed gradually and periodically with the process of the evolution. In this work, this is realized as follows:

$$w_1(t) = |\sin(2\pi t)/F|, \tag{7}$$

$$w_2(t) = 1.0 - w_1(t), \tag{8}$$

where t is the number of generation and $|\cdot|$ gives the absolute value. We can see from equation (7) that $w_1(t)$ changes from 0 to 1 periodically. The change frequency can be adjusted by F. In our study, we set $F = 400$, which means that in every 400 generations, w_1 will change from 0 to 1 and then from 1 to 0 four times. Fig.2 shows an example of how the weights change during evolution within 200 generations. We found that the results of the algorithm are not very sensitive to F, although it seems reasonable to let the weight change from 0 to 1 twice. Notice in this case, all the individuals have the same weight combination in the same generation.

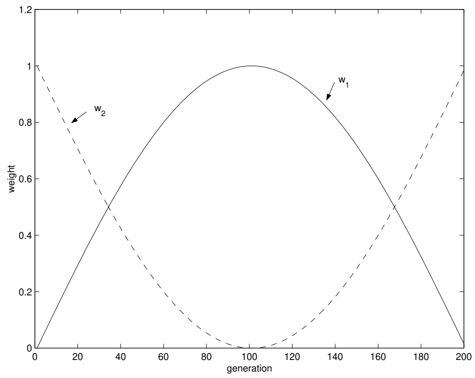


Fig. 2. Determination of weights: Generation-based periodical variation, all individuals have the same weight combination within a generation.

2.4 An Archive of Pareto Solutions

In our algorithm, the population is not able to keep all the found Pareto solutions, although it is able to approach the Pareto front dynamically. Therefore, it is necessary to record the Pareto solutions that are found so far. The idea of building a Pareto archive is inspired from [2], although we use quite a different method to aggregate and maintain the archive. The pseudo-code for building the archive is listed in Algorithm 1. The similarity is measured by the Euclidean distance in the fitness space.

3 Test Functions

To evaluate the effectiveness of the proposed algorithms, simulations are carried out on four test functions used in [1, 2].

- The first test function (F_1) used here is the second function in [2] and we extend it to an n-dimensional function:

$$f_1 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 \tag{9}$$

$$f_2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - 2.0)^2 \tag{10}$$

```
for each individual o in offspring population do
  if (o dominates an individual in parent population p) and (o is not dominated by any solutions in the archive)
     and (o is not similar to any solutions in the archive) then
    if archive is not full then
      add o to the archive
    else if o dominates any solution a in the archive then
      replace a with o
    else if any solution a1 in the archive dominates another solution a2 then
      replace a2 with o
    else
      discard o
    end if
  else
    discard o
  end if
end for
for each solution in the archive do
  if solution a1 dominates a2 then
    remove a2
  end if
end for
```

Algorithm 1: Pseudo-code for maintaining an archive of Pareto solutions.

- The second test function (F_2) is the first function in [1], which has a convex Pareto front:

$$f_1 = x_1 \tag{11}$$

$$g(x_2, ..., x_n) = 1.0 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{12}$$

$$f_2 = g \times (1.0 - \sqrt{f_1/g}) \tag{13}$$

  where $x_i \in [0, 1]$.
- The third test function (F_3) is the second function in [1], which has a concave Pareto front:

$$f_1 = x_1 \tag{14}$$

$$g(x_2, ..., x_n) = 1.0 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{15}$$

$$f_2 = g \times (1.0 - (f_1/g)^2) \tag{16}$$

  where $x_i \in [0, 1]$.
- The fourth test function (F_4) is the third function in [1], whose Pareto front consists of a number of separated convex parts:

$$f_1 = x_1 \tag{17}$$

$$g(x_2, ..., x_n) = 1.0 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{18}$$

$$f_2 = g \times (1.0 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)) \tag{19}$$

where $x_i \in [0, 1]$.

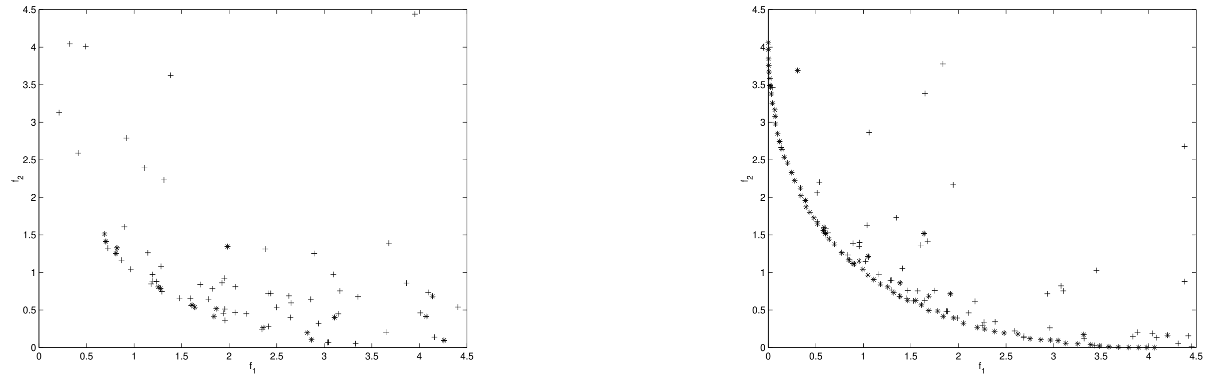
4 Case Studies

The goal of the simulations is to verify the effectiveness of the proposed algorithms using three different evolution strategies. The first part of the simulation is to demonstrate that both the random distribution of the weights among the population (hereafter noted as Method 1) and the generation based periodical variation of the weights (hereafter called Method 2) proposed in Section 2 are able to approximate the Pareto front for different low-dimensional problems, even for a concave Pareto front, which is usually considered to be impossible by aggregation methods. Please refer to our comments in Section 1. The second part of the simulation is intended to compare the effectiveness of Method 1 and Method 2 for high-dimensional problems. It is found that Method 2 is more effective for finding a complete Pareto front. Finally, we also show that different ES algorithms have different performance for multiobjective optimization with the same fitness assignment method. From these comparative studies, we conclude that the ES with Rotation Matrix Adaptation using Method 2 gives consistently satisfying results on different kinds of problems.

In all the simulations, a (15, 100)-ES is used and the maximal number of generation is 500.

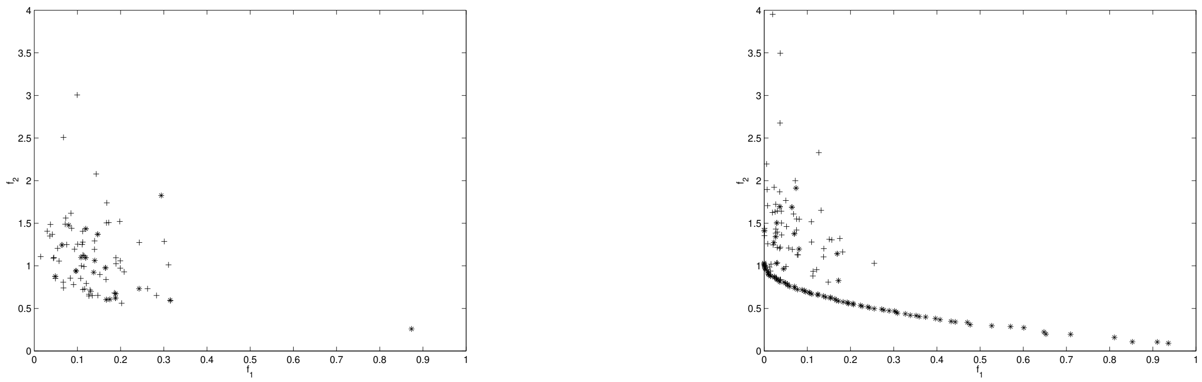
4.1 Comparison of both methods for low dimensional problems

In this part, we use the standard ES for all the four test functions with dimension $n = 2$. In the figures hereafter, the stars('$*$') denote the archived solutions and the plus signs ('$+$') denote the individuals in the offspring population. Figures 3(a) and (b) show the archived solutions and the offspring in the $10th$ and $500th$ generation, respectively, using Method 1 for F_1. It is clearly seen that the population has successfully approximated the Pareto front. Figures 4, 5 and 6 show the corresponding results for F_2, F_3 and F_4 using Method 1. From the figures, we find that Method 1 is working effectively for all different



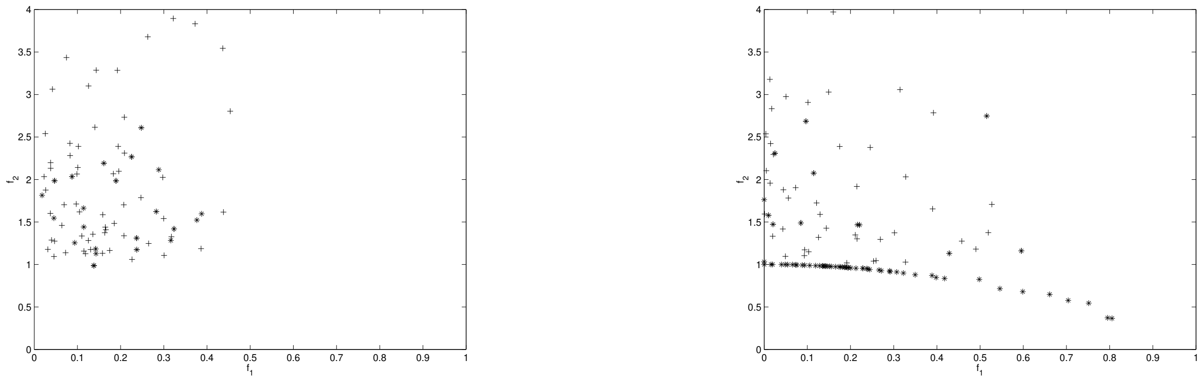
(a) (b)

Fig. 3. Method 1 for F_1 (n=2) using the standard ES: (a) generation 10, (b) generation 500.



(a) (b)

Fig. 4. Method 1 for F_2 (n=2) using the standard ES: (a) generation 10, (b) generation 500.



(a) (b)

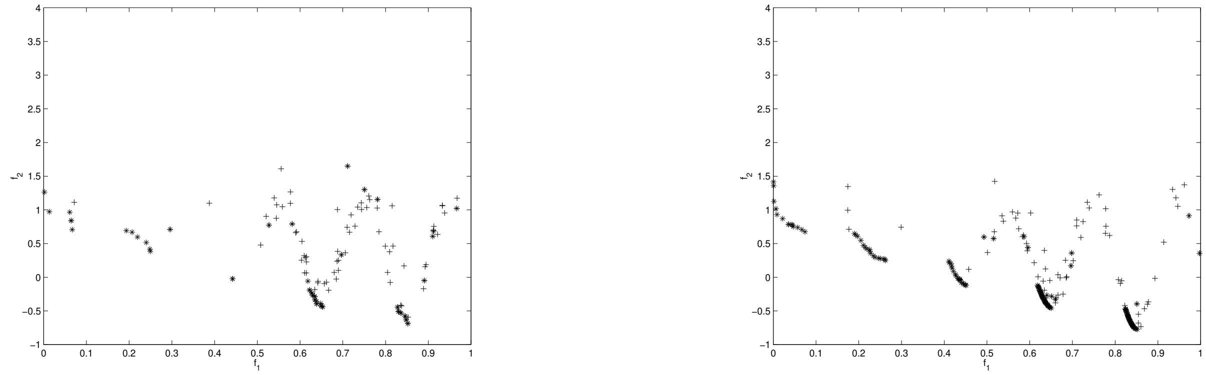
Fig. 5. Method 1 for F_3 (n=2) using the standard ES: (a) generation 10, (b) generation 500.

test functions, which shows that the idea of using a uniformly random weight distribution among the individuals of the population is for multi-objective optimization feasible.

Next, we run simulations on all four test functions with $n = 2$ using Method 2. The results are shown in Figures 7, 8, 9 and 10.

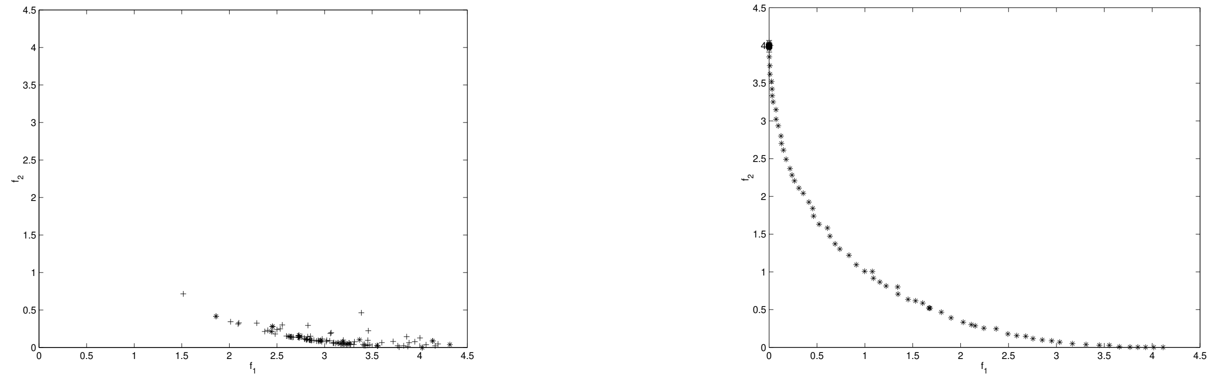
From these figures, it is demonstrated that Method 2 has been successful in obtaining a very complete Pareto front for all the four test functions. The difference between Method 1 and Method 2 is that the individuals in the population in Method 2 converged more completely to the Pareto front than the individuals in Method 1 at the end of the evolution.

The most interesting fact is that both methods have obtained very complete Pareto solutions for F_3, which has a concave Pareto front. Our empirical results show that in principle, concave Pareto solutions can be obtained by aggregation methods if the search algorithm is able to go through the concave region and if an archive is used to store the found Pareto solutions.



(a) (b)

Fig. 6. Method 1 for F_4 (n=2) using the standard ES: (a) generation 10, (b) generation 500.

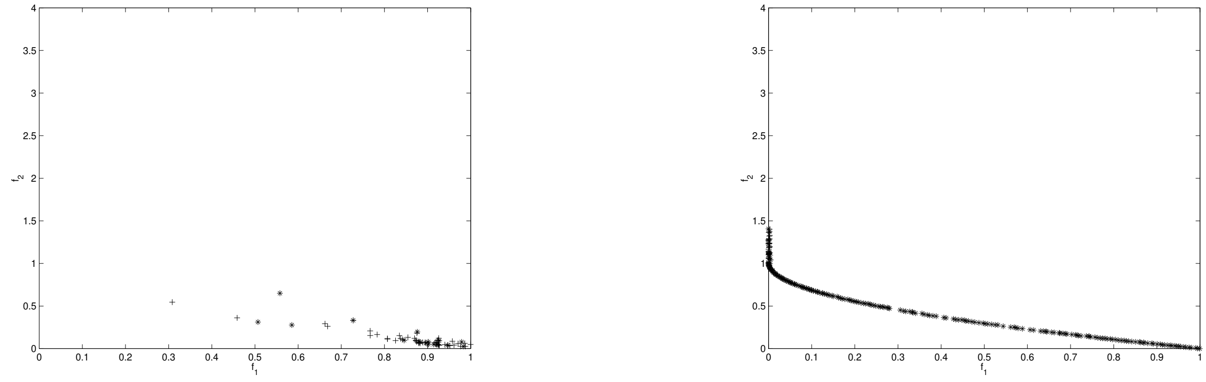


(a) (b)

Fig. 7. Method 2 for F_1 (n=2) using the standard ES: (a) generation 10, (b) generation 500.

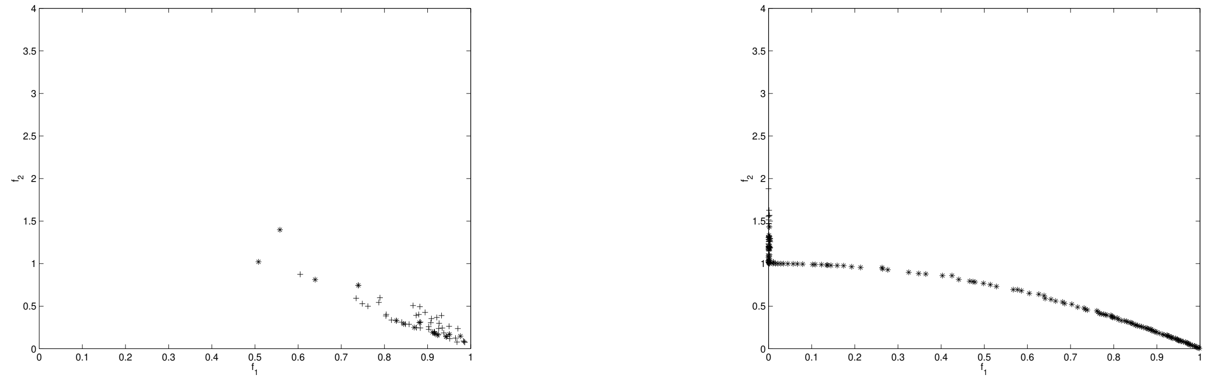
4.2 Comparison of both methods for high dimensional problems

The performance of Method 1 and Method 2 is compared in this part of the simulations. The purpose of the simulations is to investigate the efficiency of Method 1 and Method 2 for more complex problems. In the results presented in the following, the standard ES is used for the four test functions with a dimension of 10 and 500 generations are run. Figures 11 and 12 give the results using Method 1. It is seen that the performance of Method 1 on 10-dimensional problems becomes worse compared to those obtained in the two dimensional problems. In contrast, Method 2 is still able to provide very good results on the same problems, as shown in Figures 13 and 14. Notice that Method 1 shows particularly bad performance on test functions F_3 and F_4, which have a concave or discontinuous Pareto front. However, Method 2 shows quite good performance on all the four test functions.



(a) (b)

Fig. 8. Method 2 for F_2 (n=2) using the standard ES: (a) generation 10, (b) generation 500.

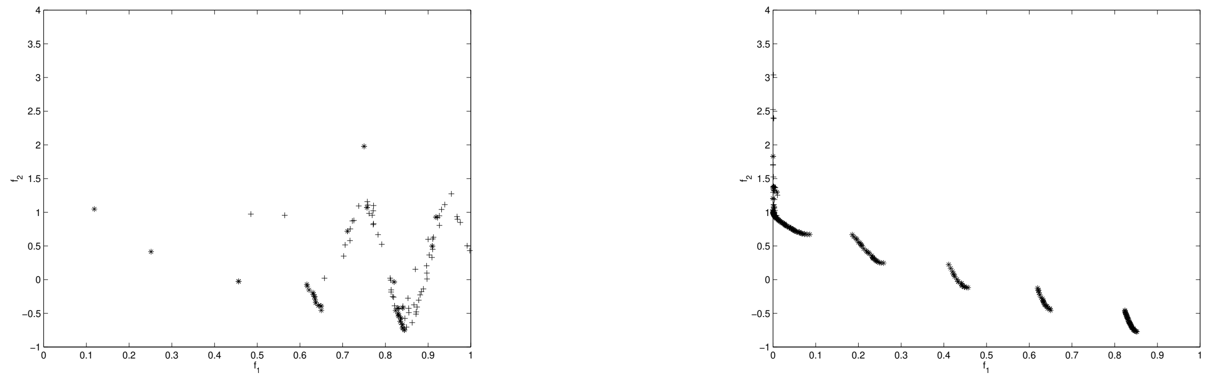


(a) (b)

Fig. 9. Method 2 for F_3 (n=2) using the standard ES: (a) generation 10, (b) generation 500.

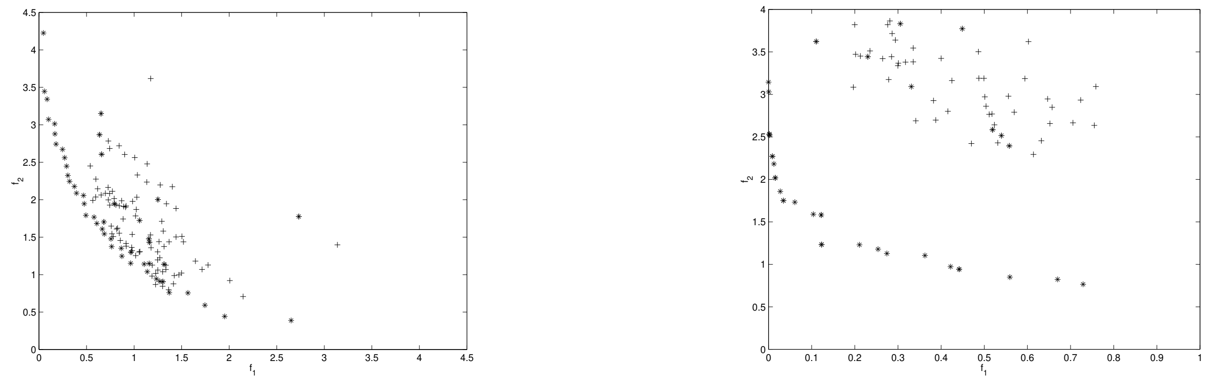
4.3 Comparison of different evolution strategies for Method 2

This part of the simulation aims at comparing the performance of different ES algorithms for more difficult problems (with higher dimension in this context). Since Method 2 exhibits much better performance in the above part of experiments, only Method 2 will be used in the following simulations. The algorithms considered in this work are the standard ES, the ES with Rotation Matrix Adaptation and the ES with Covariance Matrix Adaptation. The problems studied in this part of the simulation are the four test functions with $n = 30$. Results using the standard ES on the four functions are shown in Figures 15 and 16. The results using the ES with Rotation Matrix Adaptation are given in Figures 17 and 18. Finally, the ES with Covariance Matrix Adaptation is tested and the results are presented in Figures 19 and 20. In all the simulations, 500 generations are run. As it is shown in the above figures, the standard ES together with Method 2 can always provide quite a complete Pareto front, but unfortunately, the accuracy of the solutions is not satisfactory. On the other hand, the ES with Rotation Matrix Adaptation gives consistently good results on all the four problems, which are comparable to or even better than



(a) (b)

Fig. 10. Method 2 for F_4 (n=2) using the standard ES: (a) generation 10, (b) Generation 500.



(a) (b)

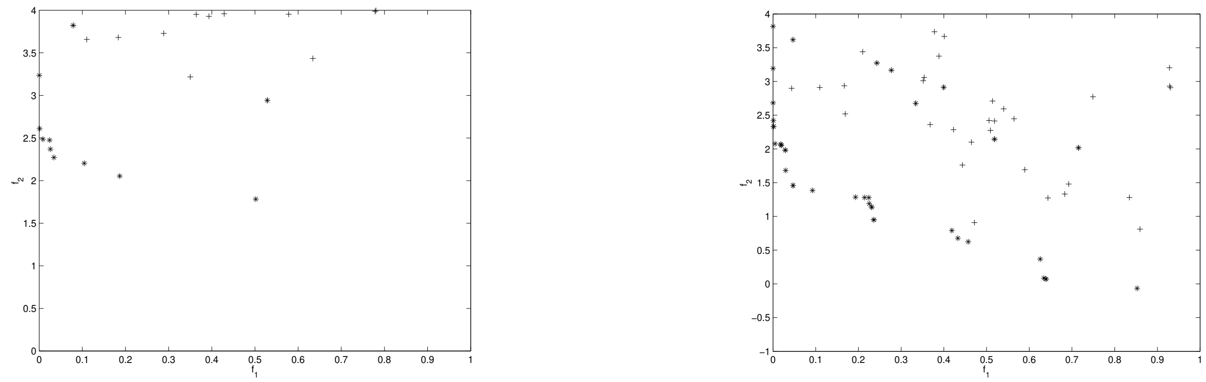
Fig. 11. Method 1 using the standard ES for (a) F_1 (n=10) and (b) F_2 (n=10)

those of the Pareto-based algorithms described in [1]. Interestingly, the ES with CMA produced very good results on F_1, but failed on F_3 and F_4. This may be ascribed to the fact that the ES with CMA is a more deterministic algorithm that converges quickly before it can explore a wider region of the search space.

At the same time, as for the low dimensional problems studied in Section 4.1 of this section, the ES with RMA is able to obtain a quite complete concave Pareto front with the dimension of 30. This was surprising taking into account the fact that it is a simple, dynamic aggregation based approach.

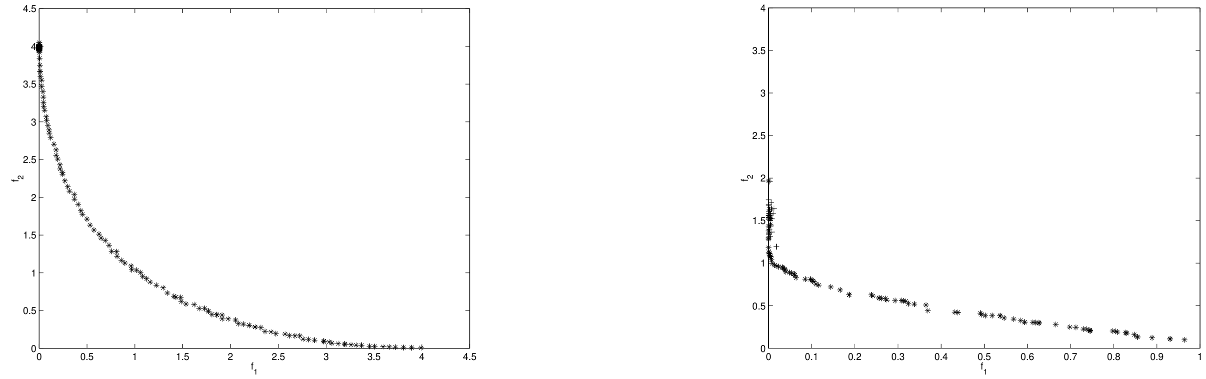
5 Conclusion

Two aggregation based methods for multiobjective optimization are proposed in this paper. The idea is to use dynamic weights instead of fixed weights to achieve the Pareto solutions. We found that both methods



(a) (b)

Fig. 12. Method 1 using the standard ES for (a) F_3 (n=10) and (b) F_4 (n=10)



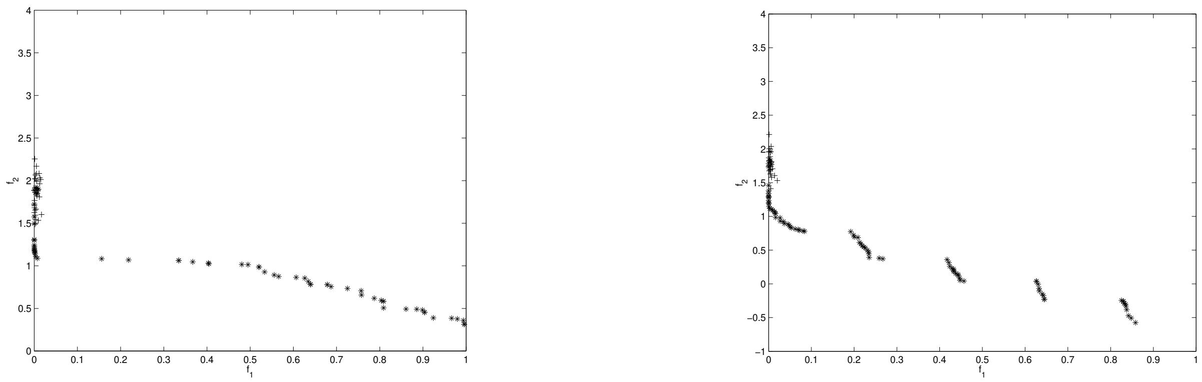
(a) (b)

Fig. 13. Method 2 using the standard ES for (a) F_1 (n=10) (b) F_2 (n=10)

work well on low-dimensional problems. However, for high-dimensional problems, the second method outperforms the first one. Furthermore, the simulation results also depend on the type of evolution strategy that is employed. On the other hand, it also depends on the performance of the evolution strategy. In our experiment, the Evolution Strategy with Rotation Matrix Adaptation gives better performance than the standard Evolution Strategy. At the same time, the Evolution Strategy with Covariance Matrix Adaptation provides very good results on smooth, high-dimensional problems, but its performance degrades seriously on problems with a discontinuous and non-convex Pareto-optimal front.

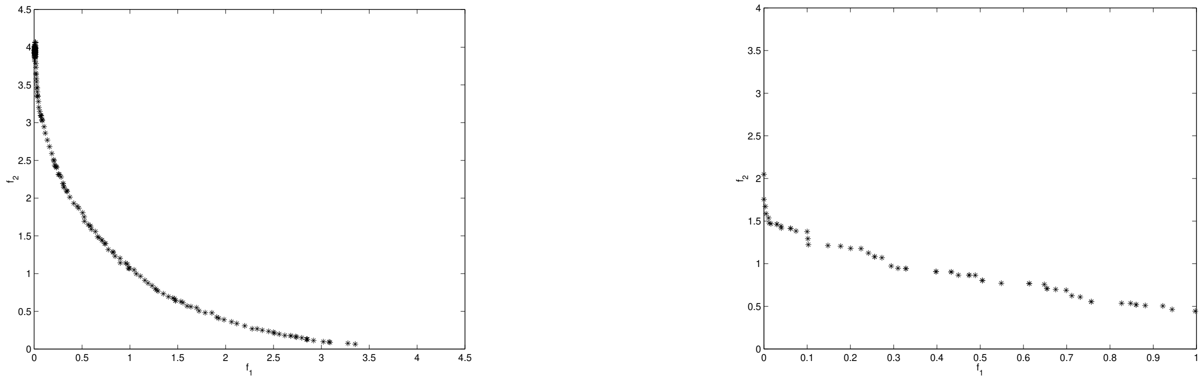
In our approach, no changes have to be made to the conventional evolutionary algorithm except for the dynamic weights and an archive of the found Pareto solutions. Therefore, the methods can straightforwardly be applied to all existing evolutionary algorithms with only minor modifications.

Another interesting phenomenon is that the proposed algorithms are able to find the Pareto solutions with a concave Pareto front. This is a very encouraging point when we are applying aggregation-based



(a) (b)

Fig. 14. Method 2 using the standard ES for (a) F_3 (n=10) and (b) F_4 (n=10)



(a) (b)

Fig. 15. Method 2 using the standard ES for (a) F_1 (n=30) and (b) F_2 (n=30)

methods to multi-objective optimization. Further investigation of this issue will be part of our future work.

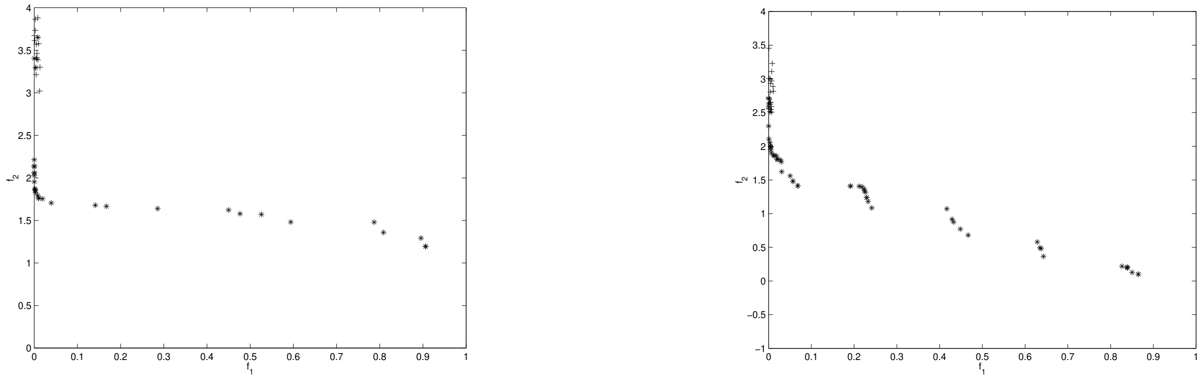
The problems studied in this paper are all two-objective ones. Theoretically, the proposed methods can be extended to problems with more than two objectives. Expected problems are the increasing complexity and the decreasing efficiency, which, however, is also true for the Pareto-based approaches.

ACKNOWLEDGMENTS

The authors would like to thank E. Körner and W. von Seelen for their support and T. Arima for his insightful comments.

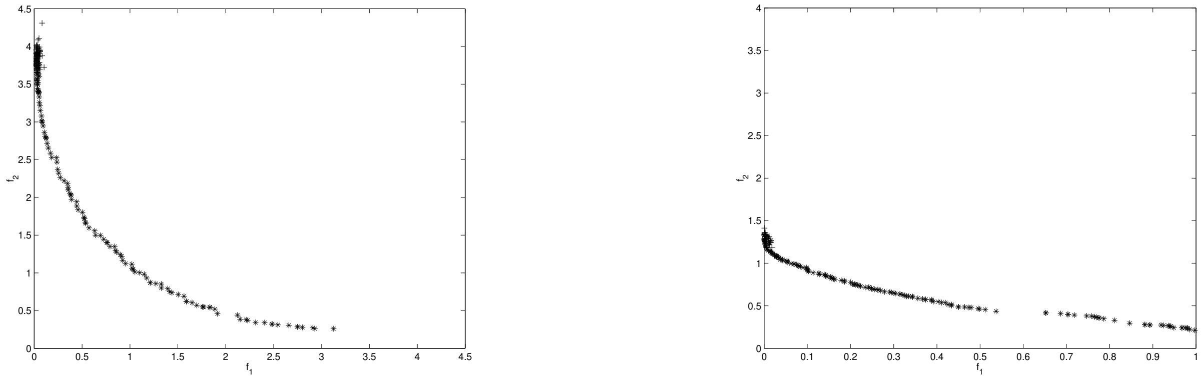
References

1. E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolution algorithms: empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.



(a) (b)

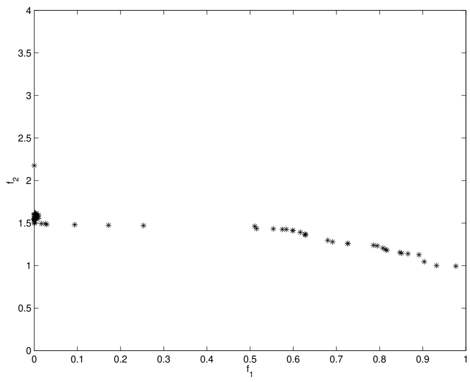
Fig. 16. Method 2 using the standard ES for (a) F_3 (n=30) and (b) F_4 (n=30)



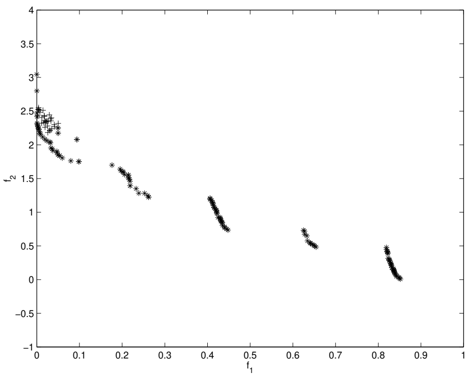
(a) (b)

Fig. 17. Method 2 using the ES with RMA for (a) F_1 (n=30) and (b) F_2 (n=30)

2. J. D. Knowles and D. W. Corne. Approximating the nondominated front using the Pareto archived evolution strategies. *Evolutionary Computation*, 8(2):149–172, 2000.
3. C.A.C. Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, 1(3):269–308, 1999.
4. C. M. Fonseca and P. J. Fleming. Multiobjective optimization. In Th. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation*, volume 2, pages 25–37. Institute of Physics Publishing, Bristol, 2000.
5. D. A. Van Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithms: Analyzing the state-of-art. *Evolutionary Computation*, 8(2):125–147, 2000.
6. P. Hajela and C. Y. Lin. Genetic search strategies in multicriteria optimal design. *Structural Optimization*, 4:99–107, 1992.
7. F. Kursawe. A variant of evolution strategies for vector optimization. In H.-P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature*, volume I, pages 193–197, 1991.
8. T. Binh and U. Korn. Multiobjective evolution strategy with linear and nonlinear constraints. In *Proceedings of the 15th IMACS World Congress on Scientific Conputation, Modeling and Applied Mathematics*, pages 357–362, 1997.

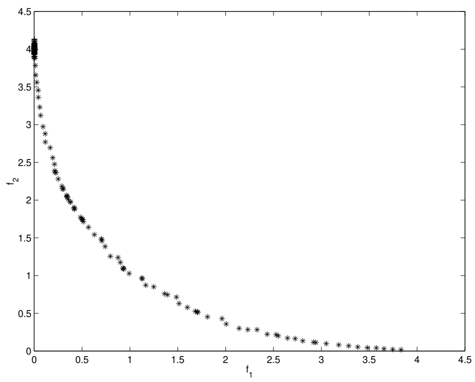


(a)

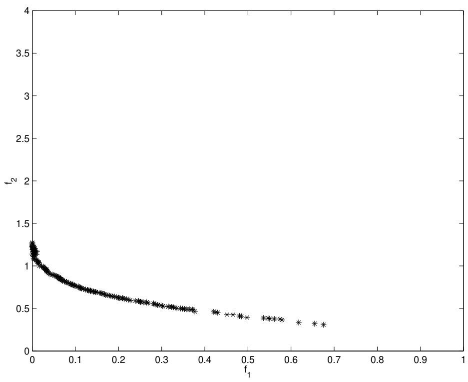


(b)

Fig. 18. Method 2 using the ES with RMA for (a) F_3 (n=30) and (b) F_4 (n=30)



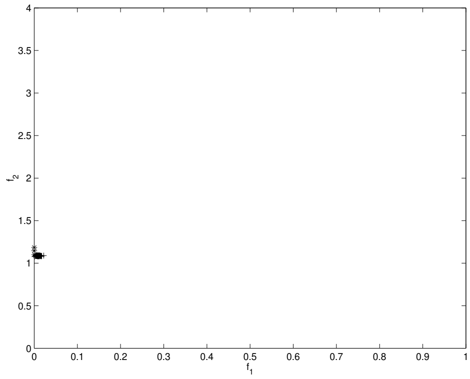
(a)



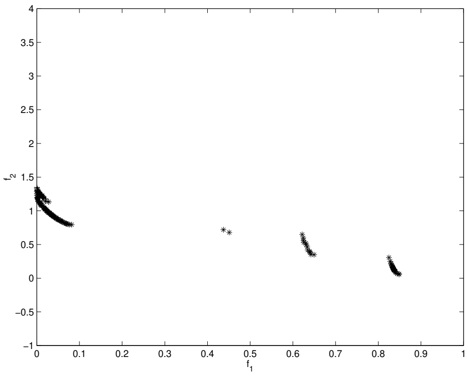
(b)

Fig. 19. Method 2 using the ES with CMA for (a) F_1 (n=30) and (b) F_2 (n=30)

9. M. Laumanns, G. Rudolph, and H.-P. Schwefel. A spatial predator-prey approach to multi-objective optimization. In A.E. Eiben, Th. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature*, volume V, pages 241–249, 1998.
10. J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedins of an International Conference on Genetic Algorithms and Their Applications*, pages 93–100, 1985.
11. H.-P. Schwefel. *Evolution and Optimum Seeking.* Sixth-Generation Computer Technologies Series. John Wiley & Sons, Inc., 1994.
12. N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaption. In *Proc. 1996 IEEE Int. Conf. on Evolutionary Computation*, pages 312–317. IEEE Press, 1996.
13. N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 2000. To appear.



(a)



(b)

Fig. 20. Method 2 using the ES with CMA for (a) F_3 (n=30) and (b) F_4 (n=30)