Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
European Journal of Operational Research xxx (2003) xxx–xxx
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/dsw

A regression study of the number of efficient extreme points in multiple objective linear programming

Ralph E. Steuer [*], Craig A. Piercy

Department of Banking and Finance, Terry College of Business, University of Georgia,
Athens, GA 30602-6253, USA

Received 29 July 2003; accepted 15 September 2003

Abstract

In this paper we employ regression analysis to construct relationships for predicting the number of efficient extreme points in MOLPs (multiple objective linear programs) with up to 120,000 efficient extreme points, and the CPU time to compute them. Principal among the factors affecting the number of efficient extreme points and CPU time are the number of objectives, criterion cone size, number of constraints, number of variables, and the nonzero density of the constraint matrix. The regression equations show the degree to which interactions are present among the factors and provide a more formal basis for understanding how the complexity of the efficient set, an indicator of the difficulty involved in solving a multiple criteria problem, increases with problem size.

Keywords: Multiple objective linear programming; Large MOLPs; Random problem generator; Efficient extreme points; Regression; Computational experiments

1. Introduction

A significant area of endeavor that falls under the rubric of multiple criteria decision making (MCDM) is research into the computational properties and characteristics of problems in multiple objective linear programming. A multiple objective linear program (MOLP) can be expressed as

$$\begin{aligned} &\max\{\mathbf{c}^1\mathbf{x} = z_1\} \\ &\vdots \\ &\max\{\mathbf{c}^k\mathbf{x} = z_k\} \\ \text{s.t.}\quad &\mathbf{x} \in S = \{\mathbf{x} \in R^n \mid \mathbf{A}\mathbf{x} \leqslant \mathbf{b}, \mathbf{b} \in R^m, \mathbf{x} \geqslant \mathbf{0}\}, \end{aligned}$$

or alternately as

$$\text{“max”}\{\mathbf{C}\mathbf{x} = \mathbf{z} \mid \mathbf{x} \in S\},$$

where k is the number of objectives and $\mathbf{C}$ is the $k \times n$ *criterion matrix* whose rows are the $\mathbf{c}^i$. In an MOLP, a solution $\bar{\mathbf{x}} \in S \subset R^n$ is an *efficient* point if and only if there does not exist an $\mathbf{x} \in S$ such that $\mathbf{c}^i\mathbf{x} \geqslant \mathbf{c}^i\bar{\mathbf{x}}$ for all i and $\mathbf{c}^i\mathbf{x} > \mathbf{c}^i\bar{\mathbf{x}}$ for at least one i. The set of all efficient points is called the *efficient*

---

[*] Corresponding author. Tel.: +1-706-5423782; fax: +1-706-5429434.
E-mail address: rsteuer@uga.edu (R.E. Steuer).

0377-2217/$ - see front matter © 2003 Published by Elsevier B.V.
doi:10.1016/j.ejor.2003.09.014

set. While the efficient set is normally a portion of the surface of the feasible region, the efficient set has the tendency to grow rapidly as problem size increases.

Many authors have studied efficient sets from a computational point of view including Armand and Malivert (1991); Benson and Sayin (1997); Climaco and Antunes (1989); Dauer (1991); Ecker and Kouada (1978); Gal (1977); Isermann (1977); Korhonen et al. (1997); Mavrotas et al. (1998); Reeves and Reid (1988); Sayin (2000); Steuer (1994); Strijbosch et al. (1991); Wiecek and Zhang (1997); Yu and Zeleny (1975); and Zionts and Wallenius (1980). Topics that have been examined include:

1. Enumeration of all efficient extreme points.
2. Computation of neighborhoods of efficient extreme points.
3. Payoff tables and minimum criterion values over the efficient set.
4. Optimizing an arbitrary function over the efficient set.
5. Obtaining discrete representations of efficient facets.

Such research has led to an increased understanding of the efficient set and of the challenges involved in the development of effective procedures for searching the efficient set for a most preferred solution (see Gardiner and Steuer, 1994 and references therein).

Because the number of efficient extreme points continues to be an intriguing question as computers become ever more powerful, the purpose of this paper is to study, in problems with up to 120,000 efficient extreme points, the statistical relationship between the number of efficient extreme points (along with CPU time) and *problem-size* characteristics of an MOLP. By problem-size characteristics we mean factors such as the number of objectives, criterion cone size, number of constraints, number of variables, **A**-matrix nonzero density, etc.

Situations in which the regression results of this paper would be of value would be in (a) dynamic array size allocation in computer codes that enumerate efficient extreme points, (b) calibrating quad-tree data structures that store large numbers of efficient extreme points for rapid retrieval as in Habenicht (1982) and Sun and Steuer (1996, 2000), (c) providing a more formal basis for understanding how the complexity of the efficient set, a primary determinant in how difficult it is to solve a multiple criteria problem, grows as problem size increases, and (d) predicting the number of efficient extreme points and the time required to compute them when such information is necessary for carrying out or validating the results of computational studies such as in connection with the five topics above.

In Sections 2 and 3 we discuss the random problem generator employed and the experimental runs conducted. Sections 4–6 discuss initial regressions, a criterion cone measure called ACONEA, and the model building process. Section 7 discusses prediction and concludes the paper.

2. Random problem generator

The workhorse software for this paper was the ADBASE (Steuer, 2000) multiple objective linear programming code for computing all efficient extreme points. Introduced in 1974, and revised every few years, ADBASE is a versatile code that runs on PCs (under DOS) and on servers.

To construct MOLP test problems, ADBASE is equipped with a random problem generator. After providing

NUMB	number of MOLPs to be randomly generated
KSEED	initial seed to random number generator

the random problem generator utilizes specifications of the following twelve *configuration* parameters:

NOBJS	number of objectives k
NCONS	number of constraints m
NVARS	number of *structural* variables n
JLC	**C**-matrix coefficient lower bound
JUC	**C**-matrix coefficient upper bound
CDEN	percent **C**-matrix nonzero density
JLA	**A**-matrix coefficient lower bound

JUA	**A**-matrix coefficient upper bound
ADEN	percent **A**-matrix nonzero density
JLB	**A**-matrix row norm lower bound
JUB	**A**-matrix row norm upper bound
IDISP	reference point for center of feasible region *S*

NOBJS, NVARS, JLC, JUC and CDEN are used to generate the **C**-matrix as follows. After the number of nonzero coefficients N_{NC} implied by CDEN where

$$N_{\mathrm{NC}} = \mathrm{rounddown}[\mathrm{NOBJS} * \mathrm{NVARS} * \mathrm{CDEN}/100]$$

is computed, a $3 \times N_{\mathrm{NC}}$ *storage array*, is formed. The storage array idea, as taken from Arbel (1993), offers two advantages. One is that it saves storage with sparse matrices, and the other is that it enables control over the dispersion of the nonzero elements across the matrix.

Continuing, the first element in each column of the $3 \times N_{\mathrm{NC}}$ storage array is a nonzero **C**-matrix coefficient uniformly selected from the nonzero integers in the interval [JLC,JUC]. The second and third elements in each column are the row and column indices of the nonzero coefficient. For example, with

$$\begin{aligned}\mathrm{NOBJS} &= 4,\\ \mathrm{NVARS} &= 5,\\ \mathrm{JLC} &= -5,\\ \mathrm{JUC} &= 10,\\ \mathrm{CDEN} &= 40.\end{aligned}$$

$N_{\mathrm{NC}} = 8$, and with KSEED = 15 431, the resulting 3×8 storage array is

−3	4	5	2	10	−2	−2	10
1	2	3	4	1	2	3	4
1	1	2	2	3	3	4	5

which defines

$$\mathbf{C} = \begin{bmatrix} -3 & 0 & 10 & 0 & 0 \\ 4 & 0 & -2 & 0 & 0 \\ 0 & 5 & 0 & -2 & 0 \\ 0 & 2 & 0 & 0 & 10 \end{bmatrix}$$

As seen in **C**, the nonzero density is 40% exactly as specified by CDEN, the nonzero elements are dispersed about the **C**-matrix, and all nonzero coefficients are from the interval [JLC,JUC] = [−5,10].

NCONS, NVARS, JLA, JUA and ADEN are used to generate the **A**-matrix of constraint coefficients in a similar fashion. After the number of nonzero coefficients N_{NA} implied by ADEN where

$$N_{\mathrm{NA}} = \mathrm{rounddown}[\mathrm{NCONS} * \mathrm{NVARS} * \mathrm{ADEN}/100]$$

is computed, a $3 \times N_{\mathrm{NA}}$ storage array is formed. For example, with

$$\begin{aligned}\mathrm{NCONS} &= 3,\\ \mathrm{NVARS} &= 5,\\ \mathrm{JLA} &= -3,\\ \mathrm{JUA} &= 9,\\ \mathrm{ADEN} &= 75.\end{aligned}$$

$N_{\mathrm{NA}} = 11$, and stemming from the same KSEED, the resulting 3×11 storage array is

4	2	6	9	2	6	−1	5	−2	−2	7
1	1	1	1	1	2	3	2	3	2	3
1	2	3	4	5	1	1	2	3	4	5

which defines

$$\mathbf{A} = \begin{bmatrix} 4 & 2 & 6 & 9 & 2 \\ 6 & 5 & 0 & -2 & 0 \\ -1 & 0 & -2 & 0 & 7 \end{bmatrix}$$

The only difference with the **A**-matrix is in the first row. To prevent unboundedness, the first row of **A** is always generated with all nonzero coefficients. This is seen in the first five columns of the storage array, and consequently in the first row of the **A**-matrix above. Also seen in **A** are that the nonzero density is $73\frac{1}{3}$% (as close as we can get to the 75% specified by ADEN) and all nonzero coefficients are from the interval [JLA,JUA] = [−3,9].

After the generation of the **A**-matrix, JLB, JUB and IDISP are used to form the right-hand side vector $\mathbf{b} \in R^m$. IDISP is first used to create a point $(\text{IDISP}, \ldots, \text{IDISP}) \in R^n$ about which the feasible region S is to be constructed. Letting $\mathbf{a}^i$ be the ith row of the **A**-matrix, each b_i is formed as follows:

1. Randomly select a nonnegative integer $\alpha_i \in [\text{JLB}, \text{JUB}]$ to form the point
   $$(\text{IDISP}, \ldots, \text{IDISP}) + \alpha_i \mathbf{a}^i$$
   which is to be on the boundary level curve of the ith constraint.
2. Form the ith constraint
   $$\mathbf{a}^i \mathbf{x} \leqslant \mathbf{a}^i[(\text{IDISP}, \ldots, \text{IDISP}) + \alpha_i \mathbf{a}^i]^T$$
   where the expression on the right is the computation for the value of b_i.

Note that if the process produces a negative value for b_i, the ith constraint would be written as a $\geqslant$ constraint if all right-hand sides are to be nonnegative.

For example, with the same 3×11 storage array for **A** but with

$$\begin{aligned} \text{NCONS} &= 3, \\ \text{NVARS} &= 5, \\ \text{JLB} &= 2, \\ \text{JUB} &= 2, \\ \text{IDISP} &= 5. \end{aligned}$$

we obtain the three calculations for $\mathbf{a}^i[(\text{IDISP}, \ldots, \text{IDISP}) + \alpha_i \mathbf{a}^i]^T$ of 127, 55 and 30. This then defines the constraints as follows:

$$\begin{aligned} &4x_1 + 2x_2 + 6x_3 + 9x_4 + 2x_5 \leqslant 127, \\ &6x_1 + 5x_2 - 2x_4 \leqslant 55, \\ &-x_1 - 2x_2 + 7x_5 \leqslant 30. \end{aligned}$$

When in random number generator mode, ADBASE can generate and solve for all efficient extreme points, one after another, any number of randomly generated MOLPs. Upon solution, outputted for each MOLP are the quantities

Ex	number of efficient extreme points of the MOLP
CPU	CPU time taken to compute the efficient extreme points

along with other technical information that may or may not be useful on a given project.

3. Experimental runs

Of the twelve parameters required to configure the ADBASE random problem generator, only the following five are considered *problem-size* parameters for predicting the number of efficient extreme points possessed by an MOLP.

NOBJS	number of objectives
JLC	lower bound of the **C**-matrix interval [JLC,JUC]
NCONS	number of constraints
NVARS	number of structural variables
ADEN	percent **A**-matrix nonzero density

NOBJS is a problem-size parameter because of the effect, often dramatic, that NOBJS usually has on the size and dimensionality of the criterion cone (where the criterion cone is the cone generated by the $\mathbf{c}^i$ gradients of the k objectives). Holding upper bound JUC fixed enables JLC to be a problem-size parameter because JLC can then be used to control the directions of the objective function gradients. NCONS and NVARS are problem-size parameters because they affect the number of extreme points of the feasible region S. Largely passed over in previous research, ADEN is included as a candidate problem-size parameter because of the possibility raised in Evans and Steuer (1973) that the **A**-matrix nonzero density might have a mild effect on the number of efficient extreme points. Of course, it is most likely that high **A**-matrix nonzero densities would only be seen in problems with smaller numbers of constraints.

In the early stages of this study, it was thought that CDEN (**C**-matrix nonzero density) might also be a problem-size parameter, but preliminary runs could not confirm any significance of its candidacy. Consequently, that conjecture has been dropped, at least in this paper, and we proceed with only the five problem-size parameters listed above.

In the experimental runs of this paper, the five problem-size parameters were varied over the values given in Table 1 to provide sufficient Ex and CPU data needed for this study. The other seven random problem generator configuration parameters, whose purpose is mainly to enhance the realism of the randomly generated MOLPs, were held fixed at the values given in Table 2.

The variable values of JLC from Table 1 and the fixed value of JUC from Table 2 are used to form the four [JLC, JUC] intervals of $[-20, 20]$, $[-10, 20]$, $[0, 20]$ and $[10, 20]$ from which the nonzero **C**-matrix coefficients in the different randomly generated MOLPs are selected. Since $[-20, 20]$ is the widest interval, it will produce on average the largest criterion cones because its generators as defined by the rows of the **C**-matrix should be the most radially dispersed. By the same token, since $[10, 20]$ is the narrowest interval, it will produce on average the smallest criterion cones. In this way, four different *categorical* sizes of criterion cones are generated for testing.

Table 1
Problem-Size parameters that are varied

Five random problem generator configuration parameters considered problem-size parameters	Values employed
NOBJS	2, 3, 4, 5, 6, 7, 8
JLC	−20, −10, 0, 10
NCONS	10, 20, 30, 40, 50, 60
NVARS	25, 50, 75, 100, 125
ADEN	20, 40, 60, 80, 100

Table 2
Random problem generator configuration parameters held fixed

Seven random problem generator configuration parameters not considered problem-size parameters	Values at which held fixed
JUC	20
CDEN	100
JLA	−20
JUA	20
JLB	50
JUB	100
IDISP	20

Utilizing all of the different values in Table 1, we have a total of $7\times4\times6\times5\times5 = 4200$ possible MOLP problem sizes. However, with a maximum of 120,000 efficient extreme points (imposed by the software) and a maximum of six minutes of equivalent run time per problem, [1] it was only possible to run 2557 of the 4200 problems sizes to completion. For instance, while it is typically possible to run both of the following two MOLPs to completion

$$\text{NOBJS} = 7, \quad \text{JLC} = 10, \quad \text{NCONS} = 40,$$
$$\text{NVARS} = 75, \quad \text{ADEN} = 100$$

$$\text{NOBJS} = 4, \quad \text{JLC} = -20, \quad \text{NCONS} = 60,$$
$$\text{NVARS} = 75, \quad \text{ADEN} = 100,$$

it is not typically possible to run either of the following two MOLPs to completion

$$\text{NOBJS} = 7, \quad \text{JLC} = -20, \quad \text{NCONS} = 20,$$
$$\text{NVARS} = 60, \quad \text{ADEN} = 100,$$

$$\text{NOBJS} = 4, \quad \text{JLC} = 0, \quad \text{NCONS} = 60,$$
$$\text{NVARS} = 125, \quad \text{ADEN} = 100.$$

The reason is that if more than one of the problem-size parameters NOBJS, JLC, NCONS or NVARS is at or near its value that puts the most pressure on the number of efficient extreme points (8 for NOBJS; −20 for JLC; 60 for NCONS; 125 for NVARS), the MOLP will probably have more than 120,000 efficient extreme points and won't complete. This is due to interaction effects among the problem-size parameters, more about which will be discussed shortly. Performing two replications per feasible problem size, 5114 valid observations containing Ex and CPU data were obtained.

---

[1] For understandability, the times reported in this paper are in 1.7 GHz Pentium 4 seconds, but in order to take advantage of the ability to run in parallel, the experiments were actually run on the University of Georgia RCR System (SGI Origin 2000 running 300 MHz R12000 processors).

4. Initial regressions

In an initial set of regressions employing the 5114 observations, the number of efficient extreme points Ex and CPU time were regressed onto NOBJS, JLC, NCONS, NVARS and ADEN. The results, with R^2_{adj} values of 14.6% and 20.7%, are shown in Figs. 1 and 2, respectively.

With poor R^2_{adj} values, unacceptable normal probability plots (not shown), and severe heteroscedasticity exhibited among the variances of the residuals (not shown), the models were disappointing. In addition, the high p-value for NVARS in Fig. 1 tends to suggest that this variable may not be needed in the model. This is clearly counter to our intuition and is not taken seriously because of the extent to which the normal probability plots and heteroscedasticity situation are in serious violation of the basic assumptions of regression analysis.

Continuing the search for models that are more in line with the assumptions, we investigated various transformations of the independent variables without obtaining improvements in the R^2_{adj}. We then turned our attention to transformations of the dependent variable. Taking the natural logarithm of Ex to form the new response variable LnEx, R^2_{adj} of the initial model jumped from 14.6% to 61.0%. Similarly, taking the natural logarithm CPU to form LnCPU, R^2_{adj} jumped from 20.7% to 58.6%. Moreover, major improvements are seen in both the normal probability plots and heteroscedasticity situation.

5. ACONEA

In connection with the fact that the size of the *criterion cone* (convex cone generated by the $\mathbf{c}^i$ rows of the $k \times n$ criterion matrix **C**) has a major effect on the number of efficient extreme points, a problem is now recognized with regard to JLC. While a necessary parameter for configuring a random problem generator, JLC does not work well with an already existing MOLP because it would not be known how to ascertain the value of JLC to go with the problem.

In situations when JLC might not be appropriate, an alternative, and potentially more accurate, indicator of criterion cone size called *average cone angle* ACONEA is introduced. ACONEA is a function of the angles among the $\mathbf{c}^i$ gradients of the objectives. ACONEA is computed by summing

```
The regression equation is
Ex = - 8797 + 1695 NOBJS - 225 JLC + 95.8 NCONS
   + 1.41 NVARS + 15.9 ADEN

Predictor      Coef    Stdev     t-ratio       p
Constant    -8797.2    617.1      -14.26   0.000
NOBJS       1694.64    71.24       23.79   0.000
JLC         -224.95    10.84      -20.76   0.000
NCONS        95.782    7.669       12.49   0.000
NVARS         1.410    3.579        0.39   0.694
ADEN         15.941    4.145        3.85   0.000

s = 8396     R-sq = 14.7%     R-sq(adj) = 14.6%

Analysis of Variance

SOURCE        DF           SS            MS       F       p
Regression     5  61913206784   12382641152 175.66  0.000
Error       5107  3.60009E+11      70493200
Total       5112  4.21922E+11
```

Fig. 1. Initial Ex regression results.

```
The regression equation is
CPU = - 22.4 + 3.51 NOBJS - 0.379 JLC + 0.177 NCONS
    + 0.0815 NVARS + 0.0266 ADEN

Predictor    Coef       Stdev      t-ratio        p
Constant -22.4175    0.9548       -23.48      0.000
NOBJS      3.5104    0.1102        31.85      0.000
JLC      -0.37930   0.01677       -22.62      0.000
NCONS     0.17729   0.01187        14.94      0.000
NVARS    0.081452  0.005538        14.71      0.000
ADEN     0.026553  0.006413         4.14      0.000

s = 12.99     R-sq = 20.8%       R-sq(adj) = 20.7%

Analysis of Variance

SOURCE       DF        SS      MS        F      p
Regression    5    225667   45133   267.44  0.000
Error      5107    861870     169
Total      5112   1087537
```

Fig. 2. Initial CPU regression results.

the angles between all $\binom{k}{2}$ pairs of the criterion cone's $\mathbf{c}^i$ generators, and then dividing by $\binom{k}{2}$.

For example, with

$$\mathbf{C} = \begin{bmatrix} 1 & -1 & 0 & -1 & 0 & 0 \\ 9 & 0 & -4 & 0 & 5 & 0 \\ 4 & 0 & -1 & 0 & 5 & 0 \\ 0 & 3 & 7 & 0 & 0 & -1 \\ 0 & 6 & 0 & 7 & 0 & 1 \end{bmatrix}$$

the angles in degrees between all pairs of $\mathbf{c}^i$ generators of the criterion cone are given in the lower triangular portion of the 5×5 array below

$$\begin{bmatrix} 61.94 & & & & \\ 69.12 & 24.76 & & & \\ 103.03 & 109.27 & 98.08 & & \\ 144.03 & 90.00 & 90.00 & 76.19 & \end{bmatrix}$$

To illustrate, the $(4, 1)$ element in the above states that the angle between $\mathbf{c}^1$ and $\mathbf{c}^4$ is 103.03°. Summing all $\binom{5}{2} = 10$ angles and dividing by 10, we have

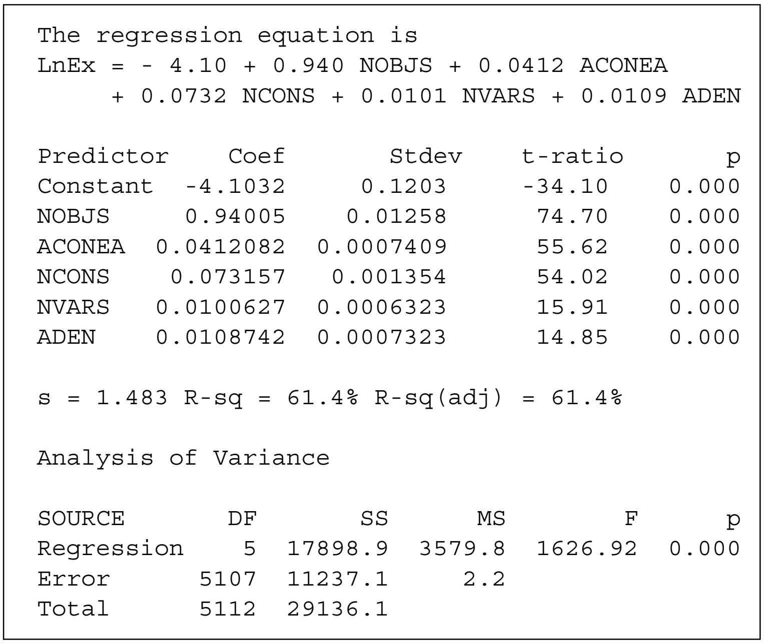
$$\text{ACONEA} = \frac{866.42}{10} = 86.64^\circ$$

While not claiming to be perfect, ACONEA is intuitive and easy to apply, the idea being that larger criterion cones will correlate with larger ACONEA values, which in turn will correlate with larger numbers of efficient extreme points and CPU times, all else held constant.

Substituting ACONEA for JLC in the regression model with the logarithmically transformed dependent variable LnEx, R^2_{adj} improves from 61.0% to 61.4% as shown in Fig. 3. Similarly, substituting ACONEA into the regression model with LnCPU, R^2_{adj} improves from 58.6% to 67.7% as shown in Fig. 4. Although there is only a slight improvement in the model for the number of efficient extreme points, the effect is more noticeable with CPU time, and given ACONEA's practical side, we will take advantage of the substitution from here forward.

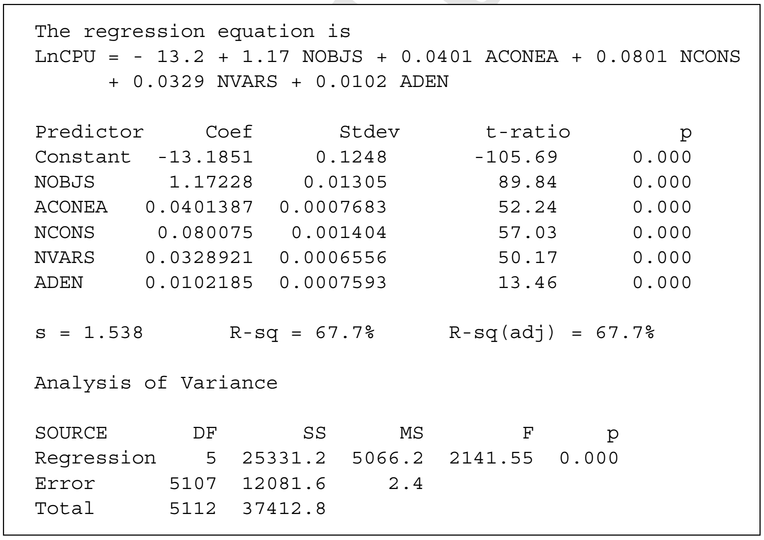
6. Model building

In this section, we build upon the logarithmic transformations of the dependent variables to include more terms in the pool of independent



```
The regression equation is
LnEx = - 4.10 + 0.940 NOBJS + 0.0412 ACONEA
     + 0.0732 NCONS + 0.0101 NVARS + 0.0109 ADEN

Predictor    Coef       Stdev    t-ratio      p
Constant  -4.1032      0.1203     -34.10    0.000
NOBJS      0.94005    0.01258      74.70    0.000
ACONEA  0.0412082  0.0007409      55.62    0.000
NCONS    0.073157   0.001354      54.02    0.000
NVARS   0.0100627  0.0006323      15.91    0.000
ADEN    0.0108742  0.0007323      14.85    0.000

s = 1.483 R-sq = 61.4% R-sq(adj) = 61.4%

Analysis of Variance

SOURCE       DF       SS      MS        F      p
Regression    5  17898.9  3579.8  1626.92  0.000
Error      5107  11237.1     2.2
Total      5112  29136.1
```

Fig. 3. Results with logarithmic transformation LnEx and ACONEA.



```
The regression equation is
LnCPU = - 13.2 + 1.17 NOBJS + 0.0401 ACONEA + 0.0801 NCONS
      + 0.0329 NVARS + 0.0102 ADEN

Predictor     Coef       Stdev       t-ratio        p
Constant  -13.1851      0.1248       -105.69      0.000
NOBJS      1.17228     0.01305         89.84      0.000
ACONEA   0.0401387   0.0007683         52.24      0.000
NCONS     0.080075    0.001404         57.03      0.000
NVARS    0.0328921   0.0006556         50.17      0.000
ADEN     0.0102185   0.0007593         13.46      0.000

s = 1.538       R-sq = 67.7%      R-sq(adj) = 67.7%

Analysis of Variance

SOURCE       DF       SS      MS        F      p
Regression    5  25331.2  5066.2  2141.55  0.000
Error      5107  12081.6     2.4
Total      5112  37412.8
```

Fig. 4. Results with logarithmic transformation LnCPU and ACONEA.

variables. Because of the likelihood of interaction effects, all squared and two-variable cross-product terms of NOBJS, ACONEA, NCONS, NVARS and ADEN are added to the pool. This increases the size of the pool of independent variable terms to 20 (five first-order, five squared, and ten cross-product). To identify which terms produce the best models, we pursued a Best-Subsets approach that employed the Hamiltonian Walk technique within Minitab (1995). In this technique, all $2^q - 1$ possible subsets, where q is the number of terms in the pool of independent variables, are evaluated in at most $2^q - 1$ steps. This is accomplished because in the Hamiltonian Walk procedure the subsets are ordered in such a way that each successor subset is derived from its predecessor by the addition or deletion of only one variable. In our case, with $q = 20$ terms in the pool of independent variables, there are $2^q - 1 = 1,048,575$ possible subsets.

With regard to the number of efficient extreme points, we have Fig. 5 which reports the five models of highest R^2_{adj} value for each individual number of independent variable terms. Selecting from the “nondominated frontier” the model of highest R^2_{adj} with eight independent variable terms, we have the model of Fig. 6.

$$\begin{aligned}\ln\text{Ex} = {} & -4.45 + 1.27k + 0.135m + 0.0116d \\ & - 0.148k^2 - 0.00162m^2 + 0.0139km \\ & + 0.0036kn + 0.0112ka\end{aligned}$$

where a is ACONEA and d is ADEN, as our best recommended expression for predicting the number of efficient extreme points possessed by an MOLP. For this model, $R^2_{\text{adj}} = 81.4\%$ which is a nice improvement over the 61.4% of the model in Fig. 3, and in a normal probability plots analysis (not shown), $r = 0.994$.

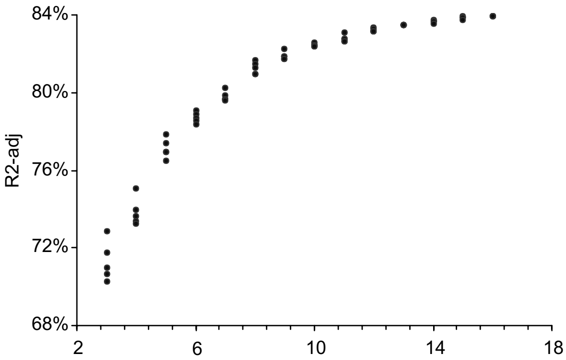


Fig. 5. R^2_{adj} versus number of independent variable terms.

```
The regression equation is
LnEx = - 4.45 + 1.27 NOBJS + 0.135 NCONS - 0.148 NOBJS2
     - 0.00162 NCONS2 + 0.0139 NOBJS*NCONS
     + 0.00360 NOBJS*NVARS + 0.0112 NOBJS*AC
     + 0.0116 ADEN

Predictor         Coef       Stdev      t-ratio        p
Constant       -4.4470      0.1472       -30.22    0.000
NOBJS          1.26804     0.05383        23.55    0.000
NCONS         0.135183    0.005234        25.83    0.000
NOBJS2       -0.148125    0.004749       -31.19    0.000
NCONS2     -0.00162442  0.00006223       -26.11    0.000
NOBJS*NCONS  0.0139399   0.0005950        23.43    0.000
NOBJS*NVARS  0.0036040   0.0001042        34.59    0.000
NOBJS*AC     0.0111759   0.0001233        90.64    0.000
ADEN         0.0115562   0.0005091        22.70    0.000

s = 1.031         R-sq = 81.4%       R-sq(adj) = 81.4%

Analysis of Variance

SOURCE        DF        SS       MS         F       p
Regression     8   23715.2   2964.4   2791.10   0.000
Error       5104    5420.9      1.1
Total       5112   29136.1
```

Fig. 6. Best recommended model for number of efficient extreme points.

An additional data set of 2030 observations was generated in order to validate the model in accordance with the procedure described in Neter et al. (1990). When fitting the chosen model to the validation set the R^2_{adj} value was found to be 82% with coefficients close to those of the model. A mean squared prediction error (MSPR) was calculated to be 1.217 which is very close to the $\text{MSE} = 1.2$ of the regression fit. A MSPR that is near the value of the MSE for the model building set means that the model is not seriously biased and provides valid predictions.

Similarly, with regard to CPU time and the same eight independent variable terms, we have the model of Fig. 7

$$\begin{aligned}\ln\text{CPU} = {} & -12.2 + 1.10k + 0.160m + 0.011d \\ & - 0.136k^2 - 0.00176m^2 + 0.012km \\ & + 0.0087kn + 0.0114ka\end{aligned}$$

as our best recommended expression for predicting CPU time. For this model it is seen that R^2_{adj} nicely improves from 67.7% to 83.8%.

7. Prediction and conclusions

An important use of regression analysis is the ability to predict values (responses) of a dependent variable. Predictions can be in the form of a point or interval estimate. A point estimate provides one number as the predicted value of a quantity as a consequence of a given combination of values for the independent variable terms. A confidence interval goes a step further by taking estimated variation into account. A confidence interval (CI) for a mean response provides two numbers between which a mean response is expected. A prediction interval (PI) is similar to a confidence interval but provides two numbers between which a new observation of the response is estimated. While centered at the same value as the confidence interval, a prediction interval for a new observation is typically wider. Equations for calculating point estimates, confidence intervals, and prediction intervals are standard and discussions of them can be found in most any applied statistics text (for instance Neter et al., 1990).

```
The regression equation is
LnCPU = - 12.2 + 1.10 NOBJS + 0.160 NCONS + 0.0110 ADEN
      - 0.136 NOBJS2 - 0.00176 NCONS2 + 0.0120 NOBJS*NCONS
      + 0.00870 NOBJS*NVARS + 0.0114 NOBJS*ACONEA

Predictor          Coef         Stdev      t-ratio         p
Constant       -12.2003        0.1555       -78.46     0.000
NOBJS           1.10141       0.05688        19.36     0.000
NCONS          0.159684      0.005531        28.87     0.000
ADEN          0.0110175     0.0005379        20.48     0.000
NOBJS2        -0.135915      0.005018       -27.08     0.000
NCONS2      -0.00175951    0.00006575       -26.76     0.000
NOBJS*NCONS   0.0119592     0.0006287        19.02     0.000
NOBJS*NVARS   0.0087003     0.0001101        79.02     0.000
NOBJS*ACONEA  0.0114109     0.0001303        87.58     0.000

s = 1.089      R-sq = 83.8%             R-sq(adj) = 83.8%

Analysis of Variance

SOURCE       DF       SS      MS         F       p
Regression    8  31360.0  3920.0   3305.56   0.000
Error      5104   6052.7     1.2
Total      5112  37412.8
```

Fig. 7. Best recommended model for predicting CPU time.

Recall that the response variables of interest, Ex and CPU, have been transformed using the natural logarithm to obtain the response variables LnEx and LnCPU that have been used in the best recommended regression models. Therefore, to make inferences about the values of Ex or CPU for various combinations of the problem-size parameters, we must first make inferences for LnEx and LnCPU and then convert these predictions with a reverse transformation.

Selected 90% confidence intervals for the mean number, and 90% prediction intervals for new observations, of efficient extreme points are presented in Table 3. Corresponding intervals for the CPU time are presented in Table 4. CPU times are reported using 1.7 GHz Pentium 4 seconds. For both tables, the regression equations contained the exact same problem-size parameters with ADEN being held constant at 100%.

From the preceding analysis and tables we can observe several interesting things:

(a) The point estimates for number of efficient extreme points in Table 3 vary from 10 to 92,420.
(b) The upper limits for 90% prediction intervals for the number of efficient extreme points in Table 3 vary from 55 to 505,347. This indicates that the set of efficient extreme points can grow quite large.

Table 3
Confidence and prediction intervals for number of efficient extreme points

Size	ACONEA	Point estimate	90% CI mean	90% PI new Obs
2×10×25	90	10	(9, 11)	(2, 55)
2×60×125	90	245	(225, 267)	(45, 1338)
3×10×25	90	59	(55, 63)	(11, 321)
3×60×125	90	4108	(3764, 4484)	(752, 22447)
4×10×25	90	255	(237, 275)	(47, 1393)
4×40×100	90	20217	(18794, 21749)	(3703, 110382)
5×10×25	90	822	(760, 890)	(151, 4489)
5×50×25	90	6379	(55304, 65920)	(11051, 329885)
6×10×25	90	1970	(1808, 2145)	(361, 10761)
6×50×25	75	92420	(83458, 102355)	(16902, 505347)
7×10×25	90	3510	(3184, 3870)	(642, 19187)
7×30×25	75	31121	(28424, 34074)	(5695, 170059)
8×10×25	90	4651	(4110, 5262)	(850, 25469)
8×50×125	15	39817	(32699, 48484)	(7221, 219542)

Table 4
Confidence and prediction intervals for CPU time

Size	ACONEA	Point estimate	90% CI mean	90% PI new Obs
2×10×25	90	0.01	(0.009, 0.011)	(0.001, 0.129)
2×60×125	90	15.03	(13.272, 17.014)	(1.192, 189.394)
3×10×25	90	0.03	(0.030, 0.037)	(0.003, 0.416)
3×60×125	90	48.53	(42.735, 55.104)	(3.850, 611.693)
4×10×25	90	0.11	(0.096, 0.118)	(0.008, 1.342)
4×40×100	90	13.88	(12.622, 15.268)	(1.103, 174.745)
5×10×25	90	.34	(0.312, 0.380)	(0.027, 4.334)
5×50×25	90	8.47	(7.614, 9.426)	(0.673, 106.680)
6×10×25	90	1.11	(1.003, 1.232)	(0.088, 13.998)
6×50×25	75	14.98	(13.464, 16.674)	(1.190, 188.676)
7×10×25	90	3.59	(3.217, 4.007)	(0.285, 45.220)
7×30×25	75	9.75	(8.819, 10.788)	(0.775, 122.809)
8×10×25	90	11.59	(10.275, 13.083)	(0.920, 146.106)
8×50×125	15	377	(324.4, 438.2)	(28.97, 4759)

(c) The predictions for number of efficient extreme points tend to increase with the number of objectives. This is evident by looking at the first entry for each number of objectives in Table 3. For these entries, the number of objectives is varied while all other values are held constant.
(d) The predictions for the number of efficient extreme points tend to increase dramatically with increases in the numbers of constraints and variables. This can be seen by comparing the pair of entries for each number of objectives value in Table 3.
(e) The point estimates for CPU time in Table 4 vary from 1 to 377.03 seconds.
(f) The upper limits for 90% prediction intervals for the CPU time in Table 4 vary from 0.129 to 4,758.7 seconds. This indicates that the computer time required for enumerating the set of efficient extreme points can be extensive.
(g) As the point estimates for number of efficient extreme points in Table 3 and CPU time in Table 4 increase so does the confidence and prediction interval widths. This is a result of the increase in variance detected for MOLPs with high-valued configuration parameters.

In this paper, we have experimented with MOLPs possessing, to our knowledge, larger numbers of efficient extreme points than have ever been reported on before. In this endeavor our purpose has been to develop "best recommended models" to identify the problem-size parameters and their interactions that characterize efficient extreme point behavior as we enter the world of larger and larger MOLPs. Whereas up until recently larger MOLPs were either out of reach or only solvable on large machines, the advances in small computers that we have all witnessed now enable larger MOLPs, as indicated by the times in Table 4, to be solvable on PCs in reasonable time.

References

Arbel, A., 1993. Exploring Interior-Point Linear Programming: Algorithms and Software, Foundations of Computing Series. MIT Press, Cambridge, Massachusetts.

Armand, P., Malivert, C., 1991. Determination of the efficient set in multiobjective linear programming. Journal of Optimization Theory and Applications 70 (3), 467–490.

Benson, H.P., Sayin, S., 1997. Towards finding global representations of the efficient set in multiple objective mathematical programming. Naval Research Logistics 44 (1), 47–67.

Climaco, J., Antunes, C., 1989. Implementation of a user-friendly software package—a guided tour of TRIMAP. Mathematical and Computer Modelling 12 (10–11), 1299–1309.

Dauer, J.P., 1991. Optimization over the efficient set using an active constraint approach. Zeitschrift für Operations Research 35 (3), 185–195.

Ecker, J.G., Kouada, I.A., 1978. Finding all efficient extreme points for multiple objective linear programs. Mathematical Programming 14 (2), 249–261.

Evans, J.P., Steuer, R.E., 1973. A revised simplex method for linear multiple objective programs. Mathematical Programming 5 (1), 54–72.

Gal, T., 1977. A general method for determining the set of all efficient solutions to a linear vectormaximum problem. European Journal of Operational Research 1 (5), 307–322.

Gardiner, L.R., Steuer, R.E., 1994. Unified interactive multiple objective programming. European Journal of Operational Research 74 (3), 391–406.

Habenicht, W., 1982. Quad trees: A datastructure for discrete vector optimization problems. In: Lecture Notes in Economics and Mathematical Systems, vol. 209. Springer-Verlag, Berlin, pp. 136–145.

Isermann, H., 1977. The enumeration of the set of all efficient solutions for a linear multiple objective program. Operational Research Quarterly 28 (3), 711–725.

Korhonen, P., Salo, S., Steuer, R.E., 1997. A heuristic for evaluating nadir criterion values in multiple objective linear programming. Operations Research 45 (5), 751–757.

Mavrotas, G., Diakoulaki, D., Assimacopoulos, D., 1998. Bounding MOLP objective functions: Effect on efficient set size. Journal of the Operational Research Society 49 (5), 549–577.

Minitab, Inc., 1995. MINITAB Release 10.51 Xtra, State College, Pennsylvania.

Neter, J., Wasserman, W., Kutner, M.H., 1990. Applied Linear Statistical Models, third ed., Richard D. Irwin, Homewood, Illinois.

Reeves, G., Reid, C., 1988. Minimum values over the efficient set in multiple objective decision making. European Journal of Operational Research 36 (3), 334–338.

Sayin, S., 2000. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. Mathematical Programming 87 (3), 543–560.

Strijbosch, L.W.G., van Doorne, A.G.M., Selen, W.J., 1991. A simplified MOLP algorithm: The MOLP-S procedure. Computers and Operations Research 18 (8), 709–716.

Steuer, R.E., 1994. Random problem generation and the computation of efficient extreme points in multiple objective linear programming. Computational Optimization and Applications 3 (4), 333–347.

Steuer, R.E., 2000. Manual for the ADBASE multiple objective linear programming package. Terry College of Business, University of Georgia, Athens, Georgia USA.

Sun, M., Steuer, R.E., 1996. Quad-Trees and linear lists for identifying nondominated criterion vectors. Journal on Computing 8 (4), 367–375.

Sun, M., Steuer, R.E., 2000. Quad-tree data structures for use in large-scale discrete alternative multiple criteria problems. In: Shi, Y., Zeleny, M. (Eds.), New Frontiers of Decision Making for the Information Technology Era. World Scientific Publishing, Singapore, pp. 48–71.

Wiecek, M., Zhang, H., 1997. A parallel algorithm for multiple objective linear programs. Computational Optimization and Applications 8 (1), 41–56.

Yu, P.L., Zeleny, M., 1975. The set of all non-dominated solutions in linear cases and a multicriteria simplex method. Journal of Mathematical Analysis and Applications 49 (2), 430–468.

Zionts, S., Wallenius, J., 1980. Identifying efficient vectors: Some theory and computational results. Operations Research 28 (3), 785–793.