Generation of efficient solutions in Multiobjective Mathematical Programming problems using GAMS.

Effective implementation of the ε-constraint method

George Mavrotas

Lecturer, Laboratory of Industrial and Energy Economics, School of Chemical Engineering
National Technical University of Athens, Zografou Campus, Athens 15780, Greece.
Tel: +30 210-7723202, fax: +30 210 7723155, e-mail: mavrotas@chemeng.ntua.gr

Abstract: According to the most widely accepted classification the Multiobjective Mathematical Programming (MMP) methods can be classified as a priori, interactive and a posteriori, according to the decision stage in which the decision maker expresses his/her preferences. Although the a priori methods are the most popular, the interactive and the a-posteriori methods convey much more information to the decision maker. Especially, the a-posteriori (or generation) methods inform the decision maker about the whole context of the decision alternatives before his/her final decision. However, the generation methods are the less popular due to their computational effort and the lack of widely available software. The basic step towards further penetration of the generation methods in MMP applications, is to provide appropriate codes for Mathematical Programming (MP) solvers that are widely used by people in engineering, economics, agriculture etc. The present work is an effort to effectively implement the ε-constraint method for producing the efficient solutions in a MMP. We propose a variation of the method (augmented ε-constraint method-AUGMECON) that produces only efficient solutions (no weakly efficient solutions) and also avoids redundant iterations as it can perform early exit from the relevant loops (that lead to infeasible solutions), accelerating the whole process. Finally, we implement the method in an adjustable GAMS model using an example from the energy sector, describing in detail the necessary code.

1. Multiobjective Mathematical Programming and efficient solutions

The solution of Mathematical Programming (MP) problems with only one objective function is a straightforward task. The output is the optimal solution and all the relevant information about the values of the decision variables, shadow prices etc. In Multiobjective Mathematical Programming (MMP) there are more than one objective functions and there is no single optimal solution that simultaneously optimizes all the objective functions. In these cases the decision makers are looking for the "most preferred" solution. In MMP the concept of optimality is replaced with that of efficiency or Pareto optimality. The efficient (or Pareto optimal, nondominated, non-inferior) solutions are the solutions that cannot be improved in one objective function without deteriorating their performance in at least one of the rest. The mathematical definition of the efficient solution is the following (without loss of generality assume that all the objective functions f_i, i=1…p are for maximization): A feasible solution **x** of a MMP problem is efficient if there is no other feasible solution **x'** such as $f_i(\mathbf{x'}) \geq f_i(\mathbf{x})$ for every i=1, 2, …,p with at least one strict inequality. Every efficient solution corresponds to a nondominated or non-improvable vector in the criterion space. If we replace the condition $f_i(x') \geq f_i(x)$ with $f_i(x') > f_i(x)$ we obtain the weakly efficient solutions. Weakly efficient solutions are not usually pursued in MMP because they may be dominated by other efficient solutions. The rational decision maker is looking for the most preferred solution among the efficient solutions of the MMP. In the absence of any other information, none of these solutions can be said to be better than the other. Usually a decision maker is needed to provide additional preference information and to identify the "most preferred" solution.

2. Classification of the MMP methods

According to Hwang and Masud (1979) the methods for solving MMP problems can be classified into three categories according to the phase in which the decision maker involves in the decision making process expressing his/her preferences: The a priori methods, the interactive methods and the generation or a posteriori methods. In a priori methods the decision maker expresses his/her preferences before the solution process (e.g. setting goals or weights for the objective functions). The criticism about the a priori methods is that it is very difficult for the decision maker to know beforehand and to be able to accurately quantify (either by means of goals or weights) his/her preferences. In the interactive methods phases of dialogue with the decision maker are interchanged with phases of calculation and the process usually converges after a few iterations to the most preferred solution. The decision maker progressively drives the search with his answers towards the most preferred solution. The drawback is that he never sees the whole picture (the set of efficient solutions) or an approximation of it. Hence, the most preferred solution is "most preferred" in relation to what he/she has seen and compare so far. In a posteriori methods (or generation methods) the efficient solutions of the problem (all of them or a sufficient representation) are generated and then the decision maker involves, in order to select among them, the most preferred one (see e.g. the interactive filtering process proposed by Steuer, 1986).

3. Generation methods

The generation methods are the less popular due to their computational effort (the calculation of the efficient solutions is usually a time consuming process) and the lack of widely available software. However, they have some significant advantages. The solution process is divided into two phases: First, the generation of the efficient solutions and subsequently the involvement of the decision maker when all the information is on the table. Hence, they are favourable whenever the decision maker is hardly available and the interaction with him is difficult, because he is involved only in the second phase, having at hand all the possible alternatives (the efficient solutions of the MMP). Besides, the fact that none of the potential solutions has been left undiscovered, reinforces the decision maker's confidence on the final decision.

For special kind of MMP problems (mostly linear problems) of small and medium size, there are also methods that produce the entire efficient set (see e.g. Steuer 1986, Mavrotas 1998, Miettinen 1999). Here we will focus on the general case, where relatively large MMP problems can be tackled. In general, the most widely used generation methods are the weighting method and the ε-constraint method. These methods are used to provide a representative subset of the efficient set. problems. Assume the following MMP:

$\max (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_p(\mathbf{x}))$

st

$\mathbf{x} \in S$

where $\mathbf{x}$ is the vector of decision variables, $f_1(\mathbf{x}), \ldots f_p(\mathbf{x})$ are the p objective functions and S is the feasible region.

3.1 The weighting method

In the weighting method, the weighted sum of the objective functions is optimized. The problem is stated as follows:

$$\max (w_1 \times f_1(\mathbf{x}) + w_2 \times f_2(\mathbf{x}) + \ldots + w_p \times f_p(\mathbf{x}))$$

$$\text{st} \qquad (1)$$

$$\mathbf{x} \in S$$

By varying the weights w_i we obtain different efficient solutions.

3.2 The ε-constraint method

In the ε-constraint method we optimize one of the objective functions using the other objective functions as constraints, incorporating them in the constraint part of the model as shown below:

$$\max f_1(\mathbf{x})$$

$$\text{st}$$

$$f_2(\mathbf{x}) \geq e_2$$

$$f_3(\mathbf{x}) \geq e_3 \qquad (2)$$

$$\ldots$$

$$f_p(\mathbf{x}) \geq e_p$$

$$\mathbf{x} \in S$$

By parametrical variation in the RHS of the constrained objective functions (e_i) the efficient solutions of the problem are obtained.

The e-constrained method has several advantages over the weighting method.

1. For linear problems, the weighting method is applied to the original feasible region and results to a corner solution (extreme solution), thus generating only efficient extreme solutions. On the contrary, the ε-constraint method alters the original feasible region and is able to produce non-extreme efficient solutions. As a consequence, with the weighting method we can spend a lot of runs that are redundant in the sense that there can be a lot of combination of weights that result in the same efficient extreme solution. On the other hand, with the ε-constraint we can exploit almost every run to produce a different efficient solution, thus obtaining a more rich representation of the efficient set.
2. The weighting method cannot produce unsupported efficient solutions in multiobjective integer and mixed integer programming problems, while the ε-constraint method does not suffer from this inadequacy (Steuer 1986, Miettinen 1999).
3. In the weighting method the scaling of the objective functions has strong influence in the obtained results. Therefore, we need to scale the objective functions to a common scale before forming the weighted sum. In the e-constrained method this is not necessary.
4. An additional advantage of the ε-constraint method is that we can control the number of the generated efficient solutions by properly adjusting the number of grid points in each one of the objective function ranges. This is not so easy with the weighting method (see point 1 above).

4. The augmented ε-constraint method (AUGMECON)

Despite its advantages over the weighting method the ε-constraint method has two points that need attention: the range of the objective functions over the efficient set (mainly the calculation of nadir values) and the guarantee of efficiency of the obtained solution. Let's take a closer look to these two points.

In order to properly apply the ε-constraint method we must have the range of every objective function at least for the *p-1* objective functions that will be used as constraints. The calculation of the range of the objective functions over the efficient set is not a trivial task (see e.g. Isermann and Steuer 1987, Reeves and Reid 1988, Steuer 1997). While the best value is easily attainable as the optimal of the individual optimization, the worst value over the efficient set (nadir value) is not. The most common approach is to calculate these ranges from the payoff table (the table with the results from the individual optimization of the *p* objective functions). The nadir value is usually approximated with the minimum of the corresponding column (see e.g. Cohon 1978, Steuer 1986, Miettinen 1999). However, even in this case, we must be sure that the obtained solutions from the individual optimization of the objective functions are indeed efficient solutions. In the presence of alternative optima the obtained by a commercial software optimal solution is not a guaranteed efficient solution. In order to overcome this ambiguity we propose the use of lexicographic optimization for every objective function in order to construct the payoff table with only efficient solutions. A simple remedy in order to bypass the difficulty of estimating the nadir values of the objective functions is to define reservation values for the objective functions. The reservation value acts like a lower (or upper for minimization objective functions) bound. Values worse than the reservation value are not allowed.

The second point of attention is that the optimal solution of problem (2) is guaranteed to be an effcient solution only if all the (p-1) objective functions' constraints are binding (Miettinen 1999, Ehrgott and Wiecek 2005). Otherwise, if there are alternative optima (that may improve one of the non binding constraints that corresponds to an objective function) the obtained optimal solution of problem (2) is not in fact efficient but it is a weakly efficient solution. In order to overcome this ambiguity we propose the transformation of the objective function constraints to equalities by explicitly incorporating the appropriate slack or surplus variables. In the same time, the sum of these slack or surplus variables is used as a second term (with lower priority) in the objective function forcing the program to produce only efficient solutions. The second term drives the search to look among the possible alternative optima of max $f_1(x)$ for the one that maximizes the sum. The new problem becomes:

$$\max (f_1(\mathbf{x}) + \delta \times (s_2 + s_3 + \ldots + s_p))$$

st

$$f_2(\mathbf{x}) - s_2 = e_2$$

$$f_3(\mathbf{x}) - s_3 = e_3 \qquad (3)$$

. . .

$$f_p(\mathbf{x}) - s_p = e_p$$

$$\mathbf{x} \in S \text{ and } s_i \in R^+$$

where δ is a small number (usually between 10^{-3} and 10^{-6}).

Proposition: The above formulation (3) of the ε-constraint method produces only efficient solutions (it avoids the generation of weakly efficient solutions).

Proof: Assume that the problem (2) has alternative optima and one of them (depicted as **x'**) dominates the optimal solution (depicted as **x**) obtained from problem (3). This means that the vector (z_1, e_2+s_2, …, e_p+s_p) is dominated by the vector (z_1, e_2+s_2', …, e_p+s_p') or in other words: (remember that $z_1 = \max f_1(\mathbf{x})$ is the same for the two cases as we have alternative optima):

$$e_2 + s_2 \leq e_2 + s_2'$$

$$e_3 + s_3 \leq e_3 + s_3'$$

$$\ldots \qquad (4)$$

$$e_p + s_p \leq e_p + s_p'$$

with at least on strict inequality. Taking the sum of these relations and based on the fact that there is at least one strict inequality we conclude that:

$$\sum_{i=2}^{p} s_i < \sum_{i=2}^{p} s_i' \qquad (5)$$

But this contradicts the initial assumption that the optimal solution of (3) maximizes the sum of s_i. Hence, there is no solution **x'** that dominates the obtained solution **x**, or, in other words the obtained solution **x** from problem (3) is efficient. □

In order to avoid any scaling problems it is recommended to replace the s_i in the second term of the objective function by s_i/r_i, where r_i is the range of the i-th objective function (as calculated from the payoff table). Thus, the objective function of the ε-constraint method becomes:

$$\max (f_1(\mathbf{x}) + eps \times (s_2/r_2 + s_3/r_3 + \ldots + s_p/r_p)) \qquad (6)$$

The proposed version of the ε-constraint method that corresponds to model (3) with the objective function (6) will be called hereafter augmented ε-constraint method or AUGMECON method.

Practically, the ε-constraint method is implemented as follows: From the payoff table we obtain the range of each one of the *p-1* objective functions that are going to be used as constraints. Then we divide the range of the i-th objective function to q_i equal intervals using (q_i-*1*) intermediate equidistant grid points. Thus we have in total (q_i+*1*) grid points that are used to vary parametrically the RHS (e_i) of the i-th objective function. The total number of runs becomes (q_2+*1*) × (q_3+*1*) × . . . × (q_p+*1*). A desirable characteristic of the ε-constraint method is that we can control the density of the efficient set representation by properly assigning the values to the q_i. The higher the number of grid points the more dense is the representation of the efficient set but with the cost of higher computation times. A trade off between the density of the efficient set and the computation time is always advisable.

An innovative addition to the algorithm is the early exit from the nested loop when the problem (3) becomes infeasible for some combination of e_i. The early exit from the loops work as follows: The bounding strategy for each one of the objective function starts from the more relaxed formulations (lower bound for a maximization objective function or upper bound for a minimization) and move to the most strict (individual optima). In this way, when we arrive to an infeasible solution there is no need to perform the remaining runs of the loop (as the problem will become even stricter and thus remains infeasible) and we force an exit from the loop (see the example in Figure 1).

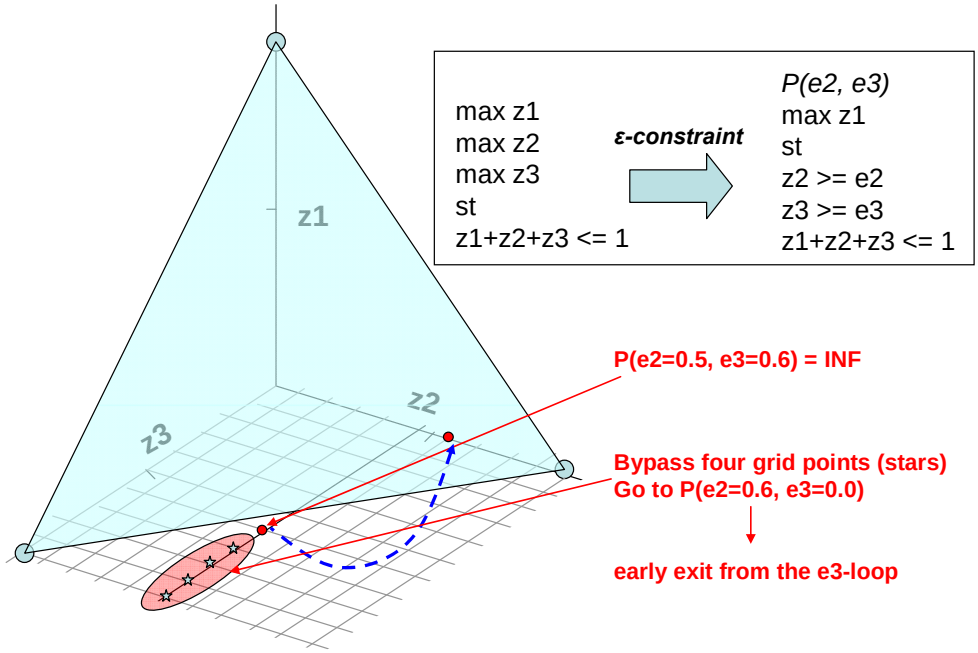

Figure 1: Graphical example for the early exit from the nested loops in a multi-objective problem with three objective functions.

The early exit saves a lot of computational time in problems with more than 2-3 objective functions. As the number of the objective functions increase the reduction in computation time is more apparent. In a real case study problem with 6 objective functions, 236 variables and 96 constraints a 45% reduction in iterations and accordingly in computation time was observed (with 5 grid point per objective function the initial 2^5 runs = 3125 were reduced to 1705 with the early exit from the loops).

4.1 Illustrative example

In order to show the weak points of the ε-constraint and the proposed remedies we use the following simple numerical example.

max f1 = X1
max f2 = 3 X1 + 4 X2
st
X1 <= 20
X2 <= 40
5 X1 + 4 X2 <= 200

The feasible region and the direction of the two objective functions are shown in Figure 2. The efficient set (or efficient frontier, Pareto set, nondominated set) for this problem is depicted with the heavy line (segment QR).

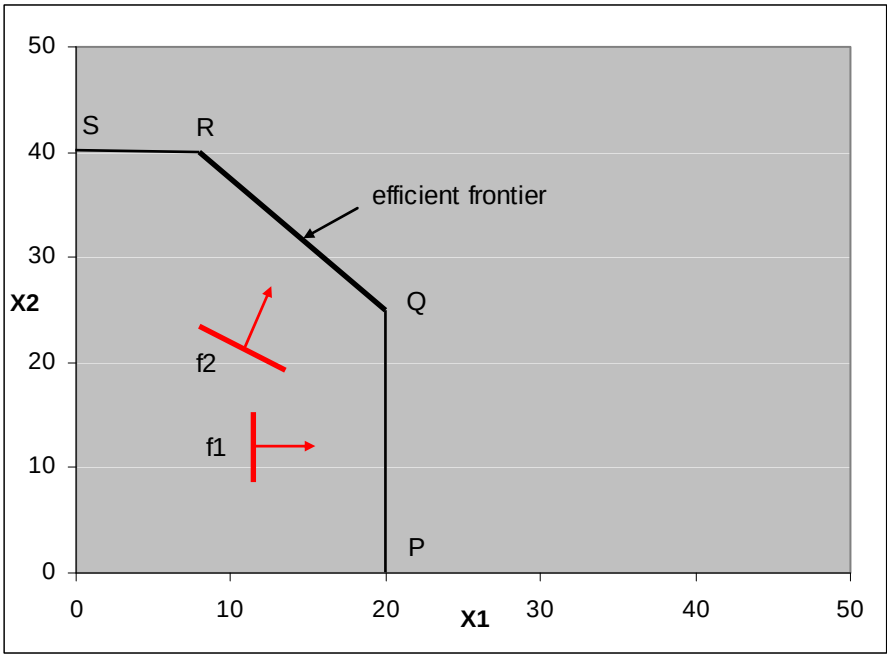

Figure 2: Feasible region and directions of objective functions

Following the conventional ε-constraint method we first calculate the payoff table by simply calculating the individual optima of the objective functions. A conventional LP optimizer will produce the payoff table shown in Table 1.

Table 1: Payoff table obtained by a conventional LP optimizer

	f_1	f_2
max f_1	20	60
max f_2	8	184

It can be noticed that the optimal solution obtained for f_1 (f_1=20, f_2=60) that corresponds to point P is a dominated solution in the problem due to alternative optima (see e.g. point Q). However, it is almost sure that a conventional LP optimizer will calculate the solution of point P first and will stop the searching giving this solution as output. In order to avoid this situation we proceed to the lexicographic optimization of the objective functions and the results are shown in Table 2.

Table 2: Payoff table obtained by the lexicographic optimization of the objective functions

	f_1	f_2
max f_1	20	160
max f_2	8	184

With the lexicographic optimization we obtain as the solution that maximizes f_1 the one that corresponds to point Q which is a non dominated solution.

In general, the lexicographic optimization of a series of objective functions is to optimize the first objective function and then among the possible alternative optima optimize for the second objective function and so on. Practically, the lexicographic optimization is performed as follows: we optimize the first objective function (of higher priority), obtaining max $f_1=z_1^*$. Then we optimize the second objective function by adding the constraint $f_1=z_1^*$ in order to keep the optimal solution of the first

optimization. Assume that we obtain max $f_2 = z_2^*$. Subsequently, we optimize the third objective function by adding the constraints $f_1=z_1^*$ and $f_2 = z_2^*$ in order to keep the previous optimal solutions and so on until we finish with the objective functions (see also Erwin Karvelagen's example in http://www.gams.com/~erwin/book/lp.pdf, page 231).

After the calculation of the payoff table we divide the ranges of the objective functions to four equal intervals and we use the five grid points as the values of e_2 in the ε-constraint method. In the first case we apply the model (2) of the conventional ε-constraint method. The results are shown in the diagram of Figure 3

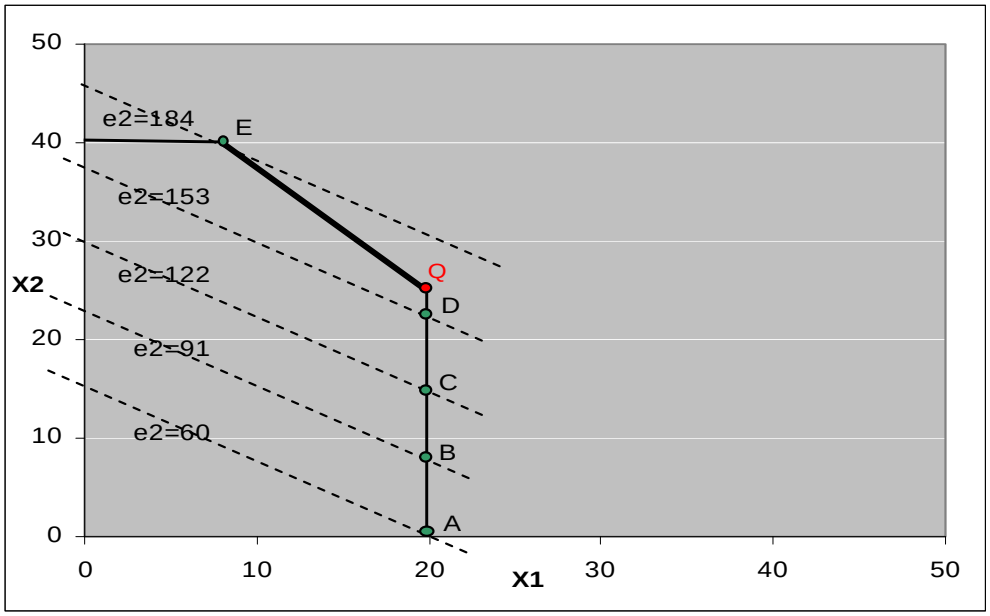

Figure 3: Results of the conventional ε-constraint method

The solutions that correspond to the points A, B, C, D, E are the output of the method. In fact only point E is efficient while the other 4 are weakly efficient point (dominated by point Q). However, if we have used (with the same payoff table) the augmented ε-constraint method (model (3)) the output would be point Q (obtained 4 times) and point E which are both non dominated points. It means that although there was a dominated solution in the payoff table the augmented ε-constraint method produces the correct results due to the corrective use of the maximization (in a second level) of the surplus variables.

Nevertheless, if we use the payoff table from the lexicographic optimization the results of the ε-constraint method are even better as we obtain a much more dense representation of the efficient set (see Figure 4). We can see that points A', B', C', D' and E' are all efficient points that adequately describe the efficient set.

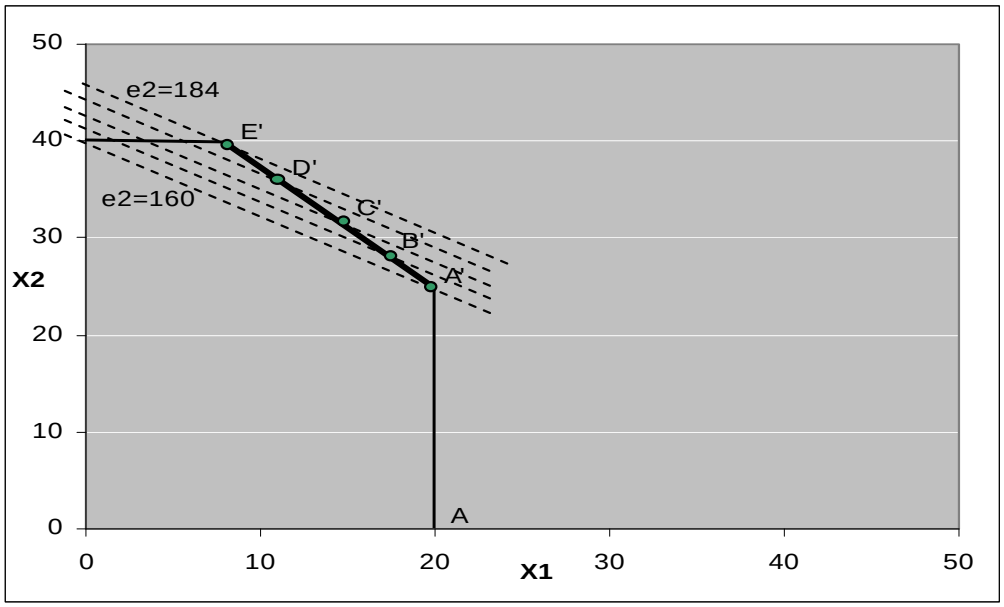

Figure 4: Results of the augmented ε-constraint method for model (3)

In the implementation of the ε-constraint method to the GAMS model we have incorporated these two features (lexicographic optimization for the calculation of the payoff table and use the augmented ε-constraint method to avoid weakly efficient solutions) in order to overcome the known shortcomings of the conventional ε-constraint method.

5. Implementation of the augmented ε-constraint method in GAMS

5.1 The example

We will show the implementation of the augmented ε-constraint method in GAMS using a simplified example from the power generation field. Assume that we have four type of power generation units in a region, namely, lignite fired, oil fired, natural gas fired and units exploiting renewable energy sources (RES) which are mostly small hydro and wind. The characteristics of these units are shown in Table 3

Table 3: Power generation characteristics

	Lignite	Oil	Natural Gas	RES
Maximum production per year (GWh)	31000	15000	22000	10000
Cost of production (€/MWh)	30	75	60	90
CO_2 emission coefficient (t/MWh)	1.44	0.72	0.45	0

The yearly demand is 64000 GWh and is characterized by a load duration curve which can be divided into three type of loads: base load (60%), medium load (30%) and peak load (10%). The lignite fired units can be used only for base and middle load, the oil fired units for middle and peak load, the RES units for base and peak load and the natural gas fired units for all type of loads. The endogenous sources are lignite and RES. We consider three objective functions: the minimization of production cost, the minimization of CO_2 emissions and the minimization of external dependence (i.e. oil and

natural gas) and we want to generate the relative efficient solutions of the problem. The multiobjective model is as follows:

MIN 30 LIGN + 75 OIL + 60 NG + 90 RES
MIN 1.44 LIGN + 0.72 OIL + 0.45 NG
MIN OIL + NG
ST
LIGN – LIGN1 – LIGN2 = 0
OIL – OIL2 – OIL3 = 0
NG – NG1 – NG2 – NG3 = 0
RES – RES1 – RES3 = 0
LIGN <= 31000
OIL <= 15000
NG <= 22000
RES <= 10000
LIGN1 + NG1 + RES1 >= 38400
LIGN2 + OIL2 + NG2 >= 19200
OIL3 + NG3 + RES3 >= 6400

The GAMS model that implements the ε-constraint method for the specific problem is presented in the model library of GAMS (no 319, name "epscm", http://www.gams.com/modlib/libhtml/epscm.htm). The concept is to split each problem into two models: One model with the specific problem's characteristics (indicated with `$STitle Example model definitions` in line 57) and one model with the required characteristics of the ε-constraint method (indicated as `$STitle eps-constraint method` in line 106). For a new problem the user must alter only the parts that describe the specific problem and properly modify the parameters that are necessary for the ε-constraint. More specifically the user alters the following lines of the GAMS code according to the specific characteristics of the MMP problem:

Line 75: The user inputs the number of the objective functions of the problem (set k)

Lines 79-80: The user inputs the direction of the objective functions (1 for maximization and -1 for minimization).

Line 171: The name of the output file with the Pareto optimal solutions

Line 174: The number of grid points for each objective function in the ε-constraint method. It can be modified for some objective functions with the dynamic set handling in line 186.

Line 214: In the output building section of the code, beside the values of the objective functions the user may add any other decision variable that wish to record for every efficient solution. In this case it is recommended to appropriately add labels to the label row in line 195.

The output of the specific GAMS model is the display of the unique Pareto optimal solutions (using some posix utilities) at the end of the *.lst file. The payoff table as well as the grid points are also displayed in the *.lst file.

5.2 Comments on the GAMS code

- The computation time for the generation of the efficient solutions in the case of 4 interval per objective function is 6 seconds in a Pentium M 1.7 Ghz. For the case of 10 intervals per objective function the computation time increases to 22 seconds.

- The user may input reservation values for the objective functions by adding upper or lower bounds to the z-variables (using the .up or .lo suffix). By the term reservation values we mean the worst acceptable value for each one of the objective functions. In other words the reservation value is a lower bound for a maximization objective function and an upper bound for a minimization. The payoff table is calculated taken into account the user input about the reservation values for the objective functions. As the reservation values for one objective function may influence the optimal values for some other objective functions (as they are incorporated as constraints in the model) a little trial and error is recommended (input reservation values, check the payoff table for the individual optima and if they are seriously deteriorated try some other reservation values and repeat this interactive procedure). By default there are no reservation values.
- The payoff table is calculated performing *k* lexicographic optimizations in lines 154-164.
- The "repeat – until" loop in line 194-206 is used to exhaustively visit all the grid points and minimize the redundant runs after an infeasible solution occurs (when the problem becomes very strict). The traversing strategy is to start from the more relaxed formulations (lower bound for a maximization objective function or upper bound for a minimization) and move to the most strict (individual optima).
- The above presented code can also be used without any changes for Multiobjective Integer Programming problems or Multiobjective Mixed Integer Programming problems as well as for the corresponding Non Linear cases. In the case of MIP problems the It must be stressed that even duality gaps in non-convex problems do not harm the ability of the ε-constraint method to produce every efficient point of the MMP problem.

6. Concluding remarks

In the present text we propose a way of implementation of the ε-constraint method in GAMS. Special care is taken in order to secure the efficiency of the obtained solutions by using the augmented ε-constraint method. The code can be easily adapted to the needs of the user (number and direction of the objective functions, density of the efficient set representation, reservation values for the objective functions). We also incorporate some acceleration issues (early exit from the loops) which are particularly useful when there are a lot of objective functions.

After the generation of the efficient solutions, the next step is to assist the decision maker in selecting his/her most preferred solution among them. The problem then is reduced to the selection among several alternatives evaluated in multiple criteria. Although this task is beyond the scope of the current paper we can say that there are a lot of Multiple Criteria Decision Analysis methods that can be useful and the interested reader can find more information in Figueira et al (2005), Belton and Stewart (2000) that extensively cover the current trends in the field.

References

Belton, V. and T. Stewart. 2000. *Multiple Criteria Decision Analysis. An Integrated Approach*, Kluwer Academic Publishers.

Cohon, J.L. (1978). *Multiobjective Programming and Planning*. Academic Press, New York.

Ehrgott M. and Wiecek M. (2005) “Multiobjective Programming” in: J.Figueira, S.Greco, M. Ehrgott (eds) *Multiple Criteria Decision Analysis. State of the Art Surveys* pp. 667-722, Springer

Figueira, J. Greco, S. and Ehrgott M. (2005) *Multiple Criteria Decision Analysis. State of the Art Surveys*, Springer

Hwang, C.L., Masud, A. (1979). *Multiple Objective Decision Making. Methods and Applications: A state of the art survey*. Lecture Notes in Economics and Mathematical Systems Vol. 164. Springer-Verlag, Berlin.

Isermann, H. and Steuer, R.E. (1987) “Computational experience concerning payoff tables and minimum criterion values over the efficient set”, *European Journal of Operational Research* 33**,** 91-97.

Mavrotas G. and Diakoulaki D. (2005), “Multi-criteria branch & bound: A vector maximization algorithm for Mixed 0-1 Multiple Objective Linear Programming”, *Applied Mathematics and Computation*, 171(1) 53-71.

Reeves, G.R. and Reid, R.C. (1988) “Minimum values over the efficient set in multiple objective decision making”, *European Journal of Operational Research* 36**,** 334-338.

Steuer, R.E. (1997) “Non-Fully Resolved Questions about the Efficient/Nondominated set” in: J. Climaco (ed) *Multicriteria Analysis* pp. 585-589, Springer.

Steuer, R.E. (1986). *Multiple Criteria Optimization.Theory, Computation and Application*, 2nd edition, Krieger, Malabar FL.

http://www.gams.com/~erwin/book/lp.pdf