A local approximation based multi-objective optimization algorithm with applications

Massimiliano Gobbi · Paolo Guarneri · Leonardo Scala · Lorenzo Scotti

Received: 11 December 2010 / Accepted: 24 December 2012 / Published online: 22 January 2013

1 Introduction

A generic multi-objective optimization problem can be formulated as follows (Miettinen 1999):

M. Gobbi (✉) · P. Guarneri · L. Scala · L. Scotti
Dept. of Mechanical Engineering, Politecnico di Milano, Milan, Italy
e-mail: massimiliano.gobbi@polimi.it

$$\begin{aligned} &\min_x \left[f_1(x) \quad f_i(x) \quad f_N(x) \right] \\ &x \in \Omega \subset \Re^n \\ &g_j(x) \leq 0 \\ &h_k(x) = 0 \end{aligned} \tag{1}$$

where f_i are the N objectives, x is the n-dimensional vector of the design variables whose domain is Ω, g_j and h_k are the inequality and equality constraints functions, respectively. The solution for this optimization problem is not unique if the objective functions are conflicting. Miettinen (1999) proposed a definition of optimality for this type of problems. A solution x^* is Pareto optimal if the following condition holds

$$\nexists x_i : \begin{cases} f_j(x_i) \leq f_j(x^*) \quad j = 1 : N \\ \exists k : f_k(x_i) < f_k(x^*) \end{cases} \tag{2}$$

This definition leads to a set of solution called Pareto optimal set which represents the set of best compromise solutions. Therefore, in order to solve the multi-objective optimization problem the Pareto set (or an approximation of the set, see Zitzler et al. (2008) for quality assessment of Pareto set approximations) must be computed.

Finding an accurate approximation of the Pareto optimal set is difficult due to the problem high dimensionality and due to the non linearity of the functions f, g and h. Often f, g and h are not analytical functions but they are the output of time consuming numerical simulations (FEM, CFD, multi-body...) and the computing effort plays a key role. Therefore, accuracy and efficiency are key features for every algorithm.

In the literature many algorithms and techniques capable of solving multi-objective optimization problems can be found. There are basically two approaches, the first one based on single objective optimization algorithms (scalarization) and the second that approximates the Pareto set at once (vector optimization).

The scalarization converts the multi-objective problem into an equivalent single-objective problem. Some examples are the weighted sum method (Mastinu et al. 2006), the ε-constraint method (Mastinu et al. 2006), the normal boundary intersection (Das and Dennis 1998) and the normal constraint method (Messac et al. 2003). These methods may fail in finding all the Pareto solutions, i.e. the weighed sum method cannot find Pareto solution when the feasible solution set in the objective domain is not convex (Das and Dennis 1997), the normal constraints and normal boundary methods in some case can leave some regions unexplored (in Messac and Mattson (2004) the algorithm is modified to overcome this issue), the ε-constraint method needs a priori information about the maximum and minimum values of the objective functions. Moreover, they compute only one Pareto solution at a time by solving the equivalent problem.

In Zhang et al. (2010) a multi-objective problem is decomposed into a set of single-objective sub-problems and a kriging (Van Beers and Kleijnen 2003) predictive model is built for each sub-problem on the basis of the solutions evaluated during the optimization procedure to reduce the computational effort. The weighted sum method is finally applied to solve the multi-objective problem.

The whole Pareto set can be obtained by changing some parameters in the algorithm (i.e. the weights for the weighted sum method) and iteratively solving an equivalent single objective problem for each Pareto-optimal solution. The opportunity to apply optimality conditions (KKT condition (Kuhn and Tucker 1951)) to solve the equivalent problem with the desired tolerance and to use efficient gradient based minimization algorithms (i.e. SQP) (Papalambros and Wilde 2000) is a relevant advantage of scalarization algorithms.

Vector optimization approaches compute the whole Pareto set by solving directly the original problem. Among these methods there are the parameter space investigation method (PSI) and evolutionary algorithms. PSI (Mastinu et al. 2006) method explores the design variables domain by means of uniformly distributed sequences. By this approach, many feasible design points are computed and the Pareto set ca be found by sorting these solutions by applying the definition (2). It is obvious that the quality of the Pareto approximation depends strongly on the uniformity of the points in the design variables domain (Statnikov and Matusov 1995; Niederreiter 1987; Tezuka 1995). The method is robust, the gradient of the functions is not required and only one parameter has to be set, the number of points. Unfortunately, the choice of the number of points influences the accuracy of the results and it is not straightforward, it depends on the sequence uniformity properties and on the Lipschitz constant (Statnikov and Matusov 1995) of the functions which is typically unknown a priori.

Evolutionary algorithms are able to approximate the Pareto set by starting from an initial set of design solutions (the so called the initial population). Among these algorithms, genetic algorithms (see Goldberg 1989 and Michalewicz 1994), simulated annealing (Kirkpatrick et al. 1983) and particle swarming (Eberhart and Kennedy 1995) are widely known. These algorithms work on an entire population by means of stochastic operators (i.e. crossover and mutation for the genetic algorithms) that lead the population towards the Pareto front. As the PSI method, evolutionary algorithms make no assumptions on the functions and on the Pareto set (convexity, continuity…). One of the advantages of genetic algorithms lies in the possibility to perform optimization with discrete design variables because the algorithm does not work directly on the design variables but on a coding of the variables themselves (i.e. binary coding, …). Moreover, the probability of being trapped in local minima is reduced by working on the entire population. The drawback lies in the large number of algorithm parameters to be set to improve convergence and accuracy.

As already stated, the computational efficiency plays a key role, because engineering optimization problems involve complex models to compute the objective and constraint functions. In this case, it is useful to rely on global approximation techniques which consist in the use of "black box" models able to reproduce the input-output relationship of the original model (Mastinu et al. 2006; Papalambros and Wilde 2000; Goel et al. 2007) and (Gobbi and Mastinu 1999). Typical approximation models are response surfaces (Myers and Montgomery 1995), artificial neural networks (Haykin 1998), kriging (Van Beers and Kleijnen 2003). These models are accurate on the whole domain (*global* approximation) while the designer wants to focus only on the Pareto region. To reduce the computational effort needed to identify these approxi-

mated models it is possible to rely on *local* approximated models which are accurate only in the neighborhood of the Pareto front (i.e., close to the area where the Pareto front is supposed to be) while they are intentionally left inaccurate far from it (Mastinu et al. 2006). There are many algorithms and approaches which have advantages and disadvantages and are suitable for some type of problems while are inefficient or do not work in other cases.

Local approximation methods are proposed to perform an approximation of the Pareto surface in the vicinity of a selected optimal solution (mainly for sensitivity analysis (Zhang 2003)). Linear (Tappeta et al. 2000) and quadratic approximations (Maginot et al. 2008) are generally considered. A method for identifying non-differentiable Pareto points has been included in Maginot et al. (2008).

In this paper an optimization algorithm based on Messac and Mattson (2004) is presented. The aim is to supply a useful tool among (and *not to replace*) the existing ones to solve efficiently multi-objective optimization problems. The algorithm is designed by taking into account efficiency and accuracy features. It is based on the normal constraint method applied to a neural network model which realizes the local approximation. At each iteration of the normal constraint algorithm, the neural network model is updated in order to move along the Pareto set. This way, the method should exploit the power of local approximation in terms of efficiency with the good uniformity properties of the normal constraint method.

In the Sect. 1 the new algorithm (named Approximate Normal Constraint, ANC) is presented and in Sect. 2 its performance is tested by means of some well known test problems. In the last paragraph of Sect. 2 the multi-objective optimization of a ground vehicle suspension system is performed by means of the ANC. Some general observations on the efficiency of the most used algorithms are reported in Sect. 3. Conclusions are provided to summarize the paper.

2 Approximate normal constraint method (ANC)

Before introducing the ANC, the normal constraint method (NC) is briefly described.

2.1 Normal constraint (NC) method

The NC is based on the minimization of a single objective function on a reduced feasible domain (Messac et al. 2003) and (Messac and Mattson 2004). The reduction of the feasible domain is performed by imposing an additional inequality constraint to the original problem (1).

Let x_i^* be the design point where the minimum of the i-th objective function is reached and F_i be the so called anchor point as defined below

$$\begin{aligned} f_i^* &= f_i\left(x_i^*\right) = \min_x f_i(x) \\ F_i &= \left[f_1(x_i^*) \quad f_i^* \quad f_N l(x_i^*) \right] \end{aligned} \tag{3}$$

The utopia plane is the hyper plane which contains all the anchor points (despite of its name, it does not contain the utopia point).

Let P a point on the utopia plane, and v_{ij} the vector joining the i-th and the j-th anchor points

$$v_{ij} = F_i - F_j \tag{4}$$

The space reduction is performed as follows

$$\begin{aligned} & x \in \Omega \\ & g(x) \leq 0 \\ & h(x) = 0 \\ & v_{ij}\big(F(x) - P\big) \leq 0 \end{aligned} \tag{5}$$

where F is a point in the objective functions domain

$$F(x) = \begin{bmatrix} f_1(x) & f_N(x) \end{bmatrix} \tag{6}$$

A single objective problem is solved on the reduced domain to find a Pareto solution

$$\begin{aligned} & \min_x f_i(x) \\ & x \in \Omega \\ & g(x) \leq 0 \\ & h(x) = 0 \\ & v_{ij}\big(F(x) - P\big) \leq 0 \quad \forall j \neq i \\ & v_{ij} = F_i - F_j \quad \forall j \neq i \end{aligned} \tag{7}$$

In Fig. 1 the domain reduction is shown for a two dimensional space ($N = 2$). By minimizing f_2 on the reduced domain, the Pareto point is computed. By changing the point P on the utopia plane the whole Pareto set can be approximated. It can be noticed that at the Pareto solution the normal constraint is active.

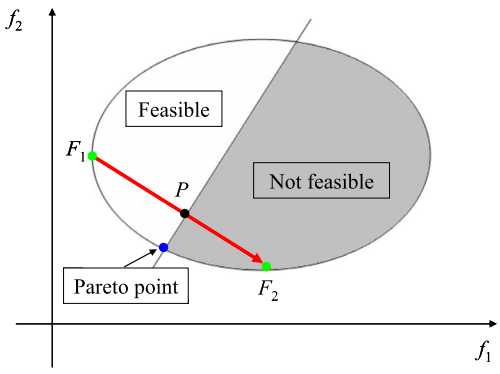


Fig. 1 Domain reduction by NC method. $F_{1,2}$ are the anchor points, P is one of the points of the utopia plan

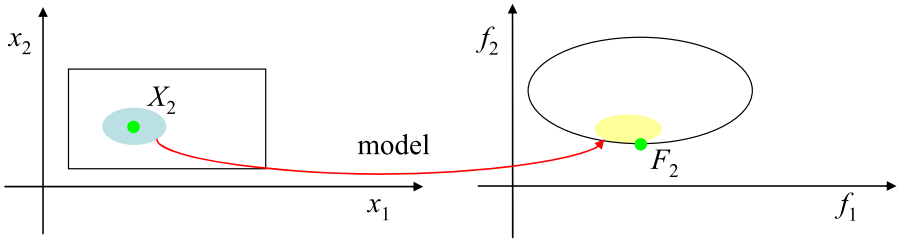


Fig. 2 ANC, step 1 and 2. The minimum (*green dot*) is computed and design solutions are sampled in its neighborhood (the *shaded area* around the minimum)

2.2 Approximate normal constraint algorithm

The ANC algorithm is based on the NC algorithm described in the previous section. The main idea behind the ANC is the use of a *local approximated* utopia plane instead of the utopia plane defined in the previous paragraph. The following steps explain how to construct this plane and how the algorithm works. Figures refer to a bi-objective problem which can be conveniently plotted.

1. Minimize *one* objective function (i.e. find point F_i, see the green point in Fig. 2). The NC requires N minimizations (N anchor points) to construct the utopia plane while the Pareto exploration of ANC starts from one single anchor point.
2. Sample the design variables domain locally around the anchor point and evaluate the constraint and objective functions (see the shaded areas around the green points in Fig. 2). The relationship between the design variables and the response (both objectives and constraints) functions is approximated in terms of metamodels. Such models use parameterized equations to relate the input to output and the parameters are tuned in order to minimize the error between the actual model responses and the metamodel response. Since the input-output pairs are sampled around the anchor point in proximity of the Pareto front, the metamodel will be a local approximation of the model equations. The metamodels will have to predict the response for values of the design variables as required by the minimization algorithm, which will be different form the sampled points. Feed-forward artificial neural networks are used due to their good generalization properties. The universal approximation theorem states that a feed-forward neural network with a single hidden layer can approximate any scalar function with any desired accuracy (Haykin 1998). The solutions in the neighbourhood of the anchor point are sampled in the design domain (blue area around X_2 in Fig. 2) and their images in the objective space are obtained (yellow area in proximity of the anchor point F_2).
3. Compute the remaining $N-1$ anchor points by using the ANN, (i.e. find $\tilde{F}_j$, see Fig. 3). A low computational effort is required. The utopia plane obtained is *local* and *approximated* due to the use of the ANN. These anchor points must be obtained in a limited region in which the neural network is accurate. For this purpose additional constraints on the variables are introduced.
4. Define and solve the equivalent approximated problem via NC algorithm with different P_k belonging to the local approximate utopia plane in order to generate

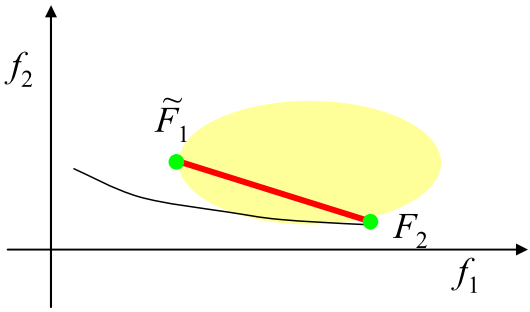


Fig. 3 ANC, step 3. The ANN is used to locate the anchor point $\tilde{F}_j$. A local utopia plane is generated through the ANN approximation

a portion of the Pareto front (see Fig. 4).

$$\begin{aligned} &\min_x \tilde{f}_i(x) \\ &x \in \Omega \\ &\tilde{g}(x) \leq 0 \\ &\tilde{h}(x) = 0 \\ &v_{ij}\left(\tilde{F}(x) - P_k\right) \leq 0 \quad \forall j \neq i \\ &v_{ij} = F_i - \tilde{F}_j \quad \forall j \neq i \end{aligned} \tag{8}$$

5. Update the ANN. The objective and constraint functions are evaluated at the (nearly) Pareto points computed through the approximated neural network model at the previous step. This procedure strongly improves the algorithm convergence. These new points are added to the previous points to train a new ANN and the region in which the approximated model is accurate moves along the Pareto set (see Fig. 5). As the algorithm moves forward, new Pareto points are computed and added to the training set.
6. Define a new equivalent approximate problem by using the new approximated functions. The new utopia local plane is defined in order to move towards the unexplored regions.
7. The algorithm continues to update the ANN and solve the locally approximated problem until a termination condition is met.

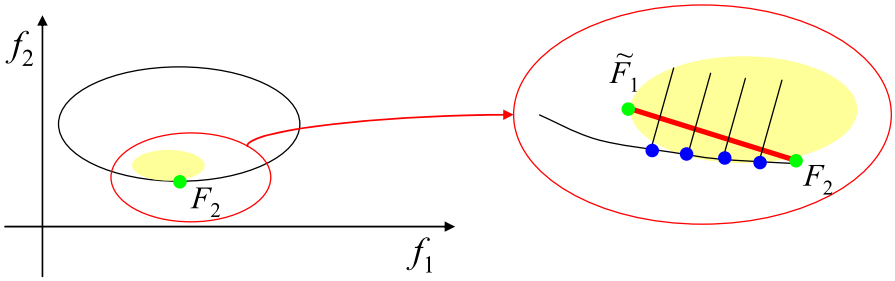


Fig. 4 ANC, step 4. The NC algorithm is applied for different points belonging to the utopia plane (see Fig. 3). The blue dots represent the Pareto optimal solutions

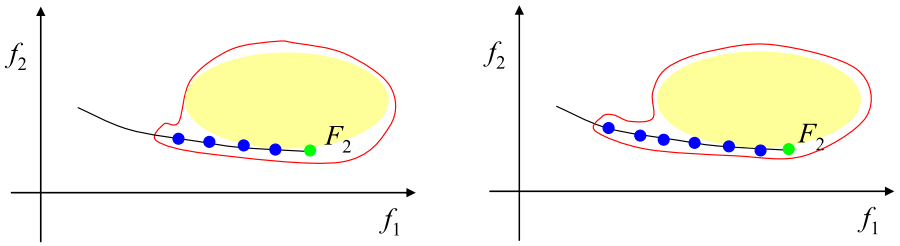
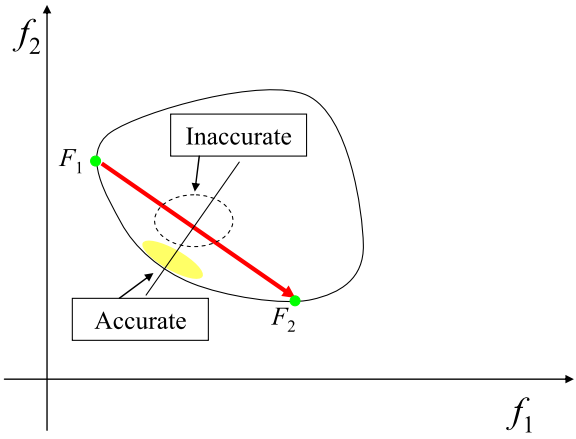


Fig. 5 ANC, phase 5. Generated Pareto optimal solutions are used to train the ANN in order to locally improve its accuracy

Fig. 6 Local approximation and the utopia plane



The main difference between Messac's ANC (Messac and Mattson 2004) lies in the utopia plane used to perform the NC algorithm. The *local* utopia plane is a *local* approximation, the actual utopia plane can be quite far from the Pareto set while the local approximation can be more accurate (see Fig. 6).

3 ANC test

The ANC algorithm is used to solve the test problems ZTD1 and ZTD2 (Deb et al. 2005). A comparison between the genetic algorithm, constraints method and the ANC is presented. The genetic algorithm belongs to the family of evolutionary algorithms which are able to find the Pareto set at once and the ε-constraint is a multi-objective method based on the scalarization. As it will be shown in the next paragraphs, the ANC is able to approximate the Pareto with high efficiency, only few function evaluations are needed to compute each Pareto solution.

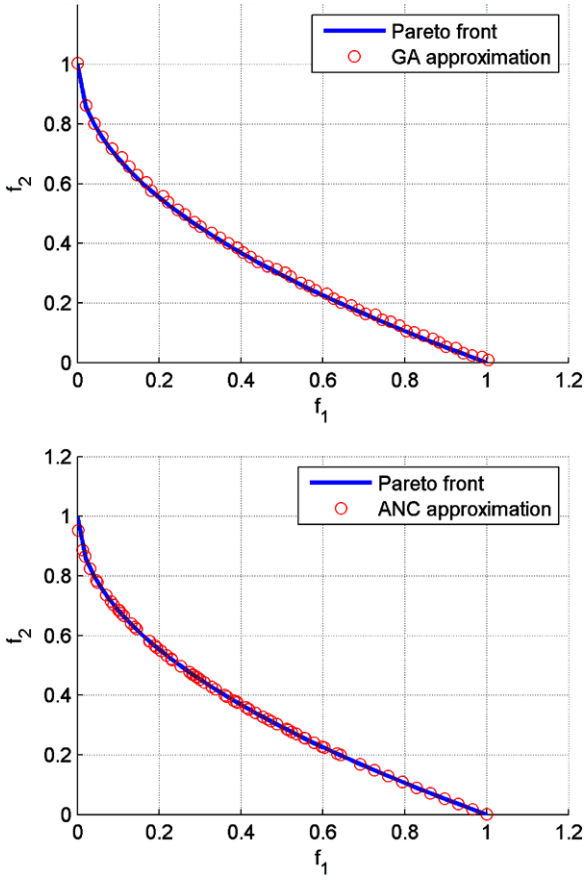


Fig. 7 GA and ANC Pareto set approximations for the ZTD1

3.1 Problem ZTD1

In this paper the ZTD1 problem with $n = 5$ design variables is considered. In Fig. 7 the objective functions domain and the Pareto set are shown.

$$\min_x \begin{bmatrix} x_1 \\ g(x)(1 - \sqrt{\frac{x_1}{g(x)}}) \end{bmatrix}$$
$$g(x) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i^2 \tag{9}$$

The results are reported in Table 1. The efficiency is the ratio between the number of function evaluations and the number of the Pareto solutions identified. The values in Table 1 are averaged, for the GA, over 10 different runs.

Some of the steps of the ANC algorithm are reported to demonstrate how the exploration of the Pareto set occurs. The first anchor point F_2 obtained by minimizing $f_2(\boldsymbol{x})$ corresponds to the point $(1, 0)$ in the objective domain. A hundred points are

Table 1 Algorithms performance—ZTD1 test problem

	Genetic algorithm			ε-constraint method	ANC
Population size	50	100	150	–	–
Computed Pareto set size	50	100	150	51	91
Function evaluations	1200	3200	5100	990	344
Efficiency	24	32	34	19.4	3.8

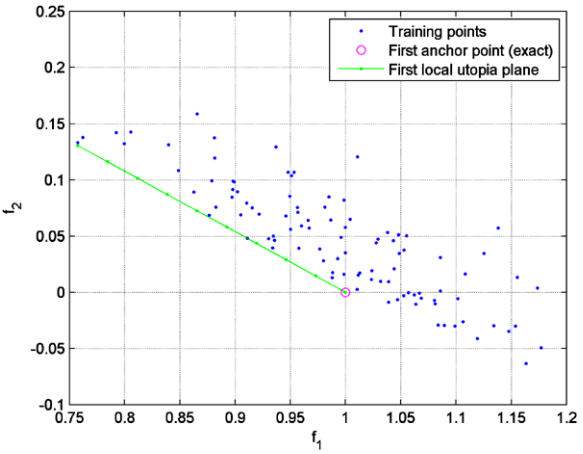


Fig. 8 ZTD1 problem. Algorithm initialization. The points are sampled around the anchor point and the local utopia plane is constructed

randomly sampled in the neighborhood of the first anchor point in the design variables domain. The values of the objective functions are computed at these points and this data is used as a training set for the neural network. The anchor point and the image of the sampled points are shown in Fig. 8.

The second anchor point required to generate the local utopia plane is generated minimizing $f_1(\boldsymbol{x})$. For this step the neural network local approximation is used (additional constraints on the variables are introduced to bound the solution to be in the region where the neural network is accurate). Once the local utopia is located, ten points are evenly placed on it (see Fig. 8).

This implies that ten single objective problems are solved generating as many (nearly) Pareto solutions (see Fig. 9). The actual values of the objectives are evaluated at these new points which are added to the neural network training set. In this way, the neural network approximation is updated by using the points in the neighborhood of the Pareto set (the approximation is local). As long as points are added to the training set, some points are removed from it to keep the size of the training set constant throughout the process. The older points, the ones that are the farthest from the exploration region are removed (points crossed in Fig. 9). The local utopia plane used to compute new Pareto points at the next iteration is shown in Fig. 9 as well.

The iterations shown in Fig. 10 demonstrate how the Pareto set is approximated iteration by iteration. The points approximating the Pareto set are plotted in Fig. 11. In the case of the simulation whose steps are shown in this section, 90 Pareto points are obtained requiring only 288 function evaluations. This result in an efficiency ratio

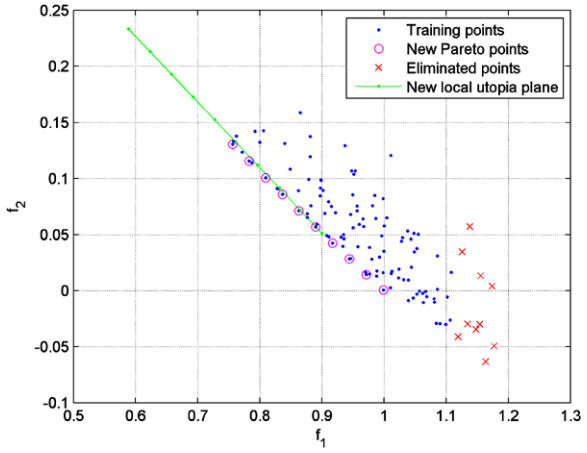


Fig. 9 ZTD1 problem. First iteration. Pareto points are computed and added to the training set. A new utopia plane is constructed to move to unexplored regions

of 3.2 function evaluations per Pareto point (in agreement with the averaged value in Table 1).

3.2 Problem ZTD2

In this paper the ZTD2 problem with $n = 5$ design variables is considered. In Fig. 8 the objective functions domain and the Pareto set are shown.

$$\min_x \begin{bmatrix} x_1 \\ g(x)(1 - (\frac{x_1}{g(x)})^2) \end{bmatrix}$$
$$g(x) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i^2 \tag{10}$$

The results are reported in Table 2. The values in Table 1 are averaged, for the GA, over 10 different runs.

As shown in Fig. 12 and partially in Fig. 7, the Pareto points obtained with the ANC algorithm are not evenly distributed on the Pareto front. The loss of uniformity is due to the local utopia plane which will have at each iterations different orientation and extension. As shown in Fig. 13, two different local utopia planes generate two sets of solutions which are not evenly spaced on the Pareto front. The partial loss of uniformity is balanced by the large increase in the computational efficiency. This is

Table 2 Algorithms performance—ZTD2 test problem

	Genetic algorithm			ε-constraint method	ANC
Population size	50	100	150	–	–
Computed Pareto set size	50	100	150	51	97
Function evaluations	2000	6000	10500	1030	430
Efficiency	40	60	70	20.2	4.4

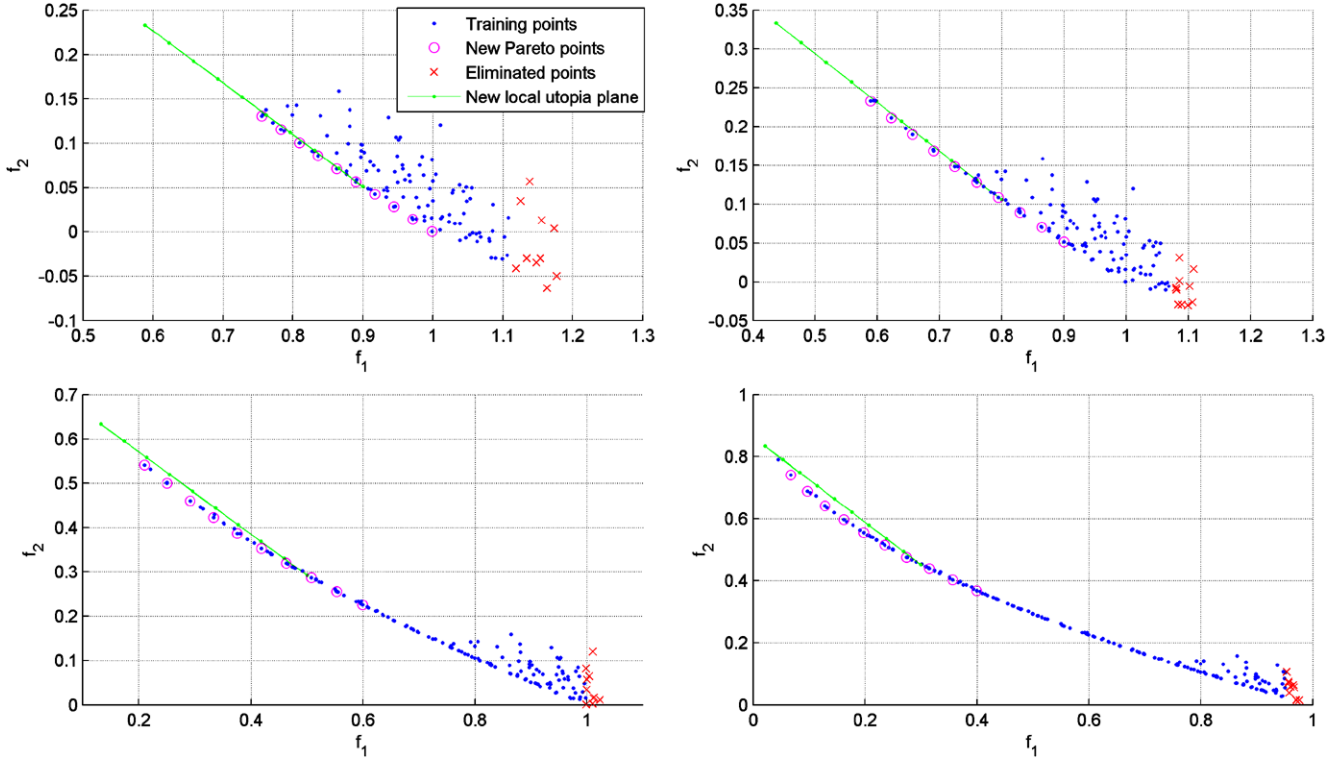


Fig. 10 ZTD1 problem. Successive iterations

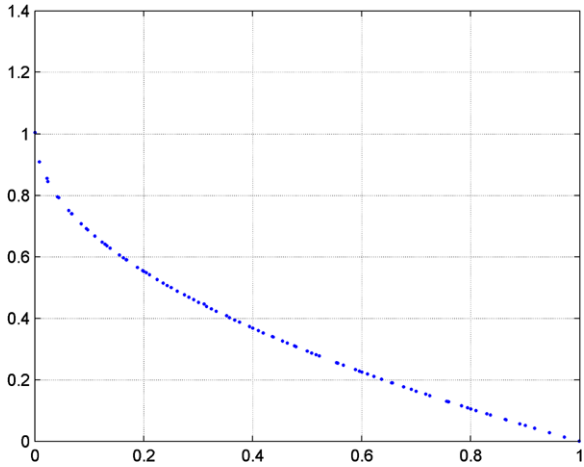


Fig. 11 The points approximating the Pareto front

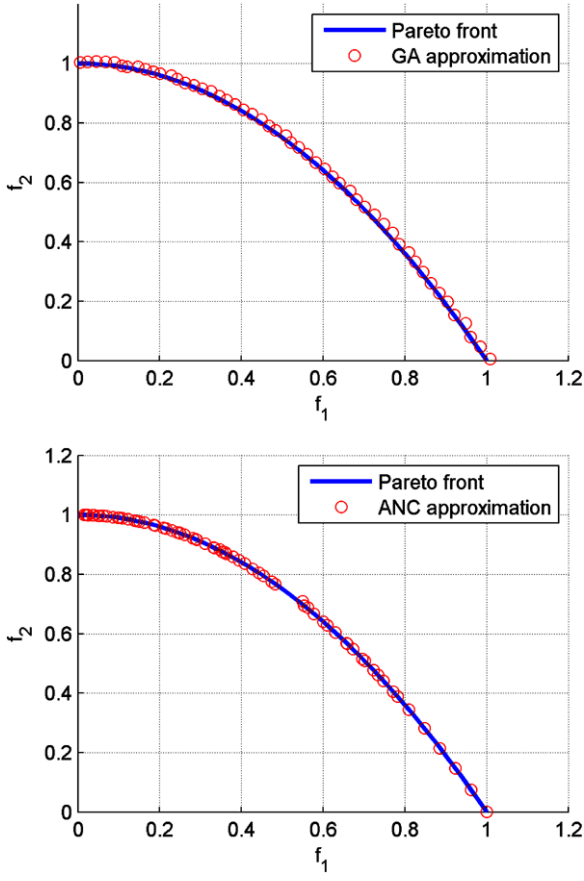


Fig. 12 GA and ANC Pareto set approximations for the ZTD2

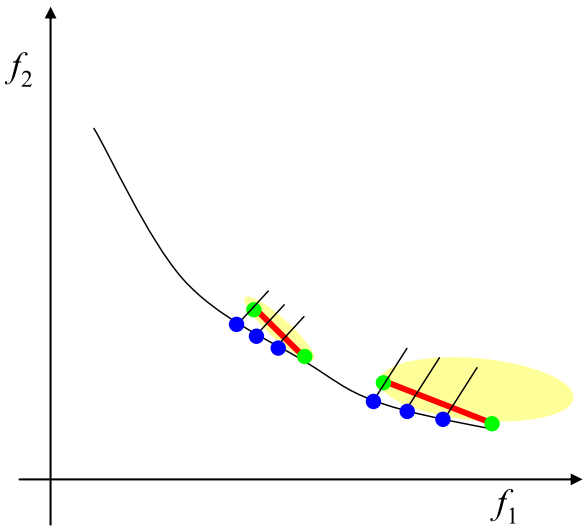


Fig. 13 Loss of uniformity of the Pareto points computed by the ANC

not a major issue since due to the objective function nonlinearity, the solutions in the design variable domain will probably be not evenly distributed (Fadel and Li 2002).

3.3 Application of the ANC: ground vehicle suspension design

In this section the ANC is used to solve a well known problem in the automotive field, the optimal choice of the suspension system stiffness and damping of a road vehicle (Mastinu 1994; Gobbi et al. 2004). The design variables affect the road holding, comfort and working space (a.k.a. rattle space) which are conflicting performance indexes. The suspension system is modeled by the "quarter vehicle" linear model (see Fig. 14) which is considered sufficiently accurate (Mitschke 1990) to capture the relationship between the considered objective functions and the design variables. The masses m_1 and m_2 are the unsprung and sprung masses respectively, k_2 and r_2 are the design variables representing the suspension stiffness and damping respectively and k_1 is the tire stiffness.

The road input ξ_1 is described by a random process with the following power spectral density

$$PSD = \frac{A_b v}{\Omega^2} \tag{11}$$

where A_b is a parameter, v is the vehicle speed and Ω is the spatial frequency (i.e. the inverse of the road irregularity wave length). This simple formula fits the measured data with sufficient accuracy (Hrovat 1993).

The performance indexes are computed as the standard deviation (Gobbi and Mastinu 2001) of the tire-road contact force (road holding *RH*), vehicle body acceleration (discomfort *DC*) and suspension deflection (working space *WS*) while the

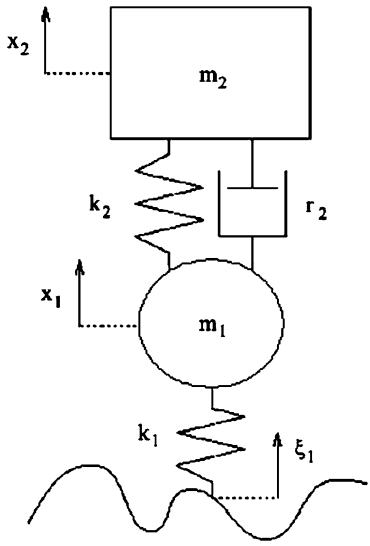


Fig. 14 Quarter vehicle linear model

vehicle is passing over an uneven road

$$DC = A \cdot \sqrt{\frac{(m_1 + m_2)}{m_2^2 r_2} k_2^2 + \frac{k_1 r_2}{m_2^2}}$$

$$RH = A \cdot \sqrt{\frac{(m_1 + m_2)^3}{m_2^2 r_2} k_2^2 - 2\frac{m_1 k_1 (m_1 + m_2)}{m_2 r_2} k_2 + \frac{k_1 r_2 (m_1 + m_2)^2}{m_2^2} + \frac{k_1^2 m_1}{r_2}}$$

$$WS = A\sqrt{\frac{m_1 + m_2}{r_2}}$$

$$\text{where } A = \sqrt{\frac{1}{2} A_b \cdot v} \tag{12}$$

As shown in Gobbi and Mastinu (2001), the Pareto set can be found by the application of the KKT optimality condition. The Pareto set is the (thin) surface (it is not a line) in Fig. 15. The Pareto solutions are a portion of the 2D design domain which will be mapped into a surface in the 3D objective domain. In Fig. 15 the Pareto set computed by the AG and the ANC respectively is shown in the objective function domain.

As in the previous section, the genetic algorithm, the constraints algorithm and the ANC are compared (see Table 3) in terms of number of function evaluations and efficiency.

3.4 Application of the ANC: ground vehicle suspension design—non-linear dampers

he design of the dampers for a passenger vehicle should be performed considering both low and high frequency motions. The former is related to the pitch and roll

Table 3 Algorithms performance—Ground vehicle suspension optimal design. Vehicle data are reported in Gobbi and Mastinu (2001)

	Genetic algorithm			ε-constraint method	ANC
Population size	50	100	150	–	–
Computed Pareto set size	50	100	150	50	85
Function evaluations	650	1300	1950	1151	500
Efficiency	13	13	13	23	5.9

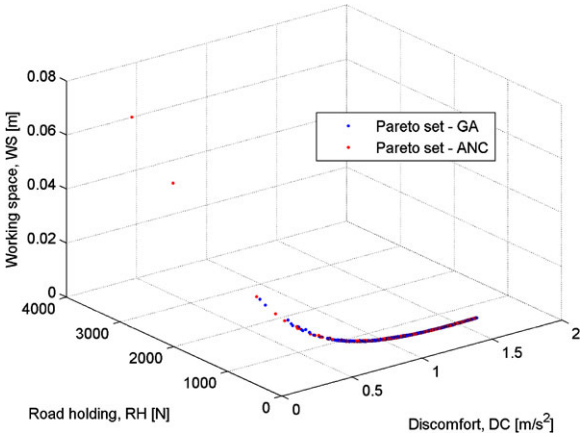


Fig. 15 Pareto surface for the 3D suspension system optimization problem solved by the GA and the ANC. Vehicle data are reported in Gobbi and Mastinu (2001)

motion of the vehicle body (about 1 Hz), the latter to the motion of the unsprung mass derived from the road unevenness (8–10 Hz). The road excitation will result in vibrations which determine the vehicle comfort (up to 100 Hz) and the road holding. For this reason the dampers have a non linear characteristic curve to take into account the different frequency ranges. The curve is also non symmetric providing different forces in compression and extension. The curve can be described by a linear piecewise function presenting different slopes in four different working conditions, low and high deflection velocity both in compression and extension (see Fig. 16).

The characteristic curve as well as the stiffness of the suspension spring and roll bars will determine the vehicle dynamics and must be optimized in two different scenarios. In this example, the running over the uneven road and the roll motion during a turn maneuver are considered.

The quarter model presented in Sect. 3.3 is used to describe the comfort and the road holding, requiring numerical integration since the damper is considered non linear as shown in Fig. 16. The model for the roll motion (see Fig. 17) is briefly described. The vehicle is modeled as a rigid body connected to the suspension systems on the left and right side. Each suspension system is described in terms of unsprung mass, spring stiffness and nonlinear damper characteristic. Since the simulation is performed by considering a smooth road surface, the front and rear suspensions are "collapsed" in a single "equivalent" suspension system at each side of the vehicle. The unsprung mass, stiffness and damping force of the suspension are given by the sum of the respective values at the front and rear axles. The vehicle turn maneuver is

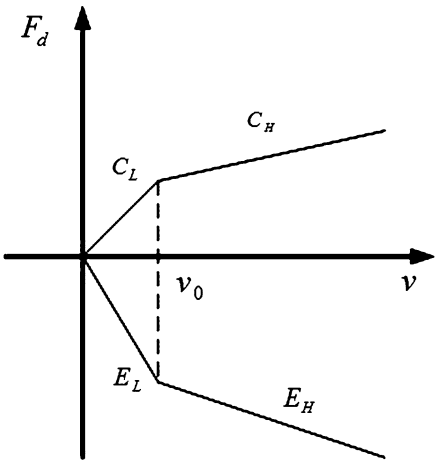


Fig. 16 Damper characteristic curve

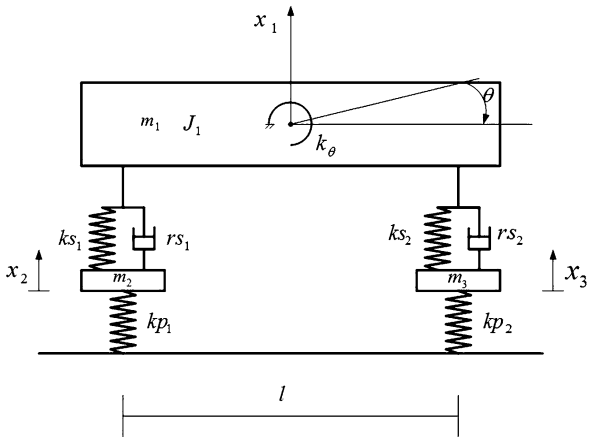


Fig. 17 Model for the vehicle roll motion

simulated by applying as input a step moment to the rigid body which represents the roll moment generated by the lateral centrifugal force.

The problem is to optimize on the following objective functions

- Rear tire road holding: this index is computed as the standard deviation of the contact force at the rear tire while passing over an uneven road. The index has to be minimized.
- Front tire road holding: this index is computed as the standard deviation of the contact force at the front tire while passing over an uneven road. The index has to be minimized.
- Discomfort: this index is computed as the standard deviation of sprung mass while passing over an uneven road. The index has to be minimized.
- Roll angle overshoot: this index represents the difference between the maximum roll angle and the steady state roll angle. The difference is normalized over the steady state value. The index has to be minimized.

Table 4 Algorithm performance—Ground vehicle suspension design

	MOGA	PSI	ANC
Computed Pareto set size	200	381	4951
Function evaluations	5600	8138	16388
Efficiency	28	22	3.3

- Response time: it is the time needed by the system to reach the 90 % of the steady state roll angle value. The index has to be minimized.

The design variables in the problem are listed below

- Stiffness of the front suspension spring.
- Stiffness of the rear suspension spring.
- Slopes of the piece-wise linear damper characteristic curve of the front suspension (4 design variables).
- Slopes of the piece-wise linear damper characteristic curve of the rear suspension (4 design variables).
- Roll stiffness.

The problem has 5 objective functions which have to be optimized with respect to 11 design variables. The ANC method presented in this paper has been applied and compared to two widely used multiobjective algorithms, the Multi-Objective Genetic Algorithm (MOGA) and the Parameter Space Investigation (PSI). The results are plotted on bidimensional projections of the objective domain space in the Appendix. The Pareto solutions computed by all the three methods are almost superimposed while the computational cost is significantly different as shown in Table 4. As previously observed on the test problems, the ANC performs better in terms of number of functions evaluations needed to compute one Pareto solution.

An order of magnitude is gained by using the ANC based on the local approximation.

4 Remarks on the efficiency of ANC optimization algorithm

The ANC, in terms of efficiency, performed better than the other tested algorithms. The reason lies in the way the local approximation is updated. The following formula gives a rough yet meaningful estimation of the number of function evaluations required by the ANC

$$N = N_{AP} + N_{NET} + N_P \tag{13}$$

The total number N is the sum of the evaluations needed to compute the first anchor point N_{AP} (see step 1), the size of the ANN training set N_{NET} (see step 2) and the number of nearly Pareto points computed at which the objective/constraint functions are evaluated to update the ANN approximation (see step 5). Being N_P the total number of Pareto points obtained, the efficiency is then easily computed as follows

$$\varepsilon_{ANC} = \frac{N}{N_P} = 1 + \frac{N_{AP} + N_{NET}}{N_P} \tag{14}$$

Efficiency improves as the number of computed Pareto solutions increases. This relation explains the capability of an approach based on the local approximation. The NC method is able to approximate the Pareto set with an almost evenly distributed set of points (Messac and Mattson 2004), therefore by increasing the number of Pareto solutions the ANC algorithm guarantees a good Pareto frontier coverage with good efficiency. The lower bound for the efficiency is

$$\bar{\varepsilon}_{ANC} = \lim_{N_P \to \infty} \frac{N}{N_P} = 1 \tag{15}$$

Similar relationships can be found for the most used algorithms. For instance, for the genetic algorithms the following estimation can be considered

$$\varepsilon_{GA} = \frac{N}{N_P} = n \frac{S}{N_P} = n \tag{16}$$

where S is the population size and n is the number of generations. At the last generation, all the individuals of the population are Pareto solutions ($S = N_P$).

Among the scalarization methods, the ε-constraints method is considered. In this case, the following relationship holds

$$\varepsilon_C = \frac{N}{N_P} = \frac{CM}{N_P} = M \tag{17}$$

where C is the number of the equivalent single objective problems that has to be solved and M is the average number of evaluations needed to perform each single objective minimization. Obviously for each problem a Pareto solution is found, therefore $C = N_P$. The number M depends on the minimization algorithm and on the tolerance on the termination condition. Typically M is proportional to the number of design variables. For a gradient based algorithm, being s the size of the design variable vector, $s + 1$ function evaluations are needed to estimate the gradients by means of finite differences.

For the global approximation approach the efficiency is

$$\varepsilon_G = \frac{N_I}{N_P} \tag{18}$$

where N_I is the size of the data set used to tune the approximated model. By increasing the number of the Pareto points N_P, the limit is

$$\bar{\varepsilon}_G = \lim_{N_P \to \infty} \frac{N_I}{N_P} = l \neq 0 \tag{19}$$

The limit l it is greater than 0 because N_I it is not independent on N_P. The number N_P should be increased to reach the desired accuracy of the Pareto set approximation. In this case an approximated model with the adequate accuracy must be used and the model accuracy is a *consequence* of the data set size N_I used to identify the model parameters.

Table 5 Efficiency of some widely used algorithms

ANC	Genetic algorithm	ε-constraint method	Global approximation
$\varepsilon_{ANC} = 1 + \frac{N_{AP}+N_{NET}}{N_P}$	$\varepsilon_{GA} = n\frac{S}{N_P} = n$	$\varepsilon_C = \frac{CM}{N_P} = M$	$\varepsilon_G = \frac{N_I}{N_P}$

An upper bound for the function f accuracy is given by the following inequality (Statnikov and Matusov 1995; Niederreiter 1987)

$$\Delta f \leq L d_N \tag{20}$$

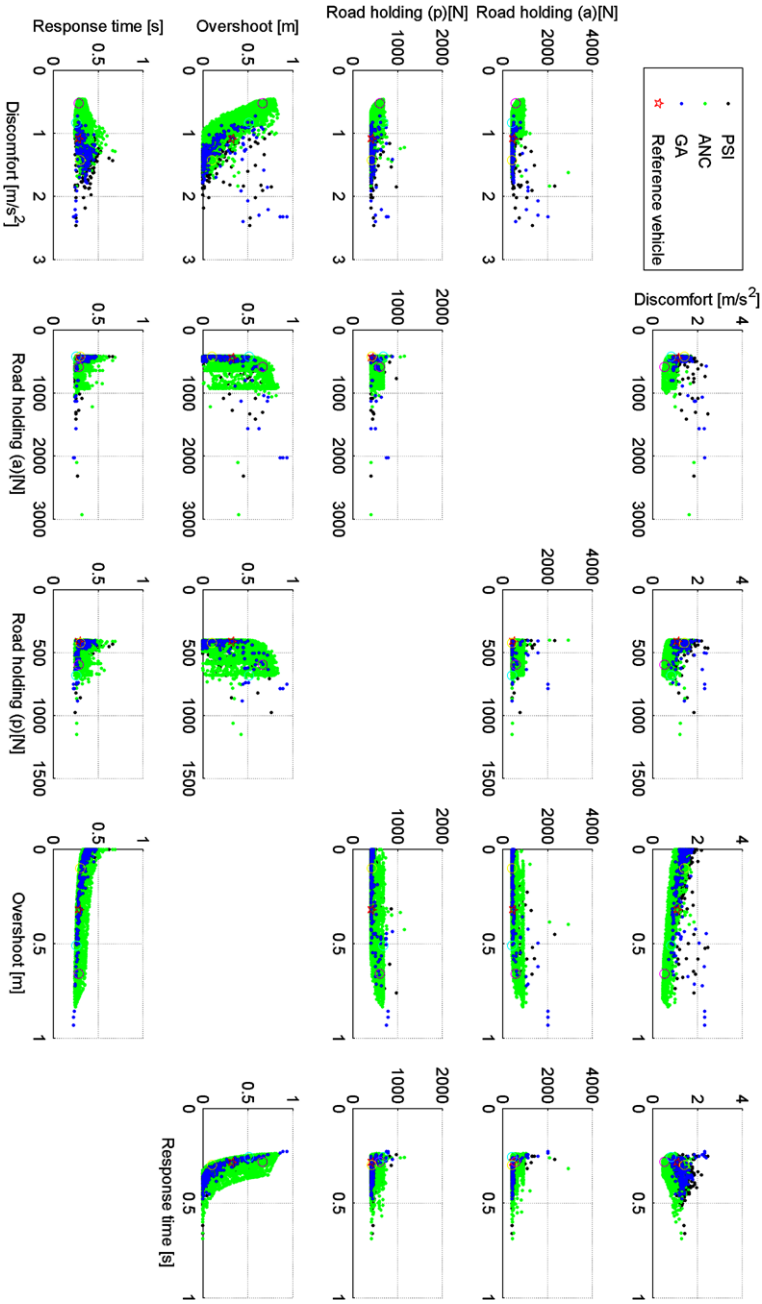
where L is the Lipschitz constant of the function f and d_N is the dispersion of the sequence used to generate the identification data set. To reduce d_N and improve the accuracy, the sequence size N_I must be increased. In the case of local approximation, the data used to update the approximated model are the Pareto (or nearly Pareto) solutions found at previous iterations. By increasing the identification data set size the efficiency it is not affected because the number of Pareto solutions N_P increases as well. These simple considerations, summarized in Table 5, explain why the ANC and more in general local approximation based algorithms can be very efficient. Obviously, ANC requires a tuning phase of the approximated model as long as other methods require different parameters to be set. This phase should be taken under consideration while analyzing the algorithm efficiency.

The exploration of the Pareto set may be more difficult if the number of objective functions increases. For bi-objective problems the ANC exploration of the Pareto set is very efficient and robust. As the number of constraints increases, the algorithm requires more effort as long as, in general, other approaches. Since the method is based on local approximation, it could be used to refine the solutions obtained with other methods which approximate the Pareto set at once (i.e., genetic algorithms and quasi Monte Carlo method).

5 Conclusion

An algorithm based on the local approximation approach is presented and tested with good results on some known test functions. The algorithm is used to determine the stiffness and damping of a passenger vehicle suspension system to optimize on conflicting objectives such as comfort, road holding. A linear quarter model vehicle is used to simulate the system while passing over an uneven road. The problem is further extended by optimizing the low frequency roll motion of the car body while turning as well as the ride comfort and road holding. Non linear dampers have been considered. The Pareto set is computed with good accuracy and higher efficiency with respect to some widely used optimization algorithms; less functions evaluations are needed to compute each Pareto solution. Through the local approximation of the objective and constraint functions many evaluations are saved because the approximated model is accurate close to the Pareto set and it is left intentionally inaccurate far from it. More in general, the results show that local approximation based algorithms, if properly designed, have an intrinsic high efficiency which is not strongly affected by the number of functions evaluations required to improve the solution accuracy.

Appendix


PSI
ANC
GA
Reference vehicle
Discomfort [m/s^2]
Road holding (a)[N]
Road holding (p)[N]
Overshoot [m]
Response time [s]

References

Das I, Dennis J (1997) A closer look at the drawbacks of minimizing weighted sum of objectives for Pareto set generation in multicriteria optimization problems. Struct Optim 14(1):63–69

Das I, Dennis J (1998) Normal boundary intersection: a new method for generating Pareto optimal points in multicriteria optimization problems. SIAM J Optim 8(4):631–657

Deb K, Thiele L, Laumanns M, Zitzler E (2005) Scalable test problems for evolutionary multiobjective optimization. In: Abraham A et al (eds) Evolutionary multiobjective optimization. Springer, London, pp 105–145

Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proc 6th int symposium on micromachine and human science, Nagoya, pp 39–43

Fadel G, Li Y (2002) Approximating the Pareto curve to help solve bi-objective design problems. Struct Optim J 23(4):280–296

Gobbi M, Mastinu G (2001) Analytical description and optimisation of the dynamic behaviour of passively suspended road vehicles. J Sound Vib 245(3):457–481

Gobbi M, Mastinu G (1999) Global approximation: performance comparison of different methods, with an application to road vehicle system engineering. In: Haque I et al (ed) Innovation in vehicle design and development. DE, vol 101. ASME, New York, pp 15–24

Gobbi M, Levi F, Mastinu G (2004) Multi-objective design of the suspension system of road vehicles. Veh Syst Dyn 41:537–546. Supplement

Goel T, Vaidyanathan R, Haftka RT, Shyy W, Queipo NV, Tucker K (2007) Response surface approximation of Pareto optimal front in multi-objective optimization. Comput Methods Appl Mech Eng 196:879–893

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison Wesley, New York

Haykin S (1998) A comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River

Hrovat D (1993) Applications of optimal control to advanced automotive suspension design. ASME J Dyn Syst Meas Control 115(2B):328–342

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680

Kuhn HW, Tucker AW (1951) Non linear programming. In: Proc 2nd Berkely symposium on mathematical statistics and probability, Berkely, pp 481–492

Maginot J, Guenov M, Utyuzhnikov S (2008) Local Pareto approximation for multi-objective optimization. Eng Optim 40(9):821–847

Mastinu G (1994) Automotive suspension design by multiple objective programming. In: Proc int symposium on advanced vehicle control (AVEC'94), Tsukuba, Japan, pp 73–78

Mastinu G, Gobbi M, Miano C (2006) Optimal design of complex mechanical systems. Springer, Berlin

Messac A, Mattson CA (2004) Normal constraint method with guarantee of even representation of complete Pareto frontier. AIAA J 42(10):2101–2111

Messac A, Ismail-Yahaya A, Mattson CA (2003) The normalized normal constraint method for generating the Pareto frontier. Struct Multidiscip Optim 25(2):86–98

Michalewicz Z (1994) Genetic algorithms + data structures = evolution programs. Springer, Berlin

Miettinen K (1999) Nonlinear multiobjective optimization. Kluwer Academic, Boston

Mitschke M (1990) Dynamik der Kraftfahrzeuge. Springer, Berlin

Myers RH, Montgomery DC (1995) Response surface methodology: process and product optimization using designed experiments. Wiley, New York

Niederreiter H (1987) Random number generation and quasi-Monte Carlo methods. Society for Industrial and Applied Mathematics, Philadelphia

Papalambros PY, Wilde DJ (2000) Principles of optimal design. Cambridge University Press, New York

Statnikov RB, Matusov JB (1995) Multicriteria optimization and engineering, 1st edn. Chapman & Hall, New York

Tappeta RV, Renaud JE, Messac A, Sundararaj GJ (2000) Interactive physical programming: tradeoff analysis and decision making in multicriteria optimization. Comput Control, AIAA J 36:219–226

Tezuka S (1995) Uniform random numbers: theory and practice. Kluwer Academic, Boston

Van Beers WCM, Kleijnen JPC (2003) Kriging for interpolation in random simulation. J Oper Res Soc 54(3):255–262

Zhang WH (2003) On the Pareto optimum sensitivity analysis in multicriteria optimization. Int J Numer Methods Eng 58(6):955–977
Zhang Q, Liu W, Tsang E, Virginas B (2010) Expensive multiobjective optimization by MOEA/D with Gaussian process model. IEEE Trans Evol Comput 14(3)
Zitzler E, Knowles J, Thiele L (2008) Quality assessment of Pareto set approximations. In: Branke J (ed) Multiobjective optimization. Springer, Berlin, pp 373–404