

UT-T00030-SSI

Don.-2012

xx(199600.1)



CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL

Laboratorio de Tecnologías de Información,
CINVESTAV-Tamaulipas

**Uso de Modelos Surrogados en
Algoritmos Evolutivos
Multiobjetivo**

Tesis que presenta:

Gerardo Montemayor García

Para obtener el grado de:

**Maestro en Ciencias
en Computación**

Director de la Tesis:
Dr. Gregorio Toscano Pulido

Cd. Victoria, Tamaulipas, México.



Diciembre, 2011

CLASS.	UT 00030
APPROB.	UT-T00030-SSA
ISSUE	24-08-2012
REVISED	Don-2012

D:199452-2001



RESEARCH CENTER FOR ADVANCED STUDY
FROM THE NATIONAL POLYTECHNIC INSTITUTE

Information Technology Laboratory,
CINVESTAV-Tamaulipas

**On the Use of Surrogated Models
in Multiobjective Evolutionary
Algorithms**

Thesis by:

Gerardo Montemayor García

as the fulfillment of the
requirement for the degree of:

**Master of Science
in Computer Science**

Thesis Director:
Dr. Gregorio Toscano Pulido

© Copyright by
Gerardo Montemayor García
2011

This research was partially funded by project number 51623 from "Fondo Mixto Conacyt-Gobierno del Estado de Tamaulipas"

The thesis of Gerardo Montemayor García is approved by:

Dr. José Gabriel Ramírez Torres

Dr. Eduardo Rodríguez Tello

Dr. Gregorio Toscano Pulido, Committe Chair

Cd. Victoria, Tamaulipas, México., December 09 2011

A mis padres.

Acknowledgements

- To my parents, for their unconditional love. Without them, this would not have been possible.
- To my siblings and old friends, for their daily words of encouragement.
- To my girlfriend, for coping with me through this two long years.
- To Dr. Gregorio Toscano Pulido, for being a great friend and an excellent advisor.
- To my reviewers, Dr. Eduardo Rodríguez Tello and Dr. José Gabriel Ramírez Torres for their time and valuable comments.
- To my new friends from Cinvestav, to make my difficult times easier, and making me feel like a brother.
- To CONACyT, for the provided economic support which allowed me to concentrate in my studies and CINVESTAV-Tamaulipas for the opportunity to pursue graduate studies.
- I also acknowledge support from CONACyT through project 105060 "Uso de técnicas evolutivas híbridas para resolver problemas de optimización multiobjetivo dinámicos y con más de tres objetivos" under the lead of Dr. Gregorio Toscano Pulido.
- Finally, I want to thank Cinvestav, for this beautiful chapter in my life.

Contents

Contents	i
List of Figures	v
List of Tables	vii
List of Algorithms	ix
Publications	xi
Resumen	xiii
Abstract	xv
Nomenclature	xvii
1 Introduction	1
1.1 Introduction	1
1.2 Motivation and hypothesis	2
1.3 Objectives	3
1.3.1 General objective	3
1.3.2 Particular objectives	4
1.4 Thesis overview	4
2 Background	7
2.1 Introduction	7
2.2 Single objective optimization.	7
2.3 Multiobjective optimization	9
2.3.1 Multiobjective techniques classification	10
2.3.2 The goal of multiobjective optimization	11
2.4 Performance measures for MOEAs	12
2.4.1 Hypervolume ratio (HVR)	13
2.4.2 Averaged Hausdorff distance (Δ_p)	14
2.5 Measuring the accuracy of predictors	15
3 Multiobjective evolutionary algorithms aided by surrogate models	17
3.1 Introduction	17
3.2 Taxonomy of multiobjective evolutionary algorithms	18
3.3 A fast elitist non-dominated sorting genetic algorithm (NSGA-II)	20
3.3.1 A fast non-dominated sorting approach.	20

3.3.2	Crowding distance	21
3.3.3	Crowded comparison operator (\prec_n)	21
3.3.4	The main loop	22
3.4	Surrogate models	23
3.5	State of the art	24
4	A study of surrogate models for their use with multiobjective evolutionary algorithms	31
4.1	Introduction	31
4.2	Methodology	32
4.3	Parameter setting	34
4.4	Adopted criteria	36
4.5	Results from the study	39
4.5.1	Accuracy, robustness and scalability	39
4.5.2	Efficiency	41
4.5.3	Suitability	42
4.6	Conclusions from the study	45
5	The Metamodel Assisted Subpopulation-based Search Algorithm (MASSA)	47
5.1	Introduction	47
5.2	Initialization of the population	48
5.3	Select exploitable areas	48
5.4	Exploit a reduced area using a subpopulation	50
5.4.1	Improving the knowledge	51
5.4.1.1	\mathcal{E} delimitation	51
5.4.2	Training and optimizing the metamodels	52
5.4.3	Choosing and evaluating the best individuals	53
5.5	Preserving diversity	54
5.6	Reducing the population size	55
5.7	Settings used	55
5.8	Results from the MASSA	56
5.9	Conclusions	68
6	The Tune-adaptive Metamodel Assisted Algorithm (TAMAAL)	69
6.1	Introduction	69
6.2	G-metric, DP and RP tournaments	70
6.3	Settings used	70
6.4	Results from the TAMAAL	71
6.5	TAMAAL conclusions	81
7	Conclusions and future work	83
7.1	Conclusions	83
7.2	Limitations and future work	85

A	Metamodeling techniques	87
A.1	Kriging	87
A.2	Radial basis functions neural networks	89
A.3	Polynomial regression	92
A.4	Support vector machines for regression	93
B	Test functions	95
B.1	ZDT1	96
B.2	ZDT2	96
B.3	ZDT3	97
B.4	ZDT4	97
B.5	ZDT6	98
B.6	Kursawe	99
B.7	DTLZ1	100
B.8	DTLZ7	100
	Bibliography	103

List of Figures

2.1	Desirable and undesirable solutions in multiobjective optimization	12
2.2	Hypervolume of PF_k solutions with a reference point r	13
2.3	Main difference between GD and IGD.	15
3.1	Non-dominated sorting.	21
3.2	The crowding distance calculation	22
3.3	NSGA-II procedure.	23
4.1	Average accuracy with the BOS and BLS.	35
4.2	Average accuracy by number of variables.	40
4.3	Average time of execution for training (a) and prediction (b) in all the problems.	42
4.4	Average RP by number of variables.	44
4.5	Average RP by number of variables.	46
5.1	Diagram that illustrates the way in which MASSA works.	49
5.2	Split in subpopulations.	50
5.3	Selecting exploitable areas.	51
5.4	Box constraints delimitation.	52
5.5	Improving the knowledge.	53
5.6	Exploitation with few evaluations in the real system.	54
5.7	Δ_p and HVR box-plots and statistical summary for ZDT1 test problem.	58
5.8	Δ_p and HVR box-plots and statistical summary for ZDT2 test problem.	60
5.9	Δ_p and HVR box-plots and statistical summary for ZDT3 test problem.	61
5.10	Δ_p and HVR box-plots and statistical summary for ZDT4 test problem.	62
5.11	Δ_p and HVR box-plots and statistical summary for ZDT6 test problem.	63
5.12	Δ_p and HVR box-plots and statistical summary for DTLZ1 test problem.	64
5.13	Δ_p and HVR box-plots and statistical summary for DTLZ7 test problem.	65
5.14	Δ_p and HVR box-plots and statistical summary for KUR test problem.	67
6.1	Δ_p and HVR box-plots and statistical summary for ZDT1 test problem.	72
6.2	Behavior of the TAMAAL using the G-metric and the $+E2$ method in ZDT1.	73
6.3	Δ_p and HVR box-plots and statistical summary for ZDT2 test problem.	74
6.4	Behavior of the TAMAAL using the G-metric and the $+E2$ method in ZDT2.	74
6.5	Δ_p and HVR box-plots and statistical summary for ZDT3 test problem.	75
6.6	Behavior of the TAMAAL using the RP and the $+E1$ method in ZDT3.	75
6.7	Δ_p and HVR box-plots and statistical summary for ZDT4 test problem.	76
6.8	Behavior of the TAMAAL using the G-metric and the $+E1$ method in ZDT4.	76
6.9	Δ_p and HVR box-plots and statistical summary for ZDT6 test problem.	77
6.10	Behavior of the TAMAAL using the G-metric and the $+E2$ method in ZDT6.	77

6.11	Δ_p and HVR box-plots and statistical summary for DTLZ1 test problem.	78
6.12	Behavior of the TAMAAL using the RP and the +E2 method in DTLZ1.	78
6.13	Δ_p and HVR box-plots and statistical summary for DTLZ7 test problem.	79
6.14	Behavior of the TAMAAL using the G-metric and the +E1 method in DTLZ7.	79
6.15	Δ_p and HVR box-plots and statistical summary for KUR test problem.	80
6.16	Behavior of the TAMAAL using the G-metric and the +E2 method in KUR.	81
A.1	Three layer topology of a RBF	90
B.1	Real Pareto front of the ZDT1 test function and 30,000 random solutions.	96
B.2	Real Pareto front of the ZDT2 test function and 30,000 random solutions.	97
B.3	Real Pareto front of the ZDT3 test function and 30,000 random solutions.	98
B.4	Real Pareto front of the ZDT4 test function and 30,000 random solutions.	98
B.5	Real Pareto front of the ZDT6 test function and 30,000 random solutions.	99
B.6	Real Pareto front of the KUR test function and 30,000 random solutions.	99
B.7	Real Pareto front of the DTLZ1 test function and 30,000 random solutions.	100
B.8	Real Pareto front of the DTLZ7 test function and 30,000 random solutions.	101

List of Tables

4.1	16 different pairs of metamodeling techniques in a bi objective problem.	33
4.2	Statistical results for the accuracy.	41
4.3	Statistical results for RP.	42
4.4	Statistical results for DP.	43
5.1	Summary table for the MASSA on the eight test problems.	66
6.1	Summary table for the TAMAAL on the eight test problems.	81

List of Algorithms

1	General scheme of EAs.	18
2	Outline of RBF algorithm	91

Publications

Gerardo Montemayor-García, Gregorio Toscano-Pulido and Eduardo Rodríguez-Tello, *A Comparison of Meta-modeling Techniques for Multiobjective Optimization*, In 10th International Conference on Artificial Evolution (EA'2011), LNCS volume to appear. Springer-Verlag, Angers, France, 2012.

Gerardo Montemayor-García, Gregorio Toscano-Pulido, *A Study of Surrogate Models for their use in Multiobjective Evolutionary Algorithms*, in 2011 8th International Conference on Electrical Engineering, Computing Science, and Automatic Control (CCE'2011), IEEE press, Mérida, Yucatán, México, October 2011.

Uso de Modelos Surrogados en Algoritmos Evolutivos Multiobjetivo

por

Gerardo Montemayor García

Maestro en Ciencias del Laboratorio de Tecnologías de Información, CINVESTAV-Tamaulipas
Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2011
Dr. Gregorio Toscano Pulido, Director

Los Algoritmos Evolutivos (AEs) son metaheurísticas inspiradas que han sido exitosamente utilizadas para resolver problemas de optimización multiobjetivo (POMs). Cuando estos problemas son computacionalmente costosos, pueden resultar intratables incluso por estas metaheurísticas. Por lo tanto se necesitan estrategias adicionales con el fin de acelerar el tiempo de respuesta de los AEs. Una estrategia comúnmente utilizada ha sido reemplazar el problema original por un modelo surrogado (metamodelo). Sin embargo, a pesar de su éxito, muy pocas comparaciones entre modelos surrogados han sido reportadas en la literatura especializada. En esta tesis, se compararon empíricamente cuatro técnicas de metamodelado con el fin de elegir el enfoque más adecuado para ser combinado un algoritmo evolutivo multiobjetivo. Los resultados de este estudio comparativo hicieron posible proponer un nuevo algoritmo llamado *Metamodeling Assisted Subpopulation Search Algorithm (MASSA)*. MASSA realiza una búsqueda en subpoblaciones, ayudada de modelos surrogados.

Los resultados indican que el enfoque propuesto necesitó un número relativamente bajo para converger al verdadero frente de Pareto de los problemas utilizados, manifestando con esto, que es una alternativa viable para tratar con problemas costosos de optimización multiobjetivo. Sin embargo, este algoritmo carece de una metodología para seleccionar *a priori* el modelo surrogado utilizar. Por ello, se propone una versión mejorada del MASSA que selecciona automáticamente la técnica de metamodelado a utilizar en un momento dado: *The Tune-Adaptive Metamodeling Assisted Algorithm (TAMAAL)*. Los resultados indican que este último enfoque mejora tanto a MASSA, como al NSGA-II en ocho problemas multiobjetivo tomados de la literatura especializada.

On the Use of Surrogated Models in Multiobjective Evolutionary Algorithms

by

Gerardo Montemayor García

Master of Science from the Information Technology Laboratory, CINVESTAV-Tamaulipas
Research Center for Advanced Study from the National Polytechnic Institute, 2011
Dr. Gregorio Toscano Pulido, Advisor

Evolutionary Algorithms (EAs) are bioinspired metaheuristics that have been successfully applied to solve multiobjective optimization problems (MOPs). When these problems are computationally expensive, they can remain intractable even by these metaheuristics. Therefore, it is necessary to employ an additional strategy in order to improve the response time of EAs when optimizing these expensive problems. Replacing the original problem with a surrogate model has been an usual strategy for time reduction. However, despite its success, few comparison among surrogate models for MOPs have been reported in the specialized literature. Additionally, only few works use simultaneously more than one surrogate model in their proposals. In this thesis, four metamodeling techniques were compared empirically with the aim to identify advantages and drawbacks of each metamodeling technique in order to choose the most suitable approach to be combined with multiobjective evolutionary algorithms. Results from this comparison made possible to propose a novel algorithm called Metamodeling Assisted Subpopulation Search Algorithm (MASSA). MASSA performs a subpopulation search enhanced by a surrogate technique. Results indicate that the proposed approach is a viable alternative to handle expensive MOPs since it needed a low number of evaluations in order to converge to the true Pareto front of the tested problems. However, this algorithm lacks of a methodology to select *a priori* the surrogate model to be used. Therefore, an improved version of MASSA that can select automatically the metamodeling technique to be used at a given time is proposed. Results indicate that the latter approach outperformed both, the former approach and the NSGA-II on eight multiobjective problems taken from the specialized literature.

Nomenclature

ANOVA	Analysis of Variance
BLS	Best Local Settings
BOS	Best Overall Setting
DACE	Design and Analysis of Computer Experiments
DM	Decision Maker
DTLZ	Deb-Thiele-Laumanns-Zitzler
EA	Evolutionary Algorithm
EC	Evolutionary Computation
EP	Evolutionary Programming
ES	Evolutionary Strategies
GA	Genetic Algorithm
GD	Generational Distance
HV	Hypervolume
HVR	Hypervolume Ratio
IGD	Inverted Generational Distance
KRG	Kriging
KUR	Kursawe
MA	Memetics Algorithms
MASSA	Metamodel Assisted Subpopulation-based Search Algorithm
MOEA	Multiobjective Evolutionary Algorithm
MOP	Multiobjective Problem
MSE	Mean Squared Error
NN	Neural Network
NSGA	Non-dominated Sorting Genetic Algorithm
pEA	Parallel Evolutionary Algorithm
PF_k	Known Pareto Front
PF_r	Real Pareto Front
pMOEA	Parallel Multiobjective Evolutionary Algorithm
PR	Polynomial Regression
RBF	Radial Basis Functions
RBFNN	Radial Basis Functions Neural Network
SVR	Support Vector Regression
TAMAAL	Tune-adaptive Metamodel Assisted Algorithm
ZDT	Zitzler-Deb-Thiele

1

Introduction

1.1 Introduction

Optimization is the procedure of finding and comparing feasible solutions until no better solution can be found [Deb, 2001]. The solutions are considered good or bad in terms of an objective, which is often the product quality, the cost of fabrication, the shape of some structures, among other factors. Although many applications in the field of optimization only consider a single objective, most real world problems have different objectives that need to be optimized simultaneously ¹ Therefore, when optimizing this sort of problems, the optimum term is redefined to find a set of good trade-off solutions where each one is an efficient solution to the problem.

Since the early 1950s, the Operations Research community has proposed several alternatives for solving multiobjective optimization problems. However, these classical optimization methods usually find (at most) one solution on each simulation run, moreover, when the search space is large, rugged,

¹Also known as multiobjective optimization problems (MOPs).

highly restricted and the functions are not differentiable, these techniques are not effective.

Evolutionary Algorithms² are bio-inspired metaheuristics that have been successfully used to solve such problems. The main advantage of these algorithms is their ability to find multiple efficient solutions in one single simulation run. Thus, EAs are an ideal alternative for solving MOPs. This explains the high number of published and widely used approaches for multiobjective evolutionary algorithms (MOEAs) [Coello Coello, 08]. The main advantage of these techniques is its ability to find solutions close to the real *Pareto front*. However, many real world optimization problems require a considerable number of objective function evaluations to find such solutions. Moreover, in many cases, a single evaluation of the objective function can take considerable time to be completed. For this reason, the EC community has adopted the use of *surrogate models*³ to approximate as much as possible the behavior of a expensive optimization function ($f(x)$), but reducing the computational effort needed to evaluate it. However, choosing correctly the best technique for a specific multiobjective problem is not a trivial task, since the multiobjective function is commonly seen as a black box, and it is not known any information about this function.

1.2 Motivation and hypothesis

A wide variety of algorithms enhanced by approximation models for MOPs have been previously proposed [Voutchkov and Keane, 2006, Emmerich et al., 2006, Knowles, 2006, Chafekar et al., 2005, Diaz-Manriquez et al., 2011]. However, most works do not perform a methodological comparison in order to select the most suitable surrogate model to be combined with.

There are some works that have developed comparative studies among metamodels [Queipo et al., 2005, Santana-Quintero et al., 2010, Diaz-Manriquez et al., 2011]. Nevertheless,

²In this document the term Evolutionary Algorithms (EAs) is used interchangeably with Evolutionary Computation (EC).

³Also known as metamodels, emulators, reduced models, approximate models, and response surface models.

most of such comparisons involve a few techniques and test problems or they select the best surrogate model based upon just one criterion. On the other hand, none of the few works that involved multiple criteria studied both the effects of the dimensionality and the possibility to use different metamodel techniques to approximate independently each of the objective functions. In this thesis work, four metamodeling techniques are compared: Radial Basis Functions (RBF), Support Vector Regression (SVR), Polynomial Regression (PR) and Kriging (KRG) on different aspects such as accuracy, robustness, efficiency, and scalability with the aim to identify advantages and drawbacks of each metamodeling technique in order to choose the most suitable one to be combined with multiobjective optimization evolutionary algorithms.

This thesis work lies on surrogate models that have been used by the multiobjective evolutionary community to speedup the convergence, but maintaining the quality of the solutions. However, nowadays there are a few scientific efforts using simultaneously different strategies of metamodeling techniques to solve computationally expensive MOPs. Moreover, the possibility to use different surrogate models simultaneously in the same MOP has not been addressed previously.

Therefore, the hypothesis that support this research is:

- "It is possible that an evolutionary algorithm enhanced by surrogated models can select a specific metamodeling technique at a given time in order to reduce the number of fitness function evaluations needed to produce quality results."

1.3 Objectives

1.3.1 General objective

To contribute with the state of the art with at least one approach that uses surrogate models to enhance the convergence rate of evolutionary algorithms such that the resulting algorithms require a lower number of fitness function evaluations in order to obtain competitive results.

1.3.2 Particular objectives

- To decrease the number of the real objective-function evaluations without sacrificing the quality of the algorithm.
- To contribute to the state of the art with:
 - A study of the behavior of four well-known surrogate approaches when increasing the number of variables.
 - A competitive multiobjective evolutionary algorithm based on surrogate models.

1.4 Thesis overview

- In Chapter 2 basic concepts of optimality and multiobjective optimization are firstly introduced. After that, both multiobjective performance measures and a performance measure to assess the accuracy of a metamodel are also explained.
- Chapter (3) introduces some Evolutionary Algorithms concepts. Then, a list of the most representative MOEAs is given. Finally, a brief revision of the MOEAs aided by surrogate models state of the art is presented.
- An empirical comparative study of surrogate models for their use with MOEAs is shown in Chapter 4. The four metamodeling techniques are evaluated using several indicators and test problems. Moreover, two novel indicators to evaluate the quality of the application of metamodels to multiobjective are presented.
- In Chapter (5), the Metamodel Assisted Subpopulation-based Search Algorithm (MASSA) is presented. Its main feature is the use of isolated subpopulations that optimize surrogated models. The discussion of the results is given at the end of this chapter.

- The Tune-adaptive Metamodel Assisted Algorithm (TAMAAL) is presented in Chapter 6 as an improvement version of MASSA. In order to select the best metamodeling technique to be used in a given time, an online adaptation method is presented in this chapter.
- Finally, Chapter 7 draws the main conclusions and possible future work.



2

Background

2.1 Introduction

This chapter begins by laying a conceptual and theoretical basis for single and multiobjective optimization. Then, several concepts and definitions related to multiobjective optimization are introduced. Next, a brief review of several performance measures adopted in this thesis work is given.

2.2 Single objective optimization

Everyday problems usually involves the optimization of a single criterion. This criterion is often referred to as product quality, fabrication cost, the shape of some structures, among others. Formally, a generic optimization problem can be stated as:

Definition 1 (Optimization problem). To find the vector \vec{x} which minimizes¹ the function $f(\vec{x})$ subject to the p inequality constraints:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \dots, p \quad (2.1)$$

and the q equality constraints:

$$h_j(\vec{x}) = 0 \quad j = 1, 2, \dots, q \quad (2.2)$$

In other words, the goal is to find among a subspace of Ω feasible solutions, the vector $\vec{x}^* \in \Omega$ that corresponds to the minimum value of the objective function in the feasible region. This vector \vec{x}^* is commonly called *global optimum*.

Definition 2 (Local minimum). Given a function $f : \Omega \subseteq \mathbb{R}^n \mapsto \mathbb{R}$, $\Omega \neq \infty$, a solution $\vec{x}^o \in \Omega$ is called *local minimum point*, if and only if:

$$\forall \vec{x} \in \Omega : f(\vec{x}^o) \leq f(\vec{x}), \quad \text{such as: } \|\vec{x} - \vec{x}^o\| < \epsilon \quad (2.3)$$

where $\epsilon > 0$ and the value $f(\vec{x}^o) > -\infty$ is called *local minimum*.

Definition 3 (Global minimum). Given a function $f : \Omega \subseteq \mathbb{R}^n \mapsto \mathbb{R}$, $\Omega \neq \infty$, for $\vec{x}^* \in \Omega$ the value $f^* = f(\vec{x}^*) > -\infty$ is called *global minimum*, if and only if:

$$\forall \vec{x} \in \Omega : f(\vec{x}^*) \leq f(\vec{x}) \quad (2.4)$$

where vector \vec{x}^* is a *global minimum point*.

¹In this thesis minimization problems are assumed without loss of generality.

2.3 Multiobjective optimization

When a particular problem involves the simultaneous satisfaction of two or more criteria, it can be referred to as multiobjective optimization problem. In words, the multiobjective optimization problem can be defined as [Osyczka, 1984]:

“find a vector of decision variables which satisfies the constraints and optimizes a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term *optimize* means finding such a solution which would give the values of all the objective functions acceptable to the decision maker”

Definition 4 (Multiobjective Optimization Problem (MOP)). Multiobjective optimization is formally defined as: find the decision vector $\vec{x}^* = [x_1^*, x_2^*, \dots, x_n^*]$ that satisfies the m inequality constraints

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \dots, m \quad (2.5)$$

the p equality constraints

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \dots, p \quad (2.6)$$

and optimizes the vector function

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \dots, f_k(\vec{x})]^T \quad (2.7)$$

where $\vec{x} = [x_1, x_2, \dots, x_n]^T$ is the vector of decision variables, n is the number of variables and k is the number of objective functions. Equation (2.5) and (2.6) determine the feasible region $\Omega \subseteq \mathbb{R}^n$ and any decision vector $\vec{x} \in \Omega$ defines a feasible solution of the MOP.

In other words, the goal is determine from among the set Ω the particular set $\vec{x}^* = [x_1^*, x_2^*, \dots, x_k^*]$ which yields the optimum values of all the objective functions. In practice, it is rarely the case where

there is a single point that simultaneously optimizes all the objective functions. Therefore, *trade-off* solutions are sought, rather than a single solution when dealing with multiobjective optimization problems. In order to describe the concept of optimality in multiobjective optimization, some basic concepts are introduced below.

Definition 5 (Pareto dominance). Let \vec{u}, \vec{v} vectors of objective values from feasible solutions with $I = \{1, 2, \dots, k\}$. Then, a vector \vec{u} dominates \vec{v} (denoted by $\vec{u} \preceq \vec{v}$) if and only if \vec{u} is partially less than \vec{v} , i.e, if $\forall i \in I, u_i \leq v_i \wedge \exists i \in I | u_i < v_i$.

Definition 6 (Pareto optimality). A vector $\vec{x}^* \in \Omega$ is a *Pareto optimal* solution if there is no other solution $\vec{y} \in \Omega$ such that $\vec{y} \preceq \vec{x}^*$.

Definition 7 (Pareto optimal set (PS)). The Pareto optimal set is the set of Pareto optimal solutions. In other words, the Pareto optimal set is formed by the corresponding decision variables of the non-dominated solutions.

Definition 8 (Pareto front (PF)). The *image* of the Pareto optimal set are called Pareto front. In this document, PF_r is used to name the real Pareto front² (the optimal solutions), and PF_k to name the known Pareto front (the solutions found by a given algorithm).

Definition 9 (Local Pareto front). A point $\vec{x}^* \in \Omega$ is said to be locally Pareto optimal if and only if there exists an open neighborhood of \vec{x}^* , $\mathbf{B}(\vec{x}^*)$, such that there is no $\vec{x} \in \mathbf{B}(\vec{x}^*) \cap \Omega$ satisfying $\vec{f}(\vec{x}) \prec \vec{f}(\vec{x}^*)$. $\vec{f}(\vec{x}^*)$ is then called locally efficient. The image of the set of locally efficient points is called local Pareto front.

2.3.1 Multiobjective techniques classification

Cohon and Marks [Cohon and Marks, 1975] proposed a classification of multiobjective optimization approaches based on the search process and the moment when the preferences are incorporated

²The terms real Pareto front and true Pareto front are used interchangeably.

by the **Decision Maker (DM)**. The preferences can be incorporated before, during and after the search process:

- **A priori approaches:** these approaches assume that either a certain desired achievable goals or a certain pre-ordering of the objectives can be performed by the decision maker prior to the search. Among the most popular methods are: Global criterion method, Goal Programming, Goal-Attainment Method, Lexicographic Method, Min-Max Optimization, Multi-attribute Utility Theory, Surrogate Worth Trade-off, ELECTRE and PROMETHEE.
- **A posteriori approaches:** these approaches make the search process before the decision maker intervenes, *i.e.* do not require prior preference information from the DM. Some examples are: Linear Combination of Weights and ϵ -Constraint Method.
- **Interactive approaches:** integrate search and decisions making (decide search). These techniques normally operate in three stages: (*i*) find a non-dominated solution, (*ii*) get the reaction of the DM regarding this non-dominated solution, and modify the preferences of the objectives accordingly, and (*iii*) repeat the two previous steps until the DM is satisfied or no further improvements are possible. Among the most popular methods are: Probabilistic Trade-Off Development Method, STEP Method and Sequential Multiobjective Problem Solving Method.

2.3.2 The goal of multiobjective optimization

Since multiobjective optimization problems can be continuous in the objective space, then, the Pareto optimal set can be boundless. In this sense, the decision maker usually requires a small but representative subset of the Pareto front. For this reason, finding a well-distributed set of solutions along the Pareto front is the goal of *a priori* multiobjective optimization.

Figure 2.1 shows the example of the process of buying a vehicle. Here, the buyer tries to minimize the cost and the instability of the vehicle. A cheap car is commonly unstable and a sport car (very

stable) commonly is very expensive. Therefore, there are a set of trade-off solutions between cost and stability. Such solutions are desirable for the DM (in this case the buyer), who must choose a solution according to their budget. The squares in this figure are undesirable because for each of them there is at least another that is cheaper and more stable. Moreover, the pentagons are undesirable because they do not belong to the representative well distributed set of solutions that could be interesting for the buyer.

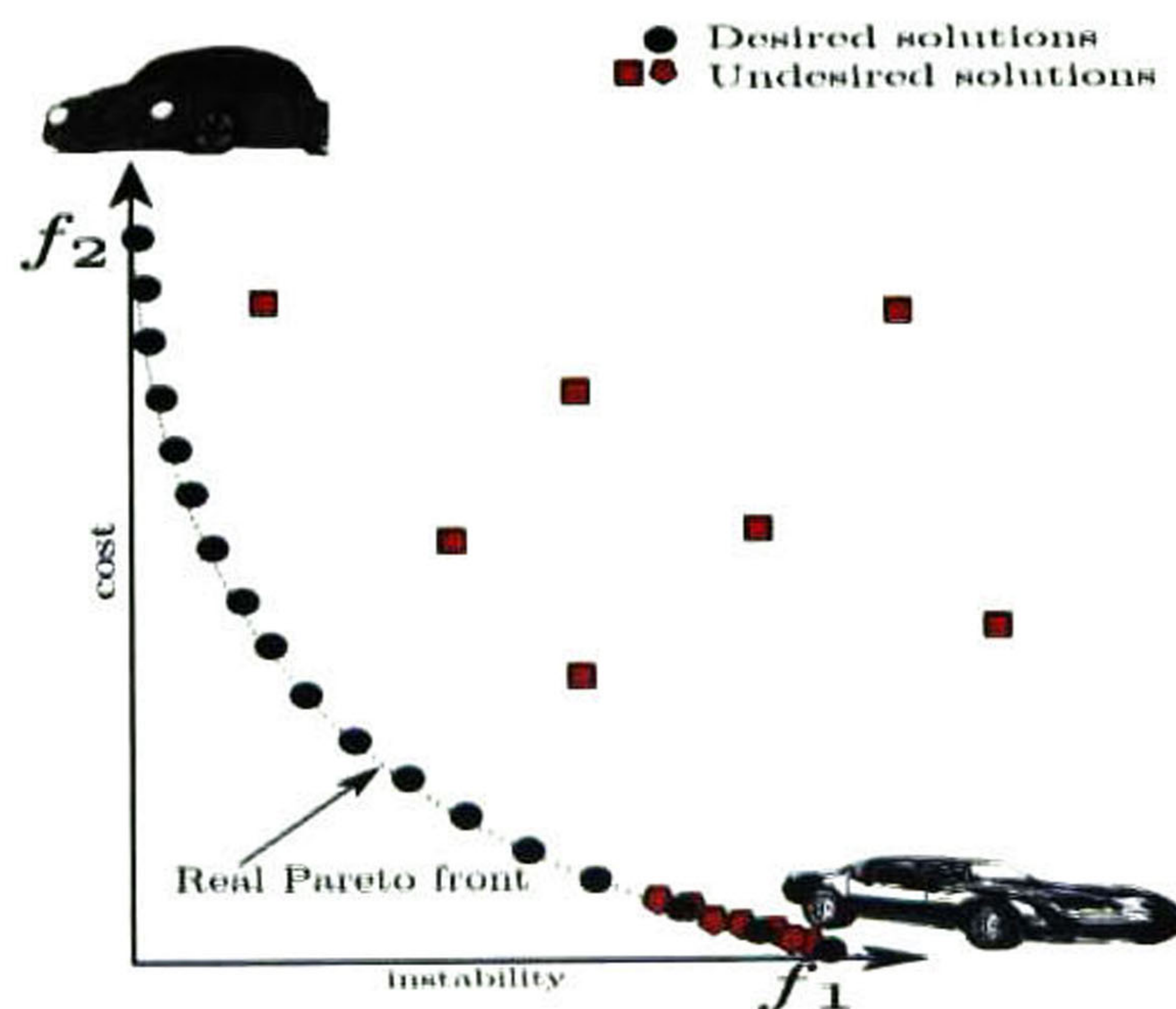


Figure 2.1: Desirable and undesirable solutions in multiobjective optimization.

2.4 Performance measures for MOEAs

As mentioned in Section 2.3.2, both convergence and diversity are quality aspects to measure quantitatively the effectiveness of a multiobjective algorithm. However, when comparing algorithms, the conclusions given for one performance measure are not necessarily the same of another. For that reason, in this thesis work two performance measures used in the multiobjective literature were adopted. One of them is a previously well-known performance measure, while the other one is

recently proposed.

2.4.1 Hypervolume ratio (HVR)

The *Hypervolume ratio (HVR)* performance indicator was proposed by Li *et al.* [Li *et al.*, 2007] as a normalization of *Hypervolume (HV)* performance indicator. The HV corresponds to the non-overlapped volume V of all the hypercubes formed by a reference point r and each solution p_i in the PF_k (see Figure 2.2). The HV is defined as:

$$HV = \bigcup_{i=1}^n V(\vec{p}_i, \vec{r}) \quad (2.8)$$

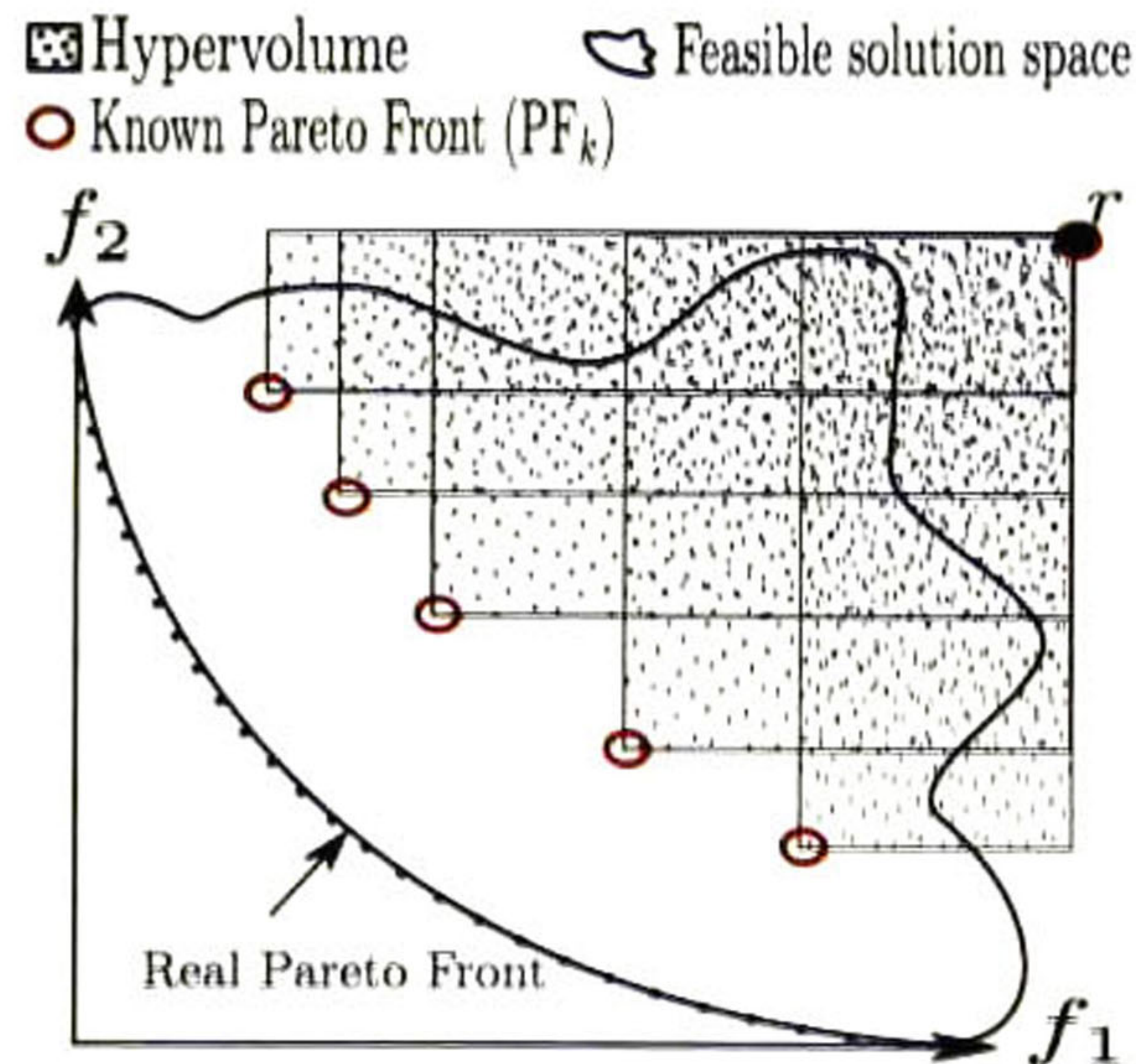


Figure 2.2: Hypervolume of PF_k solutions with a reference point r .

where n is the number of points in PF_k . Therefore, a larger HV reflects better performance for a given algorithm. In order to have the same range of results in every test problem, the Hypervolume ratio (HVR) is used. In the HVR, the closeness to the PF_r is always a value between 0 (when it is far from PF_r) and 1 (when it is over the PF_r). Equation 2.9 shows the mathematical definition of HVR.

$$HVR = \frac{HV(PF_k)}{HV(PF_r)} \quad (2.9)$$

where $HV(PF_r)$ represents the hypervolume of the real Pareto front and $HV(PF_k)$ represents the hypervolume of the known solutions.

2.4.2 Averaged Hausdorff distance (Δ_p)

The concept of *Generational Distance* (GD) was introduced by Van Veldhuizen [Veldhuizen, 1999] to measure how far is the known Pareto front (PF_k) from the real Pareto front (PF_r) (see Figure 2.3(a)). The GD performance measure is stated as follow:

$$GD = \frac{1}{n} \left(\sum_{i=1}^n d_i^p \right)^{1/p} \quad (2.10)$$

where n is the number of elements in the PF_k and d_i is the Euclidean distance between the i^{th} solution in the PF_k and the nearest solution in the PF_r . A GD closer to zero is preferred, *i.e.*, the shorter the GD , the better the accuracy is.

The *Inverted Generational Distance* (IGD) uses the same concept. However, it adopts the PF_r as the baseline solution (see Figure 2.3(b)). Thus, the IGD considers the distribution of elements and their proximity to the PF_r , while the GD only takes into account the closeness to the PF_r (see Figure 2.3).

The *Averaged Hausdorff Distance* (Δ_p) [Schüze et al., 2011] between PF_k and PF_r is composed of slight modifications of the GD and the IGD (GD_p and IGD_p). The modifications use the power mean of the considered distances. Thus, the GD_p performance indicator is stated as follow:

$$GD_p = \left(\frac{1}{n} \sum_{i=1}^n d_i^p \right)^{1/p} \quad (2.11)$$

IGD_p uses the same concept, but with the PF_r as the baseline solution.

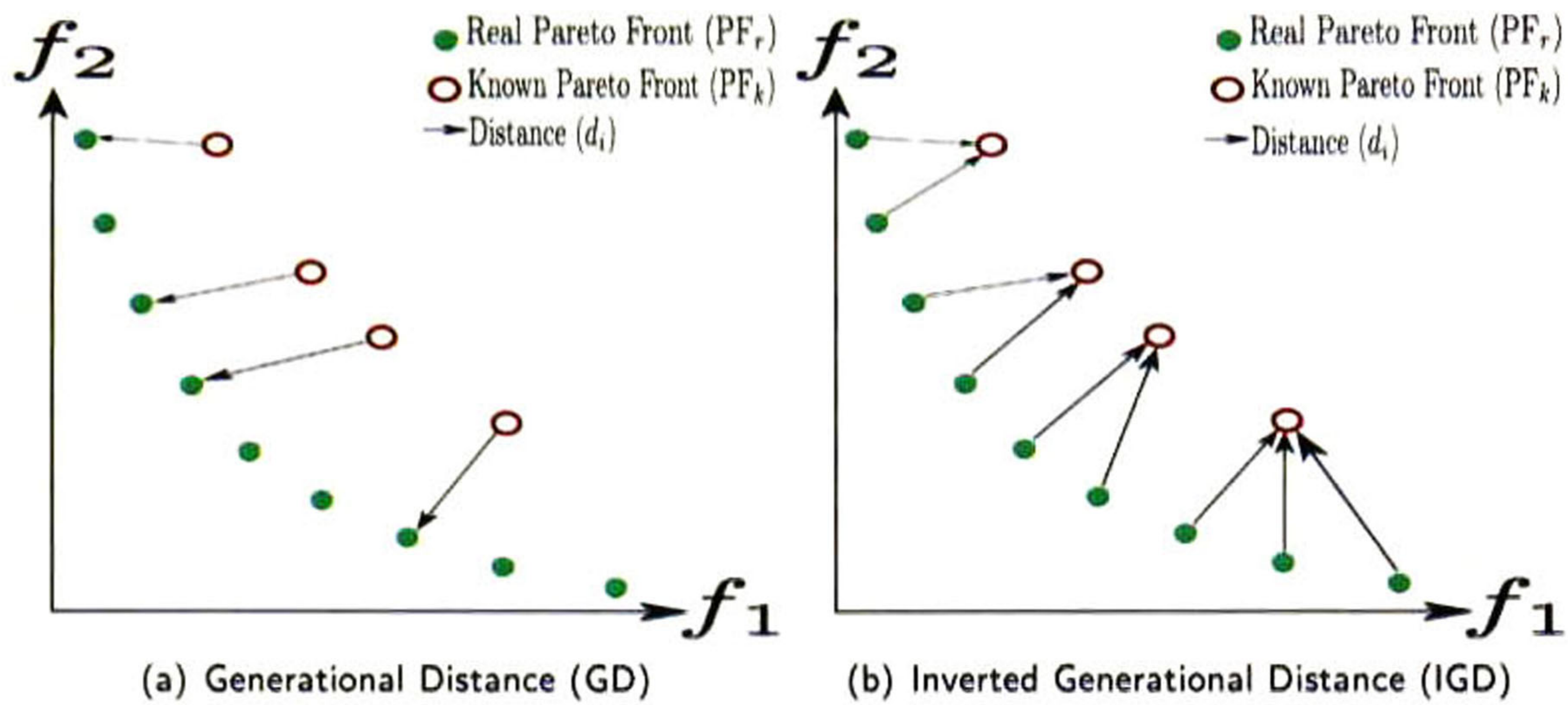


Figure 2.3: Main difference between GD and IGD.

Finally, the Δ_p indicator becomes the maximum of the GD_p and the IGD_p :

$$\Delta_p = \max(GD_p, IGD_p) \quad (2.12)$$

According to the authors, slight modification of both operators (GD_p and IGD_p) leads to more fair indicators. Therefore, Δ_p offers better metric properties than its components GD_p and IGD_p because it defines a semi-metric for all values of p and is even a pseudo-metric in case the magnitudes of the considered sets are bounded.

2.5 Measuring the accuracy of predictors

The G -metric is an indicator used in the context of statistical modeling whose main purpose is the prediction of future outcomes on the basis of other related information [Agterberg, 1984]. This metric gives an indication of how effective a prediction might be, relative to that which could have been derived from using the sample mean alone. In this thesis, the G -metric is aimed to measure the accuracy of a given predictor, and it is applied to the predicted values obtained for each objective of a MOP:

$$G_i = 1 - \frac{\sum_{j=1}^N (y_{ij} - \hat{y}_{ij})^2}{\sum_{j=1}^N (y_{ij} - \bar{y}_i)^2} = 1 - \frac{\text{MSE}}{\text{Variance}} \quad (2.13)$$

where N is the size of the validation data set, \hat{y}_{ij} is the predicted value for the objective i on the input j , and y_{ij} is the real value; \bar{y}_i is the mean of the real values on the objective j . The Mean Square Error (MSE) measures the difference between the estimator and the real value. The variance describes how far the values lie from the mean. In this metric, larger values of G are preferred since this corresponds to a more accurate predictor. Thus, for single objective problems the G_1 is in the range $[0, 1]$. However, for bi objective problems the accuracy is the sum $G_1 + G_2$, therefore is in the range $[0, 2]$.

3

Multiobjective evolutionary algorithms aided by surrogate models

3.1 Introduction

Since optimization problems are not new, it is natural to expect the existence of several optimization techniques. However, when problems present non-differentiable functions, and large, nonlinear, noisy and rugged search spaces, then they can become intractable by these traditional optimization techniques. This issue motivates the use of alternative approaches.

Evolutionary Computation¹ (EC) refers to a set of bio-inspired metaheuristics which have drawn inspiration from natural evolution (Neo-Darwinism) and adaptation that have been successfully applied to solve complex optimization problems. Neo-Darwinism asserts that the history of the vast majority of life is fully accounted for by only a very few statistical processes acting on and within

¹In this document the term Evolutionary Computation (EC) is used interchangeably with Evolutionary Algorithms (EAs).

populations and species [Hoffman, 1989]. These processes are *reproduction, mutation, competition and selection*.

EAs are population-based techniques, *i.e.*, they operate on a set of solutions instead of on only one solution (as traditional optimization methods do). Each individual of its population is evaluated on the objective function with the aim to measure its fitness. Then, a selection mechanism which tends to preserve the fittest solutions is applied. The solutions with the best fitness values have the highest probability of being recombined with other solutions in order to mix information and form new solutions. These new solutions can compete with respect to their parents in order to have a place in the next generation. This process is repeated until a termination criterion is reached. The pseudo-code of an EA is shown in Algorithm 1.

Algorithm 1 General scheme of EAs.

- 1: Initialize population (individuals) with random solutions
 - 2: Evaluate the fitness of each individual
 - 3: **repeat**
 - 4: Select the parents
 - 5: Apply the variation operators
 - 6: Evaluate the new individuals
 - 7: Select the next generation
 - 8: **until** A termination criterion is reached
-

The three main paradigms that form the Evolutionary Computation are: *Evolutionary Strategies (ES)* [Rechenberg, 1965, Schwefel, 1965], *Evolutionary Programming (EP)* [Fogel, 1964] and *Genetic Algorithms (GA)* [Holland, 1975].

3.2 Taxonomy of multiobjective evolutionary algorithms

The first indication of the possibility of using evolutionary algorithms to solve MOPs appeared in the doctoral thesis of Rosenberg in 1967 [Rosenberg, 1967] in which, however, the algorithm was reformulated as a problem of a single objective and it was solved with a GA.

David Schaffer is commonly regarded as the first to design a MOEA in the mid-1980s. His proposal was called *Vector Evaluated Genetic Algorithm (VEGA)* [Schaffer, 1985], consisting of a simple AG with a modified selection mechanism.

In the following years, there were a lot of techniques, which, as mentioned Coello [Coello Coello, 08], were divided into two generations. The first generation emphasized simplicity and among the most representative algorithms are:

- *Non-dominated Sorting Genetic Algorithm (NSGA)* [Srinivas and Deb, 1994].
- *Niched-Pareto Genetic Algorithm (NPGA)* [Horn et al., 1994].
- *Multiobjective Genetic Algorithm (MOGA)* [Fonseca and Fleming, 1993].
- *Niched-Pareto Genetic Algorithm (NPGA 2)* [Erickson et al., 2001].

While the second-generation emphasizes on efficiency, in addition this generation usually uses a mechanism that retains the elitist non-dominated global solutions generated by the MOEA. Among the most representative approaches in the second generation are:

- *Strength Pareto Evolutionary Algorithm (SPEA)* [Zitzler and Thiele, 1998].
- *Strength Pareto Evolutionary Algorithm 2 (SPEA-II)* [Zitzler et al., 2002].
- *Pareto Archived Evolution Strategy (PAES)* [Knowles and Corne, 1999].
- *Non-dominated Sorting Genetic Algorithm II (NSGA-II)* [Deb et al., 2000].
- *Pareto Envelope-based Selection Algorithm (PESA)* [Corne et al., 2000].
- *Pareto Envelope-based Selection Algorithm II (PESA-II)* [Corne et al., 2001].
- *Micro-Genetic Algorithm for Multiobjective Optimization (micro-GA)* [Coello Coello and Toscano Pulido, 2001].

- *Micro-Genetic Algorithm 2 (micro-GA2)* [Toscano Pulido and Coello Coello, 2003].

One successfully used second-generation algorithm in multiobjective optimization is the NSGA-II. It remains as one of the most competitive multiobjective evolutionary algorithms known to date. A more detailed description of the procedure of NSGA-II is provided in the next section.

3.3 A fast elitist non-dominated sorting genetic algorithm (NSGA-II)

Since this algorithm was selected to be used as baseline in the development of the proposed algorithms, then it is described below:

This algorithm was proposed by Deb *et al.* [Deb *et al.*, 2000]. Unlike other methods, which use only one elitist preservation strategy, the NSGA-II also uses an explicit mechanism for preserving diversity. This algorithm has little resemblance to its predecessor (the NSGA [Srinivas and Deb, 1994]), but the authors kept the name to highlight its origin. This algorithm has a complexity $O(MN^2)$ with M objectives and a population size N . The different modules that form part of the NSGA-II are briefly described below. However, for a better understanding of the algorithm see [Deb *et al.*, 2000].

3.3.1 A fast non-dominated sorting approach

The non-dominated sorting ranks the population in different fronts, where the first one contains only non-dominated solutions. The second front is compound by solutions dominated only by the first front, and so on until all solutions are sorted. An example of the non-dominated sorting is given in Figure 3.1. Where, the solutions have a rank equal to the front they belong. For example, the solutions from the front 1 have a $rank = 1$ (see [Deb *et al.*, 2000] for more details).

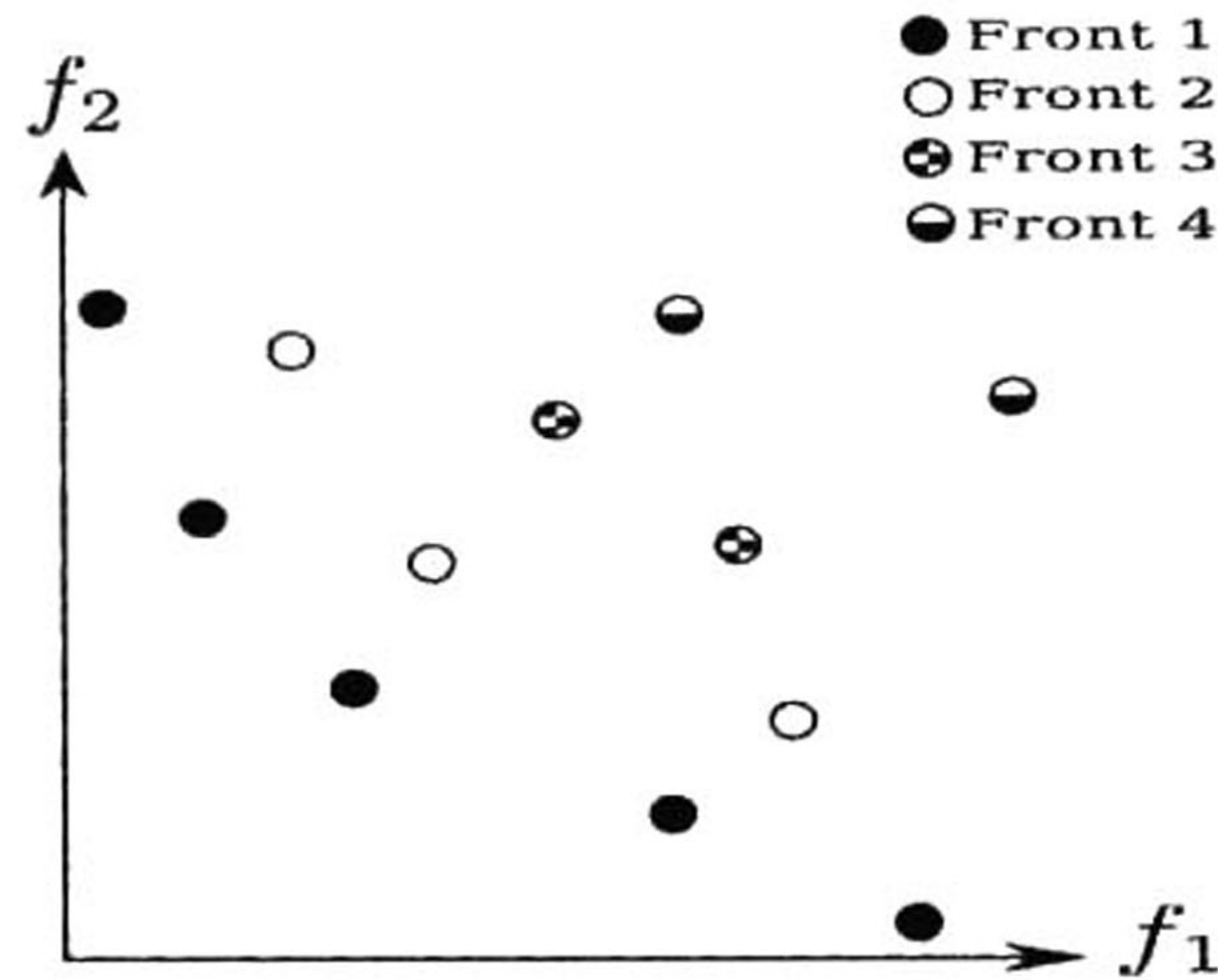


Figure 3.1: Non-dominated sorting.

3.3.2 Crowding distance

To perform an estimation of the density of solutions surrounding a particular solution i in the population, the average distance of two solutions on either side of solution i along each of the objectives is taken. This quantity is called crowding distance, and serves as an estimation of the perimeter of the cuboid formed by using the nearest neighbors as the vertices. In Fig. 3.2, the crowding distance of the i^{th} solution in its front (marked with filled circles) is the average side-length of the cuboid (shown with dashed boxes).

3.3.3 Crowded comparison operator (\prec_n)

The crowded comparison operator (\prec_n) compares two solutions and returns the winner of the tournament. It assumes that every solution i has two attributes:

1. A non-domination rank r_i in the population.
2. A local crowding distance (d_i) in the population.

Based on these two attributes, it is said that $i \prec_n j$ if: $(r_i < r_j)$ or $(r_i = r_j$ and $d_i > d_j)$.

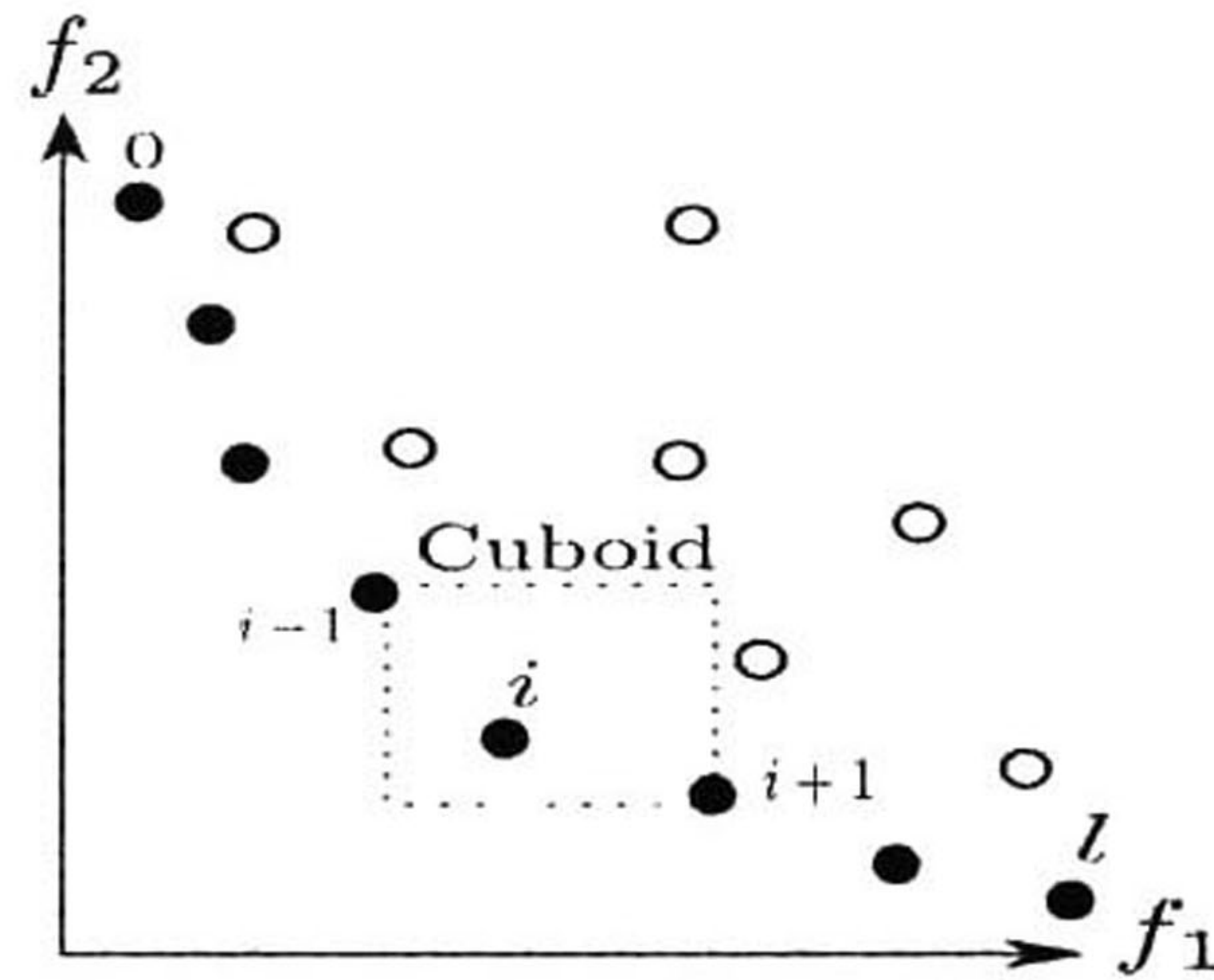


Figure 3.2: The crowding distance calculation.

The first condition ensures that the chosen solution lies on a better non-dominated front. While the second untie when both solutions belong to the same non-dominated front based on their crowding distance. The one residing in a less crowded area (with a larger crowding distance d_i) will be selected.

3.3.4 The main loop

Initially, a random parent population P_0 is created. Then, the population is sorted based on non-dominance. Afterwards, each solution is assigned to a fitness (or rank) equal to its non-dominance level (1 is the best level, 2 is the next-best level, and so on). Binary tournament selection, SBX recombination, and polynomial mutation operators are used to create a offspring population Q_0 of size N . Since elitism is introduced by comparing current population with previously found best non-dominated solutions, the procedure is different after the initial generation. The elitism procedure for the t^{th} generation starts combining the two populations to form R_t of size $2N$. Then, the **non-dominated sorting** is used to classify the entire population R_t (see Section 3.3.1). Once the population is sorted, the new population is filled by solutions of different non-dominated fronts, one at a time. The procedure starts with the best non-dominated front and continues with solutions of the second non-dominated front, and so on. Since the overall population size of R_t is $2N$, not

all fronts may be accommodated in N slots available in the new population. Therefore, the worst fronts are deleted. When the last allowed front is being considered, there may exist more solutions in the last front than the remaining slots in the new population. This scenario is illustrated in Figure 3.3. Instead of arbitrarily discarding some members from the last acceptable front, the solutions in this front are sorted using the crowded comparison operator (see Section 3.3.3), such that, the best positioned solutions will fill N individuals.

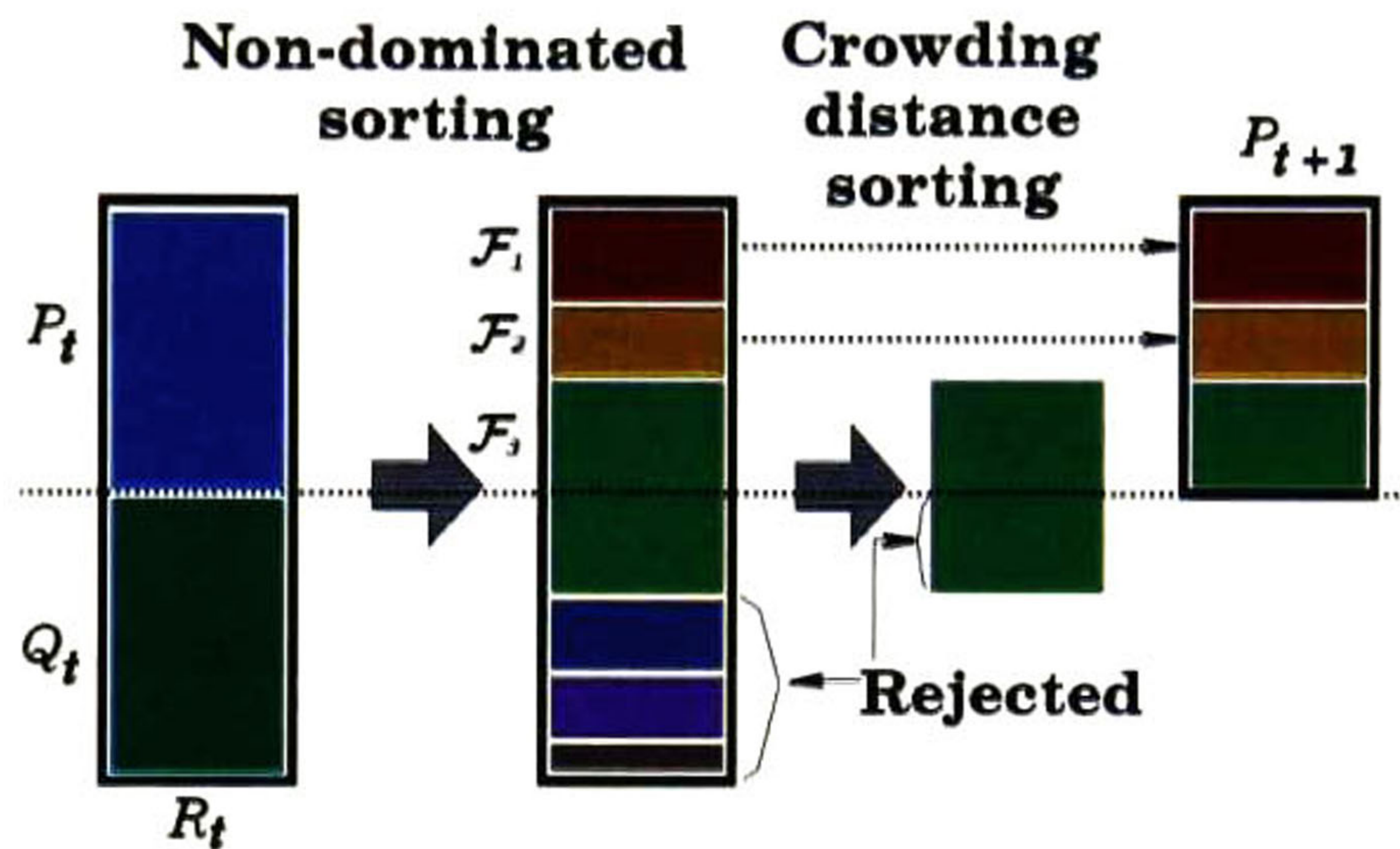


Figure 3.3: NSGA-II procedure.

Finally, binary tournament selection, recombination, and mutation operators are used again to create a offspring population Q_{t+1} , finishing the current generation.

3.4 Surrogate models

In several real world optimization problems, a single evaluation of the objective function can require long time to be evaluated. For this reason, the EC community has adopted the use *metamodels* in order to approximate as much as possible the behavior of a computationally expensive function ($f(x)$), but reducing the computational effort needed to evaluate it.

A metamodel is an approximation of a simulation used to construct simpler and lower computational cost models; if the original simulation is represented as $f(x)$, and the surrogated model is represented as $f'(x)$, then, $f'(x) = f(x) + e(x)$, where $e(x)$ is the approximated error. The internal behavior of $f(x)$ is not necessary to be known (or understood), only the input/output behavior is important². A model is constructed based on modeling the response of the simulator to a limited number of intelligently chosen data points. Metamodels generate simple models that capture relations between the relevant information of the input and output variables and not in the underlying process. Nowadays, there are a wide variety of techniques to construct metamodels. Among the most populars are *Radial Basis Function Neural Network (RBFNN)*, *Support Vector Machines for Regression (SVR)*, *Polynomial Regression (PR)* and *Kriging-DACE Models (KRG)*. For a more complete understanding of these surrogate models, please refer to Appendix A.

3.5 State of the art

In order to maintain the number of evaluations as low as possible, it is possible to combine the iterative search of EAs with surrogate models, where the objective functions need to be modeled as fitting functions through evaluated points during the optimization process. These models are used to predict the value of future search points. A description of several MOEAs enhanced with surrogate models is presented below:

- In 2005, Chafekar *et al.* proposed a variation of the GADO [Rasheed, 1998] enhanced with metamodels called OEGADO [Chafekar *et al.*, 2005]. To deal with more of one goal, the proposed algorithm executes an independent GA on each objective, and exchange information at certain intervals in order to find the set of trade-off solutions. Least-squares approximation functions were used in this work. However, only this approximation method was tested. On the other hand, since each objective function is assessed by an independent GA, then this

²what is known as *behavior modeling* or *black box modeling*

proposal can be implemented easily in a parallel architecture. For its validation, the algorithm was compared with respect to two MOEAs representative of the state of the art (NSGA-II and ϵ -MOEA). Moreover, six problems with two and three objectives with a fixed number of variables were used in order to validate its results. The proposed approach could produce (with fewer iterations) a more complete and distributed Pareto front than GADO and NSGA-II.

- In 2006, Voutchkov and Keane [Voutchkov and Keane, 2006] proposed an approach that connects the NSGA-II with independent metamodels for each objective function. The proposed algorithm executes the NSGA-II for several generations using the metamodel functions, after that, the real function is used in order to perform the selection and to update the metamodels. In this paper, twelve response surface methods were compared (including *RBF*, *Kriging* and *PR*). The main advantage of this approach is that each objective could, in theory, be modeled by a different type of response surface method. However, no comparisons with different metamodeling techniques on each objective were performed *i.e.*, the user has to pre-fix which models will be used at any given time. A study of 3 MOPs was performed. However, these problems only were used from 2 to 10 variables.
- Emmerich *et al.* [Emmerich *et al.*, 2006] proposed in 2006 a method where a new point for evaluation is chosen based on both, its predicted value and the confidence in such predictions. Their approach is a modification of the NSGA-II and uses several criteria for the selection of the points that will be evaluated in the real function. The authors compared four different evaluation criteria in problems with two and three objectives. Moreover, six test problems were tested using a fixed number of variables. Authors found improvements in all cases when comparing their results with respect to those obtained by the standard NSGA-II. The proposed algorithm depends on the confidence estimation provided by Kriging models. Thus, no other surrogate models were tested.
- Also in 2006, Knowles proposed the *Pareto Efficient Global Optimization (ParEGO)*

[Knowles, 2006] as a new alternative to the Efficient Global Optimization (EGO) [Jones et al., 1998]. ParEGO uses an approach of Analysis and Design of Computer Experiments (DACE) to model the objective functions. Initialization procedure uses latin hypercube method and its learning approach uses a Gaussian process model, which is updated after every evaluation. To create the Pareto front, ParEGO uses a series of vectors to assign weighted values to the objective functions according to their importance. On each iteration of the algorithm, a new candidate point is determined by a procedure called *expected improvement maximization*. This approach was tested on nine MOPs with 2 and 3 objectives. However, the MOPs tested had relatively few decision variables (from 2 to 8). Results showed that ParEGO could reduced the number of evaluations. Furthermore, any other metamodeling technique can not be plugged since ParEGO uses the *estimated error* (provided by the DACE) to select the new points to evaluate.

- Deb and Nain proposed the NSGA-II-ANN [Deb and Nain, 2007] in 2007, where they enhanced the NSGA-II with ANN. In this case, they used the *standard error back-propagation algorithm with sigmoidal activation function* to approximate the problem. The main procedure intersperses both, real function evaluations and metamodel evaluations. The NSGA-II-ANN procedure was tested on three test problems and a couple of real-world problems. They concluded that NSGA-II-ANN simulations could successfully save from 25% to 62% of the evaluations needed by the original NSGA-II using a fixed number of variables. Also, in this work, the authors compared two different training models: *Incremental Training NN* and *Batch Training NN*. Results indicated that the second approach behave the best.
- In 2008, M. Li et al. proposed the *Kriging assisted MOGA (K-MOGA)* [Li et al., 2008]. The main feature of this algorithm is the inclusion of a method for avoiding the real simulation where the predicted error is acceptable for a point, otherwise, the real simulation is carried out. The metamodeling technique used in this work was Kriging. Seven numerical and engineering

examples with different degrees of difficulty were used in the experiments. Results showed that their proposal could reduce the number of real simulation calls. On the other hand, the K-MOGA depends on the confidence level provided by the Kriging method. Therefore, can not be plugged any other metamodeling technique in their proposed algorithm.

- In 2011, Diaz-Manriquez *et al.* proposed a *Surrogate-based Intelligent Variation Operator (SIVO)* [Diaz-Manriquez *et al.*, 2012] for Multiobjective Optimization. The SIVO operator starts selecting a non-dominated solution to outperform, and three new solutions are searched trying to locate their positions in the variable space. Therefore, a local surrogate model is built and a new optimization algorithm is launched, which tries to find the surrogated solutions which are closest to the three previously proposed points. The final surrogate solutions are evaluated with the real function with the aim that the new solutions found can improve the original ones. Moreover, the authors proved the performance of SIVO coupling it to both a deterministic algorithm and to the NSGA-II. Only the Kriging surrogate model was used in this work, although any other can be used. On the other hand, five problems were used in order to prove the functionality of their proposal.
- The term Memetics Algorithms (MA) refers to: evolutionary algorithms that apply a local search process to refine solutions (for further exploration refer to [Moscatto, 2003]). Some Memetic Multiobjective Algorithms assisted by Metamodels are also presented below:
 - In 2009, Georgopoulou and Giannakoglou [Georgopoulou and Giannakoglou, 2009] proposed a Multiobjective Metamodel Assisted Memetic Algorithm with Strength-based Local Refinement. In their work, the metamodels perform a dual role (into global and local search). During the global search a RBF approach is trained in order to predict values for the children population. Then, the children population is sorted and the best performed individuals are re-evaluated but using the real function. Then, a local search procedure that maximizes the strength of a solution (*i.e.*, the number of solutions that the

current solution dominates to.) is performed. This latter procedure uses gradient-based descent method and it also uses the RBF approach. The candidates with higher strength must be selected with priority for their evaluation on the real function. The authors demonstrated the effectiveness of their algorithm in three mathematical test problems and two engineering applications. In this approach, the authors did not prove any other metamodeling technique.

- In 2010 Zapotecas and Coello [Zapotecas Martínez and Coello Coello, 2010] proposed a multiobjective memetic algorithm that uses SVM as local search. The algorithm uses a series of vectors to assign weighted values to the objective functions according to their importance. Then the approach uses the *Hooke-Jeeves method* to minimize each problem defined by the weighted vectors (evaluating in the SVM). After that, the *K*-means algorithm (with $K = \text{population size}$) is used to reduce the non-dominated solutions. The authors tested it proposal on problems of moderate dimensionality (10 to 30 decision variables). Results indicate that the proposed approach could obtain competitive results with respect to the NSGA-II using 1,000 evaluations of the real function.
- Since parallelization is a common strategy to tackle expensive function evaluation problems, in EC area has emerge parallel EA (pEA) [Nowostawski and Poli, 1999]. The objective of a pEA is to find solutions of equal or higher quality in less time than its serial counterpart. The following proposals are examples of parallel Multiobjective Evolutionary Algorithms (pMOEAs) enhanced by surrogate approaches:
 - In 2006, Ray and Smith proposed a *Surrogate Assisted Parallel Multiobjective Evolutionary Algorithm* [Ray and Smith, 2006]. The main motivation of the authors was to build a robust approach that maximizes the performance while minimizing the variance of its results. The algorithm uses a neural network of radial basis functions to create metamodels. The algorithm was implemented on multiple processors using a master-

slave topology. Two design optimization problems were solved to demonstrate the competitiveness of the algorithm. Where, the proposal demonstrates savings in the number of real function evaluations. However, no other metamodeling techniques were used. The algorithm was tested with a fix number of evaluations.

- Syberfeldt *et al.* proposed in 2008 the *MOPS-EA (Multiobjective Parallel Surrogate-Assisted Evolutionary Algorithm)* [Syberfeldt *et al.*, 2008]. This algorithm is based on a steady state design. It uses a surrogate model to identify promising candidate solutions and to filter out dominated solutions. To address the uncertainty associated with the approximation of the evaluations in the surrogate model, the algorithm uses a method in which surrogate objective values assigned to offspring are adjusted to consider the error of the surrogate. An Artificial Neural Network was used as metamodeling technique. On the other hand, the algorithm was evaluated on the ZDT benchmark functions with competitive results comparing with a Neural Networks assisted NSGA-II. A drawback in this paper is that the algorithm was tested with a fixed number of evaluations (3000). Moreover, the same metamodeling technique was used for all objectives.

Despite the suggested benefits of surrogate models, little effort has been placed by the MOEA community. Therefore, deeper theoretical and empirical studies about the behavior of the combination of metamodels and EAs are needed. In this direction, it is necessary to address studies about the behavior of MOEAs enhanced with metamodels taking into account the effects of the dimensionality. These studies will allow the designing of new metaheuristics that behave better on a wider number of problems.

4

A study of surrogate models for their use with multiobjective evolutionary algorithms

4.1 Introduction

Since surrogate models are not new, a wide variety of optimization techniques that use them have been previously proposed [Voutchkov and Keane, 2006, Emmerich et al., 2006, Knowles, 2006, Chafekar et al., 2005]. However, few works have performed a comparison among metamodels in order to find the most suitable approach to be used with the application at hand. Most of these works have focused just on single objective problems [Santana-Quintero et al., 2010, Queipo et al., 2005, Jin et al., 2000, Diaz-Manriquez et al., 2011], while only few works on MOPs [Santana-Quintero et al., 2008, Fang et al., 2005]. With respect to the latter subject, Fang *et al.* [Fang et al., 2005] compared metamodels using a small number of techniques and test problems and Santana *et al.* [Santana-Quintero et al., 2008] selected the best surrogate model based in just one

criteria. On the other hand, most of the works ignored the effects of the dimensionality and did not analyze the possibility to combine different techniques, in order to approximate independently each of the objective functions of a MOP.

This chapter presents a study of surrogate models for their use with multiobjective evolutionary algorithms, inspired by the study in single objective problems presented in [Diaz-Manriquez et al., 2011].

The remain of this chapter is organized as follows: the methodology used is described in Section 4.2. After that, results from the application of the methodology are shown in Section 4.5. Finally, Section 4.6 addresses conclusions from this study.

4.2 Methodology

To evaluate the performance of different techniques on MOPs, eight scalable unconstrained continuous bi-objective test problems were selected from the specialized literature (see Appendix B.1). Each objective of each test function will be approximated independently, such that, the first objective will be approximated with a metamodel \mathcal{A} and the second objective with a metamodel \mathcal{B} , where \mathcal{A} and \mathcal{B} can be any of RBF, SVR, PR or KRG.

Also, in order to allow a quantitative assessment of the performance of each metamodeling technique, five issues were considered: accuracy, robustness, efficiency (both, for training and prediction), scalability and the suitability to be combined with an evolutionary approach.

The methodology of this experiment is explain as follows:

- First, in order to void penalize the behavior of the metamodeling techniques adopted in this thesis work, a parameter setting for each approach was designed (see Section 4.3).
- Then, a full factorial design of the combination of the studied approaches were perform, the 16 pairs are shown in Table 4.1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Objective 1	RBF	RBF	RBF	RBF	SVR	SVR	SVR	SVR	KRG	KRG	KRG	KRG	PR	PR	PR	PR
Objective 2	RBF	SVR	KRG	PR	RBF	SVR	KRG	PR	RBF	SVR	KRG	PR	RBF	SVR	KRG	PR

Table 4.1: 16 different pairs of metamodeling techniques in a bi objective problem.

- After that, each combination $(\mathcal{A}, \mathcal{B})$ was executed following the steps shown below:
 1. Create a training data set with latin hypercubes [McKay et al., 1979] of size 100 (since EAs typically handle this population size).
 2. Train the two objectives using $(\mathcal{A}, \mathcal{B})$, respectively.
 3. Create the validation data set with a latin hypercube of size 500.
 4. Predict the validation data set with metamodels $(\mathcal{A}, \mathcal{B})$.
 5. Compute the performance indicators for the adopted criteria (see descriptions in Section 4.4).
 6. Repeat 31 times the steps 1 to 5 and compute the average.
- Perform a statistical analysis¹ [Demsar, 2006] for the adopted criteria according the next steps:
 1. For each pair of combinations perform the *Shapiro-Wilk* test to verify if both sets follow the Gaussian distribution.
 2. If both sets follow Gaussian distributions, then verify the homogeneity of their variances using *Levene* test.
 3. If results from both combinations fulfill the Levene test, then, apply the *ANOVA* test, otherwise, *Welch* test is performed.
 4. If *Shapiro-Wilk* is not satisfied, then the non-parametric *Kruskal-Wallis* test is used to compare the differences between the two sets.

¹Note: For all statistical tests a confidence level of 95% was considered.

Summary tables of results were built using this statistical analysis. In this tables, metamodeling techniques are compared each other (the 16 pairs of metamodeling techniques²). The tables were filled as follows: each pair is compared with each other following the statistical analysis. If one pair is statistically different and has a better value for the adopted criterion at hand, then the counter of wins is incremented in one, otherwise not increment is performed at all.

4.3 Parameter setting

KRG : Self-tuning is an important advantage of KRG method, since it can adjust its two parameters (θ 's and p 's). To perform this adjustment, KRG uses the *maximum likelihood* [Jones, 2001].

PR : PR does not need any parameter tuning at all. However, that is not the case for RBF and SVR.

RBF : This technique needs the number of centers for the hidden layer as a parameter (this parameter accepts values equal or greater than 3, therefore, this study analyzes values from 3 to 100).

To have a statistical validation, 31 executions for each value of the parameter were executed on five single objective test functions (Rosenbrock, Rastrigin, Griewangk, Sphere, and Ackley). For this experiment, 100 solutions for training and other 200 solutions for prediction were used. Since the aim is to identify if there is a single value for the parameter that can produce an outstanding behavior on RBF, two experiments were carry on. Both experiments will evaluate the performance of RBF while such parameter is modified. Therefore, the performance of each value will be evaluated using the G performance measure. The first experiment will obtain the best performance of RBF on each test function. The values that make RBF to perform the best on each test function will be referred as Best Local Setting (BLS). The second experiment

²For reasons of space in the tables, the following abbreviations were used: $K = KRG$, $P = PR$, $R = RBF$ and $S = SVR$

will evaluate the behavior induced by the parameter but in all the test functions. This latter approach will be referred as Best Overall Setting (BOS). Figure 4.1 shows the results produced by BLS and BOS using the $G - Metric$ (see Section 2.5) for RBF. From these results, it seems obvious that both sets behaved similar. However, for simplicity, BOS was preferred. The values of the parameter obtained in the BOS for the RBF approach was: 38 centers.

SVR : For the parameter setting of this approach a methodology similar than the one used for RBF was used. However, in this case there are three parameters than need to be fine-tune: C , γ and ϵ , their ranges of values are $[2^{-5}, 2^{15}]$, $[0.1, 2]$ and $[2^{-10}, 2^5]$, respectively. These ranges of values were taken from the specialized literature. In order to perform a methodological comparison, a discretization of each parameter were carry on. For this sake, 20 possible values uniformly spaced from each other were performed. Then, a full factorial design executing each combination of parameters ($20 \times 20 \times 20$) with every test function adopted were performed. Figure 4.1 shows the results produced by BLS and BOS using the $G - Metric$ for SVR. From these results, it seems obvious that both sets behaved similar. However, for simplicity, the BOS was used in the rest of the experiments in this study. The SVR's BOS combination was $C = 12072.4407895$, $\gamma = 0.1$, and $\epsilon = 0.000976$.

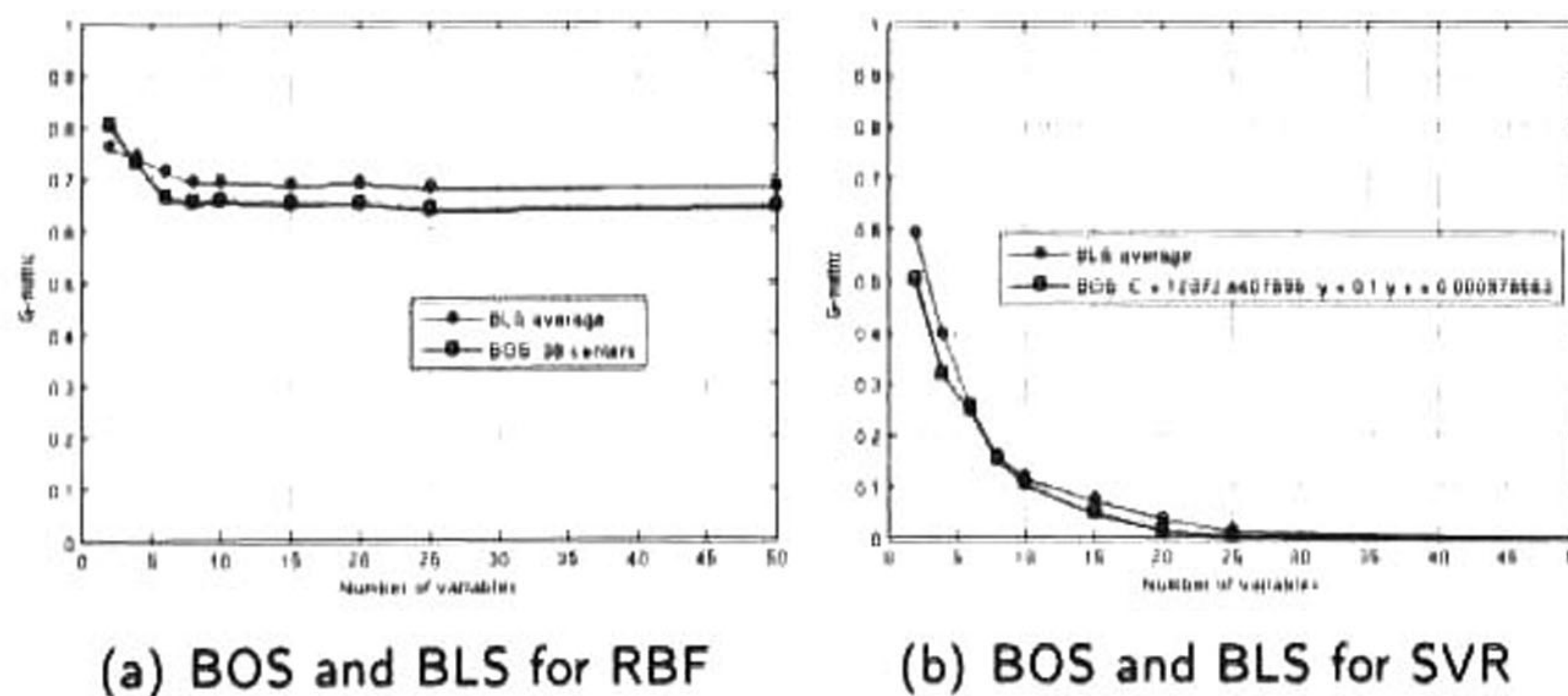


Figure 4.1: Average accuracy with the BOS and BLS for five single objective problems.

4.4 Adopted criteria

In order to assess the different techniques, five different criteria were measured:

- **Accuracy:** is the ability of a technique to make predictions close to the values given by the real system. In order to assess this criterion, the *G-Metric* was used (see Section 2.5).
- **Robustness:** refers to the ability of a technique to achieve a good accuracy on different test problems. Then, the robustness of a metamodel will be given by its average behavior for all adopted test cases.
- **Scalability:** is the ability of a technique to achieve a good accuracy taking the dimensionality into account (*i.e.*, a metamodels is more scalable if it presents good accuracy values for every problem size). To measure this criterion, the following problem sizes: $v = \{2, 4, 6, 8, 10, 15, 20, 25, 50\}$ were used.
- **Efficiency:**
 - *to train:* this criterion refers to the computational effort required for the technique to build the metamodel. For this criterion, the time needed to train a specific metamodel with 100 solutions was measured.
 - *to predict:* refers the computational effort required for the technique to predict responses to new entries. For this criterion, the time needed by a particular approach to predict 500 solutions was measured.
- **Suitability:** this criterion is defined as the degree of difficulty needed to optimize a pair of metamodels (using one metamodel for each objective function). For measuring this criterion, two approaches were proposed. The first one refers to the way in which some evolutionary multiobjective approaches (MOEAs) compare solutions. Since Pareto dominance is a common

used approach, then, this approach was selected. Therefore, the first experiment tries to identify which is the pair of metamodels that preserves better the dominance rank in a bi objective function. For this purpose, the metamodels will be trained first. Then, N well distributed points will be produced. After that, each pair of the N points will be compared using dominance rank in both, the surrogated functions and the real system. The pair of metamodels that preserves better the relation of Pareto dominance will be the most suitable one. In order to measure this criterion, the degree of ranking preservation³ (RP) is used as the ability of the metamodels to maintain the same rank of the solutions with respect to the original MOP. A combination of metamodels \vec{f}^j has a perfect **multiobjective RP** under the original functions \vec{f} if:

$$\begin{aligned}
 & (\vec{f}(\vec{x}) \prec \vec{f}(\vec{y}) \wedge \vec{f}^j(\vec{x}) \prec \vec{f}^j(\vec{y})) \\
 \forall x, y \in \mathcal{F} : & \quad \vee (\vec{f}(\vec{y}) \prec \vec{f}(\vec{x}) \wedge \vec{f}^j(\vec{y}) \prec \vec{f}^j(\vec{x})) & (4.1) \\
 & \quad \vee ((\vec{f}(\vec{x}) \not\prec \vec{f}(\vec{y}) \wedge \vec{f}(\vec{y}) \not\prec \vec{f}(\vec{x})) \wedge (\vec{f}^j(\vec{x}) \not\prec \vec{f}^j(\vec{y}) \wedge \vec{f}^j(\vec{y}) \not\prec \vec{f}^j(\vec{x})))
 \end{aligned}$$

where \mathcal{F} is the feasible region of the problem. Therefore, this performance indicator can be defined as follows:

$$RP = \left(\sum_{i=1}^N \sum_{j=i+1}^N h(i, j) \right) / \binom{N}{2} \quad (4.2)$$

where N refers to the number of solutions used to validate the model. In this thesis, $N = 500$ solutions were used for each size of all test problems. And $h(i, j)$ is:

³The ranking preservation indicator was initially proposed in [Diaz-Manriquez et al., 2011], in order to assess the predictions in single objective problems.

$$h(i, j) = \begin{cases} \mathbf{if}((\vec{f}(i) \prec \vec{f}(j) \wedge \vec{f}'(i) \prec \vec{f}'(j))) \\ 1 \quad \vee (\vec{f}(j) \prec \vec{f}(i) \wedge \vec{f}'(j) \prec \vec{f}'(i)) \\ \quad \vee ((\vec{f}(i) \not\prec \vec{f}(j) \wedge \vec{f}(j) \not\prec \vec{f}(i)) \wedge (\vec{f}'(i) \not\prec \vec{f}'(j) \wedge \vec{f}'(j) \not\prec \vec{f}'(i)))) \\ 0 \quad \mathbf{otherwise} \end{cases} \quad (4.3)$$

The second approach refers to the fact that when some MOEAs perform the optimization process on metamodels, they commonly obtain the non-dominated solutions in the metamodel, then, such solutions are evaluated in the real MOP. Finally, if solutions are non-dominated with respect to an elitist population, then, they are retained, the remain solutions are discarded. Therefore, it is necessary to know the percentage of non-dominated solutions on the real MOP that are also non-dominated on the metamodels (dominance preservation). A combination of metamodels \vec{f}' has a perfect **dominance preservation** under the original functions \vec{f} if:

$$\forall x \in \mathcal{F} : \vec{f}(x) \text{ is non-dominated} \wedge \vec{f}'(x) \text{ is non-dominated} \quad (4.4)$$

where \mathcal{F} is the feasible region of the problem. Therefore, the performance indicator can be defined as follows:

$$DP = \left(\sum_{i=1}^{NDS} h(i) \right) / NDS \quad (4.5)$$

where NDS is the number of non-dominated solutions in the real function and $h(i)$ is:

$$h(i) = \begin{cases} 1 \quad \mathbf{if}((\vec{x}_i \text{ is non-dominated in } \vec{f}) \wedge (\vec{x}_i \text{ is non-dominated in } \vec{f}')) \\ 0 \quad \mathbf{otherwise} \end{cases} \quad (4.6)$$

For these performance indicators, larger values of RP and DP are preferred.

4.5 Results from the study

4.5.1 Accuracy, robustness and scalability

In this thesis, problems with $v \leq 10$ are considered of low dimensionality while the rest are considered to be of high dimensionality.

Figure 4.2 shows the results produced by the application of the G-metric to the eight test problems. In general, it is clear from this figure, that the worst performance approach for high dimensional problems was (PR, PR) . However, for problems ZDT1, ZDT2, ZDT3 and DTLZ7 this pair achieved good accuracy on low dimensionality problems. Even in specific instances such as ZDT4 where $v = \{2, 4, 6\}$, ZDT3 where $v = \{4, 6\}$ and DTLZ7 where $v = \{6, 8, 10\}$, (PR, PR) this proposal had the best accuracy. Similarly, (KRG, KRG) obtained an erratic behavior for high dimensional problems, but its general behavior was considerably better than (PR, PR) . Although the behavior of (SVR, SVR) was not good in ZDT4 and Kursawe test problems, it outperforms the other proposals in problems ZDT1, ZDT2, DTLZ1 and DTLZ7. In ZDT4 (see Figure 4.2(d)) (RBF, RBF) presented an erratic behavior. However, in the rest of the problems, this pair shown a good performance.

Table 4.2 shows that (KRG, KRG) was the best pair for $v = 2$. Moreover, (PR, SVR) was the pair that showed the best performance for $v = \{4, 6\}$. (PR, RBF) outperform the others when $v = 8$. However, for high dimensional problems every combinations of PR obtained bad results. On the other hand, (RBF, RBF) had good results for $v > 10$. Finally, for $v = \{20, 25, 50\}$ the approach that behaved the best was (SVR, SVR) . For these reasons, RBF and SVR are selected as the most scalable techniques, since both presented their best behavior on high dimensional problems. The worst performance for this indicator was for (PR, PR) . Finally, since (RBF, RBF) was not the

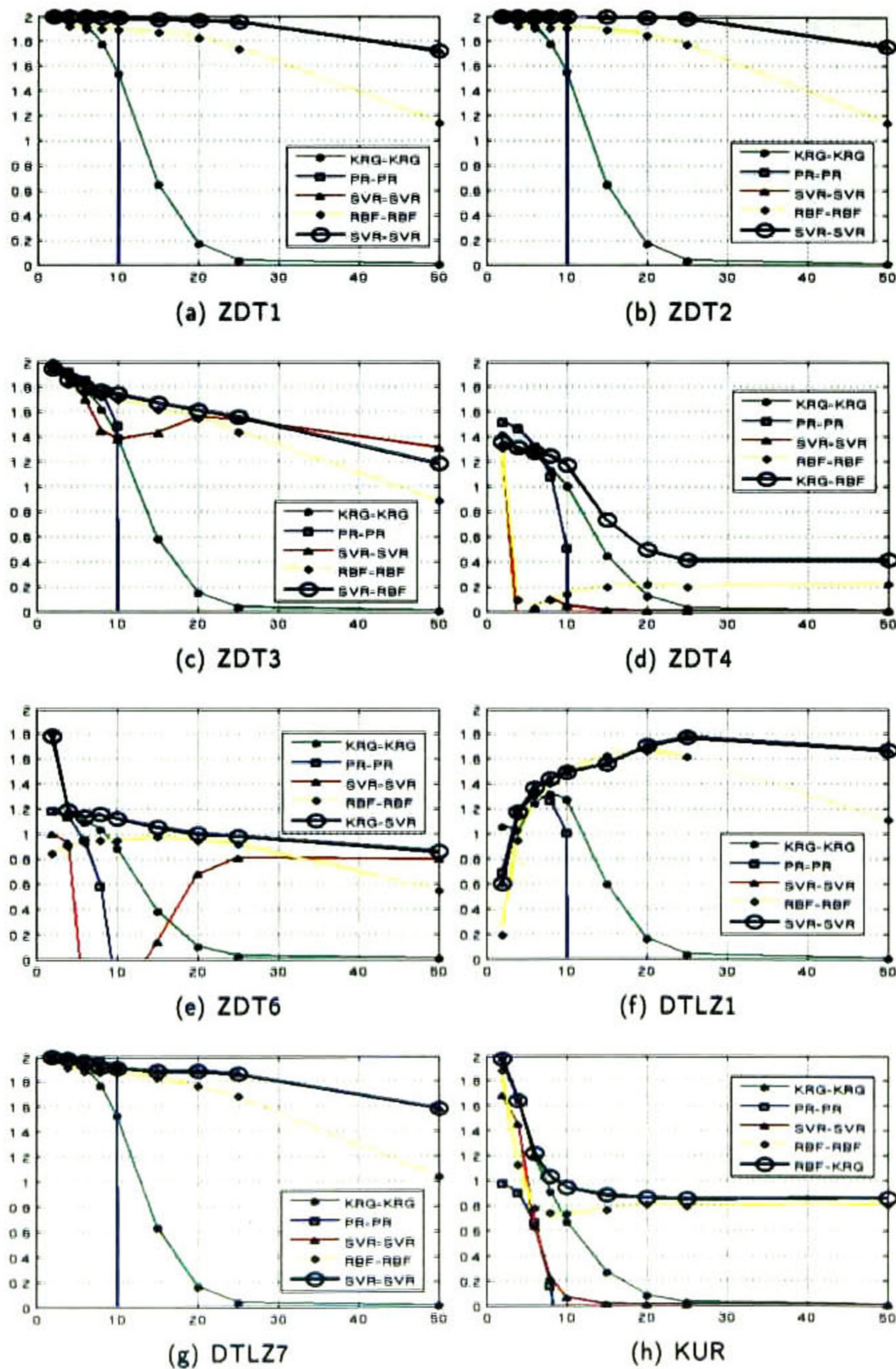


Figure 4.2: Average accuracy (for all plots: y -axis is the G-metric) by number of variables.

best in all test problems, but in most of them it presented a stable behavior, and *SVR* performed bad in problems ZDT4, ZDT6 and KUR. *RBF* is said to have a slightly better robustness than *SVR*.

It is important to note, that for some problems, the best pair of methods was composed of two different techniques (*(SVR, RBF)* in ZDT3, *(KRG, RBF)* in ZDT4, *(KRG, SVR)* in ZDT6 and *(RBF, KRG)* in Kursawe). These results are comprehensible since each fitness landscape in a MOP may have a different landscape. Therefore, might be interesting for an evolutionary algorithm to use simultaneously different metamodeling techniques for each objective.

	(R, R)	(R, S)	(R, K)	(R, P)	(S, R)	(S, S)	(S, K)	(S, P)	(K, R)	(K, S)	(K, K)	(K, P)	(P, R)	(P, S)	(P, K)	(P, P)
2	9	11	33	20	21	54	73	61	33	58	82	68	21	59	75	0
4	5	21	23	22	22	65	60	72	28	61	65	69	33	74	71	0
6	14	35	22	38	36	53	52	69	31	59	46	70	47	72	63	0
8	50	53	36	50	61	53	37	59	43	46	36	41	68	62	47	0
10	69	61	43	41	60	56	39	46	50	47	32	33	63	61	45	0
15	93	90	71	3	88	86	68	3	73	75	63	3	18	18	18	0
20	91	92	68	3	91	93	71	3	68	75	59	3	18	18	18	0
25	91	92	66	3	91	95	72	3	65	75	58	3	18	18	18	0
50	87	92	64	3	91	98	70	3	64	73	54	3	18	18	18	0
sum	509	547	426	183	561	653	542	319	455	569	495	293	304	400	373	0

Table 4.2: Statistical results for the accuracy. For short, $K = KRG$, $P = PR$, $R = RBF$ and $S = SVR$.

4.5.2 Efficiency

This criterion was measured using two experiments : *a)* the time needed to train a metamodel and *b)* the time that a metamodel needs to predict solutions. Results (see Figure 4.3) indicate that *RBF* and *SVR* required short periods of time in both, training and prediction. On the other side, *PR* needed large training times, but it presented a fast prediction response. Contrary, *KRG* presented the worst performance for both experiments of this performance indicator.

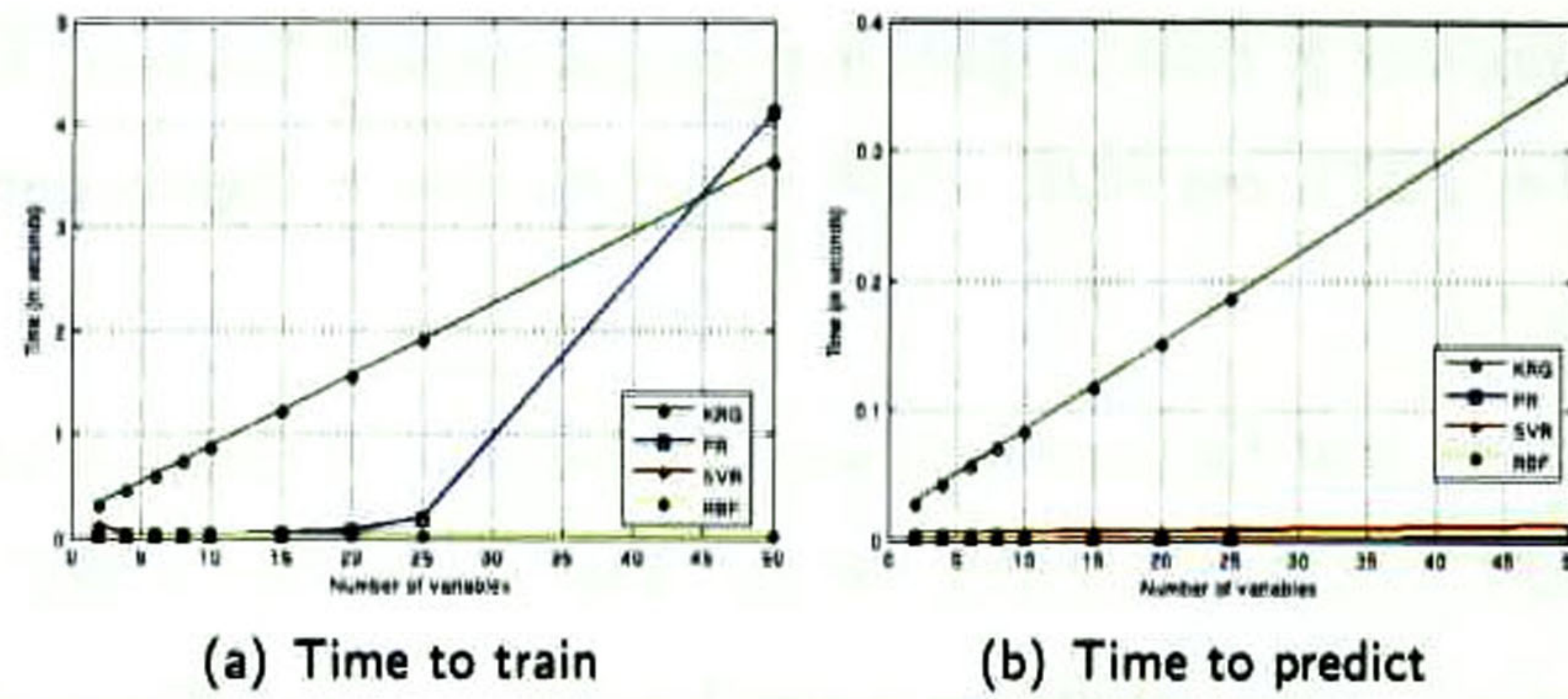


Figure 4.3: Average time of execution for training (a) and prediction (b) in all the problems.

4.5.3 Suitability

An important feature of RP is that the higher its value, the lower the probability of adding a false optimum to the problem. This feature is an important characteristic in EAs since most of the approaches attempt to avoid false optima. However, in most cases, it is required to repeat the training of the metamodels, which leads to a reduction in efficiency. It can be seen in Table 4.3 that (KRG, KRG) outperformed the other techniques where $v = 2$. Also it is possible to see that when using PR in the first objective, most of the pairs produced good results for low dimensional problems, as is easy to see in (PR, SVR) , (PR, SVR) , (PR, RBF) and (PR, RBF) were the approaches outperform the others for $v = \{4, 6, 8, 10\}$, respectively. The best pair for $v = 15$ was (RBF, RBF) . Finally, when $v = \{20, 25, 50\}$ the pair that behaved the best was (SVR, SVR) .

	(R, R)	(R, S)	(R, K)	(R, P)	(S, R)	(S, S)	(S, K)	(S, P)	(K, R)	(K, S)	(K, K)	(K, P)	(P, R)	(P, S)	(P, K)	(P, P)
2	7	9	39	8	22	57	82	57	41	59	88	61	26	64	84	0
4	6	24	23	19	33	69	67	69	35	60	69	61	46	84	83	0
6	17	28	24	28	45	59	62	66	30	52	49	54	56	81	73	0
8	41	39	35	45	61	50	41	56	37	37	38	42	72	60	63	0
10	61	52	44	42	62	52	38	47	47	37	41	34	77	65	60	0
15	92	86	67	21	84	76	57	16	73	70	65	21	17	11	11	0
20	89	85	63	27	82	90	52	18	73	67	61	17	21	14	11	0
25	89	84	60	24	80	98	54	12	70	63	59	14	33	25	20	0
50	84	81	33	45	85	100	42	43	32	42	83	12	42	36	7	0
sum	486	488	388	259	554	651	495	384	438	487	553	316	390	440	412	0

Table 4.3: Statistical results for the Ranking Preservation. For short, $K = KRG$, $P = PR$, $R = RBF$ and $S = SVR$.

Figure 4.4 shows the average results for this indicator in the eight test problems adopted. This Figure highlights the bad performance of (PR, PR) for high dimensional problems in RP. But in low dimensional problems this pair had an acceptable behavior. In this performance indicator, the best robustness behavior was for (RBF, RBF) , since it obtained good results on all problems. Moreover, (RBF, RBF) also was the most scalable pair. On the other hand, (SVR, SVR) outperformed the other pairs in ZDT1, ZDT2, DTLZ1 and DTLZ7 test functions. In ZDT3 and ZDT6, (SVR, SVR) also obtained excellent results. However, the behavior of this pair was very poor in ZDT4 and KUR. Finally, (KRG, KRG) had acceptable results with few variables, as is easy to see in ZDT4, ZDT6 and KUR, where the technique helped to achieve the best results despite their high dimensionality.

Results from Dominance Preservation (DP) (see Table 4.4) indicate that it differ slightly from the conclusions obtained with RP. Here, (PR, KRG) produced the best results for $v = 2$. Moreover, the best pair of techniques when $v = \{4, 6, 8, 10\}$ was (PR, SVR) . Finally, for high dimensional problems, combinations of SVR and RBF were the techniques that perform the best.

	(R, R)	(R, S)	(R, K)	(R, P)	(S, R)	(S, S)	(S, K)	(S, P)	(K, R)	(K, S)	(K, K)	(K, P)	(P, R)	(P, S)	(P, K)	(P, P)
2	7	17	23	15	22	51	61	47	26	54	59	52	25	51	65	0
4	3	21	13	15	23	60	51	54	7	44	21	39	32	71	64	0
6	3	23	9	22	25	49	33	54	11	30	17	30	47	71	54	0
8	9	20	7	22	32	49	25	55	13	31	15	28	58	68	51	0
10	23	34	11	33	34	48	17	49	21	30	12	27	68	70	46	0
15	63	58	36	6	72	60	36	19	55	49	26	8	24	20	5	0
20	61	66	28	19	57	65	34	12	50	45	16	7	33	32	4	0
25	70	56	28	9	58	67	34	11	60	40	12	10	29	33	8	0
50	59	73	12	12	63	89	3	16	9	34	36	29	41	36	26	0
sum	298	368	167	153	386	538	294	317	252	357	214	230	357	452	323	0

Table 4.4: Statistical results for the Dominance Preservation. For short, $K = KRG$, $P = PR$, $R = RBF$ and $S = SVR$.

Figure 4.5 shows the average results for DP in the eight test problems adopted. As in the case of RP, the pair that performed the worst for DP in high dimensional problems was (PR, PR) . However, this pair presented an acceptable behavior in some problems when $v \leq 10$. Again, the most scalable and the most robust pair was (RBF, RBF) . On the other hand, the pair (SVR, SVR) reached the best results on ZDT1, ZDT2, ZDT3 and DTLZ1. However, this pair again was the one that

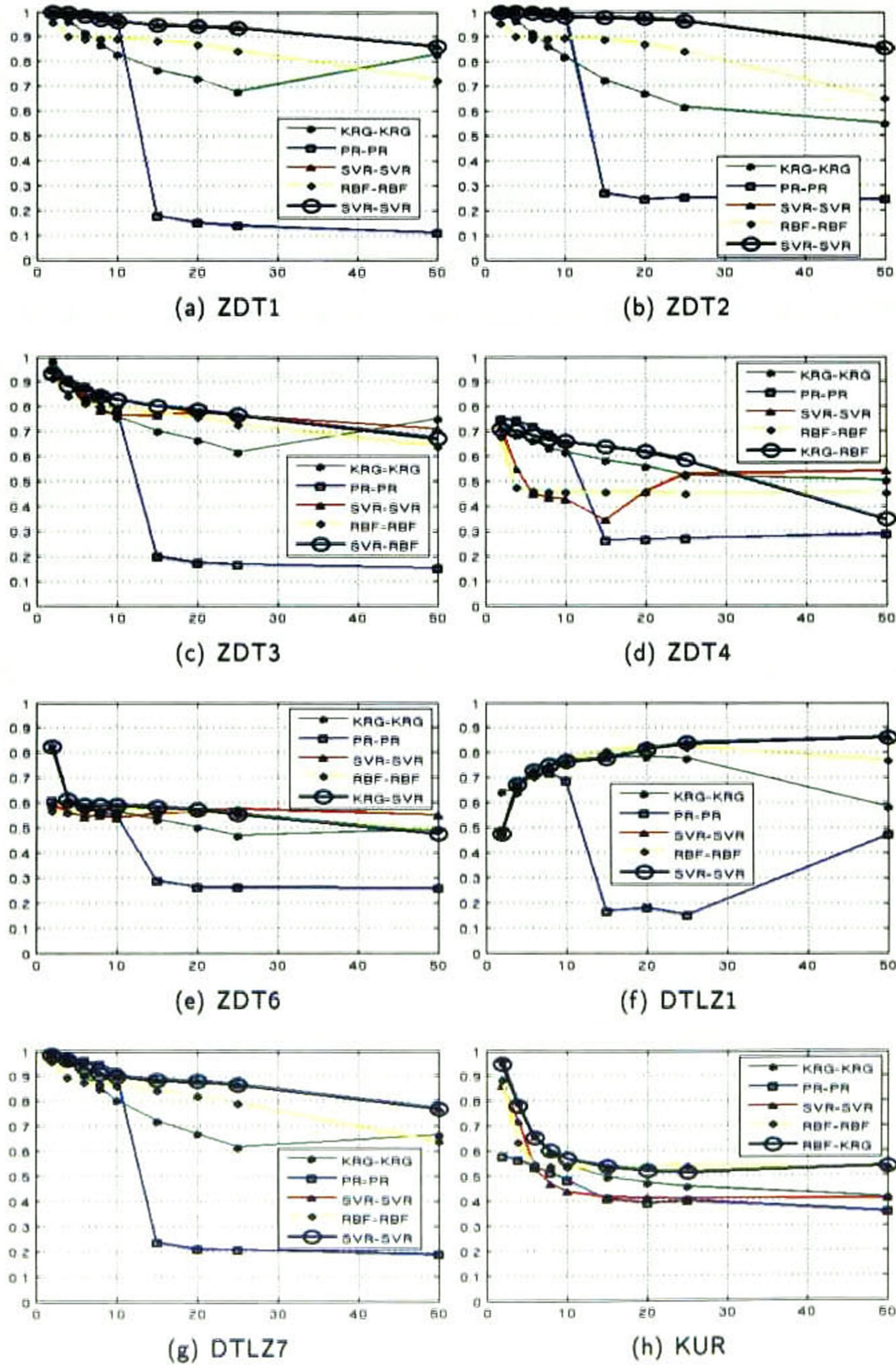


Figure 4.4: Average RP (for all plots: y -axis is the RP) by number of variables.

performed the worst in ZDT4 and KUR. The pair (KRG, KRG) alone was not successful, but combining it with another metamodeling technique achieved good results. That is the case for KUR and ZDT6 problems.

4.6 Conclusions from the study

The purpose of the current study was to compare metamodeling techniques and evaluate them under multiple aspects in order identify their suitability to be used with MOEAs. The study presented in this chapter has provided results about the performance of PR, KRG, RBF and SVR techniques in several test problems with different features.

From the results derived on this study, it is possible to conclude that the best approaches to be used in problem sizes ≤ 6 , are KRG or SVR . If the problem size is greater than 6 but lower than 15, then KRG , or RBF can be used. Moreover, if the problem size is equal or greater than 15, then, the most obvious technique to be used is SVR . Nevertheless, a combination of two different metamodels often remain over using a single metamodel for both objectives.

With respect to efficiency, RBF and SVR were the approaches with an outstanding performance. However, PR also obtained a good efficiency when predicting new solutions. The worst behavior on this criterion was presented by KRG , since it obtained the worst times on both, training and predicting.

An important issue is that if $DP = 1$, all non-dominated solutions in the artificial function will be also non-dominated in the real function. In addition, it should be noted that not necessarily the best accurate metamodel can produce the best results in the optimization process. Therefore, from this results, SVR presented the best results among the four approaches tested.

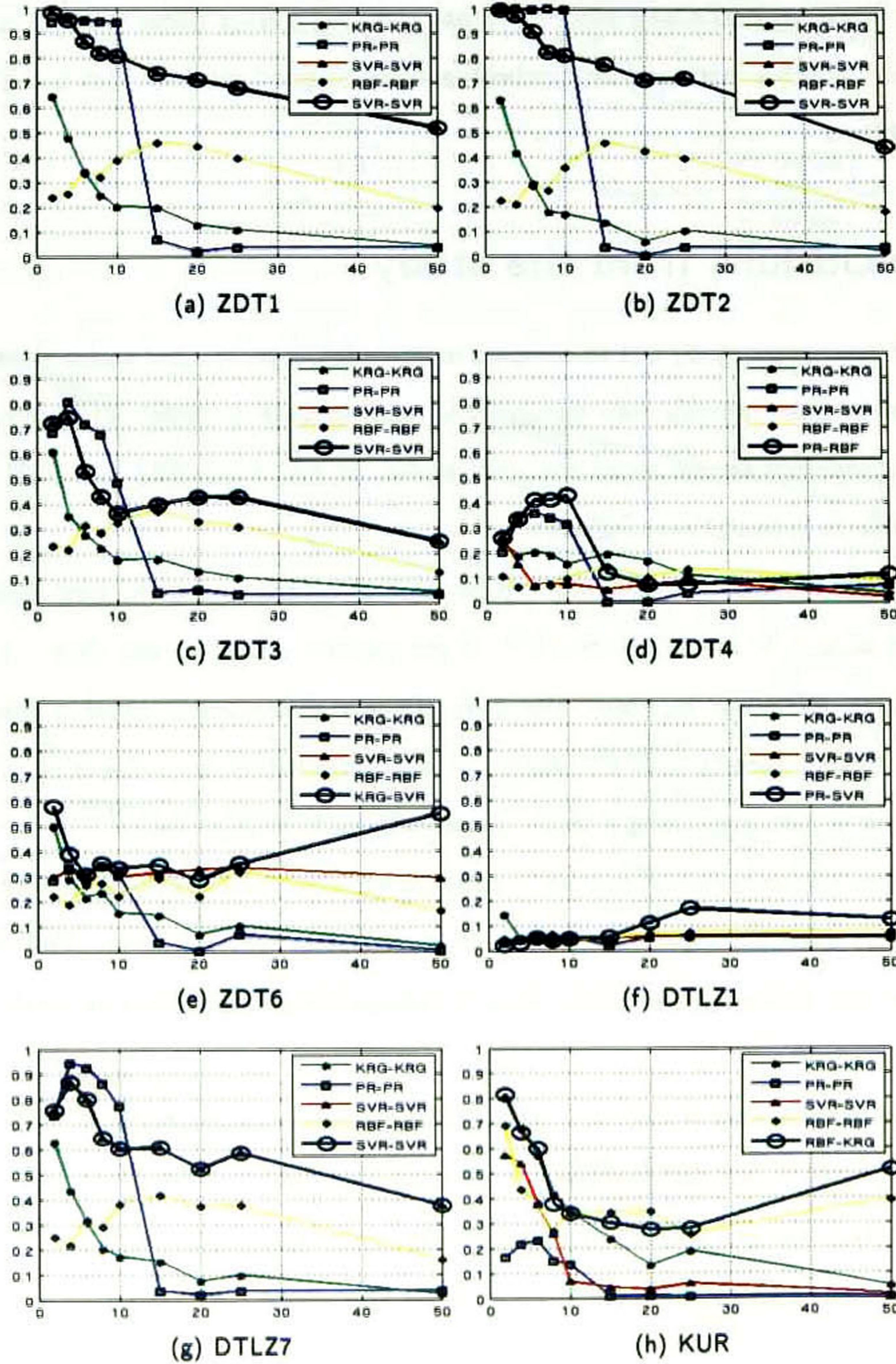


Figure 4.5: Average DP (for all plots: y – axis is the DP) by number of variables.

5

The Metamodel Assisted Subpopulation-based Search Algorithm (MASSA)

5.1 Introduction

The aim of metamodels is to perform similar than the real system but with a lower computational effort. Therefore, when optimizing expensive objective functions, these metamodels can accelerate the response time of EAs with the replacement of function evaluations.

Perhaps the most direct method for using a metamodel with EAs, would be create a metamodel with a fixed number of pre-tested solutions and optimize the metamodel using an evolutionary algorithm. Nonetheless, a metamodel is less accurate in less populated regions, so if there are not enough samples in the regions where the optimum is, this metamodel may not get the desired results.

Another common approach to use a metamodel is to intersperse its use with the real objective function, and using the new real evaluated solutions for feedback the metamodel, improving its

accuracy.

The latter scheme is the one used in this thesis work. In this chapter the Metamodel Assisted Subpopulation-based Search Algorithm (MASSA) is proposed. This algorithm was built taking into account the conclusions derived in Chapter 4, where no one metamodeling technique outperforms the others in all cases. Therefore, the proposal approach should be flexible to use any surrogate model.

The basis goal of MASSA is to perform exploitation in a certain region with a low number of evaluations. Therefore, the aim of this chapter is to describe the main components of MASSA. A general illustration about how the algorithm works is shown in Figure 5.1.

5.2 Initialization of the population

Evolutionary algorithms usually feed their initial population with randomly-generated solutions. Also, metamodels require a proper distribution of points such that its accuracy can be good. In this sense, Koehler and Owen [McKay et al., 1979] described three methods for selecting values of input variables. The Latin hypercube method (LHS) was the approach with the best results among the proved. Therefore, this algorithm was adopted in this thesis work to create an initial global population (P_0) of size N .

5.3 Select exploitable areas

Since the aim of the proposed approach is to diminish the computational effort, then, special attention was placed on the selection of promissory regions. Therefore, a subpopulation scheme was adopted in order to improve the focus on special regions.

Also the use of subpopulations prevents the whole algorithm to get trapped in local optima, since in case that one subpopulation get trapped, the other subpopulations are able to continue the search independently.

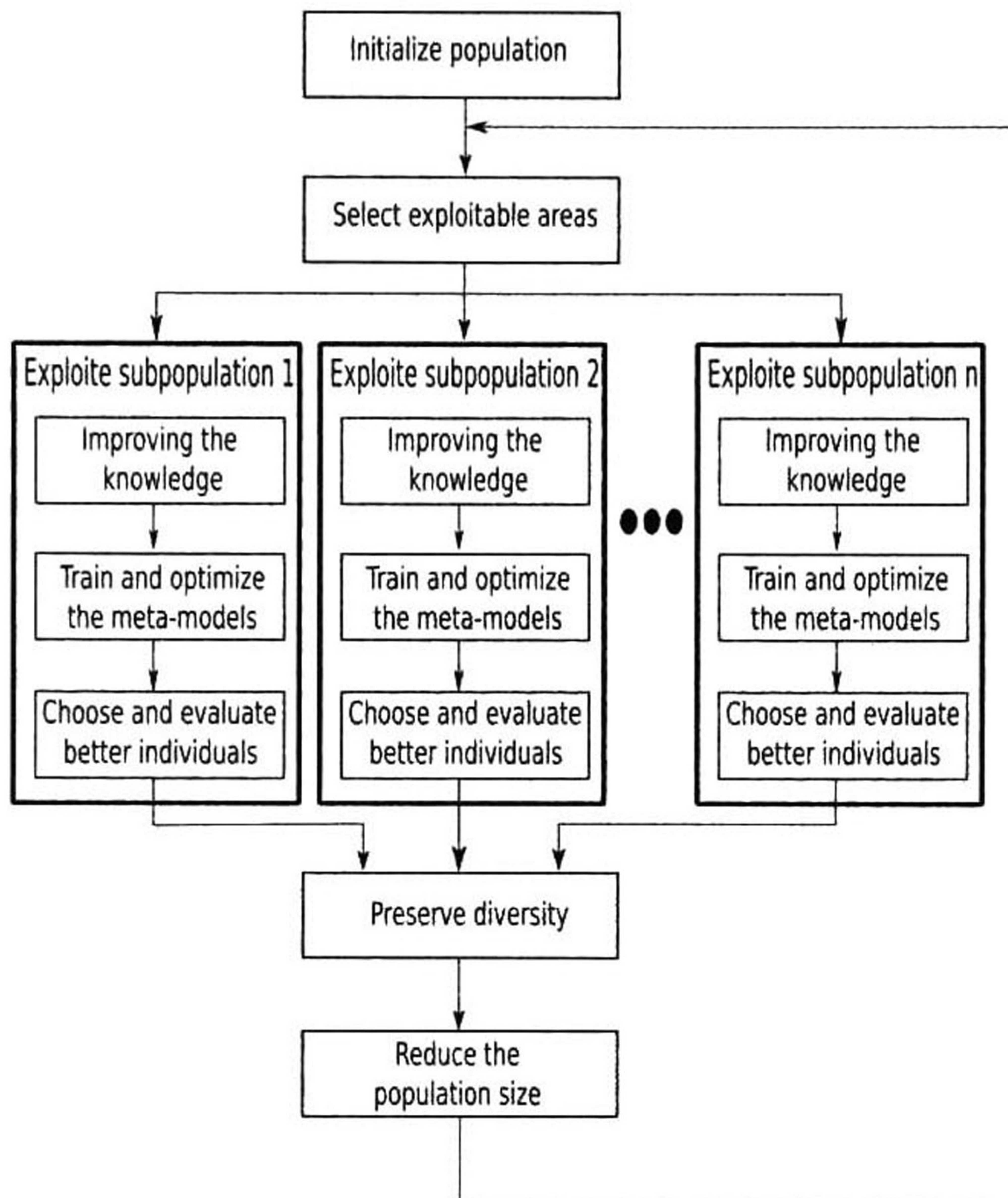


Figure 5.1: Diagram that illustrates the way in which MASSA works.

This approach clusters the main population using the K-means algorithm [MacQueen, 1967] in the variable decision space (see Figure 5.2). If a cluster has promissory solutions, then it is selected to become a subpopulation to perform exploitation, otherwise the cluster is discarded.

The Λ promissory solutions are chosen according to the crowded comparison operator [Deb et al., 2000]. This procedure will ensure the selection of non-dominated solutions well distributed on the known Pareto front. The Λ solutions are considered as indicators of promising

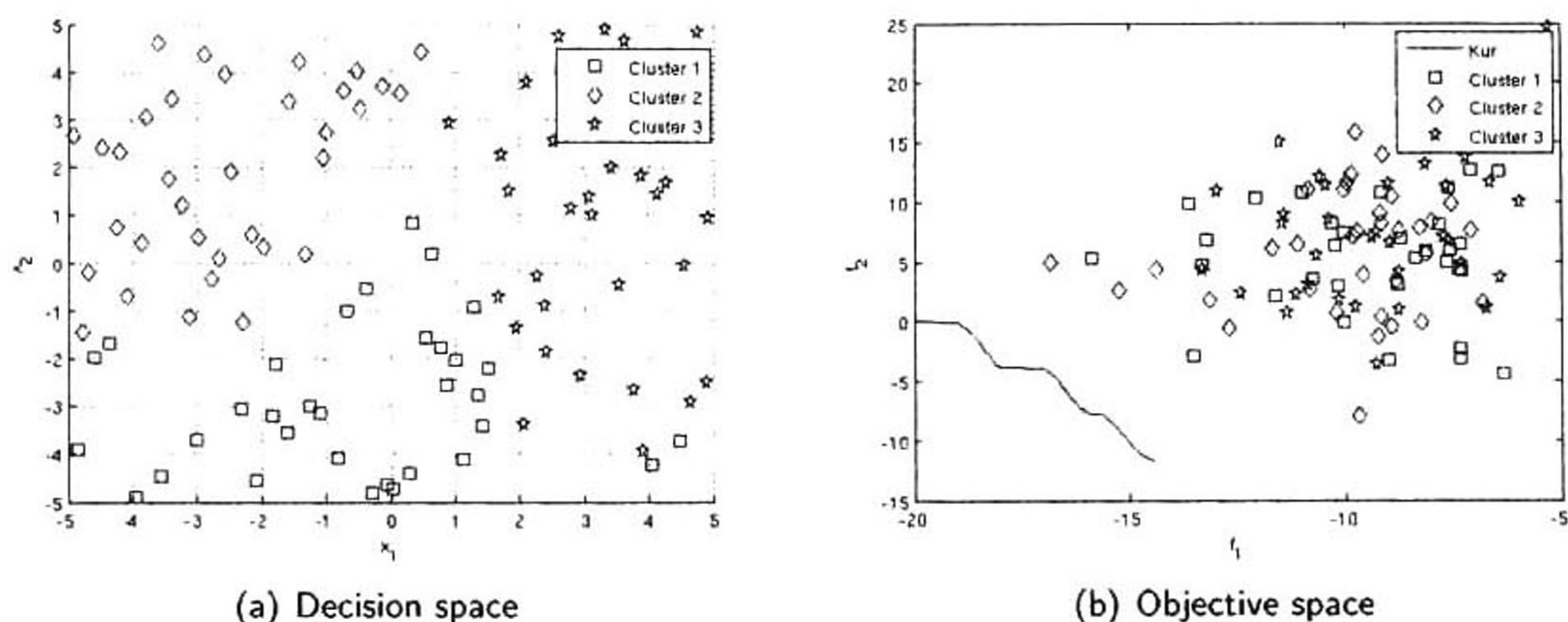


Figure 5.2: 100 solutions of Kursawe test function using latin hypercubes. The K -means ($K = 3$) was used on the search space (a), however the solutions do not necessarily have to be clustered in the objective space (b).

regions (exploitable areas).

An example of exploitable areas selection is shown in Figure 5.3. Here, three clusters using K -means ($K = 3$) were used. Moreover, three promising individuals ($\lambda = 3$) were selected. The cluster 1 ("squares") was selected once. Moreover, the cluster 2 ("diamonds") was selected twice. This achievement means that exploitation process is performed two times in the "diamonds" region. However, the cluster 3 ("stars") was not selected, then, no exploitation process will be held in the "stars" region.

5.4 Exploit a reduced area using a subpopulation

Once identified the promising regions, an independent exploitation process will be held on each subpopulation S_i . This process begins by *improving the knowledge* of the subpopulation. Then, the metamodel is trained and then optimized, the best individuals found by the optimization process will be evaluated in the real function and added to the main population. A more detailed of these processes description is given below:

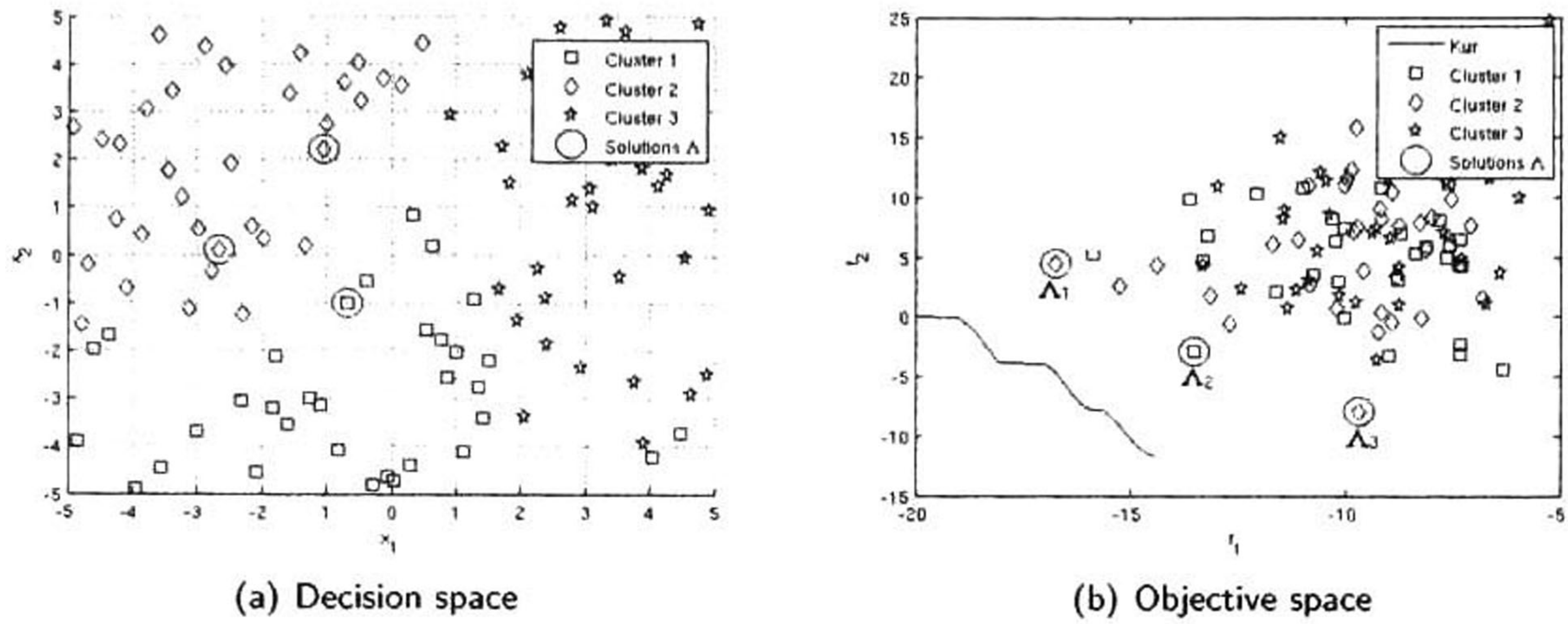


Figure 5.3: Selecting exploitable areas.

5.4.1 Improving the knowledge

This process refers to populate an specific area with additional solutions which are close to the subpopulation at hand. With this, the knowledge of a supopulation is enriched with additional data (previously stored and evaluated in the real MOP).

For this process, a box constraint is created around the exploitation area \mathcal{R}_i . Such a box refers to an extension of 10% (see Figure 5.4) on the current search space of the exploitation area. This operation looks for a wider opening on the search, such that better results are obtained. For this sake, an external archive \mathcal{E} (see Figure 5.5(a)) that contains solutions evaluated in past generations is used. Then, a new subpopulation Q_i is filled using $S_i \cup \{\text{solutions of } \mathcal{E} \text{ inside } \mathcal{R}_i\}$ (an example can be devised in Figure 5.5(b)). The way in which \mathcal{E} is managed is explained below:

5.4.1.1 \mathcal{E} delimitation

At beginning of MASSA, the external archive (\mathcal{E}) is filled with the same solutions than the population P . Furthermore, this archive is feeded each time that an individual is evaluated in the real function. Nevertheless, since the memory of the computer is limited, the size of this external file is fixed. Then,

since it is important to maintain a fixed size for this external archive, two methods were tested to perform this operation, the \mathcal{E} -methods. The first \mathcal{E} -method (+E1) will to remove older individuals (those who exceed the maximum capacity of \mathcal{E}). The second \mathcal{E} -method (+E2) will maintain a distributed population using the K -means algorithm. In this way, this algorithm is executed with $K = \{\text{the maximum capacity of } \mathcal{E}\}$. Afterwards, for each centroid its nearest solution is associated (no solution and no centroid are selected more than once). Then, the associated solutions are retained while the others are deleted from this archive.

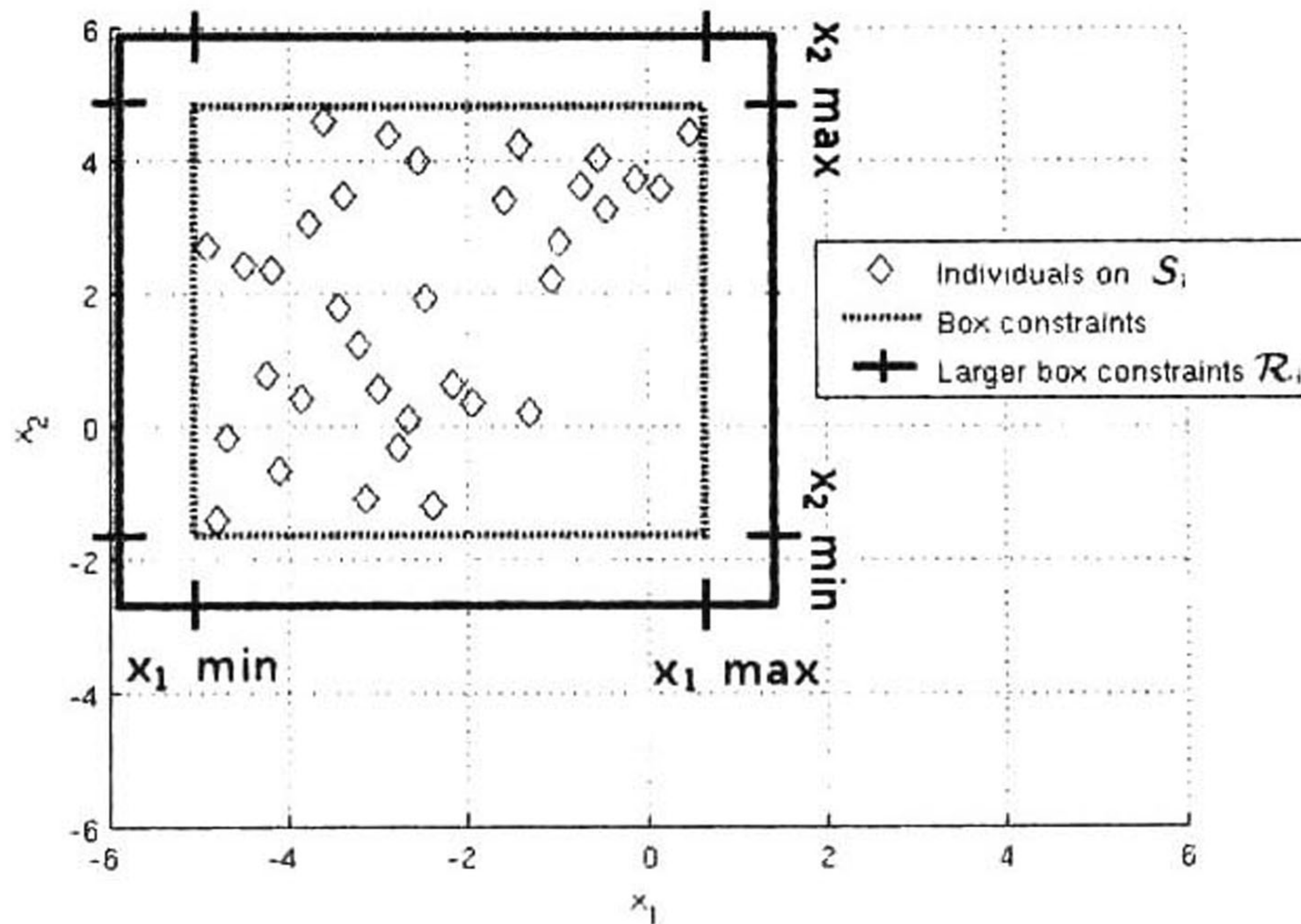


Figure 5.4: With the box constraints \mathcal{R}_i MASSA delimit the region around the cluster that will be refilled with old information to enrich the knowledge for training $\vec{f}_i'(\vec{x})$.

5.4.2 Training and optimizing the metamodels

First, the surrogate models $\vec{f}_i'(\vec{x})$ are created using the Q_i solutions (one metamodel for each objective). For this operation, any metamodeling technique can be used.

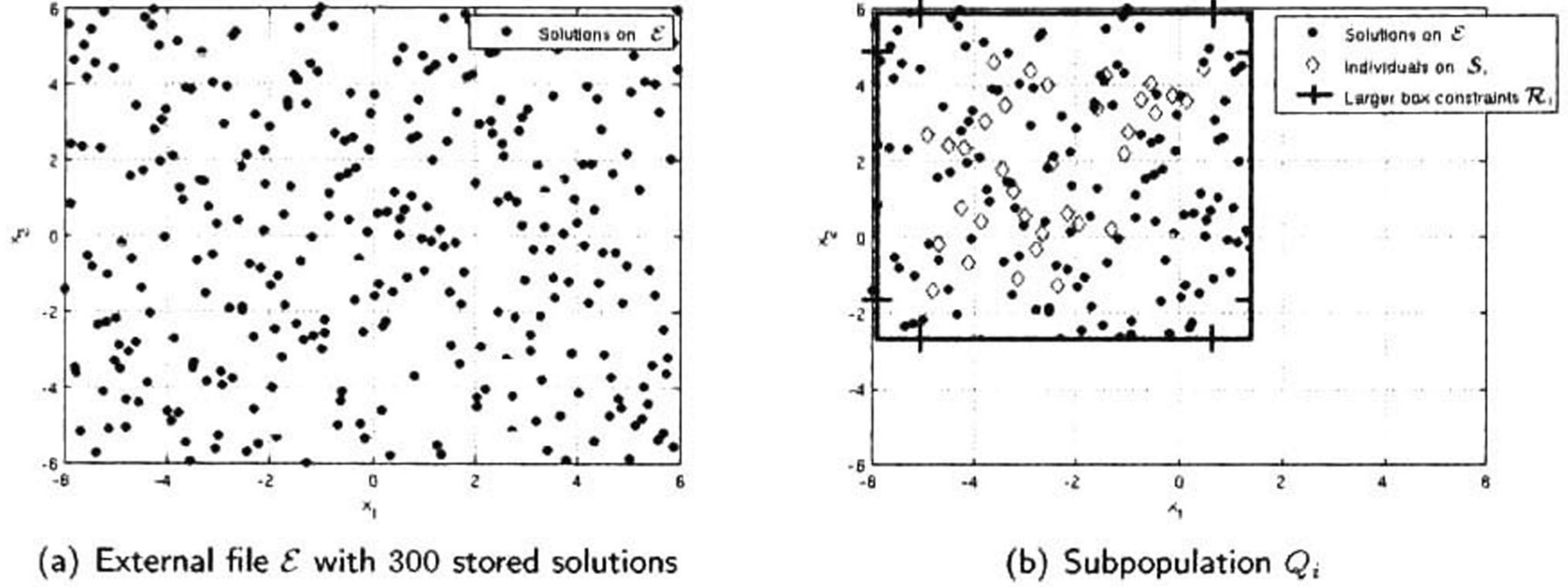


Figure 5.5: A new subpopulation Q_i is filled using $S_i \cup$ solutions of \mathcal{E} inside \mathcal{R}_i . This procedure attempt to use solutions evaluated in previous generations (\mathcal{E}) that can help to fill lonely places (the lower right corner of \mathcal{R}_i).

It is important to comment that both Q_i solutions and \mathcal{R}_i constraints are used only for training $\vec{f}'_i(\vec{x})$. In further process will not be taken into account.

Now, the metamodels $\vec{f}'_i(\vec{x})$ can be exploited using the subpopulation S_i . This process could be performed with any multiobjective optimization algorithm. In this case, the NSGA-II is used and it is executed for a few iterations.

Although in the MASSA the error in the predictions is an important factor, there is not a strong effect in the results. Since if the use of $\vec{f}'_i(\vec{x})$ in the current generation had a big error, there will exist more stored solutions in \mathcal{E} , helping to reduce the error in the long run.

5.4.3 Choosing and evaluating the best individuals

After apply the surrogated NSGA-II using subpopulation S_i as input, the best solutions (Γ_i) will be selected and evaluated in the real $\vec{f}(\vec{x})$. The Γ_i solutions are chosen using the crowded comparison operator (a small $|\Gamma_i|$ is recommended). If a solution of Γ_i is into \mathcal{P} or \mathcal{E} then it is deleted from Γ_i , otherwise it is evaluated in $\vec{f}(\vec{x})$ (see Figure 5.6 to devise an example) and added to the main

population P and the external archive \mathcal{E} .

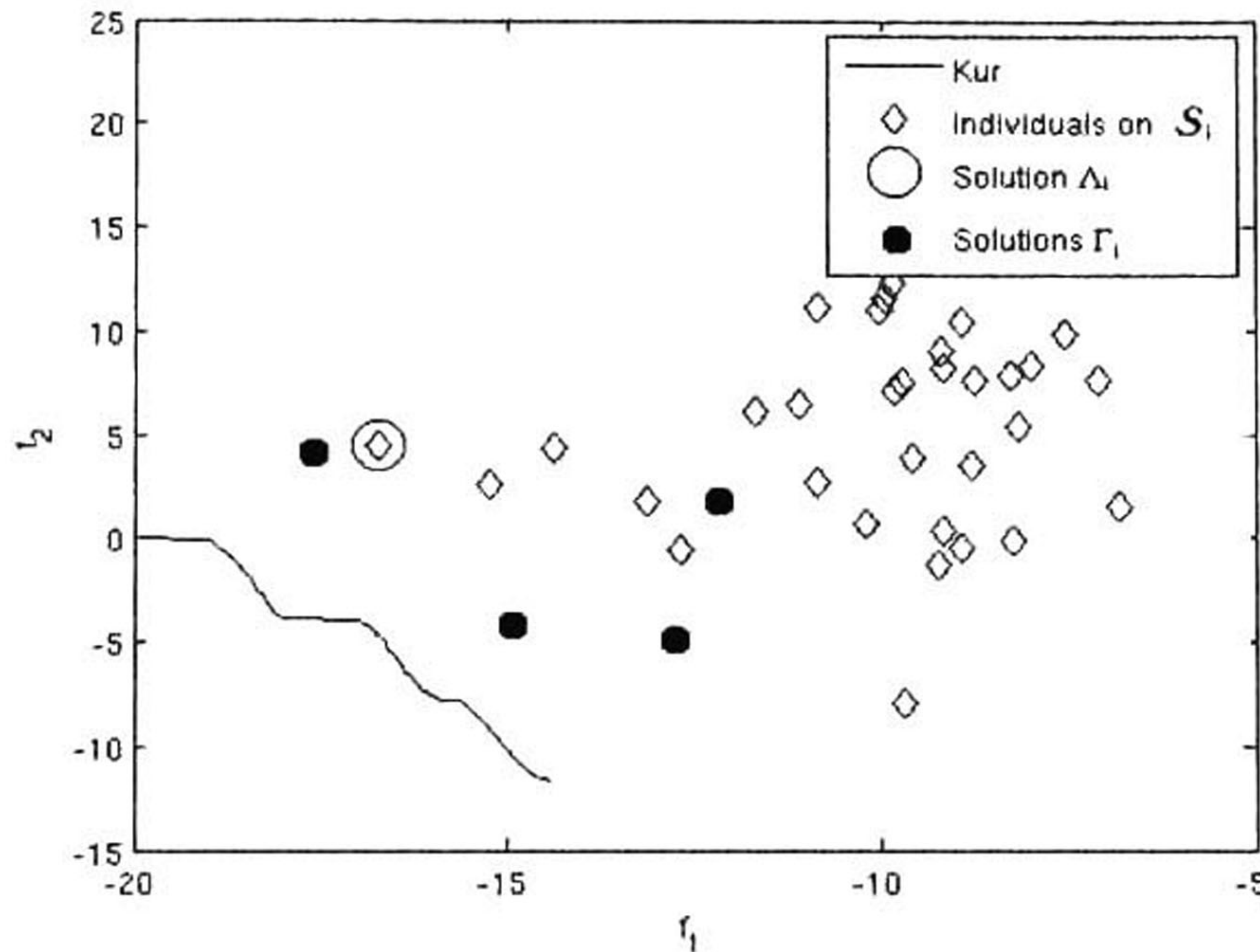


Figure 5.6: In this example the filled points are the result of the optimization of metamodels using the NSGA-II and S_i . That process has a cost of only four evaluations in the real function.

5.5 Preserving diversity

Since the population tends to converge fastly, then a variation operator is needed. In this case, a real-parameter mutation operator is chosen as source of diversity in P . For this stage, the best solutions Ω from P are mutated. Then, these solutions are evaluated in real system and added to both, the main population and the external archive. Therefore, the size of P now is approximately $N + (|\Lambda| \times |\Gamma_i|) + |\Omega|$ (since some solutions were deleted from Γ_i).

5.6 Reducing the population size

Finally, in order to maintain a fixed population size, the solutions in P are ranked with the crowded comparison operator. The best N solutions are kept.

5.7 Settings used

Parameters used in this thesis were tuning as follows:

- **Initial size of P (N):** 100 individuals.
- **Maximum size of \mathcal{E} (msE):** 300 individuals¹
- **Size of Λ (λ):** 5 individuals.
- **Size of Γ_i (γ):** 4 individuals.
- **Size of Ω (ω):** 5 individuals.
- **Number of demes (K):** 5.
- **NSGA-II parameters:** the proposed by the original NSGA-II [Deb et al., 2000].
- **Termination criterion:** until to exceed 2000 evaluations in $\vec{f}(\vec{x})$.
- **Combination of metamodeling techniques (M):** the same metamodeling technique was used for all objectives. For example, if SVR is chosen, then the combination in a problem with three objectives is $M \leftarrow (SVR, SVR, SVR)$.

¹ $|\mathcal{E}| = 300$ was used since the time required to train a metamodel (especially PR and KRG) with many solutions is very large. For more accurate results a larger $|\mathcal{E}|$ is recommended.

5.8 Results from the MASSA

Results may vary depending on the metamodeling technique in use. Therefore, in order to construct the metamodels² RBF, SVR, KRG and PR were explored.

Since the external archive is an important issue on MASSA, then both approaches for this subject were studied. Additionally, the performance of the algorithm itself but without any archive approach is revised. Nomenclature for these approaches are $+E1$ for the first method, $+E2$ for the second and $-E$ for the approach that does not use any external archive.

Moreover, the MASSA is compared with respect to the NSGA-II. The implementation of NSGA-II used in this thesis work is available for download from Deb's KanGAL web page³, and the default settings were used with this EA:

- **Population size:** 100
- **Maximum generations:** 20
- **Crossover probability:** 0.9
- **Real-value mutation probability:** $1/\text{number of dimensions}$
- **Real-value SBX parameter:** 15 for ZDT's, 15 for DTLZ's and 10 for KUR.
- **Real-value mutation parameter:** 20 for ZDT's, 20 for DTLZ's and 10 for KUR.

The MASSA was tuned to perform 2000 real function evaluations (as the same was performed for the NSGA). Moreover, in order to analyze the behavior of both algorithms, the statistics for 500, 1000 and 1500 evaluations were also computed.

²Metamodels settings are the same explained in Chapter 4. Nevertheless, the number of centers in the RBF was changed to $\lceil (\text{subpopulation size})/2 \rceil$ since it was not possible to use the 38 centers obtained in Chapter 4 in all cases (since the subpopulation size may be less than 38).

³<http://www.iitk.ac.in/kangal>

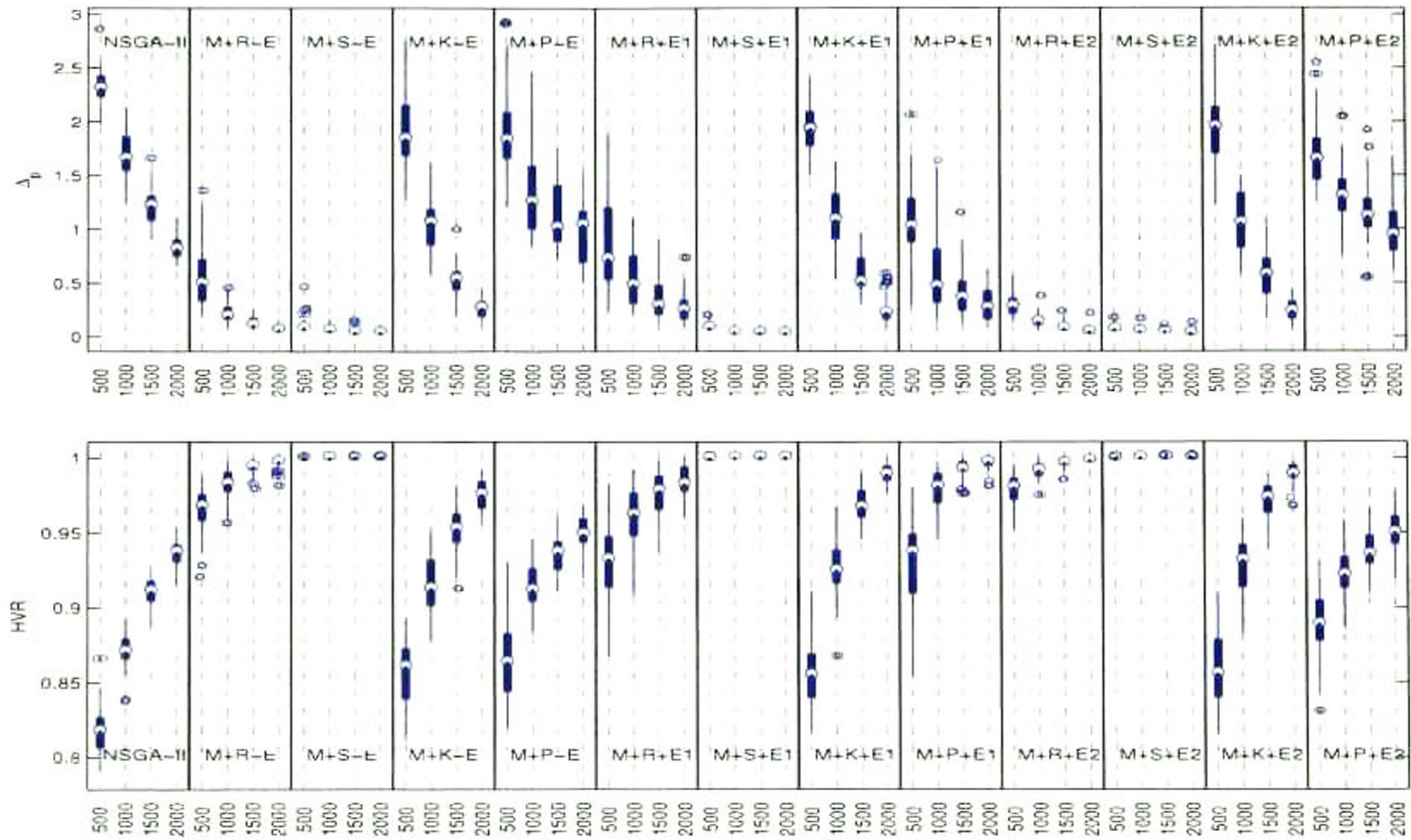
Box-plots were selected for results presentation since they show different descriptive measures. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually. The size of the box indicates the variability of the algorithm. The smaller the box, the more robust the algorithm is. Due space limitations, in box-plot images the following abbreviations were used: $R = RBF$, $S = SVR$, $K = KRG$ and $P = PR$. For example, the MASSA with RBF as metamodel, which do not uses the external file solutions is abbreviated as M+R-E and the MASSA with SVR as metamodel which uses the +E2 method is abbreviated as M+S+E2.

In this section, the box-plot figures are divided in 13 columns. The first one is for the NSGA-II results. The following four refer to those approaches that do not use the external archive. (M+R-E, M+S-E, M+K-E and M+P-E). While, the next four refer to the approaches that use the first external archive method (M+R+E1, M+S+E1, M+K+E1 and M+P+E1). And, the last four columns involve the approaches that use the second external archive method (M+R+E2, M+S+E2, M+K+E2 and M+P+E2).

Additionally, in order to provide more confidence results, summary tables were build using the statistical analysis [Demsar, 2006]. These tables are also split in 13 columns with the same order than the box-plot images (using the same abbreviations). In these tables, each approach is compared with respect to 12 remaining. The methodology to fill these tables is described as follows: On each comparison, one point is added at the pair that remains winner (in case of tie, no points are added). This analysis was performed for each number of evaluations used (this is shown in the rows). Then, a cell contains the number of times that an approach (column) outperformed the others 12 in a specific number of evaluations (row).

For ZDT1 test function, both Δ_p and HVR performance indicators show that the best metamodeling technique was SVR. Where there are no significant differences among M+S-E, M+S+E1 and M+S+E2 (see Figure 5.7). However, in statistical tables it can be seen that M+S+E1 is slightly better than M+S+E2 and M+S-E in Δ_p with 1000 evaluations. Finally, it is possible

observe observed that whatever version of the MASSA with RBF and SVR outperformed the NSGA-II. Actually, the NSGA-II was the worst algorithm for this problem, followed by the M+P-E.



Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	10	1	1	7	10	1	6	9	10	1	5
1000	0	8	10	3	1	6	11	3	6	9	10	3	1
1500	0	8	10	3	1	6	10	3	6	9	10	3	0
2000	2	8	9	3	0	3	9	3	3	9	9	3	0
total	2	32	39	10	3	22	40	10	21	36	39	10	6

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	10	1	1	6	10	1	6	9	10	1	5
1000	0	7	10	1	1	6	10	3	7	9	10	3	2
1500	0	7	10	3	1	5	10	4	7	9	10	4	1
2000	0	7	10	3	1	4	10	5	7	9	10	4	1
total	0	29	40	8	4	21	40	13	27	36	40	12	9

Figure 5.7: Δ_p and HVR box-plots and statistical summary for ZDT1 test problem.

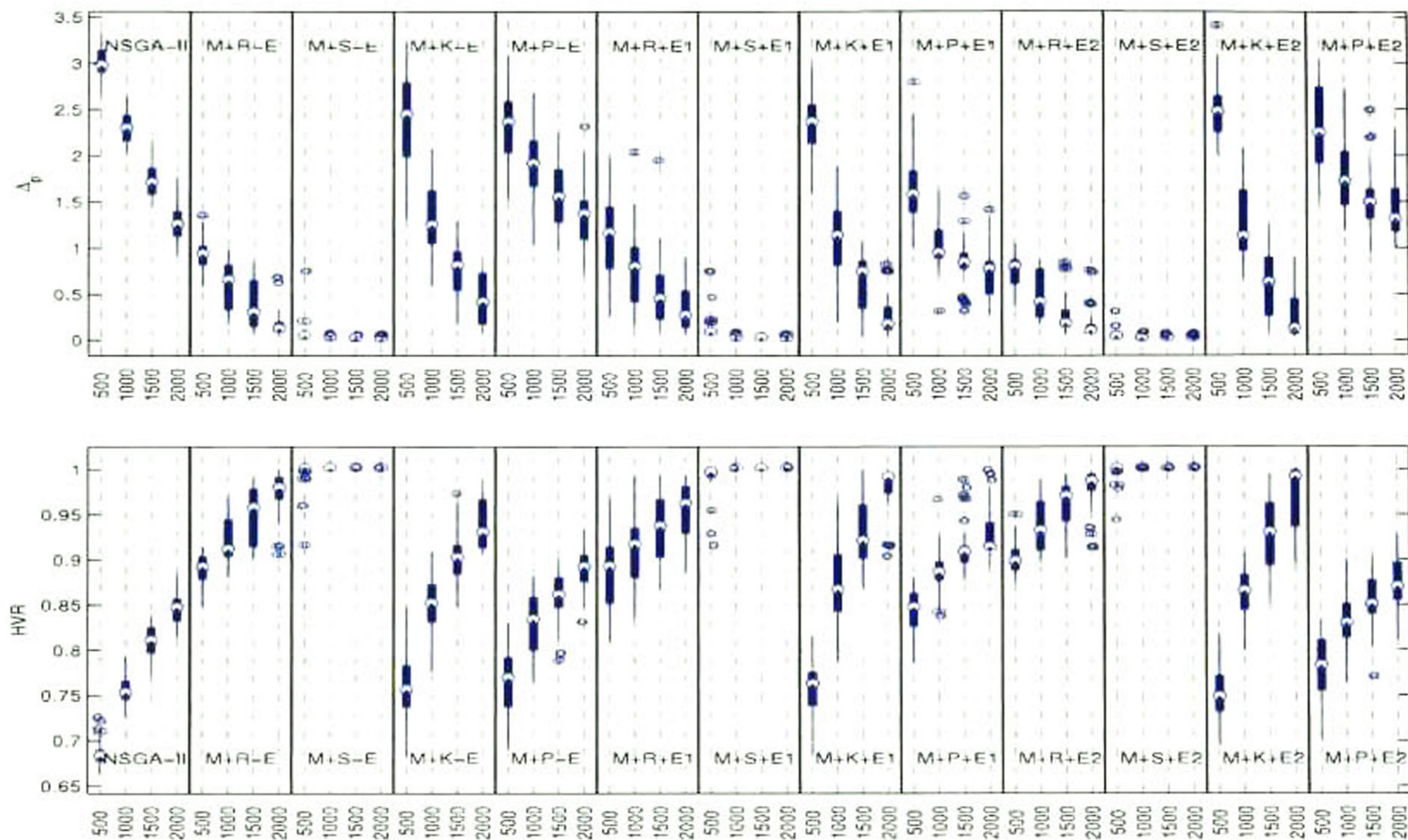
Figure 5.8 shows a similar behavior for ZDT2, where SVR is again the best metamodeling technique in both Δ_p and HVR. Actually, significant differences are not visible in the box-plots. However, tables show that M+S-E and M+S+E2 had the best results in Δ_p , and M+S+E2 was slightly better in HVR. As in ZDT1, 500 evaluations of any MASSA with SVR are better than 2000 evaluations in the NSGA-II. Actually, the NSGA-II did not outperform any other algorithm on any number of evaluations.

Once again, SVR was the best metamodeling technique for ZDT3 test problem in both, Δ_p and HVR (see Figure 5.9). There are no visible differences among M+S-E, M+S+E1 and M+S+E2. Nevertheless, in statistical tables it is possible to observe that M+S-E and M+S+E2 were slightly better than M+S+E1. As in ZDT1 and ZDT2, 500 evaluations of MASSA variations with SVR were better than the NSGA-II (even with 2000 evaluations). Again, the worst algorithm in this problem was the NSGA-II, followed by the M+P-E.

Figure 5.10 shows that KRG was outstanding metamodeling technique for ZDT4, for both, Δ_p and HVR. However, visible differences in the box-plots are shown, since the best algorithm here was M+K+E1, which is confirmed by the summary tables. In this problem, the improvement to the NSGA-II was small compared to the previous problems. Nevertheless, 500 evaluations of M+K+E1 are comparable with 1500 evaluations of the NSGA-II in Δ_p performance measures, and 1000 evaluations of M+K+E1 are better than 2000 evaluations in the NSGA-II in HVR. In this problem the worst algorithm was M+P-E, followed by the NSGA-II.

Figure 5.11 shows the results in ZDT6 test function. Here, all MASSA versions with RBF, SVR and KRG are outperform the NSGA-II. Statistical tables show that in Δ_p indicator with 500 and 1000 evaluations the best algorithm was M+K+E1, but with 1500 and 2000 the outstanding approach was M+R+E2. Moreover, in HVR with 500, 1000 and 1500 evaluations the best performance approach was M+K+E1, who tied with M+K+E2 in 2000 evaluations. The worst algorithm in this problem was the NSGA-II on both, HVR and Δ_p indicators.

For DTLZ1 test function KRG was the metamodeling technique that perform the best. While in

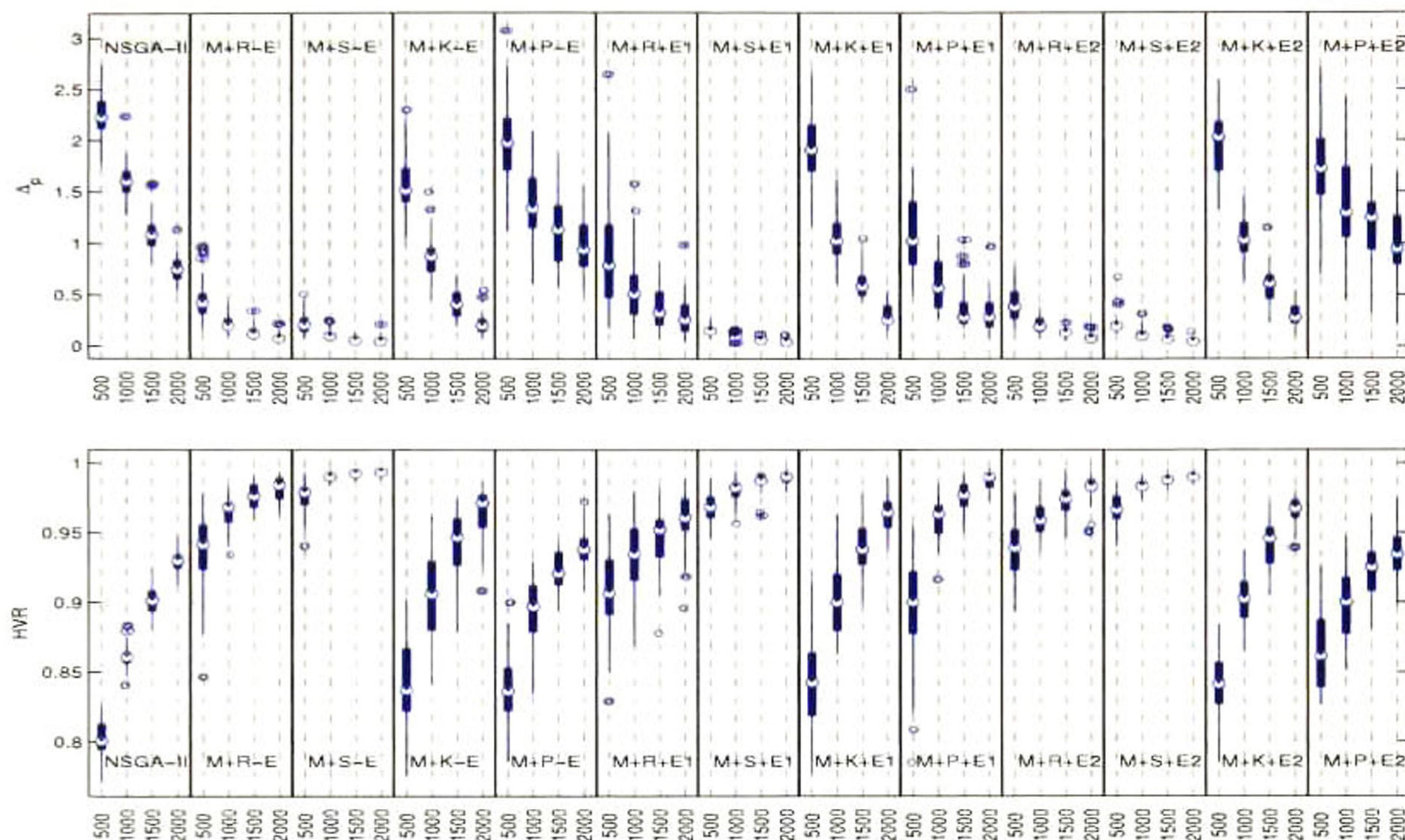


Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	7	11	1	1	7	10	1	6	9	11	1	1
1000	0	7	10	3	1	7	10	3	5	9	10	3	1
1500	0	7	10	3	1	5	10	5	3	8	10	4	1
2000	0	6	10	4	0	4	10	5	3	7	10	5	0
total	0	27	41	11	3	23	40	14	17	33	41	13	3

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	7	11	1	2	7	10	1	6	9	11	1	4
1000	0	7	10	3	1	7	10	4	5	8	11	3	1
1500	0	5	10	3	1	4	10	5	3	8	10	4	1
2000	0	6	10	3	2	4	10	6	3	6	10	6	1
total	0	25	41	10	6	22	40	16	17	31	42	14	7

Figure 5.8: Δ_p and HVR box-plots and statistical summary for ZDT2 test problem.

both, Δ_p and HVR indicators there was no appreciable differences among M+K-E, M+K+E1 and M+K+E2 (see Figure 5.12). Nevertheless, the statistical summary shows that M+K+E2 was the

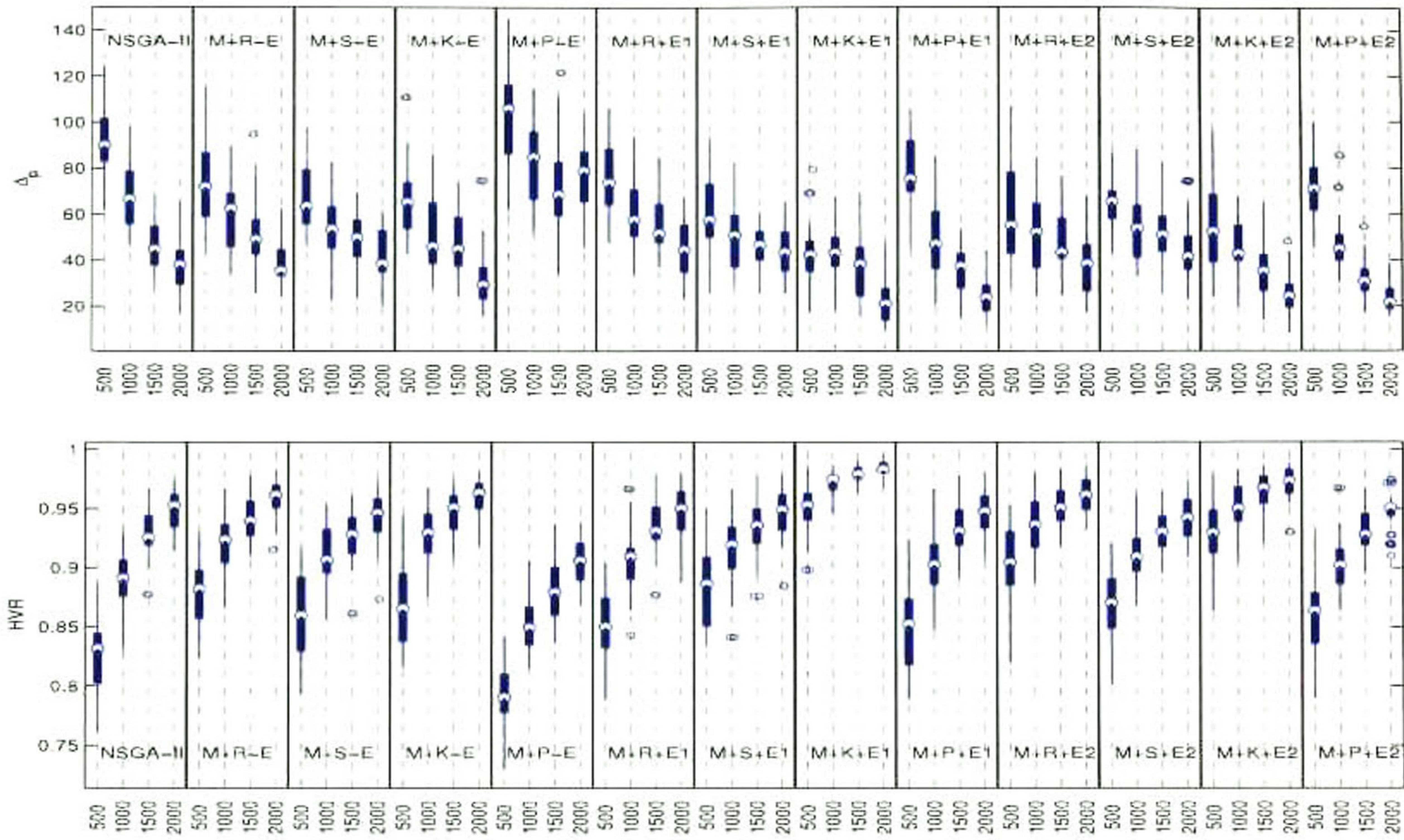


Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	10	4	1	7	12	1	6	8	10	1	3
1000	0	8	10	5	1	6	10	3	6	8	10	3	1
1500	0	8	10	5	0	5	10	3	6	8	10	3	0
2000	2	8	10	6	0	3	10	3	3	8	10	3	0
total	2	32	40	20	2	21	42	10	21	32	40	10	4

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	12	1	1	6	10	1	6	8	10	1	5
1000	0	7	12	1	1	6	10	1	7	7	10	1	1
1500	0	7	12	3	1	3	10	3	7	7	10	3	1
2000	0	7	12	3	1	3	9	3	9	7	9	3	0
total	0	29	48	8	4	18	39	8	29	29	39	8	7

Figure 5.9: Δ_p and HVR box-plots and statistical summary for ZDT3 test problem.

best for all number of evaluations. However, on this time the worst results were for the algorithms composed by SVR and PR.

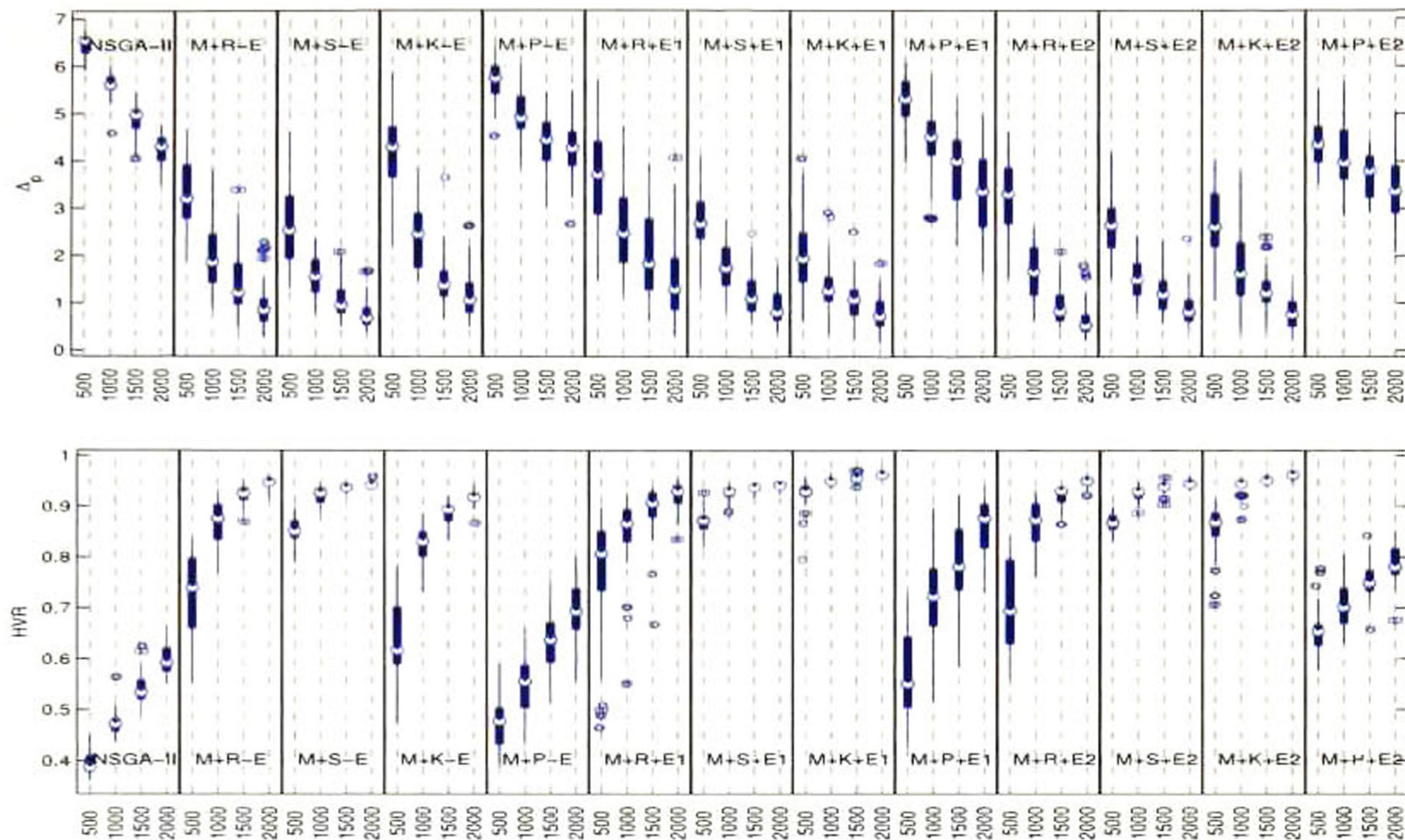


Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	2	4	4	0	2	6	12	2	6	5	9	3
1000	1	2	3	4	0	2	4	8	4	3	2	6	6
1500	2	1	2	2	0	1	3	9	9	2	1	9	9
2000	2	4	1	7	0	1	1	9	9	1	1	9	9
total	5	9	10	17	0	6	14	38	24	12	9	33	27

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	1	6	2	4	0	2	6	12	2	10	4	11	2
1000	1	5	2	7	0	2	4	12	1	9	2	11	2
1500	1	4	1	8	0	1	1	12	1	9	1	11	1
2000	1	7	1	8	0	1	1	12	1	8	1	11	1
total	4	22	6	27	0	6	12	48	5	36	8	44	6

Figure 5.10: Δ_p and HVR box-plots and statistical summary for ZDT4 test problem.

Figure 5.13 shows the box-plots results for the three objective DTLZ7 test function for Δ_p and HVR. In this problem, SVR had the best results over all techniques. Where there is no appreciable

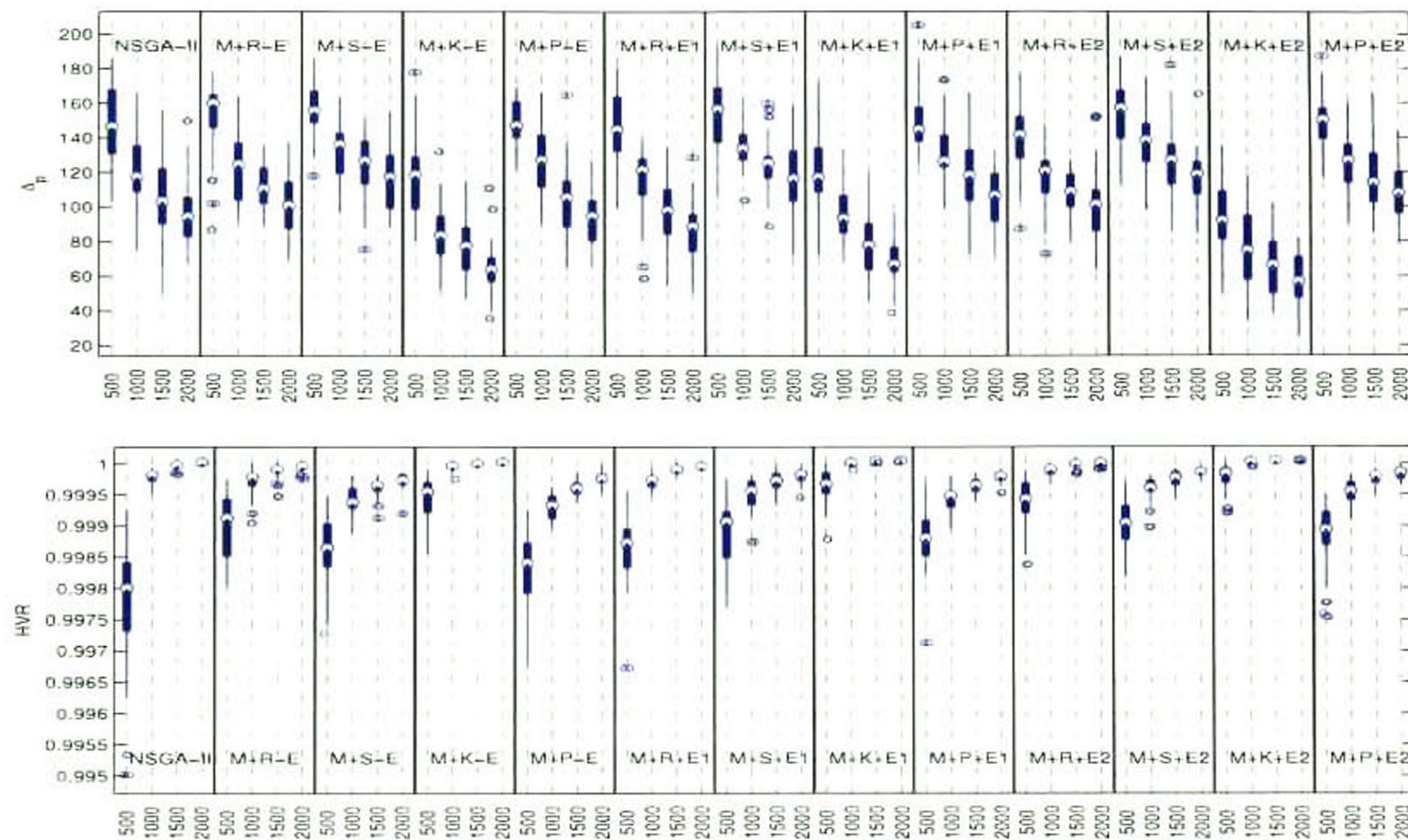


Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	5	8	3	1	5	8	12	2	5	8	8	3
1000	0	5	8	4	1	4	6	10	2	6	8	6	2
1500	0	5	8	5	1	4	6	6	2	10	6	5	2
2000	0	6	7	4	0	4	5	6	2	10	6	6	2
total	0	21	31	16	3	17	25	34	8	31	28	25	9

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	5	8	3	1	7	9	12	2	5	9	8	3
1000	0	5	8	4	1	5	8	12	2	5	8	11	2
1500	0	6	8	4	1	5	8	12	3	6	8	11	2
2000	0	6	6	4	1	4	6	11	3	9	6	11	2
total	0	22	30	15	4	21	31	47	10	25	31	41	9

Figure 5.11: Δ_p and HVR box-plots and statistical summary for ZDT6 test problem.

differences among M+S-E, M+S+E1 and M+S+E2. However, summary tables shows a tie in Δ_p for 500 and 1000 evaluations among these three algorithms. But in 1500 and 2000 evaluations



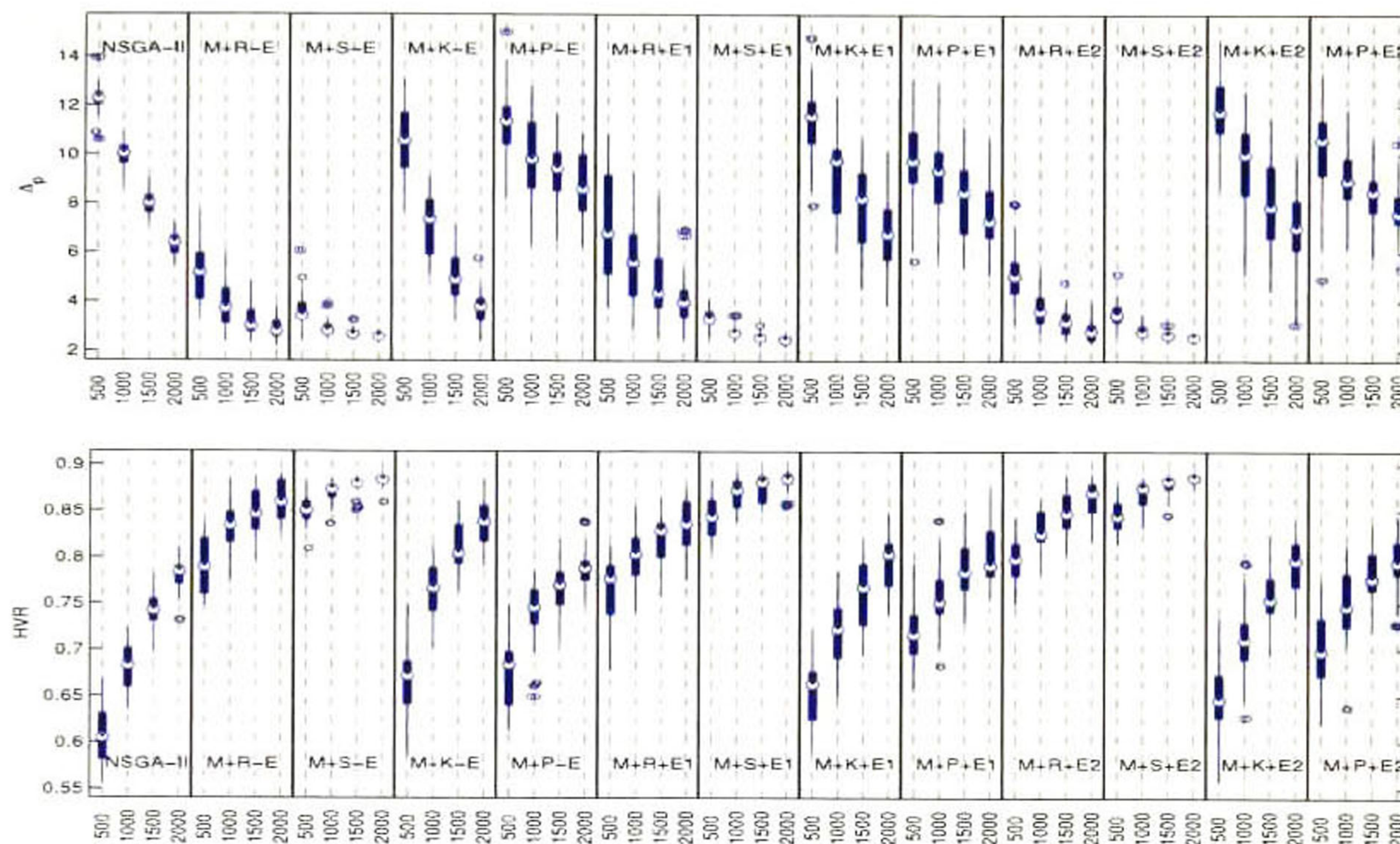
Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	0	0	10	1	1	0	10	1	4	0	12	0
1000	4	2	0	11	1	4	0	10	0	5	0	11	2
1500	3	3	0	10	5	7	0	10	0	3	0	12	0
2000	4	3	0	10	6	8	0	10	3	3	0	11	2
total	11	8	0	41	13	20	0	40	4	15	0	46	4

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	4	2	9	1	2	2	11	2	9	5	12	2
1000	7	6	0	10	0	6	2	11	1	9	3	12	3
1500	8	6	0	9	0	6	2	11	0	8	3	12	3
2000	8	6	0	9	0	6	1	11	0	8	4	12	3
total	23	22	2	37	1	20	7	44	3	34	15	48	11

Figure 5.12: Δ_p and HVR box-plots and statistical summary for DTLZ1 test problem.

M+S+E1 was the approach with the best performance. Finally, in HVR also shows a tie in 500, 1000 and 1500 evaluations, and M+S+E2 was the best in 2000. In this problem, the M+P-E was

the worst in the HVR performance indicator while the NSGA-II was the worst in Δ_p performance indicator.



Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	10	3	1	7	10	1	4	8	10	0	3
1000	0	8	10	6	0	7	10	1	2	8	10	0	2
1500	1	8	10	6	0	6	11	1	1	8	10	1	1
2000	3	8	9	6	0	6	12	3	1	8	10	2	1
total	4	32	39	21	1	26	43	6	8	32	40	3	7

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	10	1	3	7	10	1	5	8	10	1	5
1000	0	8	10	4	3	7	10	1	3	8	10	1	3
1500	0	8	10	6	1	6	10	1	4	8	10	1	2
2000	0	8	10	6	0	6	10	1	2	8	11	0	1
total	0	32	40	17	7	26	40	4	14	32	41	3	11

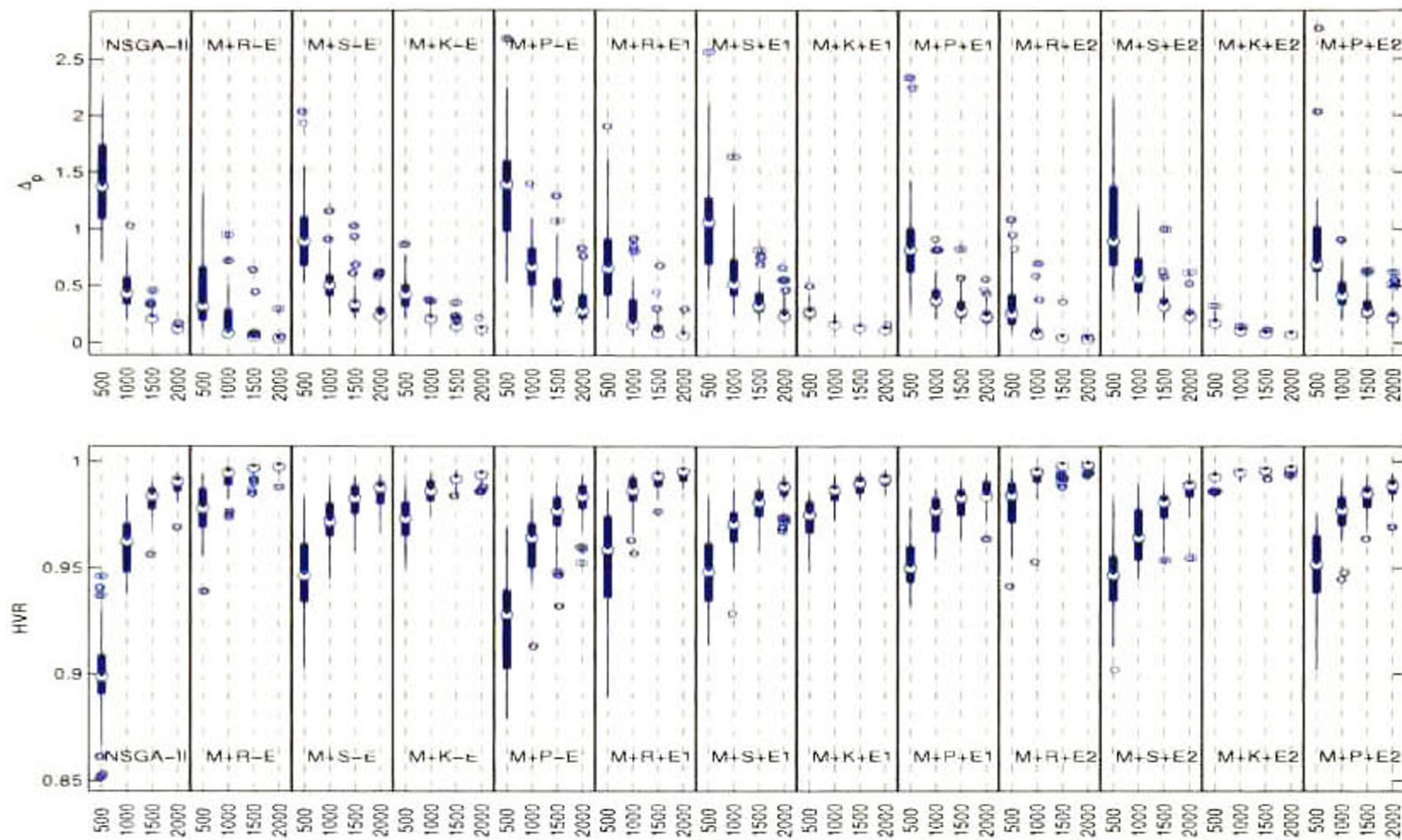
Figure 5.13: Δ_p and HVR box-plots and statistical summary for DTLZ7 test problem.

Figure 5.14 shows the behavior in KUR test function with both, Δ_p and HVR. In this problem KRG and RBF were the best metamodeling techniques. Nevertheless, the differences are unclear in the box-plot. In this sense, in statistical tables it is possible to observe that the best algorithm in Δ_p and HVR with 500 evaluations was M+K+E2. In addition, with 1000 or more evaluations M+R-E and M+R+E2 were the winners in Δ_p , and M+R+E2 was the best in HVR. In this problem, 500 evaluations in M+R+E2 are equal to or better than 1500 evaluations in the NSGA-II. However, the worst results were for the algorithms that used SVR and PR.

Finally, Table 5.1 summarizes the number of times an algorithm outperforms the others. From this table, it is easy to identify that the best metamodeling technique was SVR, since it obtained good results in $-E$, $+E1$ and $+E2$. However, SVR is not very robust, since it had very bad results in some problems such as DTLZ1 and KUR. On the other hand, this table also shows that the use of an archive improves the overall performance of the algorithm. In this sense, the best method was $+E2$ with 48 points, followed by $+E1$ with 40 and last $-E$ with 31.

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
ZDT1	0	0	7	0	0	0	8	0	0	1	7	0	0
ZDT2	0	0	7	0	0	0	5	0	0	0	8	0	0
ZDT3	0	0	7	0	0	0	4	0	0	0	3	0	0
ZDT4	0	0	0	0	0	0	0	8	2	0	0	2	2
ZDT6	0	0	0	0	0	0	0	6	0	2	0	1	0
DTLZ1	0	0	1	0	0	0	0	0	0	0	0	8	0
DTLZ7	0	0	5	0	0	0	7	0	0	0	0	6	0
KUR	0	4	0	0	0	0	0	0	0	5	0	3	0
Total by algorithm	0	4	27	0	0	0	24	14	2	8	18	20	2
Total by \mathcal{E} - method			31				40				48		

Table 5.1: Summary table for the MASSA on the eight test problems.



Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	2	8	0	6	2	9	2	9	2	12	4
1000	3	11	1	7	0	7	0	8	4	11	0	10	4
1500	6	11	0	7	0	9	0	7	3	11	0	9	1
2000	6	12	0	6	0	9	0	6	1	11	0	9	1
total	15	42	3	28	0	31	2	30	10	42	2	40	10

Evals	NSGA-II	M+R-E	M+S-E	M+K-E	M+P-E	M+R+E1	M+S+E1	M+K+E1	M+P+E1	M+R+E2	M+S+E2	M+K+E2	M+P+E2
500	0	8	2	8	1	2	2	8	2	10	2	12	2
1000	0	10	3	7	0	7	2	7	4	10	0	10	4
1500	1	11	1	7	0	8	0	7	0	12	0	10	2
2000	3	11	0	8	0	9	1	6	0	12	1	10	2
total	4	40	6	30	1	26	5	28	6	44	3	42	10

Figure 5.14: Δ_p and HVR box-plots and statistical summary for KUR test problem.

5.9 Conclusions

This chapter has provided results about the suitability of PR, KRG, RBF and SVR techniques to be combined with a MOEA. The resulting approach was called MASSA.

An important issue is that the right selection of a metamodeling technique can help to reduce the number of evaluations. In problems ZDT1, ZDT2, ZDT3 and DTLZ7, SVR had excellent results improving clearly the others (the NSGA-II is included). Similarly, KRG was specially useful in ZDT4, ZDT6 and DTLZ1 test problems. Finally RBF outperformed the others in KUR test problem.

Therefore, the robustness of any version of MASSA is low, since excellent results showed with a metamodeling technique on a specific test problem do not guarantee good results in other test problem. For example, as mentioned above SVR had excellent results in four different test problems. However, in DTLZ1 it showed the worst performance.

In this sense, it would be interesting to perform an online-adaptation of MASSA, such that the algorithm itself can identify the best metamodeling technique to be used at any given moment. This is the idea behind the Tune-adaptive Metamodeling Assisted Algorithm (TAMAAL), which will be discussed in Chapter 6.

Finally, the use of the ε archive is convenient, because both, $+E2$ and $+E1$ methods showed better results than $-E$.



The Tune-adaptive Metamodel Assisted Algorithm (TAMAAL)

6.1 Introduction

Results from Chapters 4 and 5 suggest that there is not a single metamodeling technique that outperform the others on all test problems. In this sense, the Tune-adaptive Metamodel Assisted Algorithm (TAMAAL) is proposed as an improved version of MASSA.

Unlike MASSA, TAMAAL performs an online adaptation on the selection of metamodels that are improving each subpopulation. This mechanism finds the best surrogate technique to be used at a given time in a given problem.

The way in which the TAMAAL finds the best surrogate model is through a tournament selection among the four metamodeling techniques. For this purpose, three different approaches were evaluated.

- G-metric tournament
- Ranking preservation tournament
- Dominance preservation tournament

A detailed description of such approaches is given below.

6.2 G-metric, DP and RP tournaments

Given a subpopulation Q_i , the procedure to perform a tournament is as follows: First, Q_i is split in both, a testing set ($(10\% \text{ of } |Q_i|) + 1$) and a training set ($(90\% \text{ of } |Q_i|) - 1$). Afterwards, the training set is used to train the four metamodeling techniques on each objective (4 training processes for each objective). Then, the prediction of each individual is performed using the testing set on each objective with the four metamodeling techniques (4 prediction processes for each objective). Additionally, a full factorial design of the prediction processes is performed. The full factorial gives 4^k different combinations of metamodels $(C_1, C_2, \dots, C_{4^k})$ all of them of size k . For example, a bi-objective problem ($k = 2$) has 4^2 different combinations of size 2 $((RBF, RBF), (RBF, SVR), (RBF, KRG), (RBF, PR), (SVR, RBF), \dots, (PR, KRG) \text{ and } (PR, PR))$, where (RBF, SVR) means that the first objective of the function is predicted with RBF and the second one is predicted with SVR. For each combination, the selected indicator (G-metric, DP or RP) is calculated. Then, the combination of metamodels that wins the tournament (the one with the highest value of the adopted indicator) is chosen to perform the surrogated optimization process in the Q_i subpopulation.

6.3 Settings used

Parameters used in this algorithm were tuning as used in MASSA, with the following differences.

- M is not an input parameter because it is computed automatically.

- Type of tournament to do (T): any of G-metric, RP or DP.

6.4 Results from the TAMAAL

Since the \mathcal{E} – methods (+E1 and +E2) achieved good results in the MASSA, then, this methods were adopted in order to perform the comparison of results. Moreover, results from NSGA-II are shown again with the purpose of compare this proposal with respect to an algorithm representative of the state of the art. Box-plots and the statistical analysis for each test function similar to the ones presented in Chapter 5 are used to validate the results.

Due space limitations, abbreviations for the box-plot images and statistical tables were used. For example, the TAMAAL with G-metric as tournament method which uses the +E2 external file method is abbreviated as T+G+E2. Then, the TAMAAL with ranking preservation (RP) as tournament method and +E1 as \mathcal{E} – method is abbreviated as T+RP+E1.

The box-plot figures and statistical tables are divided in 8 columns. The first one is for the NSGA-II. The next one is for the best version of MASSA for the problem at hand with a specific metric. Last six are T+G+E1, T+RP+E1, T+DP+E1, T+G+E2, T+RP+E2 and T+DP+E2.

In addition to the box-plot figures and statistical tables, a new kind of figures is introduced in this chapter. These figures try to highlight the way in which the TAMAAL was selecting the combination of metamodels (M) at any given time. In these figures the averages of 31 runs were plotted. In each run were performed 19 samples. The first sample was taken between [100, 200] evaluations, the second one was between [200, 300], and so on until sample between [1900, 2000] evaluations. The graph shows four lines which correspond to the four metamodeling techniques adopted in this thesis work. A fifth line is also shown. This line indicates total of tournaments performed.

Figure 6.1 shows the results obtained by the eight algorithms in ZDT1 test function. Here, any TAMAAL version was better than the NSGA-II in both, Δ_p and HVR indicators. The differences with the winner MASSA version (M+S+E1) is not easily visible. Nevertheless, in the statistical

summary shown in Figure 6.1, it is possible to see that T+RP+E2 outperforms the others for Δ_p , while T+G+E2 obtained the second place. Moreover, in HVR the best algorithm was T+G+E2. Afterwards, in this problem the \mathcal{E} - method that performed the best was +E2 in both indicators, while the best approach for tournament was the RP in Δ_p and the G-metric in HVR. Figure 6.2 shows the behavior of the tournaments performed with the algorithm T+G+E2. It can be seen that in the first objective SVR lie near to the TOTAL line, which means that it won almost all the tournaments. These technique was selected from the beginning to the end consistently. In the second objective SVR also was the technique that outperforms the others most of times. Nevertheless, in this objective RBF behaves well in some tournaments, especially when the number of evaluations were increased.

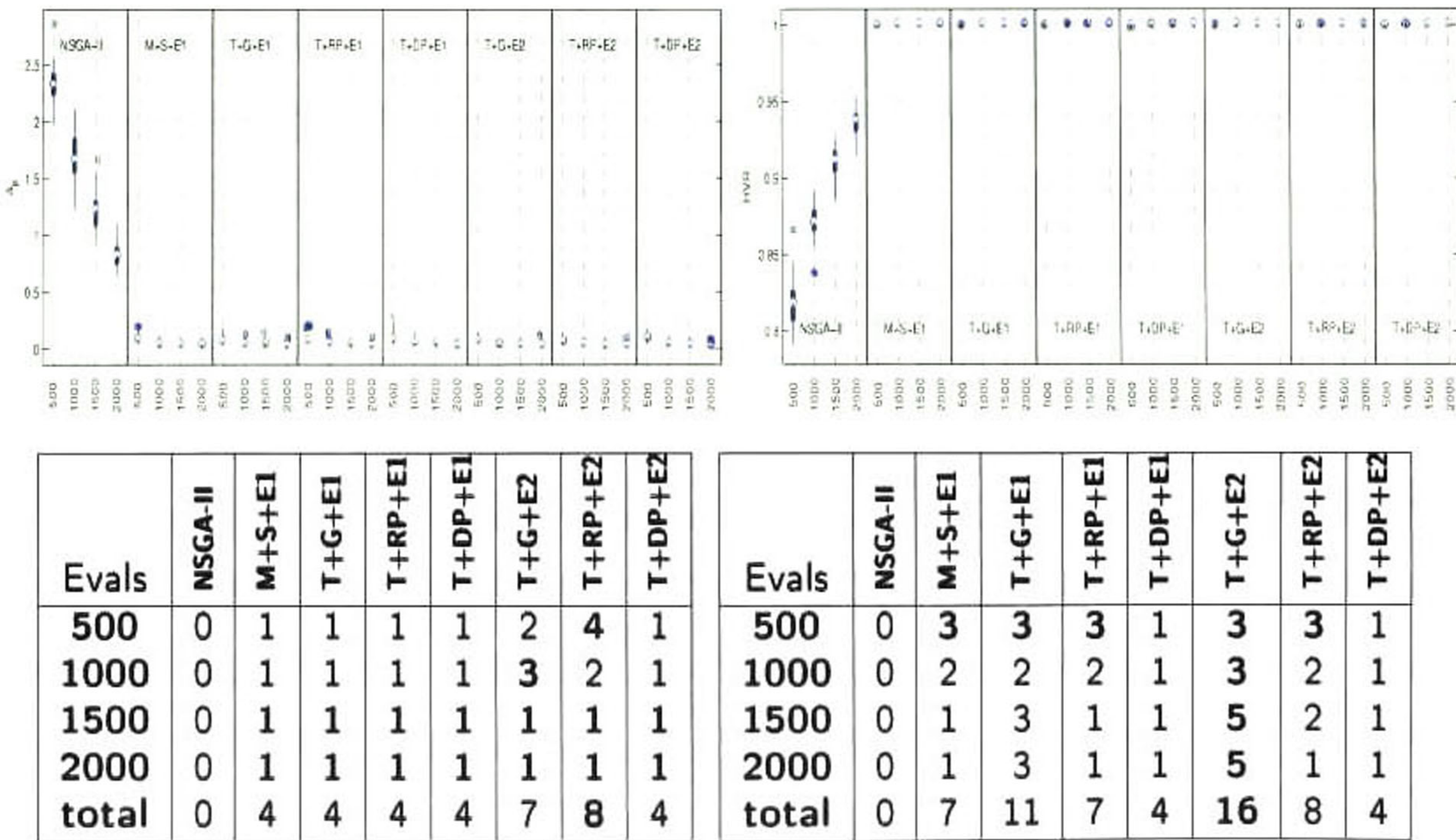


Figure 6.1: Δ_p and HVR box-plots and statistical summary for ZDT1 test problem.

The results for ZDT2 test function are shown in Figure 6.3. From this figure, it can be seen that the NSGA-II results were outperformed by the results from the other algorithms. As can be appreciate in the statistical tables, the winner in Δ_p was T+G+E2. Nevertheless, T+RP+E2 also

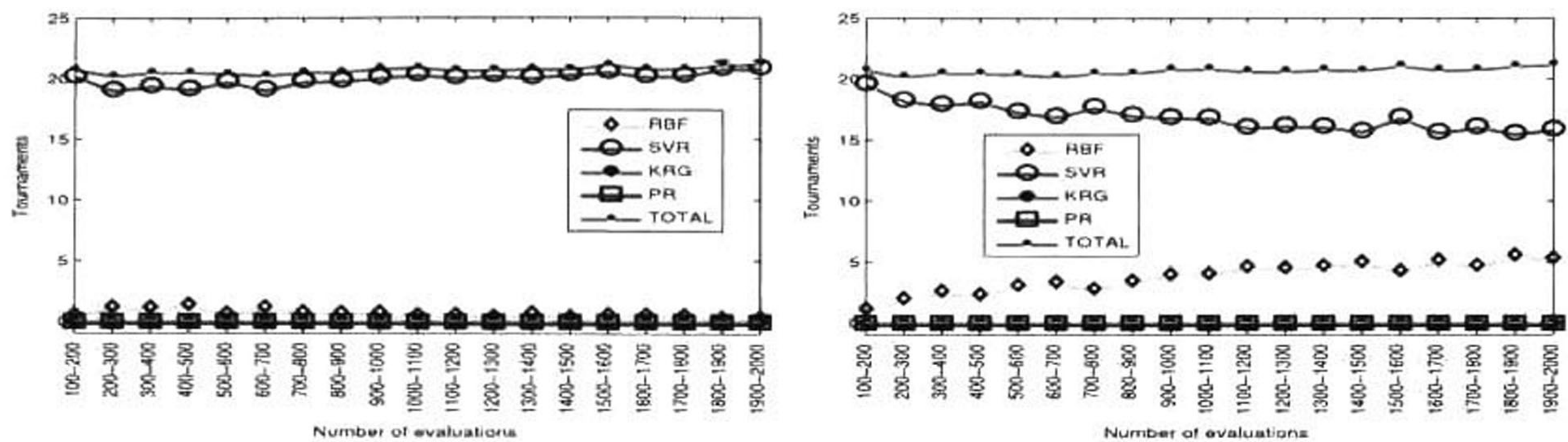


Figure 6.2: Behavior of the TAMAAL using the G-metric and the $+E2$ method (T+G+E2) in ZDT1.

had outstanding results. While in HVR, these both algorithms behaved similarly. Then, for this problem best \mathcal{E} – method was $+E2$ in both indicators, while the best tournament method was the G-metric in Δ_p and the G-metric and RP for HVR. The behavior of the tournaments performed with the T+G+E2 algorithm are shown in Figure 6.4. Here, SVR also was the most selected technique in both objectives, winning almost all the tournaments. In fact, SVR was selected from the beginning to the end consistently in both objectives.

Once again, in ZDT3 test problem all TAMAAL versions outperform the NSGA-II in both, Δ_p and HVR indicators (see Figure 6.5). For this problem, T+G+E1 was the best performance approach in Δ_p , while M+S-E obtained the best results for HVR. The best performance approach in HVR performance measure was T+RP+E1, while the most used were the G-metric for Δ_p and for RP in HVR. Therefore, the RP in the tournaments figures is shown (see Figure 6.6). Where SVR again was the winner in almost all the tournaments (the SVR line is close to the TOTAL line). These technique was selected from the beginning to the end consistently in both objectives.

For ZDT4 test function the best algorithm was T+G+E1 in both, Δ_p and HVR performance indicators (see Figure 6.7). And once again, the algorithm that behave the worst in this test function was the NSGA-II in both indicators. Afterwards, the $+E1$ and the G-metric were the best approaches in this test problem (see Figure 6.8). It is obvious from this figure that KRG was the most selected approach in the second objective. However, in the first objective is more complicated. On the

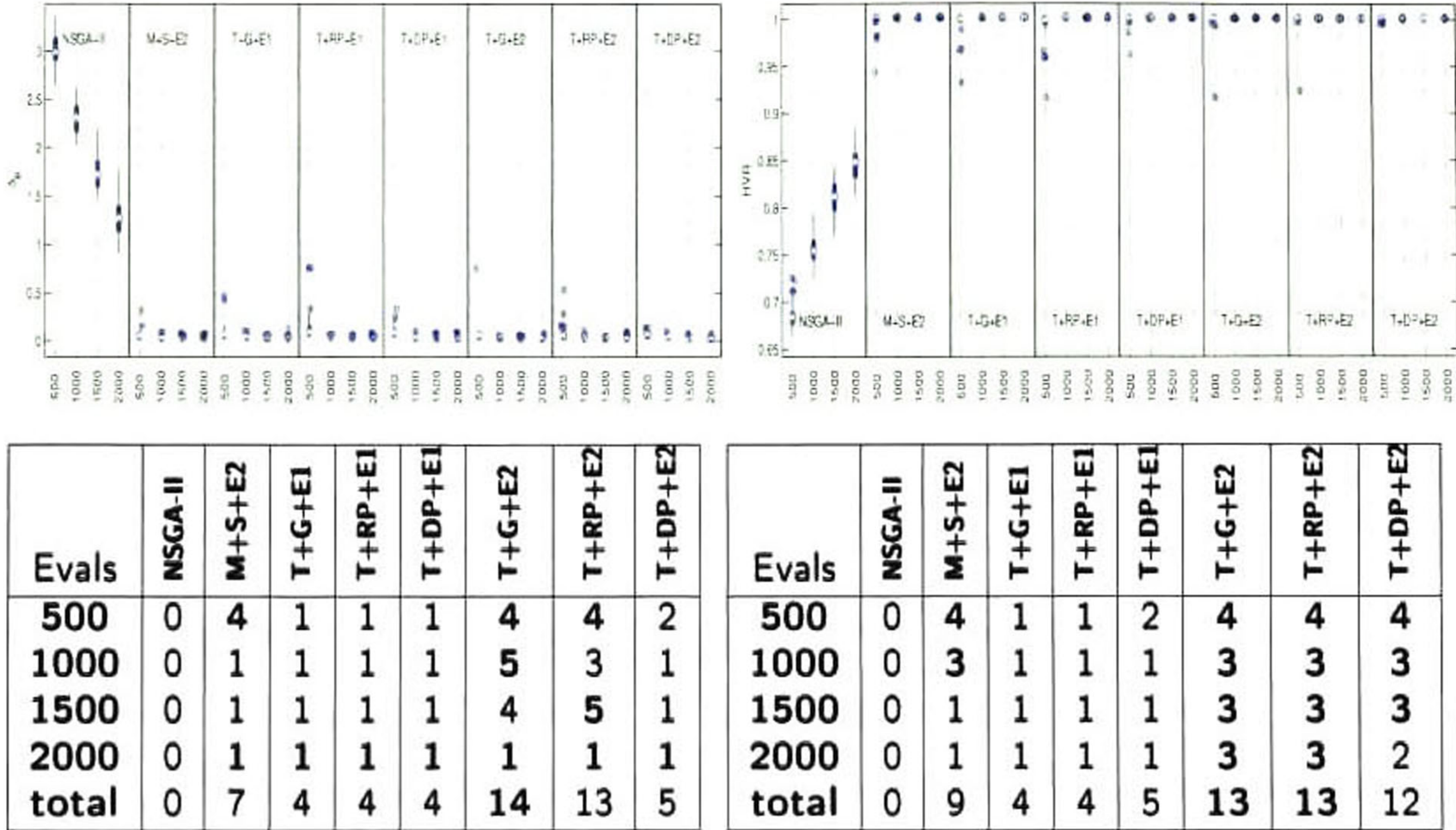


Figure 6.3: Δ_p and HVR box-plots and statistical summary for ZDT2 test problem.

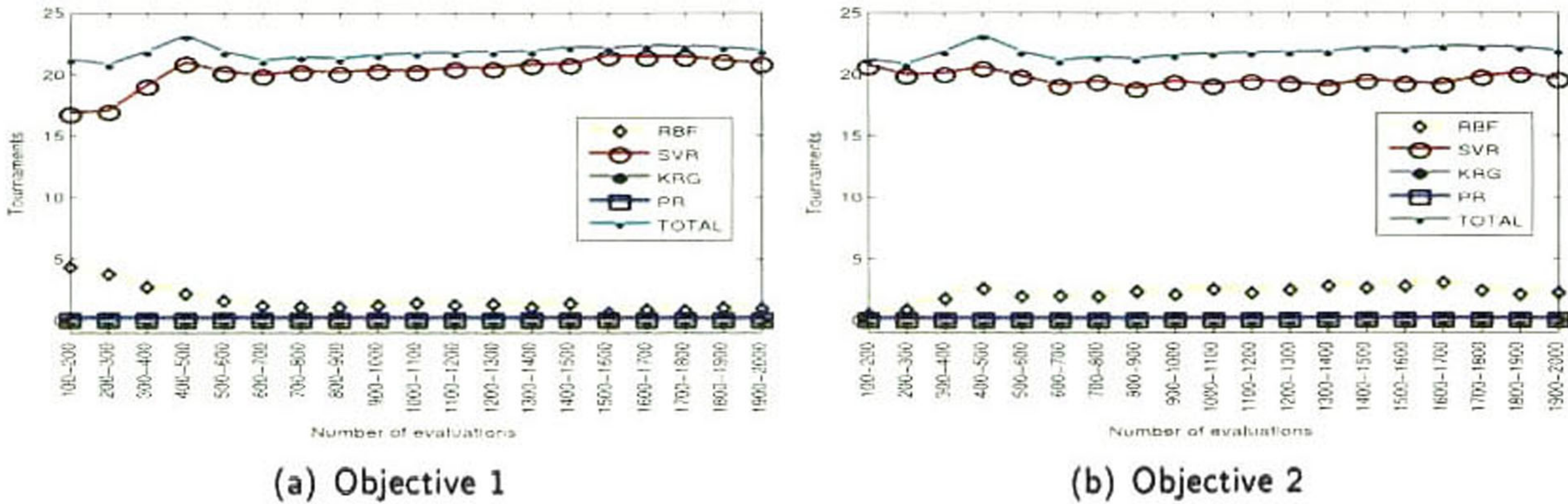


Figure 6.4: Behavior of the TAMAAL using the G-metric and the +E2 method (T+G+E2) in ZDT2.

first evaluations both, KRG and PR were selected. But exceeding 1100 evaluations SVR was the preferred. This interesting behavior can be the reason of the excellent results in this problem.

Clearly, in ZDT6 the NSGA-II produced the worst results in both, Δ_p and HVR (see box-plot of Figure 6.9). In this problem, no differences of the version winner of MASSA compared to TAMAAL versions are visible in the box-plots. However, the MASSA version had the best results over all

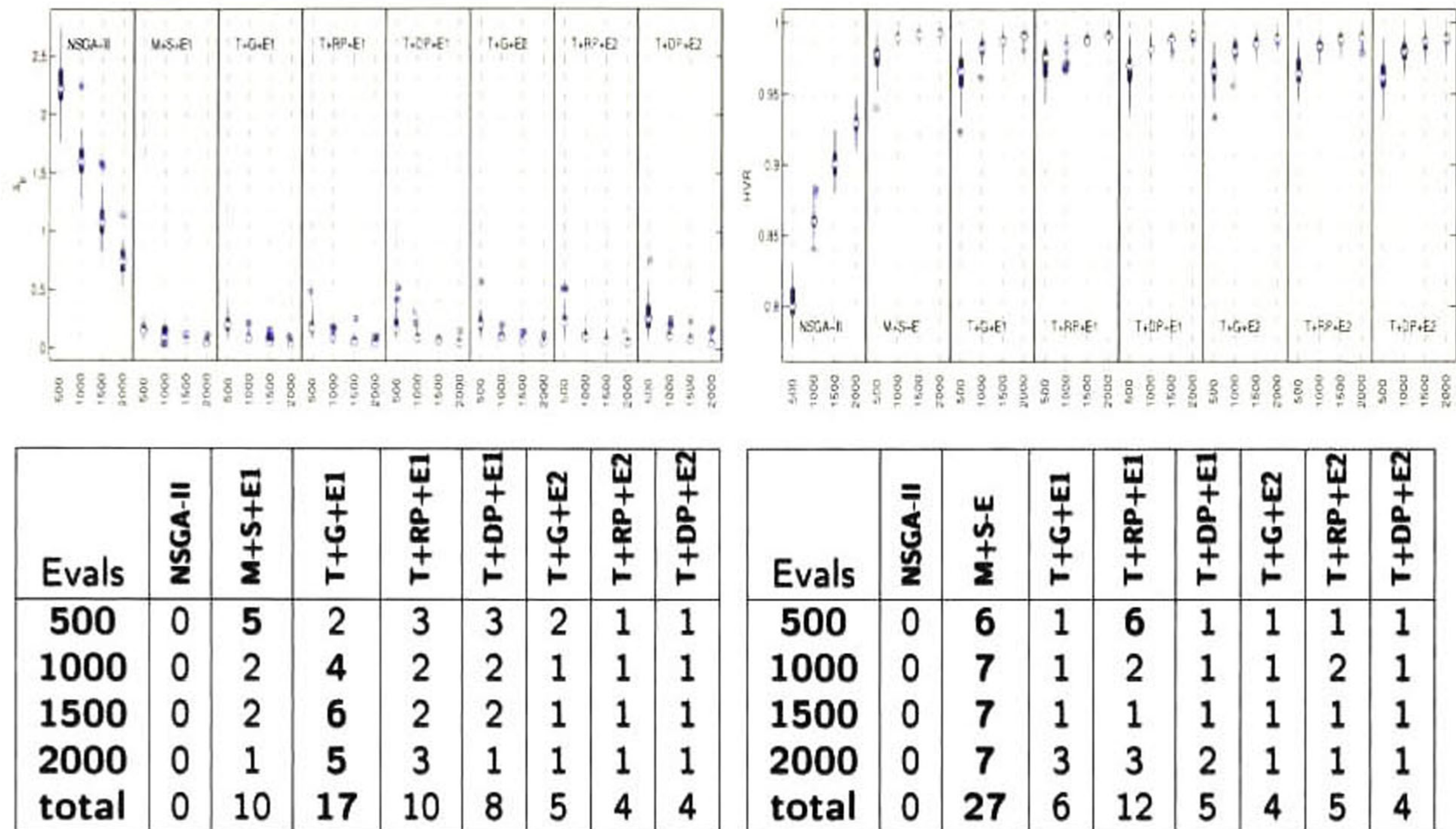


Figure 6.5: Δ_p and HVR box-plots and statistical summary for ZDT3 test problem.

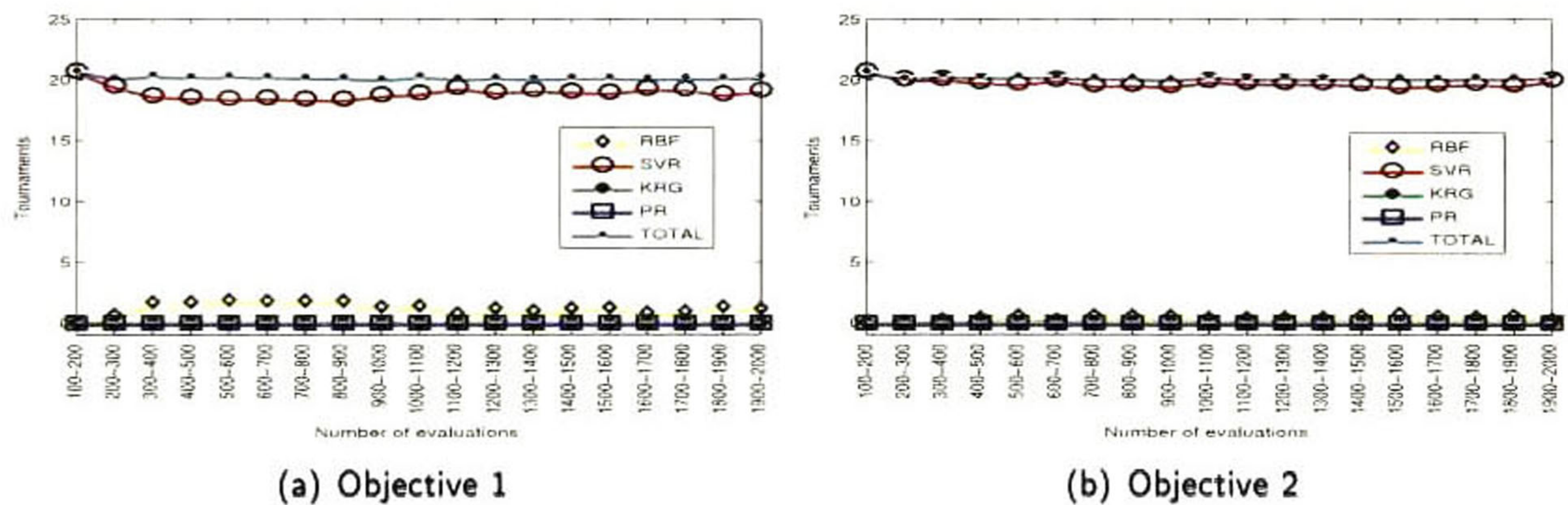


Figure 6.6: Behavior of the TAMAAL using the RP and the +E1 method (T+RP+E1) in ZDT3.

algorithms (see tables on Figure 6.9). Moreover, the best tournament method was the G-metric in both problems, and the best \mathcal{E} - methods were +E2 in Δ_p , and +E1 in HVR, although this last approach was only slightly superior than +E2. Since T+G+E2 is the had the best compromise, then results from this approach are shown in Figure 6.10. In objective 1, the surrogated method most used was KRG, although SVR also was used by a considerable number of tournaments. By contrast,

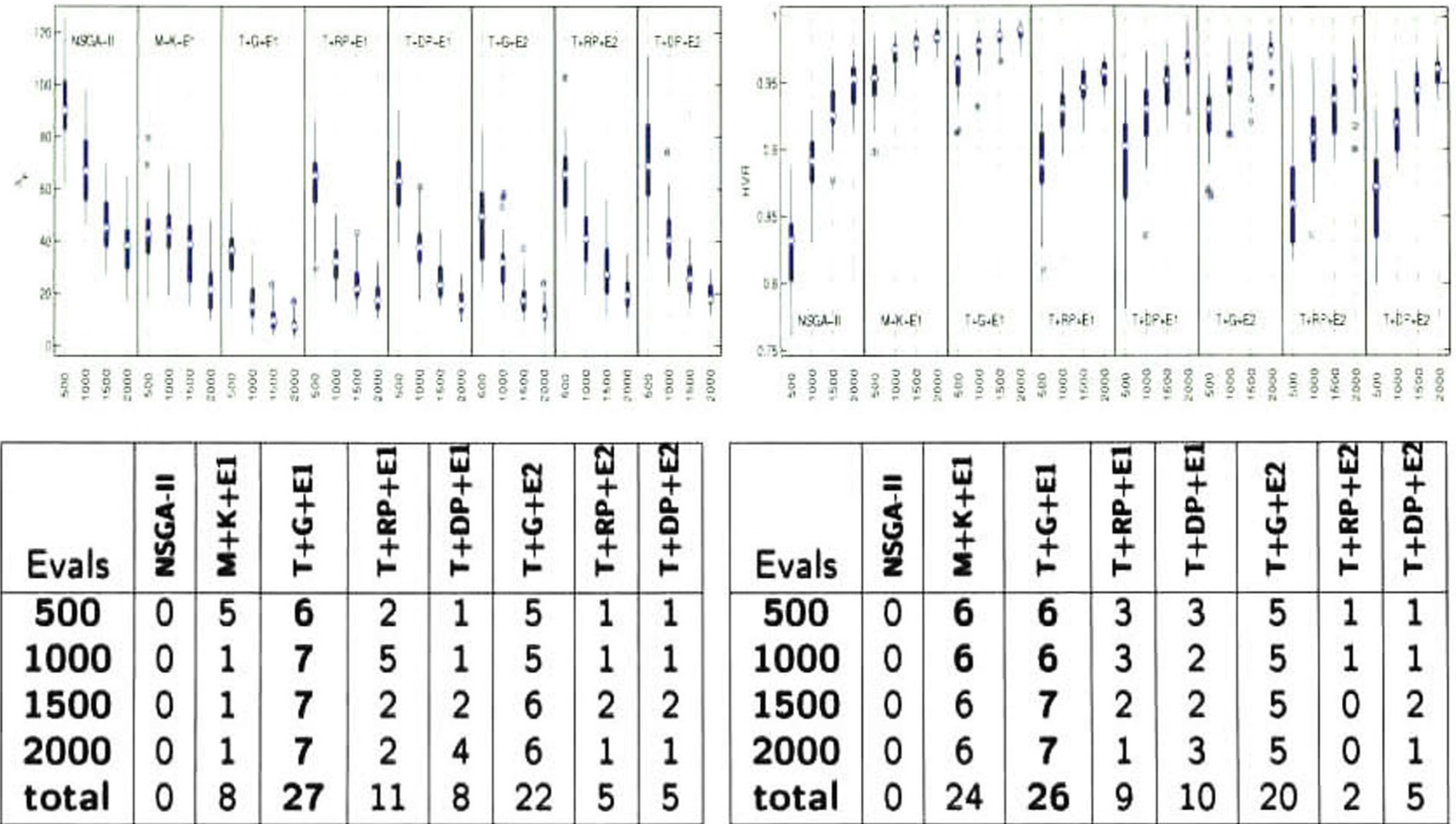


Figure 6.7: Δ_p and HVR box-plots and statistical summary for ZDT4 test problem.

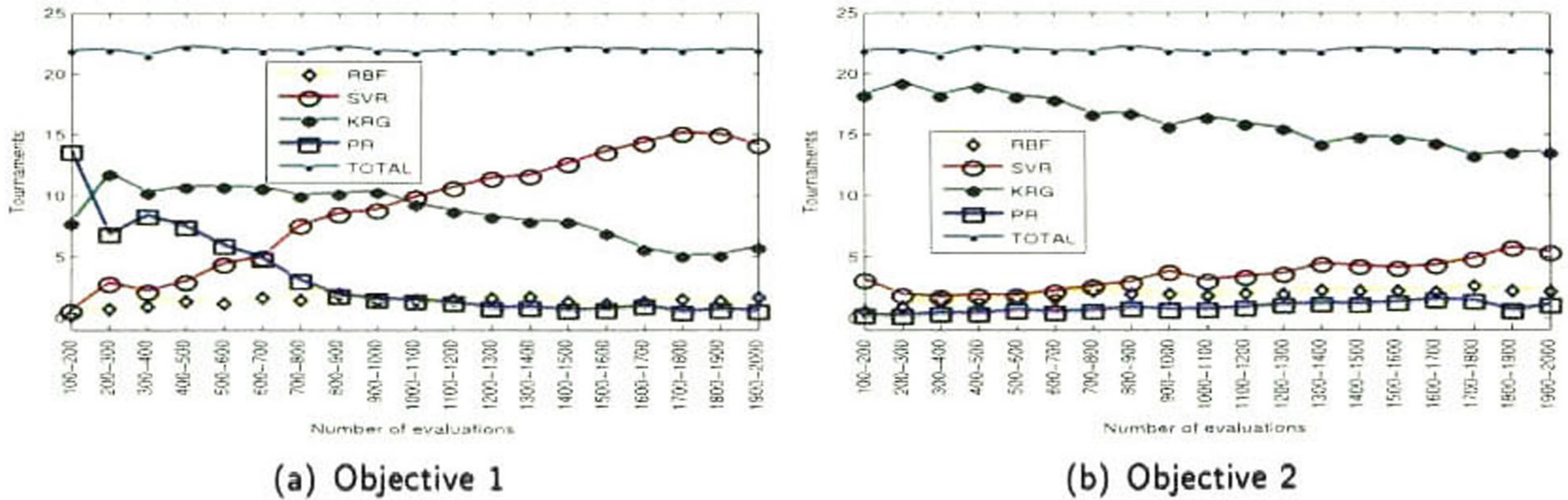


Figure 6.8: Behavior of the TAMAAL using the G-metric and the +E1 method (T+G+E1) in ZDT4.

in the objective 2 the preferred technique was SVR, and KRG was the second place.

Figure 6.11 shows the results for the DTLZ1 test function (according to Δ_p and HVR performance measures). In this problem, the MASSA version M+K+E2 had the best results over all algorithms. The differences between the NSGA-II and the TAMAAL versions are so small that it was not possible to identify who was the best in the box-plots. These results are reinforced in the statistical tables,

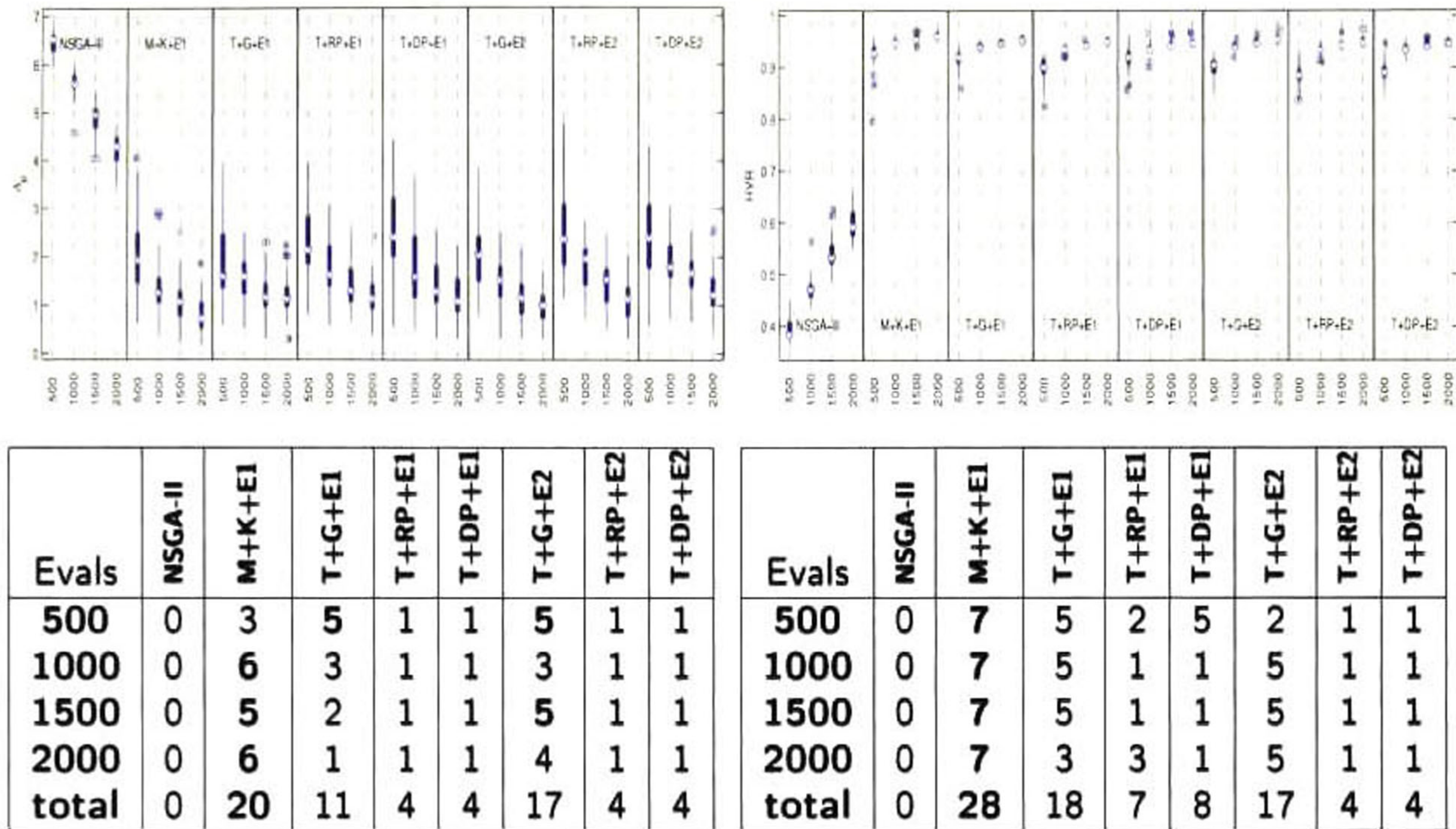


Figure 6.9: Δ_p and HVR box-plots and statistical summary for ZDT6 test problem.

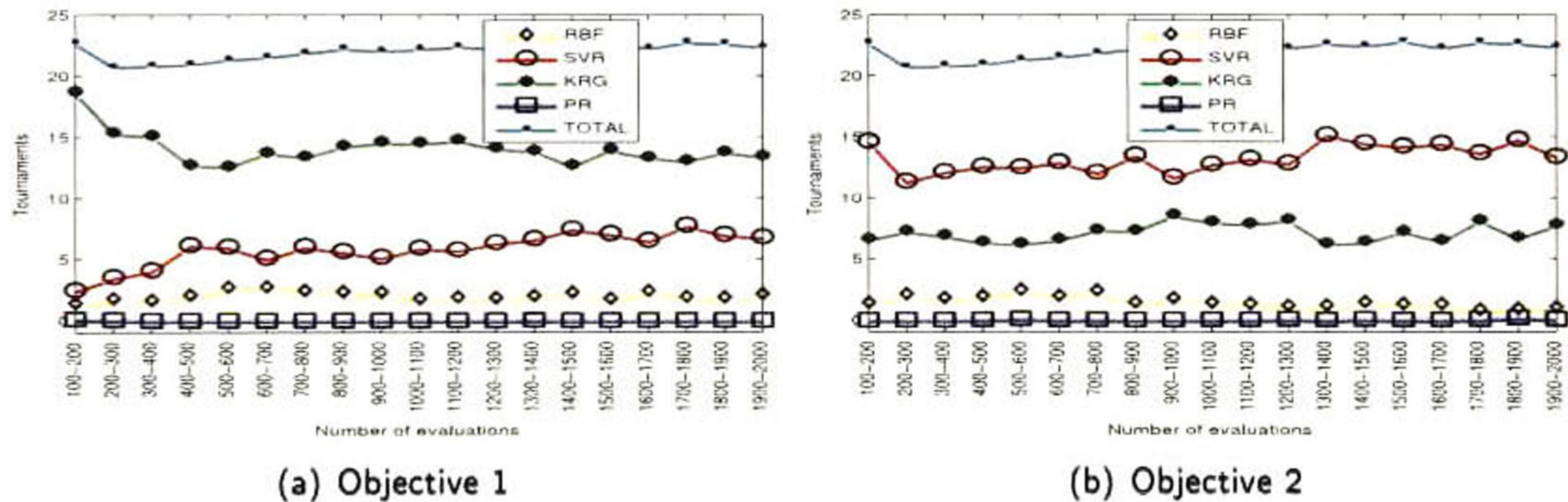


Figure 6.10: Behavior of the TAMAAL using the G-metric and the +E2 method (T+G+E2) in ZDT6.

since from this tables it is clear that algorithms that performed worst were T+G+E1, T+DP+E1 and T+DP+E2. Furthermore, both indicators show that the best tournament method was RP, and the best \mathcal{E} - method was +E2, then results from this approach are shown in Figure 6.12. Such a figure shows that the preferred metamodeling technique at early evaluations was KRG, but in the

subsequent evaluations both approaches behaved similar: SVR and KRG.

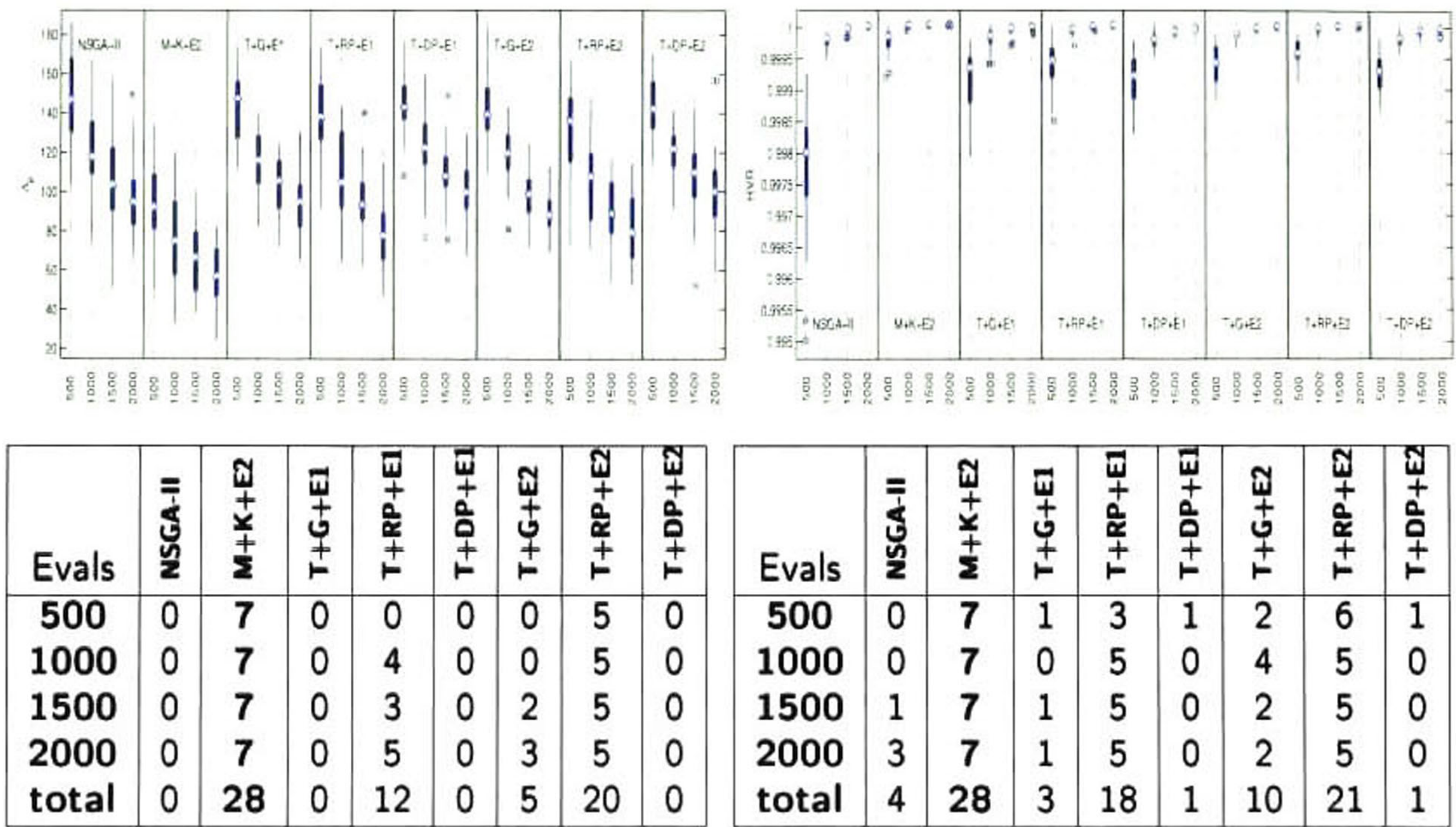


Figure 6.11: Δ_p and HVR box-plots and statistical summary for DTLZ1 test problem.

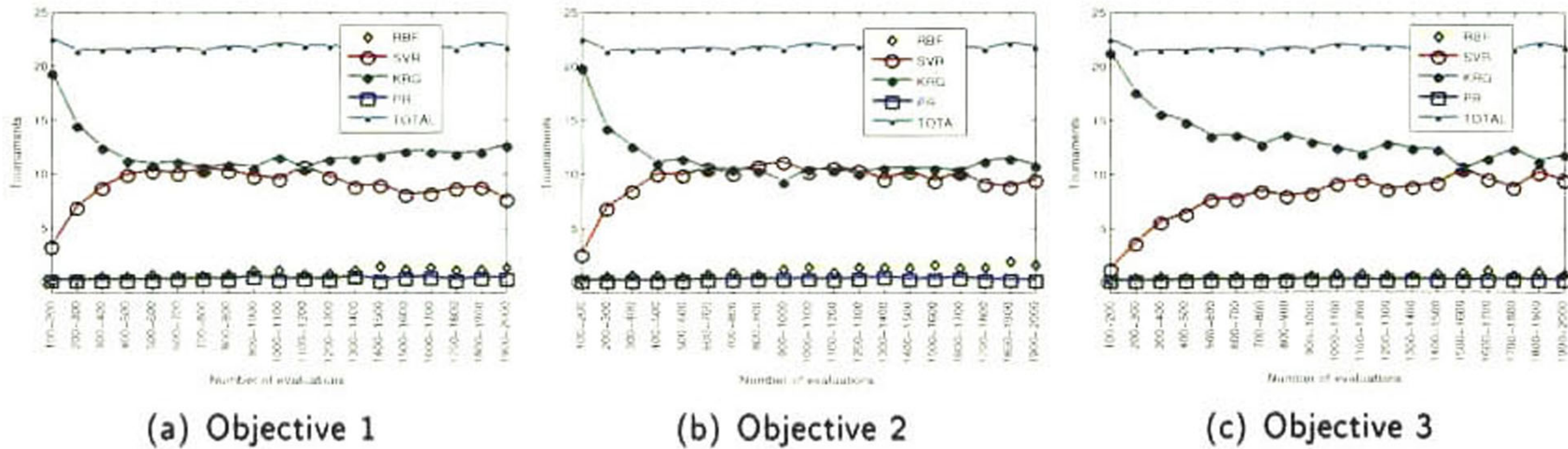


Figure 6.12: Behavior of the TAMAAL using the RP and the +E2 method (T+RP+E2) in DTLZ1.

In DTLZ7 test problem, results from TAMAAL and MASSA were outstanding, while NSGA-II behaved the worst as is easy to see in Figure 6.13. If statistical tables were analyzed, it is easy to find that here the differences between MASSA and TAMAAL are small. However, in Δ_p , the best approach was the MASSA version M+S+E1, but T+G+E1 and T+RP+E1 were very close. In HVR

there was a tie among M+S+E2, T+G+E1, T+DP+E1 and T+DP+E2. Taking into account both indicators, it can be say that the best combination of methods was the G-metric and +E1. Then, in Figure 6.14 the behavior of T+G+E1 in the tournament selection is shown. From this figure, it is easy to see that SVR was the preferred approach in all the objectives.

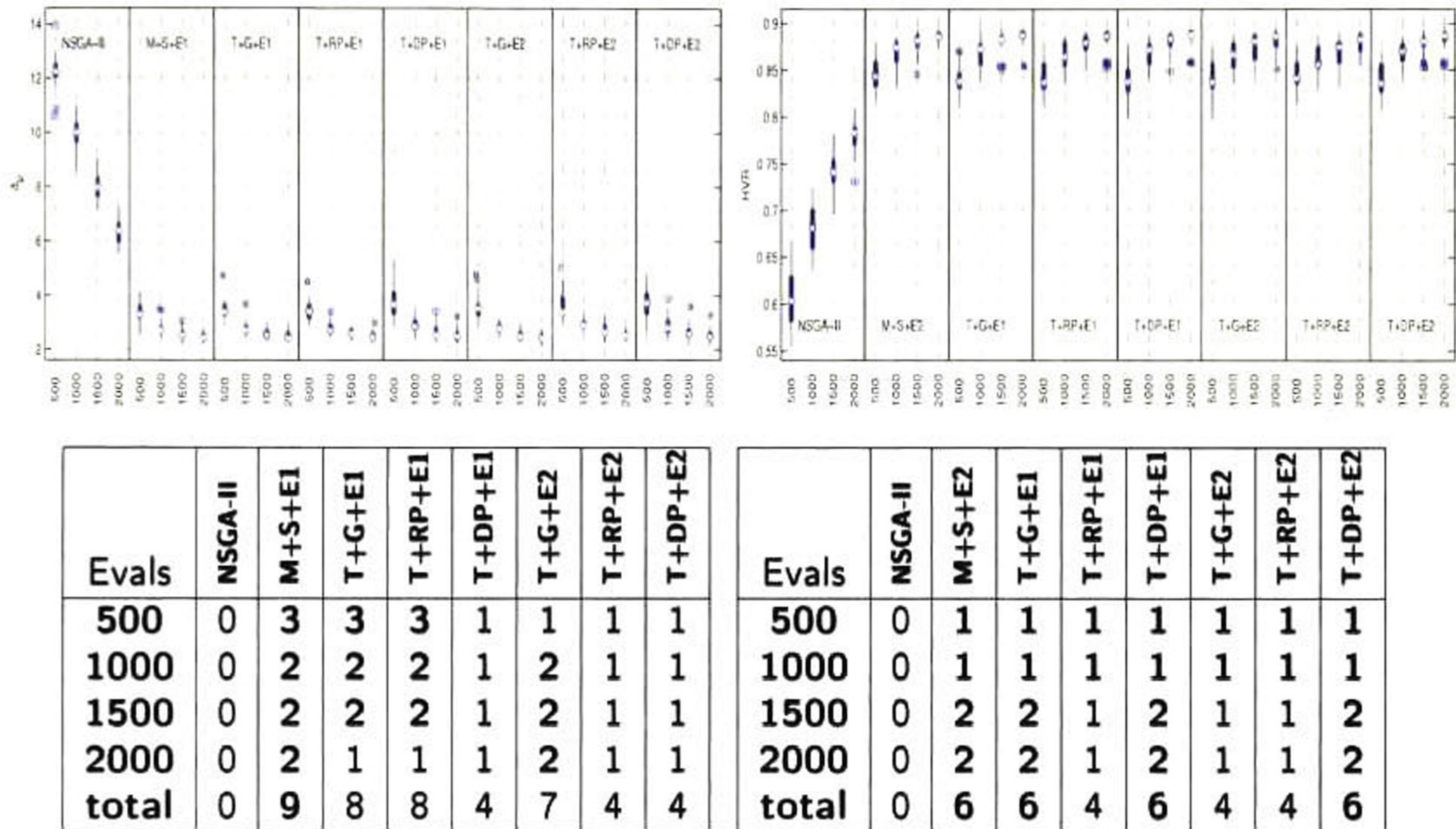


Figure 6.13: Δ_p and HVR box-plots and statistical summary for DTLZ7 test problem.

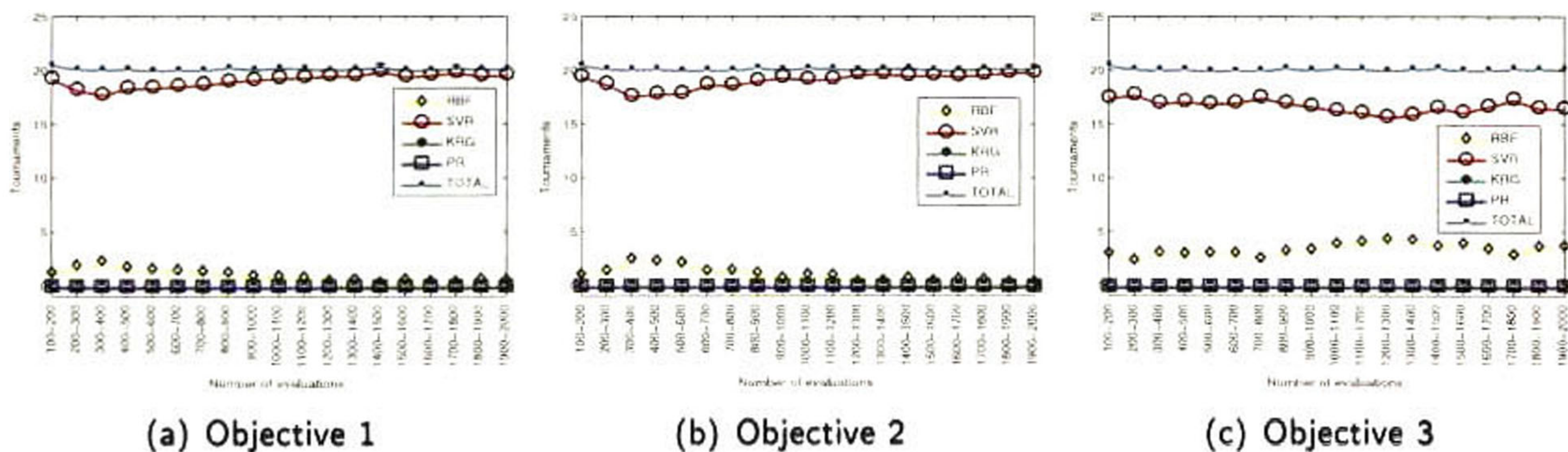
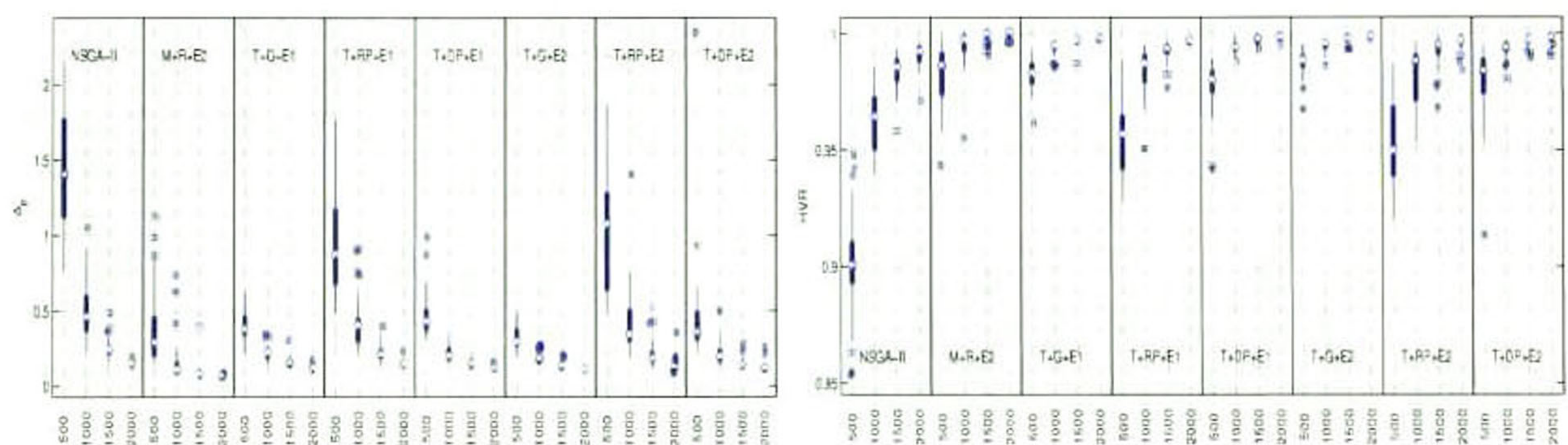


Figure 6.14: Behavior of the TAMAAL using the G-metric and the +E1 method (T+G+E1) in DTLZ7.

Finally, Figure 6.15 show the results for KUR test function in both, Δ_p and HVR. It is clear

that every version of TAMAAL outperformed the NSGA-II. Nevertheless, in box-plots can not see differences regarding to the winner version of the MASSA, but statistical tables show that the T+G+E2 outperform the others in 500 evaluations. However, the MASSA version was the winner in 1000, 1500 and 2000 evaluations. The T+G+E2 is shown in the tournament behavior figures because this algorithm was the best version of TAMAAL (see Figure 6.16). In this sense, in the beginning of the search, KRG was the most selected in both objectives, but in the long run SVR was the preferred approach.



Evals	NSGA-II	M+R+E2	T+G+E1	T+RP+E1	T+DP+E1	T+G+E2	T+RP+E2	T+DP+E2
500	0	4	3	1	3	6	1	3
1000	0	7	3	1	3	5	1	3
1500	0	7	3	1	3	5	1	3
2000	0	7	2	0	2	5	1	3
total	0	25	11	3	11	21	4	12

Evals	NSGA-II	M+R+E2	T+G+E1	T+RP+E1	T+DP+E1	T+G+E2	T+RP+E2	T+DP+E2
500	0	3	3	1	3	6	1	3
1000	0	6	3	1	3	3	1	3
1500	0	7	3	1	3	3	1	3
2000	0	7	3	1	3	4	1	3
total	0	23	12	4	12	16	4	12

Figure 6.15: Δ_p and HVR box-plots and statistical summary for KUR test problem.

Finally, Table 6.1 summarizes the results of the four different number of evaluations and the two indicators, showing the number of the times that an algorithm outperform the others. This table shows that the best \mathcal{E} - method was the +E2 with a score of 46, followed by the +E1 with a score of 40. The best tournament selection method was the G-metric (in both +E1 and +E2). Nevertheless, in some cases if the best versions of MASSA is chosen manually, this will be better than the TAMAAL. However, the best surrogated technique to be used in a given problem is not possible

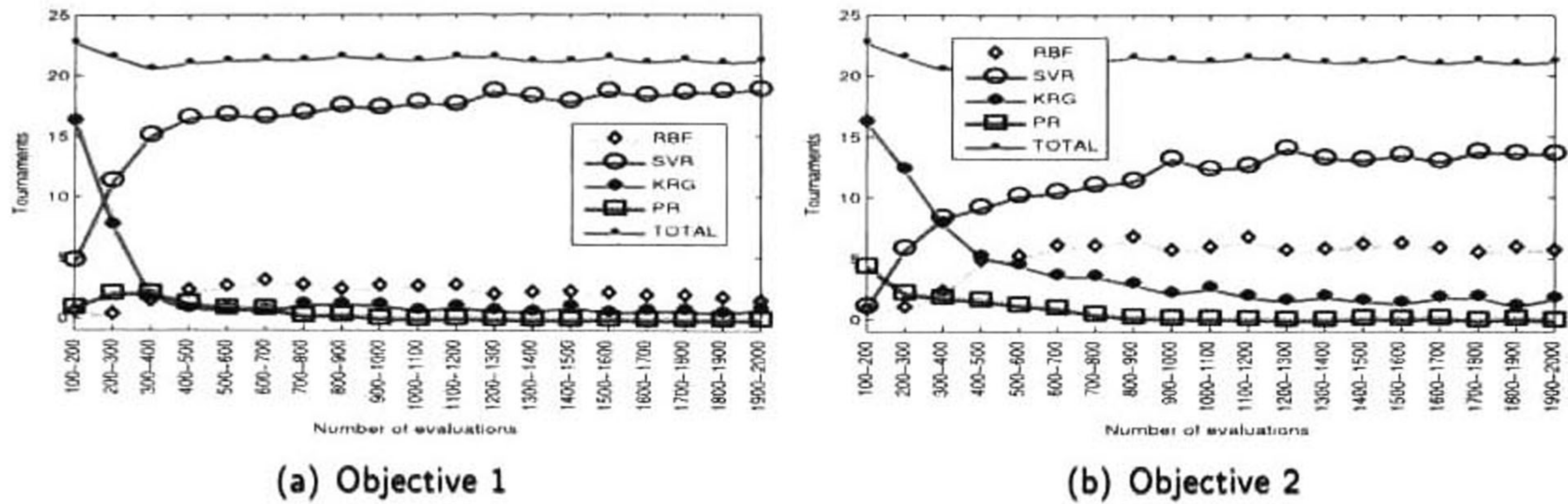


Figure 6.16: Behavior of the TAMAAL using the G-metric and the +E2 method (T+G+E2) in KUR.

to know a-priori. In this way, the T+G+E2 approach was chosen as the representative version of the TAMAAL, which was the most robust algorithm.

Evals	NSGA-II	MASSA	T+G+E1	T+RP+E1	T+DP+E1	T+G+E2	T+RP+E2	T+DP+E2
ZDT1	0	3	3	3	2	7	4	2
ZDT2	0	4	1	1	1	7	7	4
ZDT3	0	5	3	1	0	0	0	0
ZDT4	0	2	8	0	0	0	0	0
ZDT6	0	7	1	0	0	2	0	0
DTLZ1	0	8	0	0	0	0	0	0
DTLZ7	0	8	7	5	4	5	2	4
KUR	0	6	0	0	0	2	0	0
Total by algorithm	0	43	23	10	7	23	13	10
Total by \mathcal{E} - method			40			46		

Table 6.1: Summary table for the TAMAAL on the eight test problems.

6.5 TAMAAL conclusions

This chapter introduced the Tune-adaptive Metamodel Assisted Algorithm, which is an improved version of the MASSA.

Three methods were tested in order to select a surrogated method at a given moment, The G-metric approach, the Ranking Preservation and Dominance Preservation. Results indicate that the G-metric tournament approach produced the best performance among all, and $+E2$ method was the best version. Therefore, T+G+E2 was chosen to be the representative version of TAMAAL algorithm.

Furthermore, results also show that the online-adaptation proposal was successful applied, since it allowed TAMAAL to have outstanding results. This approach will be specially useful in real world problems, since the shape of the landscape is not known *a priori*. Moreover, the objectives of a MOP can have a different landscapes, such that, use separate algorithms is a competitive advantage. Such is the case of ZDT4 problem, where in the second objective the most selected surrogated technique was KRG, while in the first objective SVR was the preferred metamodel.

7

Conclusions and future work

7.1 Conclusions

Results indicate that metamodeling techniques can improve the performance of evolutionary algorithms. However, the selection of the surrogate algorithm plays a key role on the success of the collaboration. In this thesis, four different techniques to approximate functions were studied: Radial Basis Functions (RBF), Support Vector Regression (SVR), Kriging-DACE (KRG) and Polynomial Regression (PR). The main conclusions derived from this study are:

- RBF is a good approach to approximate MOPs, since it showed robust behavior on most of the test problems adopted. This approach behave remarkably well with high dimensionality.
- SVR was the method with the best accuracy in many test problems. However, its robustness was affected by the lack of capacity to approximate a few problems.
- Results from KRG method indicate that this approach presents a poor scalability, since its

performance become severely diminished with the increment in the number of decision variables. However, this method behaved very well when problems with 10 dimensions or less were used.

- PR method obtained the worst results for scalability purposes, since it only could behave moderately acceptable when optimizing problems with less than 10 variables.

Due to the nature of multiobjective optimization, it was possible to evaluate a method for approximating functions not only for its accuracy, but also using alternative techniques such as Ranking Preservation and Dominance Preservation.

Moreover, identify areas where it is worth that the surrogate algorithm perform the exploitation is an important issue. These areas should be modeled accurately using well-distributed solutions (since the metamodel is not reliable in poorly populated areas).

The preliminary study made possible to propose the Metamodel Assisted Subpopulation-based Search Algorithm (MASSA). Results from this algorithm show that previous evaluated solutions strengthen the knowledge of the current population and improve the accuracy of metamodeling techniques. Two different methods were tested in order to identify the best approach to maintain previous solutions in the archive (the \mathcal{E}). The first approach removes the oldest solutions when the \mathcal{E} is full, while the second method retains those solutions better distributed throughout the variable space. Results indicate that the latter method prevailed.

Furthermore, as long as the metamodeling technique is right chosen, it is possible to reduce the number of evaluations in the real function. However, this is not a task that can be performed by hand, since when optimizing real world problems the shape of the fitness landscape is not known *a priori*.

An important observation when comparing different metamodeling techniques, is that there is not a single approach that outperforms the others for all test functions. Therefore, a scheme for the automatic selection of the metamodel to be used at any given time was needed. The Tune-adaptive Metamodel Assisted Algorithm tries to solve this requirement (TAMAAL). TAMAAL

implement a tournament in order to select the best of four methods for constructing metamodels. The selected method is responsible for the prediction the values of individuals in the surrogated process. Three tournament methods were tested (through the G-metric, Dominance Preservation and Ranking Preservation). For this experiment, results indicate that the G-metric perform the best.

The proposed TAMAAL approach could reduced the number of evaluations required to achieve the Pareto front in the eight test problems (ZDT1-4, ZDT6, DTLZ1, DTLZ7 and KUR).

7.2 Limitations and future work

There are some ideas that for time limitations were not completed:

- An obvious limitation in these proposals is that there is not any explicit mechanism for handle constraints. Therefore, this is an extension needed.
- Another major limitation of MASSA and TAMAAL is the response time. That is, the time to train metamodeling techniques (specifically KRG and PR) is very large, specially when the number of dimensions is high. Therefore, it is necessary to evaluate other approaches to approximate functions.
- Moreover, an area in which evolutionary computing research have supported to speed the response time on problems whose objective functions are computationally expensive is parallelization. Due the way that MASSA and TAMAAL are built, both can easily take advantage of parallel architecture. Therefore a parallel implementation is suggested.
- In literature there is not any research that studied metamodeling techniques when the number of objectives is raised. Therefore, a study covering this issue could be very useful in the multiobjective optimization area.
- The set of problems chosen to evaluate the proposals of this thesis do not include computationally expensive objective functions. In this sense, it would be interesting to conduct

a study with real world problems with expensive objective functions that allow to observe the efficiency of the proposed approaches.



Metamodeling techniques

Below, a brief description of the most popular techniques to construct metamodels is presented.

A.1 Kriging

Kriging is a spatial prediction method based on the mean squared error (MSE) minimization. Design and Analysis of Computer Experiments (DACE) is a parametric regression model developed by Sacks *et al.* [Sacks *et al.*, 1989] that emerged as an extension of the Kriging method to handle more than two objectives.

Suppose that a deterministic function of k variables has to be evaluated in n points. In Kriging method, it is assumed that the correlation between errors is related to the distance between the corresponding points, *i.e.*, assuming that the function that has been modeled is continuous, the real values in the points \vec{x}_i and \vec{x}_j ($y(\vec{x}_i)$ y $y(\vec{x}_j)$) tend to be close if the distance $\|\vec{x}_i - \vec{x}_j\|$ is small. This can be modeled statistically saying that the random variables $\epsilon(\vec{x}^{(i)})$ and $\epsilon(\vec{x}^{(j)})$ are highly correlated if $\|\vec{x}_i - \vec{x}_j\|$ is small. Thus, it is assumed that the correlation between two random variables is:

$$\text{corr}(\epsilon(\vec{x}^{(i)}), \epsilon(\vec{x}^{(j)})) = \exp[-d(\vec{x}^{(i)}, \vec{x}^{(j)})] \quad (\text{A.1})$$

where the distance $d(\vec{x}^{(i)}, \vec{x}^{(j)})$ is:

$$\sum_{h=1}^k \theta_h |x_h^{(i)} - x_h^{(j)}|^{p_h} \quad (\theta_h \geq 0, p_h \in [1, 2]) \quad (\text{A.2})$$

This correlation function has the property that if the point $\vec{x}^{(i)} = \vec{x}^{(j)}$, then the correlation is 1. Similarly, when $\|\vec{x}^{(i)} - \vec{x}^{(j)}\| \rightarrow \infty$, the correlation tends to zero. The θ_h parameter determines how fast the correlation falls when moving to the h^{th} direction. Larger values of θ_h are used to model functions that are highly active in the h^{th} variable. For such variables, function values can change quickly even at small distances. The p_h determines the smoothness of the function in h^{th} direction. Values of p_h closer to 2 help to smooth the function, while values closer to 0 make it more rough.

The Kriging method needs $2k + 2$ parameters ($\mu, \sigma^2, \theta_1, \dots, \theta_k$ and p_1, \dots, p_k), which describe how the objective function typically behaves. An advantage in this technique is that the $2k + 2$ parameters can be estimated choosing those parameters that maximize the following *likelihood function* of the observed data:

$$\frac{1}{(2\pi)^{n/2} (\sigma^2)^{n/2} \det(\mathbf{R})^{1/2}} \exp\left(-\frac{(\vec{y} - \vec{1}\mu)' \mathbf{R}^{-1} (\vec{y} - \vec{1}\mu)}{2\sigma^2}\right) \quad (\text{A.3})$$

Note that Equation A.3 involves a vector $\vec{y} = (y^{(1)}, \dots, y^{(n)})$ of observed values and another $n \times 1$ vector of ones $\vec{1}$. Note also that it is involved a correlation matrix \mathbf{R} , which denotes a $n \times n$ matrix whose value (i, j) is $\text{corr}(\epsilon(\vec{x}^{(i)}), \epsilon(\vec{x}^{(j)}))$. This matrix involves the $2k + 2$ parameters.

In practice it is more convenient to choose the model parameters by maximizing the log-likelihood function:

$$-\frac{n}{2} \log(\sigma^2) - \frac{1}{2} \log(\det \mathbf{R}) - \frac{(\vec{y} - \vec{1}\mu)' \mathbf{R}^{-1} (\vec{y} - \vec{1}\mu)}{2\sigma^2} \quad (\text{A.4})$$

Setting the derivatives of this expression with respect to μ and σ^2 to zero and solving, the optimal values of σ^2 and μ can be expressed as functions of \mathbf{R} :

$$\hat{\mu} = \frac{\bar{\mathbf{1}}' \mathbf{R}^{-1} \bar{\mathbf{y}}}{\bar{\mathbf{1}}' \mathbf{R}^{-1} \bar{\mathbf{1}}} \quad (\text{A.5})$$

$$\hat{\sigma}^2 = \frac{(\bar{\mathbf{y}} - \bar{\mathbf{1}} \hat{\mu})' \mathbf{R}^{-1} (\bar{\mathbf{y}} - \bar{\mathbf{1}} \hat{\mu})}{n} \quad (\text{A.6})$$

Substituting Equations (A.5) and (A.6) into Equation (A.4) the so-called *concentrated log-likelihood* function is derived. Ignoring constant terms, the concentrated log-likelihood function is:

$$-\frac{n}{2} \log(\sigma^2) - \frac{1}{2} \log(\det \mathbf{R}) \quad (\text{A.7})$$

The concentrated log-likelihood function depends only on \mathbf{R} and, hence, on the correlation parameters θ 's and p 's. In practice, this is the function that is maximized to obtain estimates of $\hat{\theta}_h$ and \hat{p}_h ($h = 1, \dots, k$).

Once found the $2k$ values that maximize the likelihood function, it is possible to predict the value of a point $\bar{\mathbf{x}}^*$ without having to evaluate in the original function. This is done as follows:

$$\hat{y}(\bar{\mathbf{x}}^*) = \hat{\mu} + \bar{\mathbf{r}}' \mathbf{R}^{-1} (\bar{\mathbf{y}} - \bar{\mathbf{1}} \hat{\mu}) \quad (\text{A.8})$$

where element i of $\bar{\mathbf{r}}$ is $r_i(\bar{\mathbf{x}}^*) = \text{corr}(\epsilon(\bar{\mathbf{x}}^*), \epsilon(\bar{\mathbf{x}}^{(i)}))$.

Readers interested in a more detailed explanation of the Kriging method may consult [Sacks et al., 1989, Jones, 2001].

A.2 Radial basis functions neural networks

Artificial Neural Networks (ANN) or simply Neural Networks (NN) represent a simplified approach to model biological neural systems. The NN are based on learning the characteristics of the samples of the input space which stores in its structure. The procedure by which the network is capable of learning is called training or learning algorithm.

Broomhead and Lowe [Broomhead, 1988] were the first in explore the radial basis functions

(RBF) as activation functions in NN. However, the current topology of a Radial Basis Function Neural Network (RBFNN) was exposed in the work of Poggio and Girosi [Poggio and Girosi, 1989]. The RBFNN has an architecture without feedback and a single hidden layer (as shown in Figure A.1). The number of nodes in the input layer is equal to k independent variables of the problem. The hidden layer consists of m neurons with RBFs ϕ_i as activation functions, whose connections with the input nodes encode the centers of the RBFs. Thus, the definition of the weights of the hidden layer can be seen intuitively as the position of nodes in the space of input data. The r neurons in the output layer just perform a weighted sum of the activations of the hidden layer, with weights defined by the vector w .

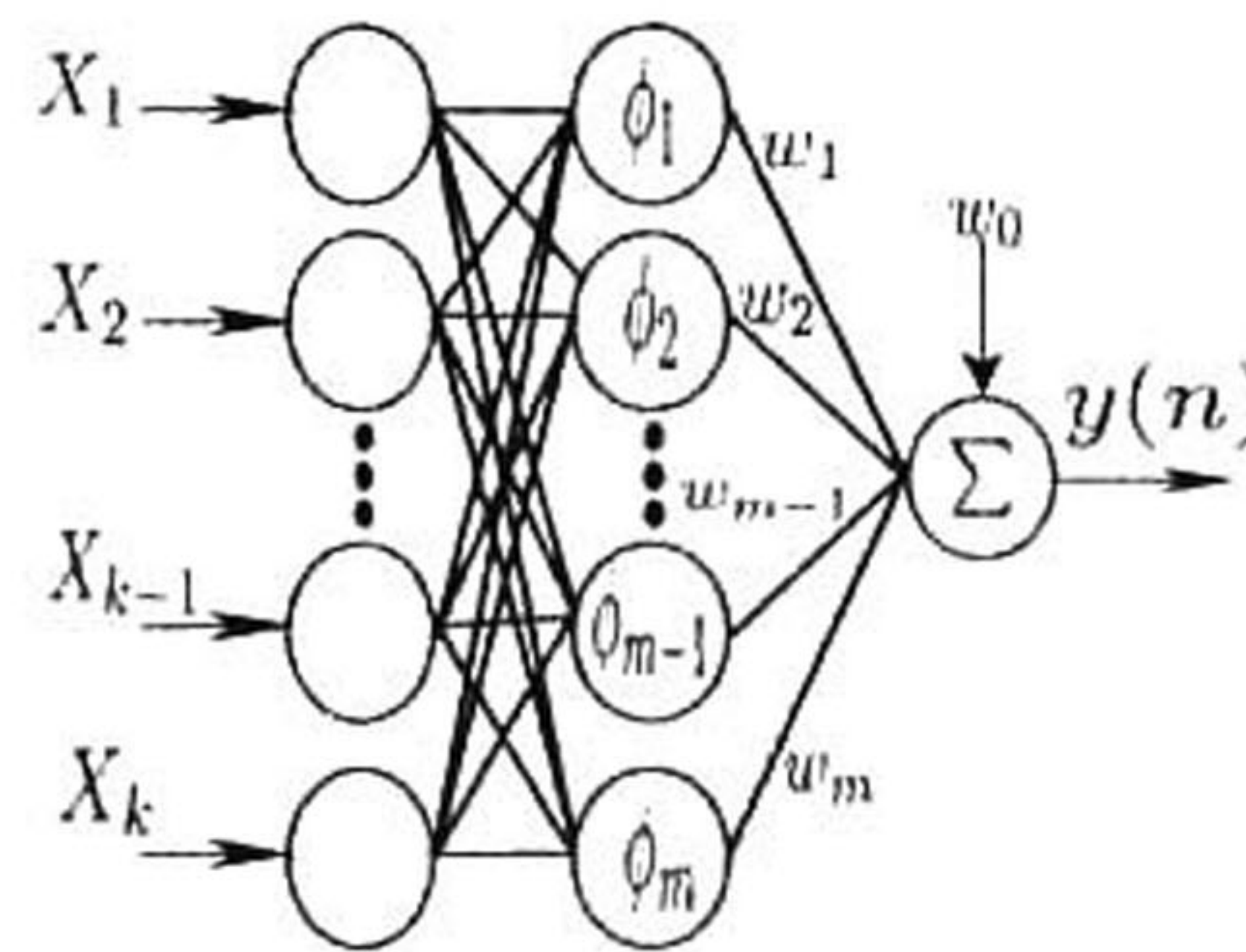


Figure A.1: Three layer topology of a RBF

The RBF's output can be obtained with the expression shown in Equation A.9:

$$\hat{y}(n) = w_0 + \sum_{i=1}^m w_i \phi_i(n), \quad (\text{A.9})$$

where w_0 is the bias of the output layer and ϕ_i is the radial basis function, which is usually the Gaussian function.

In the algorithm 2, the outline to implement a classical RBF is shown. This approach has two initial parameters. First, the number of centers in the hidden layer and second the location of each center.

To determine the width σ^2 of the RBF, it is possible to use the average distance to the p nearest neighbors, defined as:

Algorithm 2 Outline of RBF algorithm

- 1: Select the initial number of centers.
- 2: Select the location of each center.
- 3: For each input vector input/center calculate the activation value.
- 4: Once the activation values were calculated for each input vector, calculate the weights of the connections between the hidden layer and output layer using a supervised learning method.
- 5: Calculate the weighted sum defined in Equation (A.9).

$$\sigma_i^2 = \frac{1}{p} \sum_{j=1}^p \|C_i - C_j\| \quad (\text{A.10})$$

where $C_j (j = 1, \dots, p)$ are the p centers closer to the corresponding center C_i . For purposes of this thesis a value of $p = 2$ is used.

In the second phase of training, a supervised learning method is used to calculate the weights of the connections between hidden and output layers. The pseudo-inverse matrix method is used in this thesis. This method provides a direct solution to the optimization problem and is given by the following matrix expression:

$$W = G^+ S = (G^T G)^{-1} G^T S \quad (\text{A.11})$$

where W is the weighted and *bias* matrix of the RBF, of size $(m + 1) \times r$, such that:

$$W = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1r} \\ w_{21} & w_{22} & \cdots & w_{2r} \\ \vdots & \vdots & & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mr} \\ u_1 & u_2 & \cdots & u_r \end{pmatrix},$$

G is a matrix of size $N \times (m + 1)$ containing the activations of hidden neurons for the samples of the input space:

$$G = \begin{pmatrix} \phi_1(1) & \phi_2(1) & \cdots & \phi_m(1) & 1 \\ \phi_1(2) & \phi_2(2) & \cdots & \phi_m(2) & 1 \\ \vdots & \vdots & \cdot & \vdots & \vdots \\ \phi_1(N) & \phi_2(N) & \cdots & \phi_m(N) & 1 \end{pmatrix},$$

where $\phi_i(n)$ is the activation of hidden neuron, i , for the input sample $x(n)$; and S is the desired output matrix of size $N \times r$:

$$S = \begin{pmatrix} s_1(1) & s_2(1) & \cdots & s_r(1) \\ s_1(2) & s_2(2) & \cdots & s_r(2) \\ \vdots & \vdots & \cdot & \vdots \\ s_1(N) & s_2(N) & \cdots & s_r(N) \end{pmatrix},$$

where $s_l(n)$ is the coordinate l of the desired output for the input sample $x(n)$.

A.3 Polynomial regression

The regression analysis is a methodology that studies the quantitative association between a function of interest f and N_{BF} basis functions z_j , where there are N_s sample values of the function of interest f_i for a set of basis functions $z_j^{(i)}$. For each observation i a linear equation is formulated:

$$f_i(\mathbf{z}) = \sum_{j=1}^{N_{BF}} \beta_j z_j^{(i)} + \varepsilon_i, E(\varepsilon_i) = 0, V(\varepsilon_i) = \sigma^2 \quad (\text{A.12})$$

where the errors ε_i are considered independent with expected value equal to zero, variance σ^2 , and $N_{BF} = (k+1)(k+2)/2$.

In the case of a second order polynomial, the response to the k input variables can be seen as:

$$\hat{y}(\vec{x}) = \beta_0 + \sum_{i=1}^k \beta_i x_i + \sum_{i=1}^k \sum_{j=1}^k \beta_{ij} x_i x_j \quad (\text{A.13})$$

where $\vec{\beta}$ is unknown. However, the set of equations specified in (A.12) can be expressed in matrix

form as:

$$\vec{y} = X\vec{\beta} + \varepsilon, E(\varepsilon_i) = 0, V(\varepsilon_i) = \sigma^2\mathbf{I} \quad (\text{A.14})$$

where \mathbf{X} is an $N_s \times N_{BF}$ matrix of basis functions with the design variable values for sampled points. Solving $\vec{\beta}$ through the pseudo-inverse matrix method:

$$\hat{\vec{\beta}} = (X^T X)^{-1} X^T \mathbf{y} \quad (\text{A.15})$$

Thus, substituting the coefficients of the vector $\hat{\vec{\beta}}$ of Equation (A.15) in Equation (A.13) the approximate response value for a not evaluated vector \mathbf{x} is obtained.

A.4 Support vector machines for regression

Support vector machines (SVMs) were introduced by Vladimir Vapnik *et al.* in [Cortes and Vapnik, 1995]. SVMs are supervised learning methods that analyze data and recognize patterns. The standard SVM takes a set of input data and predicts, for each given input, which possible classes it corresponds. The SVM are inspired from statistical learning theories, and its major advantage is that there is not local minimum during learning and the generalization error does not depend on the dimensions of the space. SVMs can also be applied to regression problems by the introduction of an alternative loss function [Gunn, 1998]. A widely used loss function is $\varepsilon - SVR$, whose goal is to find a function $f(x)$ that has at most ε deviation from the corresponding targets y_i for all the training data and at the same time is as flat as possible.

Consider the problem of approximating the set of data with a linear function,

$$f(x) = \langle w, x \rangle + b \text{ with } w \in X, b \in \mathbb{R} \quad (\text{A.16})$$

the formulation of the regression problem is given by:

$$\begin{aligned} & \text{minimize} \quad \left(\frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) \right) \\ & \text{subject to} \quad \begin{cases} y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \end{aligned} \quad (\text{A.17})$$

where C is a pre-specified value and ξ, ξ^* are slack variables representing upper and lower constraints on the outputs of the system.

B

Test functions

To evaluate the performance of the proposals, eight scalable unconstrained global multiobjective test problems were selected from the specialized literature taking into account both, the number of local minima and the shape of the Pareto front, containing characteristics that are representative of what can be considered as “difficult” in multiobjective optimization problems:

- Zitzler-Deb-Thiele test suite [Zitzler et al., 2000]: ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6.
- Two Deb-Thiele-Laumanns-Zitzler test functions [Deb et al., 2002]: DTLZ1 and DTLZ7.
- The Kursawe test function [Kursawe, 1991]: KUR.

The eight test functions are to be minimized. KUR and the ZDTs test functions are scalable in the design variable space, and DTLZs functions are also scalable in the objective function space. These functions were selected taking into account the number of local minima and the shape of the Pareto front, characteristics that are representative of what can be considered as “difficult” in multiobjective optimization. A brief description of the eight functions is given below.

For each test function a summary of the main characteristics, the formal definition of the function, and a figure of the true Pareto front as well as 30,000 random solutions are presented.

B.1 ZDT1

This test function was proposed by *Zitzler et al* in [Zitzler et al., 2000]. It is a uni-modal bi-objective test function, the shape of the Pareto front is convex, and its range of values are in $[0, 1]$. The formal definition of the test problem is presented in Equation B.1. Figure B.1 shows the graphical results produced by 30,000 randomly generated solutions (with circles) and the true Pareto front (shown in stars).

$$\begin{aligned} f_1(\vec{x}) &= x_1 \\ f_2(\vec{x}) &= g(\vec{x}) \times h(f_1(\vec{x}), g(\vec{x})) \end{aligned}$$

such as:

$$\begin{aligned} g(\vec{x}) &= 1 + \frac{9}{n-1} \sum_{i=2}^n x_i \\ h(f_1, g, \vec{x}) &= 1 - \sqrt{f_1(\vec{x})/g(\vec{x})} \end{aligned} \tag{B.1}$$

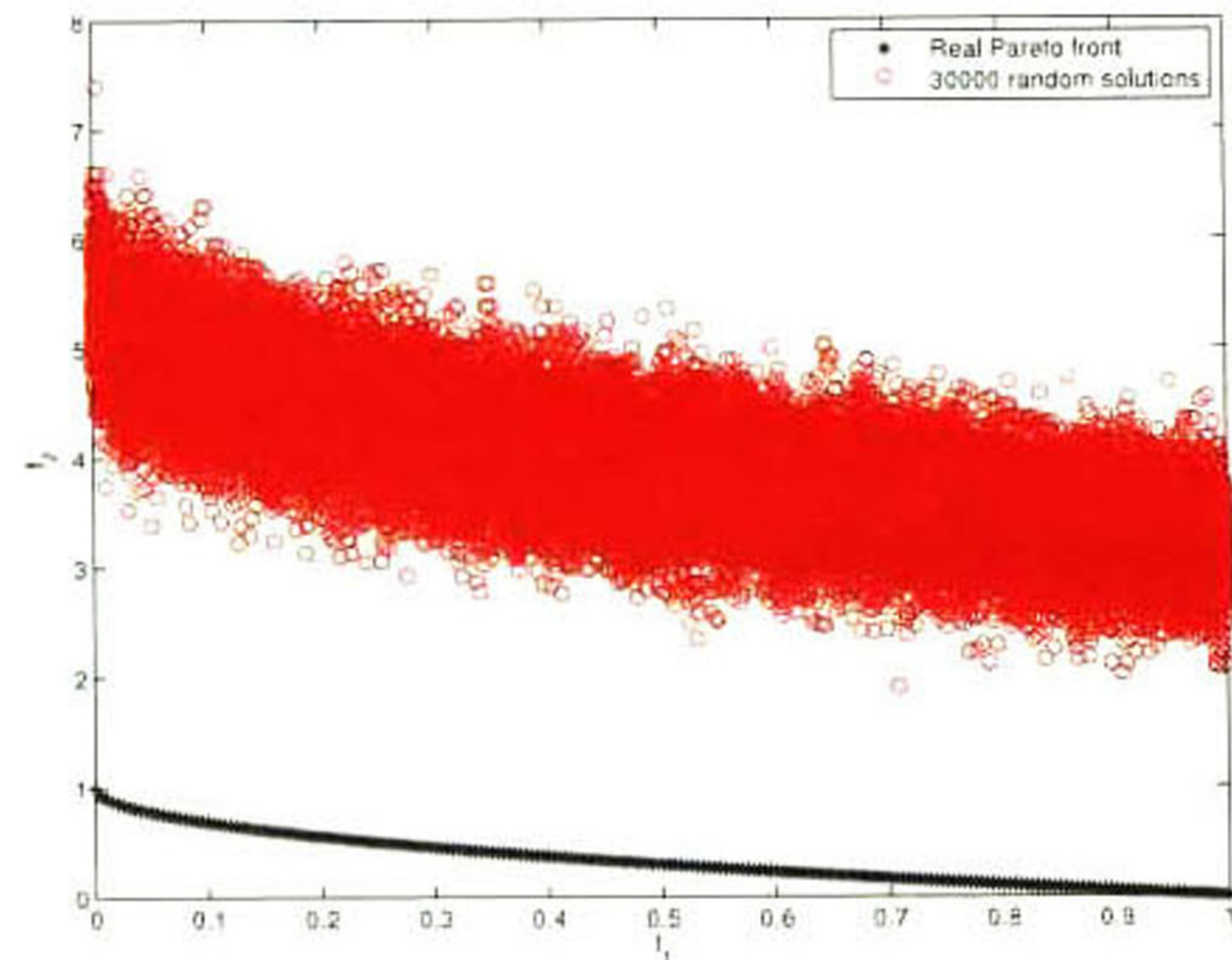


Figure B.1: Real Pareto front of the ZDT1 test function and 30,000 random solutions.

B.2 ZDT2

This unimodal test function was proposed by *Zitzler et al* in [Zitzler et al., 2000]. Its range of values are in the domain $[0, 1]$. The formal definition of functions is presented in Equation B.2. Figure B.1 shows the graphical results produced by 30,000 randomly generated solutions (with circles) and the true Pareto front (shown in stars).

$$f_1(\vec{x}) = x_1$$

$$f_2(\vec{x}) = g(\vec{x}) \times h(f_1(\vec{x}), g(\vec{x}))$$

such as:

$$g(\vec{x}) = 1 + \frac{9}{n-1} \sum_{i=2}^n x_i$$

$$h(f_1, g, \vec{x}) = 1 - (f_1(\vec{x})/g(\vec{x}))^2$$
(B.2)

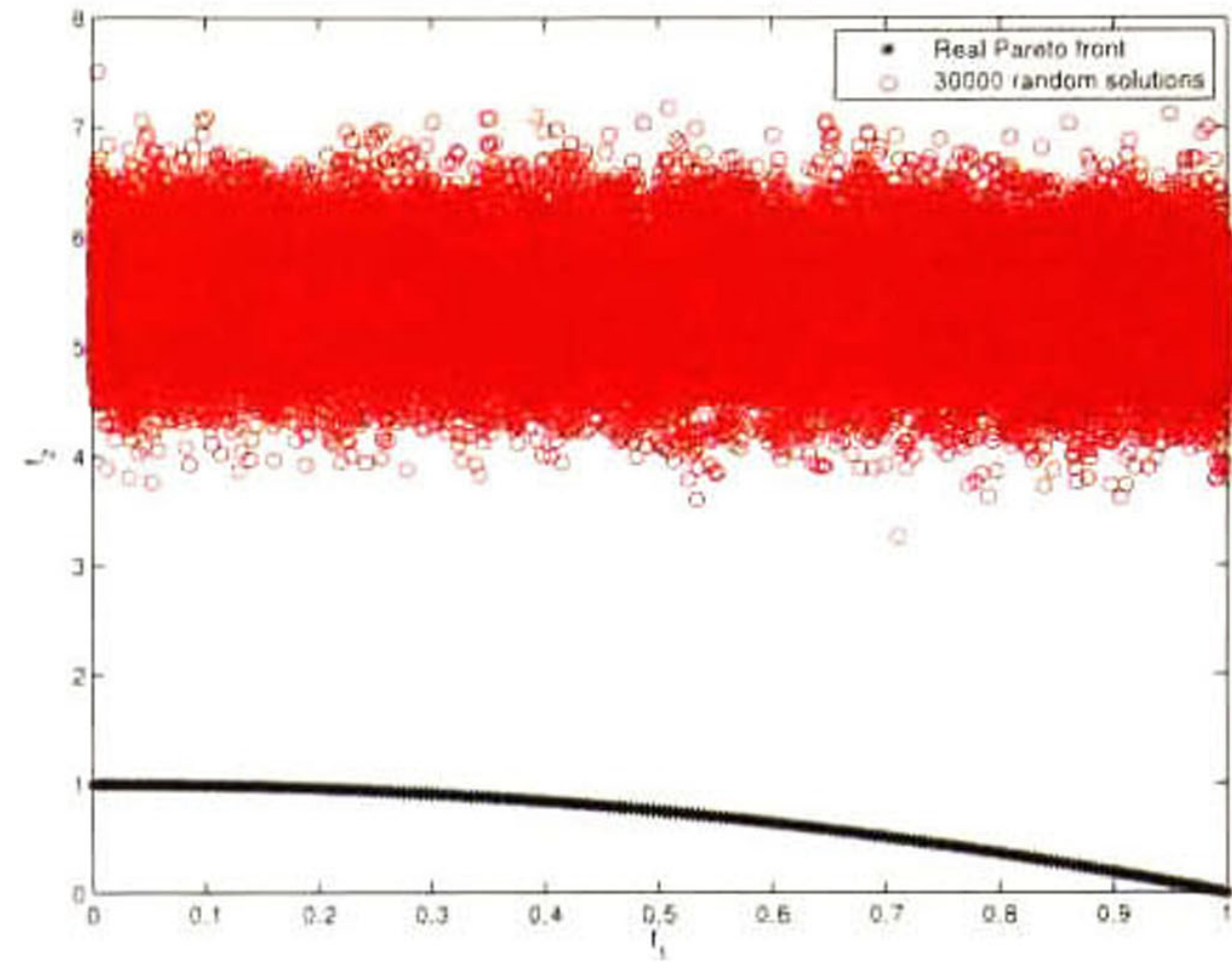


Figure B.2: Real Pareto front of the ZDT2 test function and 30,000 random solutions.

B.3 ZDT3

The ZDT3 function [Zitzler et al., 2000] is uni-modal in the first objective and multi-modal in the second objective. Its true Pareto front consists of five disconnected convex parts as is easy to see in Figure B.3. This figure also shows 30,000 randomly generated solutions (plotted in circles). Its range of values are also in the domain $[0, 1]$. Its formal definition is presented in Equation B.3.

B.4 ZDT4

The ZDT4 test function [Zitzler et al., 2000] is uni-modal in the first objective and multi-modal in the second one. It has a convex and continuous Pareto front, and its range of values are in the domain $[0, 1]$. Moreover, this function has 21^9 local Pareto fronts. In Equation B.4, the formal definition is shown. In this test function, the random solutions are far from the real Pareto front (see Figure B.4). For this reason, a zoom at solutions belonging to the real Pareto front is included in the bottom right side of that image.

$$f_1(\vec{x}) = x_1$$

$$f_2(\vec{x}) = g(\vec{x}) \times h(f_1(\vec{x}), g(\vec{x}))$$

such as:

$$g(\vec{x}) = 1 + \frac{9}{n-1} \sum_{i=2}^n x_i$$

$$h(f_1, g, \vec{x}) = 1 - \sqrt{f_1(\vec{x})/g(\vec{x})} - (f_1(\vec{x})/g(\vec{x})) \sin(10\pi f_1(\vec{x}))$$

(B.3)

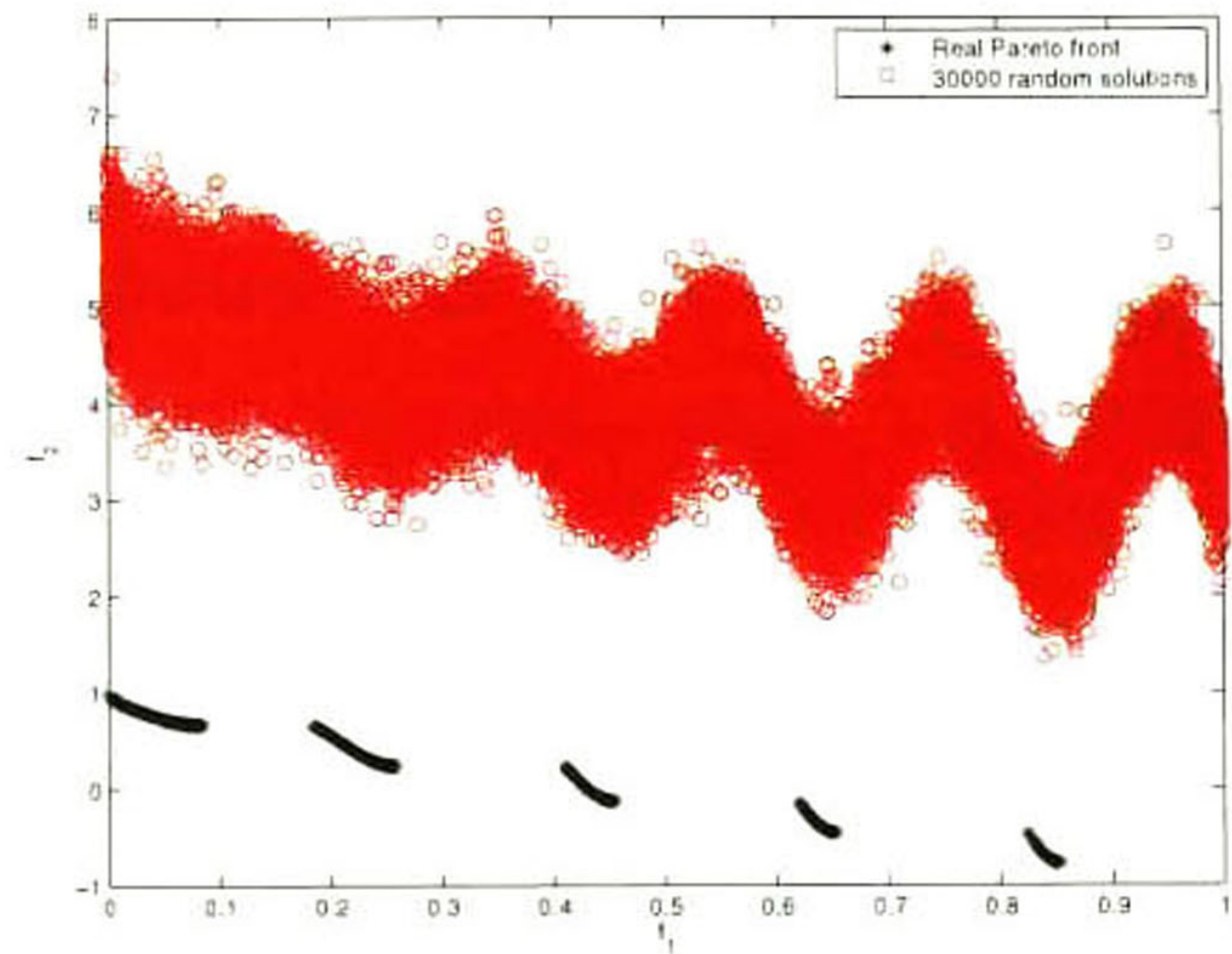


Figure B.3: Real Pareto front of the ZDT3 test function and 30,000 random solutions.

$$f_1(\vec{x}) = x_1$$

$$f_2(\vec{x}) = g(\vec{x}) \times h(f_1(\vec{x}), g(\vec{x}))$$

such as:

$$g(\vec{x}) = 1 + 10(n-1) + \sum_{i=2}^n (x_i^2 - 10 \cos(4\pi x_i))$$

$$h(f_1, g, \vec{x}) = 1 - \sqrt{f_1(\vec{x})/g(\vec{x})}$$

(B.4)

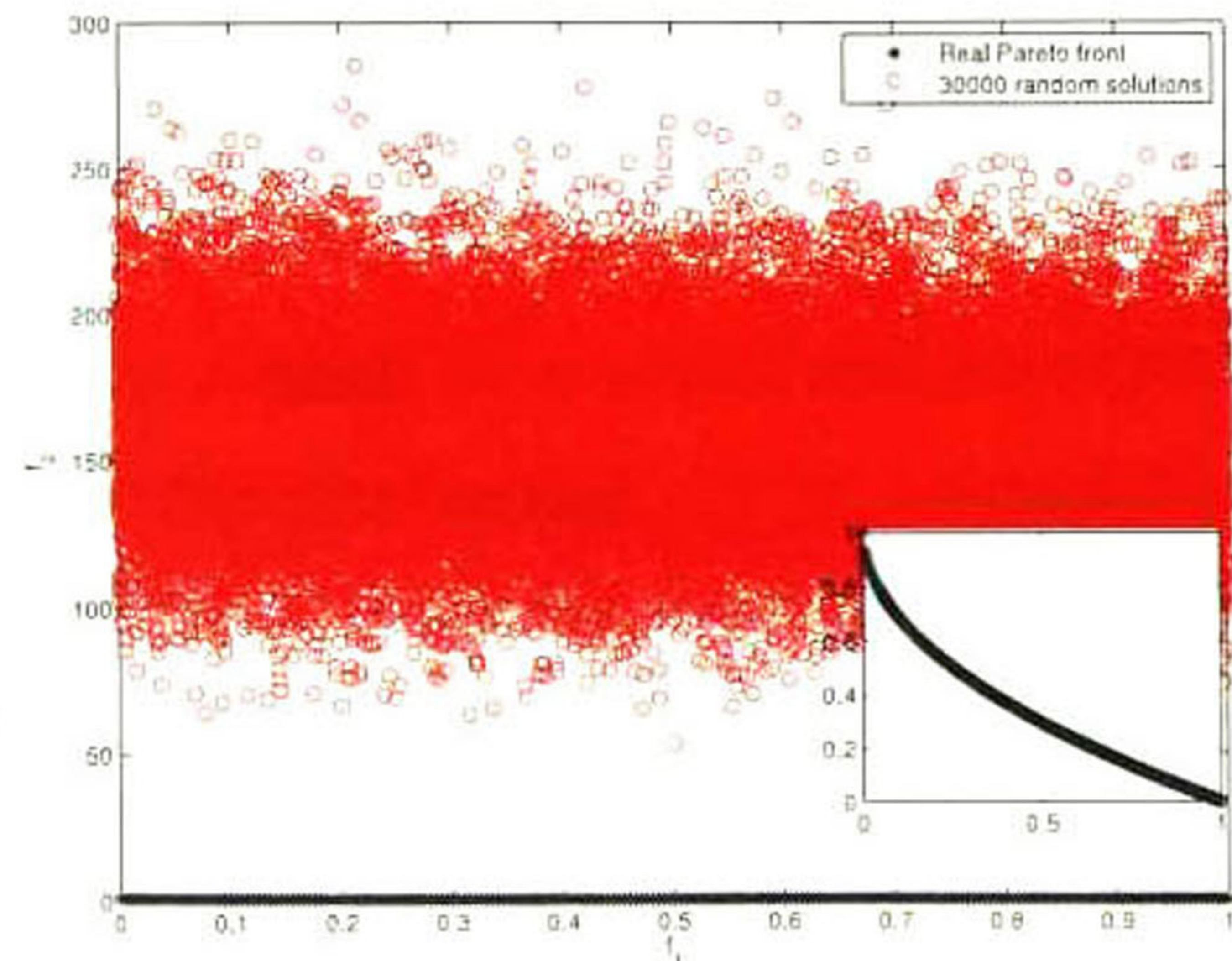


Figure B.4: Real Pareto front of the ZDT4 test function and 30,000 random solutions.

B.5 ZDT6

The ZDT6 test function is multi-modal in both objectives, its shape on the objective space is convex as is easy to see in Figure B.5. This figure also shows 30,000 randomly generated points. Its range of values are in the domain $[0, 1]$. The formal definition is presented in Equation B.5.

$$f_1(\vec{x}) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$$

$$f_2(\vec{x}) = g(\vec{x}) \times h(f_1(\vec{x}), g(\vec{x}))$$

such as:

$$g(\vec{x}) = 1 + 9 \left(\frac{1}{n-1} \sum_{i=2}^n x_i \right)^{0.25}$$

$$h(f_1, g, \vec{x}) = 1 - (f_1(\vec{x})/g(\vec{x}))^2 \tag{B.5}$$

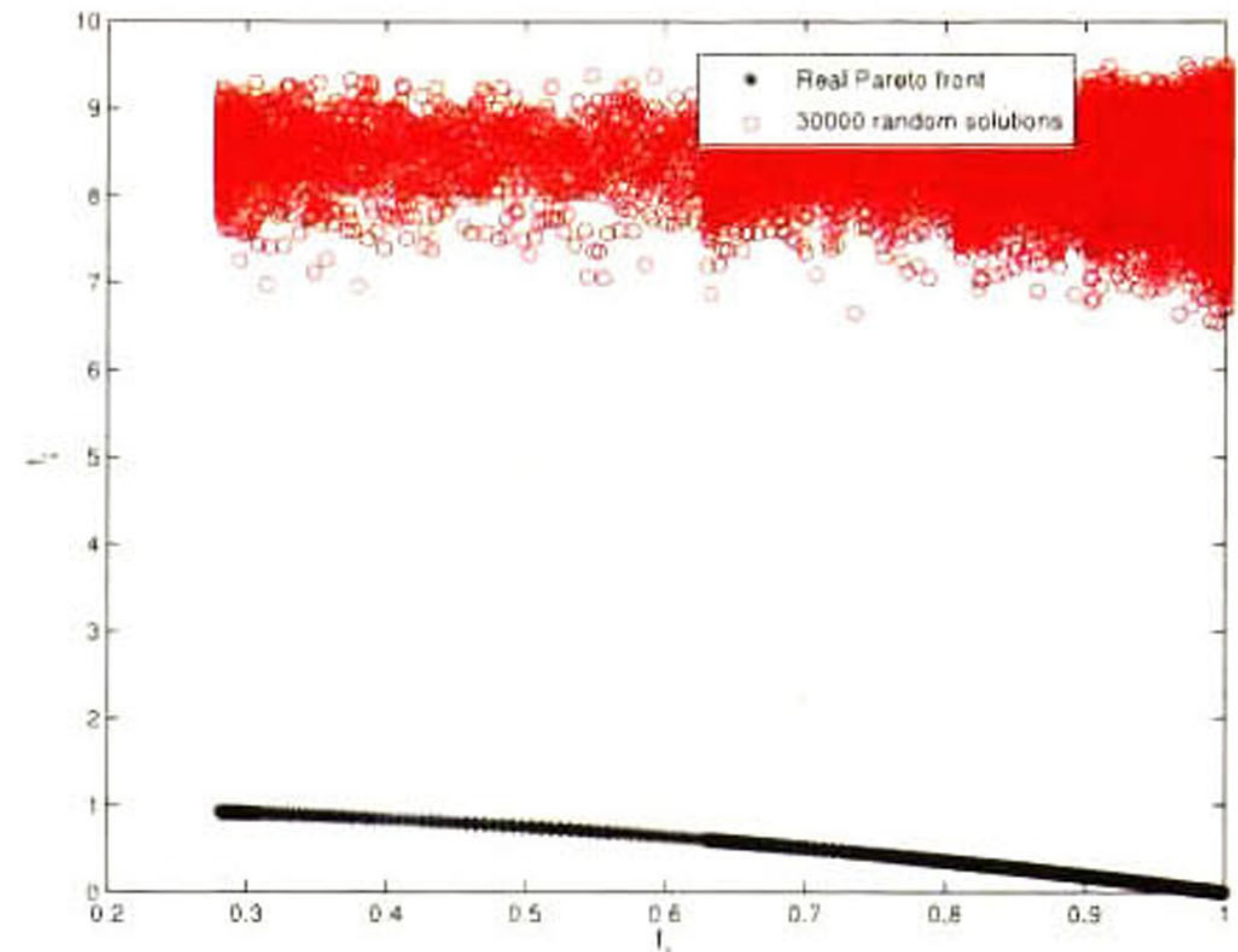


Figure B.5: Real Pareto front of the ZDT6 test function and 30,000 random solutions.

B.6 Kursawe

The bi-objective Kursawe function (KUR) has a disconnected Pareto front with four parts (see Figure B.6). It is uni-modal in the first objective and multi-modal in the second one. Its range of values are in the domain $[0, 1]$. In Equation B.6, the formal definition is presented.

$$f_1(\vec{x}) = \sum_{i=1}^{n-1} -10 \exp^{-0.2 \sqrt{x_i^2 + x_{i+1}^2}}$$

$$f_2(\vec{x}) = \sum_{i=1}^{n-1} (|x_i|^{0.8} + 5 \sin(x_i^3)) \tag{B.6}$$

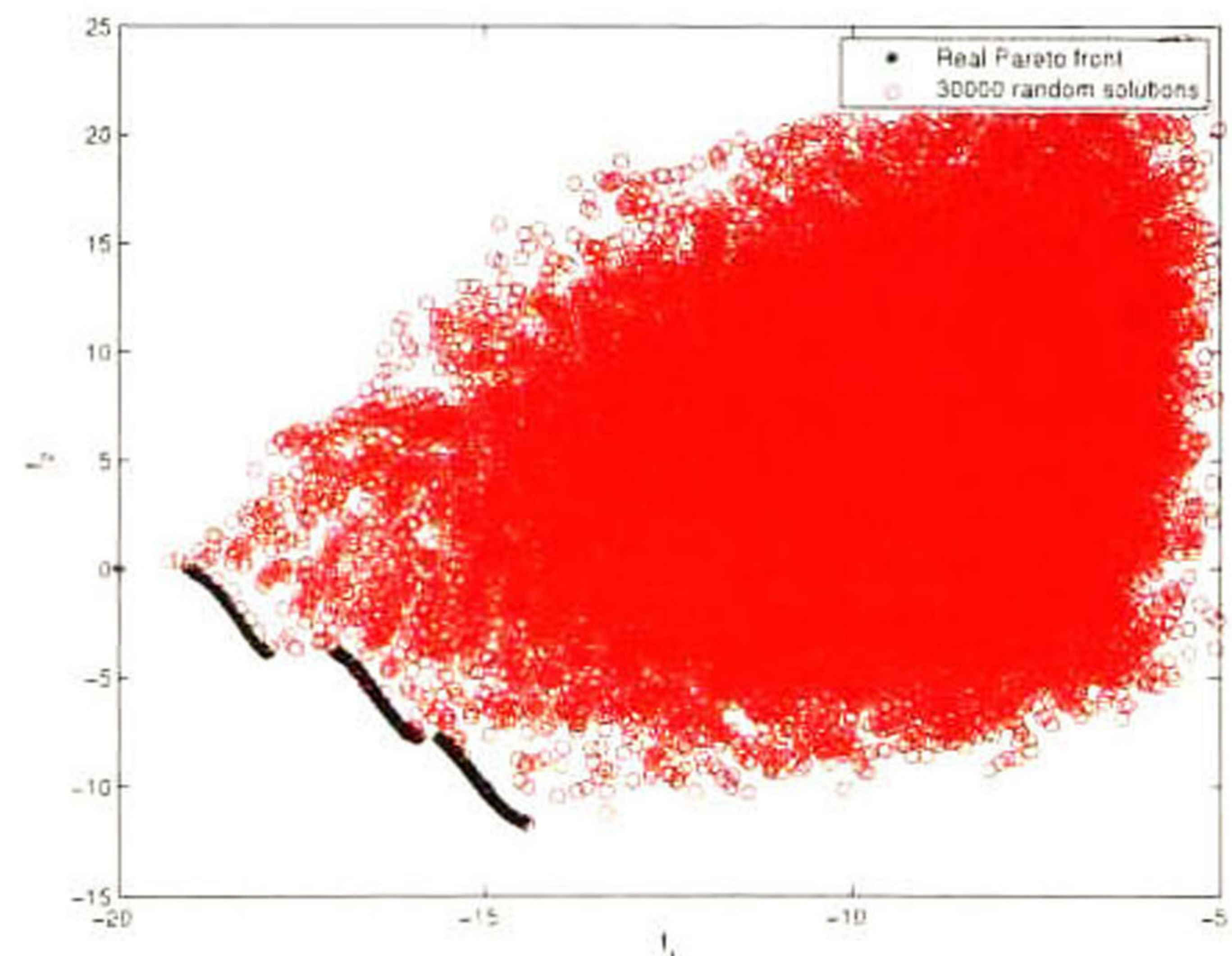


Figure B.6: Real Pareto front of the KUR test function and 30,000 random solutions.

B.7 DTLZ1

In addition to being scalable in the number of variables, the DTLZ family is also scalable in the space of objective functions. DTLZ1 test function was proposed by *Deb et al* in [Deb et al., 2002]. The test problem presents a big search space and $11^5 - 1$ local attractors, therefore (it is multi-modal in all objectives). Its range of values are also in the domain $[0, 1]$, and the formal definition is presented in Equation B.7. The initial random solutions are far from the real Pareto front, for this reason a zoom at solutions belonging to the real Pareto front is included in the bottom right side of the image (see Figure B.7).

$$f_1(\vec{x}) = 0.5(1 + g(\vec{x}))\left(\prod_{i=1}^{K-1} x_i\right)$$

$$f_{k=2:K-1}(\vec{x}) = 0.5(1 + g(\vec{x}))\left(\prod_{i=1}^{K-k} x_i\right)$$

$$(1 - x_{K-k+1})$$

$$f_K(\vec{x}) = 0.5(1 + g(\vec{x}))(1 - x_1)$$

such as:

$$g(\vec{x}) = 100[(n - M + 1) + \sum_{i=K}^n ((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)))]$$

(B.7)

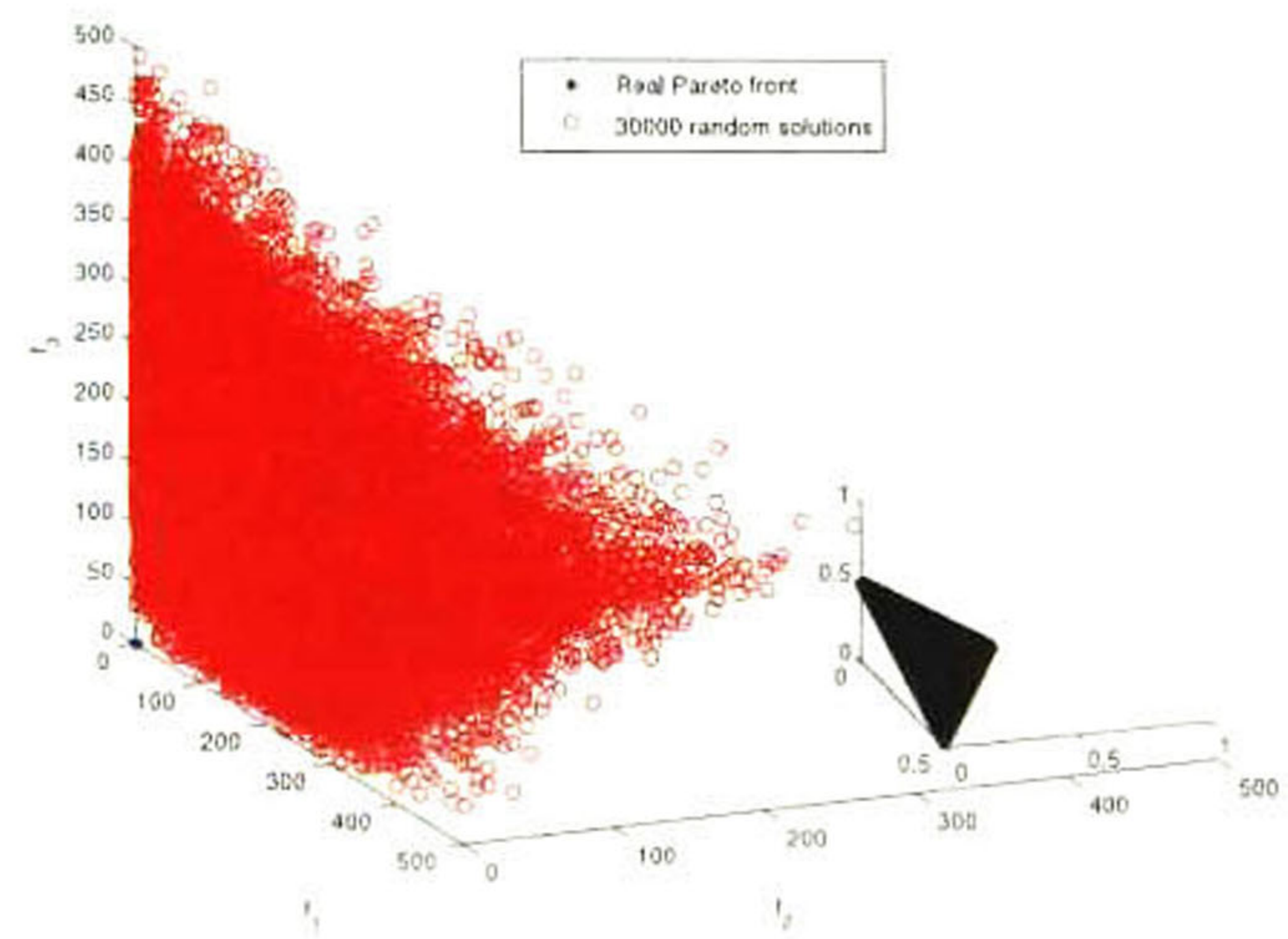


Figure B.7: Real Pareto front of the DTLZ1 test function and 30.000 random solutions.

B.8 DTLZ7

The DTLZ7 test function proposed by *Deb et al* in [Deb et al., 2002] has several disconnected Pareto optimal regions (see Figure B.8). It is uni-modal in the first objective and multi-modal in the second one. Its range of values are in the domain $[0, 1]$. The formal definition is presented in Equation B.8.

$$\begin{aligned}
 f_{k=1:K-1}(\vec{x}) &= x_m \\
 f_K(\vec{x}) &= (1 + g(\vec{x})) \\
 &\left(K - \sum_{i=1}^{K-1} \left(\frac{f_i(\vec{x})}{1 + g(\vec{x})} (1 + \sin(3\pi f_i(\vec{x}))) \right) \right)
 \end{aligned}$$

such as:

$$g(\vec{x}) = 1 + \frac{9}{n - M + 1} \sum_{i=K}^n x_i \quad (\text{B.8})$$

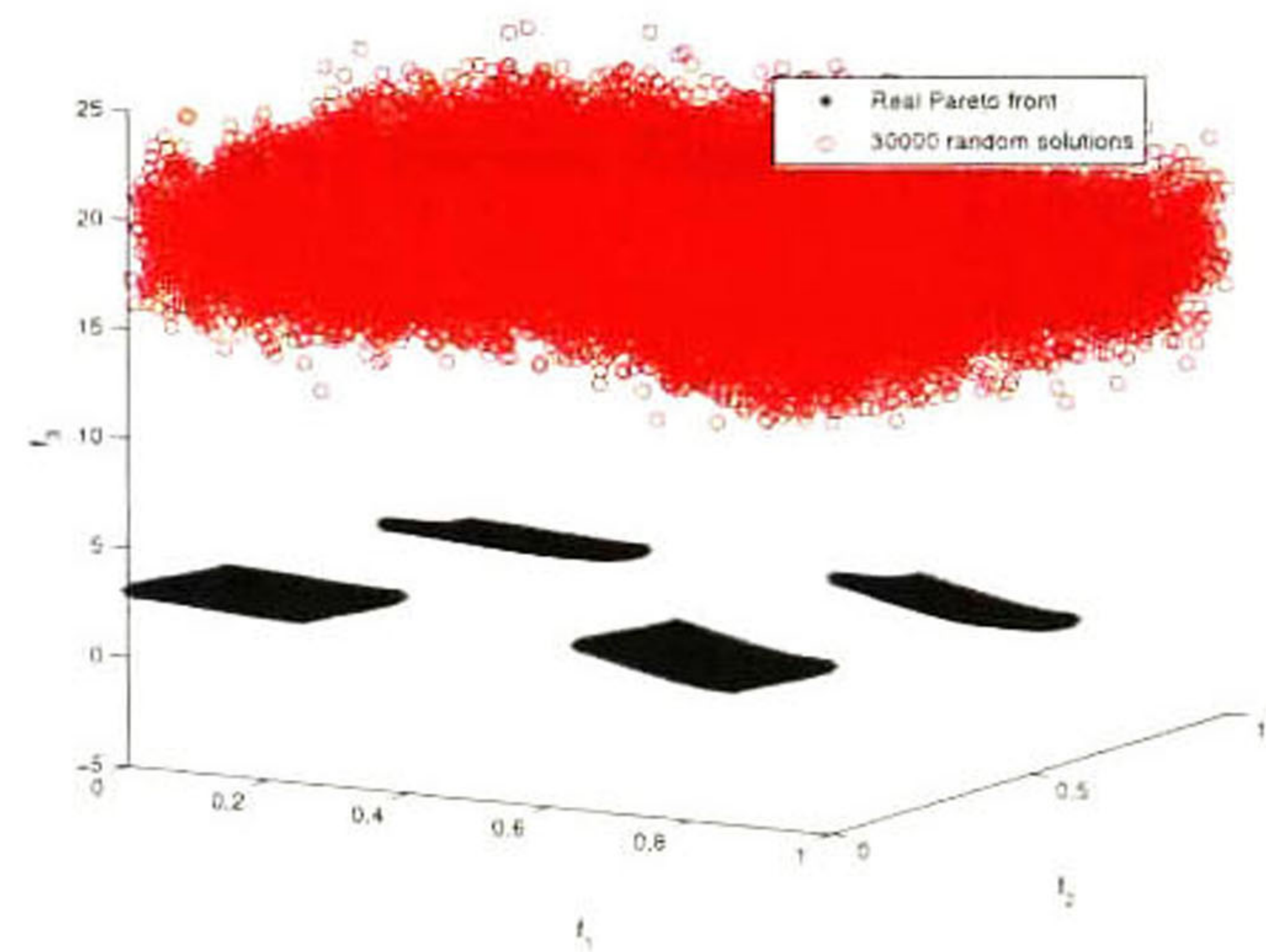


Figure B.8: Real Pareto front of the DTLZ7 test function and 30,000 random solutions.

Bibliography

- [Agterberg, 1984] Agterberg, F. (1984). Trend Surface Analysis. *Spatial statistics and models*, pages 147–171.
- [Broomhead, 1988] Broomhead, D. S., D. L. (1988). Multi-variable functional interpolation and adaptive networks. In *Complex Systems*, pages 321–355. Vol. 2.
- [Chafekar et al., 2005] Chafekar, D., Shi, L., Rasheed, K., and Xuan, J. (2005). Multiobjective GA Optimization Using Reduced Models. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 35(2):261–265.
- [Coello Coello, 0 8] Coello Coello, C. A. (2006, ISBN 0-9787135-0-8). 20 Years of Evolutionary Multi-Objective Optimization: What Has Been Done and What Remains to be Done. In Yen, G. Y. and Fogel, D. B., editors, *Computational Intelligence: Principles and Practice*, chapter 4, pages 73–88. IEEE Computational Intelligence Society, Vancouver, Canada.
- [Coello Coello and Toscano Pulido, 2001] Coello Coello, C. A. and Toscano Pulido, G. (2001). A Micro-Genetic Algorithm for Multiobjective Optimization. In Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D., editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [Cohon and Marks, 1975] Cohon, J. and Marks, D. (1975). A Review and Evaluation of Multiobjective Programming Techniques. *Water Resources Research*, 11(2):208–220.
- [Corne et al., 2001] Corne, D. W., Jerram, N. R., Knowles, J. D., and Oates, M. J. (2001). PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. In Spector, L.,

- Goodman, E. D., Wu, A., Langdon, W., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. H., and Burke, E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 283–290, San Francisco, California. Morgan Kaufmann Publishers.
- [Corne et al., 2000] Corne, D. W., Knowles, J. D., and Oates, M. J. (2000). The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 839–848, Paris, France. Springer. Lecture Notes in Computer Science No. 1917.
- [Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. In *Machine Learning*, pages 273–297.
- [Deb, 2001] Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK. ISBN 0-471-87339-X.
- [Deb et al., 2000] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, Paris, France. Springer. Lecture Notes in Computer Science No. 1917.
- [Deb and Nain, 2007] Deb, K. and Nain, P. (2007). An Evolutionary Multi-objective Adaptive Meta-modeling Procedure Using Artificial Neural Networks. In Yang, S., Ong, Y.-S., and Jin, Y., editors, *Evolutionary Computation in Dynamic and Uncertain Environments*, volume 51, chapter 13, pages 297–322. Springer Berlin Heidelberg, Berlin, Heidelberg.

- [Deb et al., 2002] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002). Scalable Multi-objective Optimization Test Problems. In *Congress on Evolutionary Computation(CEC' 2002)*, pages 825–830.
- [Demsar, 2006] Demsar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7(Jan):1–30.
- [Diaz-Manriquez et al., 2011] Diaz-Manriquez, A., Toscano-Pulido, G., and Gomez-Flores, W. (2011). On the Selection of Surrogate Models in Evolutionary Optimization Algorithms. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 2155 –2162.
- [Diaz-Manriquez et al., 2012] Diaz-Manriquez, A., Toscano-Pulido, G., and Landa-Becerra, R. (2012). A Surrogate-based Intelligent Variation Operator for Multiobjective Optimization. In *In 10th International Conference on Artificial Evolution (EA'2011), LNCS volume to appear, Springer-Verlag,, Angers, France.*
- [Emmerich et al., 2006] Emmerich, M. T., Giannakoglou, K. C., and Naujoks, B. (2006). Single- and Multiobjective Evolutionary Optimization Assisted by Gaussian Random Field Metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439.
- [Erickson et al., 2001] Erickson, M., Mayer, A., and Horn, J. (2001). The Niche Pareto Genetic Algorithm 2 Applied to the Design of Groundwater Remediation Systems. In Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D., editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 681–695. Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [Fang et al., 2005] Fang, H., Rais-Rohani, M., Liu, Z., and Horstemeyer, M. (2005). A comparative study of metamodeling methods for multiobjective crashworthiness optimization. *Computers & Structures*, 83(25-26):2121 – 2136.

- [Fogel, 1964] Fogel, L. J. (1964). *On the Organization of Intellect*. PhD thesis, University of California, California, Los Angeles.
- [Fonseca and Fleming, 1993] Fonseca, C. M. and Fleming, P. J. (1993). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California. University of Illinois at Urbana-Champaign, Morgan Kaufman Publishers.
- [Georgopoulou and Giannakoglou, 2009] Georgopoulou, C. A. and Giannakoglou, K. C. (2009). A Multi-objective Metamodel-assisted Memetic Algorithm with Strength-based Local Refinement. *Engineering Optimization*, 41(10):909–923.
- [Gunn, 1998] Gunn, S. R. (1998). Support Vector Machines for Classification and Regression. Technical report, Faculty of Engineering, Science and Mathematics School of Electronics and Computer Science, Zurich, Switzerland.
- [Hoffman, 1989] Hoffman, A. (1989). *Arguments on Evolution: A Paleontologist's Perspective*. Oxford University Press. ISBN 019-50-4443-6.
- [Holland, 1975] Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Michigan.
- [Horn et al., 1994] Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A Niche Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey. IEEE Service Center.
- [Jin et al., 2000] Jin, R., Chen, W., and Simpson, T. W. (2000). Comparative studies of metamodeling techniques under multiple modeling criteria. *Structural and Multidisciplinary Optimization*, 23:1–13.

- [Jones, 2001] Jones, D. (2001). A Taxonomy of Global Optimization Methods Based on Response Surfaces. *J. of Global Optimization*, 21(4):345–383.
- [Jones et al., 1998] Jones, D., Schonlau, M., and Welch, W. (1998). Efficient Global Optimization of Expensive Black-box Functions. *Journal of Global optimization*, 13:455–492.
- [Knowles, 2006] Knowles, J. (2006). ParEGO: A Hybrid Algorithm With On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66.
- [Knowles and Corne, 1999] Knowles, J. D. and Corne, D. W. (1999). The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Washington, D.C. IEEE Service Center.
- [Kursawe, 1991] Kursawe, F. (1991). A Variant of Evolution Strategies for Vector Optimization. In *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, volume 496, pages 193–197. Springer-Verlag. Lecture Notes in Computer Science.
- [Li et al., 2008] Li, M., Li, G., and Azarm, S. (2008). A Kriging Metamodel Assisted Multi-Objective Genetic Algorithm for Design Optimization. *Journal of Mechanical Design*, 130(3):031401–1 – 031401–10.
- [Li et al., 2007] Li, X., Branke, J., and Kirley, M. (2007). On Performance Metrics and Particle Swarm Methods for Dynamic Multiobjective Optimization Problems. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 576 –583.
- [MacQueen, 1967] MacQueen, J. B. (1967). Some Methods for Classification and Analysis of MultiVariate Observations. In Cam, L. M. L. and Neyman, J., editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

- [McKay et al., 1979] McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245.
- [Moscato, 2003] Moscato, P. (2003). A Gentle Introduction to Memetic Algorithms. In *Handbook of Metaheuristics*, pages 105–144. Kluwer Academic Publishers.
- [Nowostawski and Poli, 1999] Nowostawski, M. and Poli, R. (1999). Parallel Genetic Algorithm Taxonomy. *1999 Third International Conference on KnowledgeBased Intelligent Information Engineering Systems Proceedings Cat No99TH8410*, pages 88–92.
- [Osyczka, 1984] Osyczka, A. (1984). *Multicriterion Optimization in Engineering with FORTRAN Programs*. Ellis Horwood Limited. ISBN 9780470200193.
- [Poggio and Girosi, 1989] Poggio, T. and Girosi, F. (1989). A Theory of Networks for Approximation and Learning. Technical report, Massachusetts Institute of Technology, Artificial Intelligence Laboratory and Center for Biological Information Processing, Whitaker College., Zurich, Switzerland.
- [Queipo et al., 2005] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Kevin (2005). Surrogate-based Analysis and Optimization. *Progress in Aerospace Sciences*, 41(1):1–28.
- [Rasheed, 1998] Rasheed, K. M. (1998). Gado: A genetic algorithm for continuous design optimization. Technical Report Tech. Rep. DCS-TR-352, Dept. Comput. Sci., Rutgers Univ., New Brunswick, NJ.
- [Ray and Smith, 2006] Ray, T. and Smith, W. (2006). A Surrogate Assisted Parallel Multiobjective Evolutionary Algorithm for Robust Engineering Design. *Engineering Optimization*, 38(8):997–1011.

- [Rechenberg, 1965] Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. In *Royal Aircraft Establishment Translation No. 1122*, B. F. Toms, Trans. Ministry of Aviation, Royal Aircraft Establishment, Farnborough Hants.
- [Rosenberg, 1967] Rosenberg, R. S. (1967). *Simulation of genetic populations with biochemical properties*. PhD thesis, University of Michigan, Ann Harbor, Michigan.
- [Sacks et al., 1989] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, 4:409–423.
- [Santana-Quintero et al., 2008] Santana-Quintero, L. V., Coello Coello, C. A., and G., A. (2008). Hybridizing surrogate techniques, rough sets and evolutionary algorithms to efficiently solve multi-objective optimization problems. In *Genetic and Evolutionary Computation Conference (GECCO'2008)*, pages 763–764.
- [Santana-Quintero et al., 2010] Santana-Quintero, L. V., Montaña, A., and Coello, C. A. (2010). A Review of Techniques for Handling Expensive Functions in Evolutionary Multi-Objective Optimization. *Computational Intelligence in Expensive Optimization Problems*, 2:29–59.
- [Schaffer, 1985] Schaffer, J. D. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum.
- [Schüze et al., 2011] Schüze, O., Esquivel, X., Lara, A., and Coello, C. A. C. (2011). Using the Averaged Hausdorff Distance as a Performance Measure in Evolutionary Multi-Objective Optimization. *IEEE Transactions on Evolutionary Computation*. (accepted).
- [Schwefel, 1965] Schwefel, H.-P. (1965). *Kybernetische Evolution als Strategie der Experimentellen Forschung in der Stromungstechnik*. PhD thesis, Technical University of Berlin.

- [Srinivas and Deb, 1994] Srinivas, N. and Deb, K. (1994). Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248.
- [Syberfeldt et al., 2008] Syberfeldt, A., Grimm, H., Ng, A., and John, R. I. (2008). A Parallel Surrogate-Assisted Multi-Objective Evolutionary Algorithm for Computationally Expensive Optimization Problems. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pages 3176–3183, Hong Kong. IEEE Service Center.
- [Toscano Pulido and Coello Coello, 2003] Toscano Pulido, G. and Coello Coello, C. A. (2003). The Micro Genetic Algorithm 2: Towards Online Adaptation in Evolutionary Multiobjective Optimization. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 252–266, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.
- [Veldhuizen, 1999] Veldhuizen, D. A. V. (1999). *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio.
- [Voutchkov and Keane, 2006] Voutchkov, I. and Keane, A. (2006). Multiobjective Optimization using Surrogates. In Parmee, I., editor, *Adaptive Computing in Design and Manufacture 2006. Proceedings of the Seventh International Conference*, pages 167–175, Bristol, UK. The Institute for People-centred Computation.
- [Zapotecas Martínez and Coello Coello, 2010] Zapotecas Martínez, S. and Coello Coello, C. A. (2010). A Multi-objective Meta-model Assisted Memetic Algorithm with Non Gradient-based Local Search. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation, GECCO '10*, pages 537–538, New York, NY, USA. ACM.

- [Zitzler et al., 2000] Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical results. In *Evolutionary Computation*, volume 8, pages 173–195.
- [Zitzler et al., 2002] Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In Giannakoglou, K., Tsahalis, D., Periaux, J., Papailou, P., and Fogarty, T., editors, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece.
- [Zitzler and Thiele, 1998] Zitzler, E. and Thiele, L. (1998). An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.



CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL IPN

UNIDAD TAMAULIPAS

Cd. Victoria, Tamaulipas, a 9 de Diciembre de 2011

Los abajo firmantes, integrantes del jurado para el examen de grado que sustentará el C. GERARDO MONTEMAYOR GARCIA, declaramos que hemos revisado la tesis titulada:

“Uso de los Modelos Surrogados en Algoritmos Evolutivos Multiobjetivo”

Y consideramos que cumple con los requisitos para obtener el grado de Maestro en Ciencias en Computación.

Atentamente,


Dr. Gregorio Toscano Pulido



Dr. Eduardo Arturo Rodríguez Tello



Dr. José Gabriel Ramírez Torres





CINVESTAV - IPN
Biblioteca Central



SSIT0010851