Scientific Journal of Mechanical and Industrial Engineering

***Sci. J. Mech. Ind. Eng.*, 2(3): 57-60 (2013)**

Differential Evolution Algorithm for Measuring Efficiency in DEA

Rahimian M[1], Mahmoodirad A[1], Molla-Alizadeh-Zavardehi S[2*]

[1]*Department of Mathematics, Masjed Soleiman Branch, Islamic Azad University, Masjed Soleiman, Iran*
[2]*Department of Industrial Engineering, Masjed Soleiman Branch, Islamic Azad University, Masjed Soleiman, Iran*

*Corresponding author's E-mail: saber.alizadeh@gmail.com

Abstract

Data Envelopment Analysis (DEA) is one of the successful evaluation methods for measuring the relative efficiency of Decision Making Units (DMUs) that utilizes techniques of mathematical programming. In DEA models, for measuring the relative efficiency of DMUs, for a large dataset with many inputs/outputs would need to have a long time with a huge computer. This paper proposed and developed the Differential evolution (DE) for DEA. DE requirements for computer memory and CPU time are far less than that needed by conventional DEA methods and can therefore be a useful tool in measuring the efficiency of large datasets. Since the operators have important roles on the fitness of the algorithms, all the operators and parameters are calibrated by means of the Taguchi experimental design in order to improve their performances.

Original Article

Received 10 Jun. 2013
Accepted 15 Sep. 2013

Keywords:
Data Envelopment analysis,
Differential evolution,
Taguchi experimental design.

INTRODUCTION

Data envelopment analysis (DEA), first introduced by Charnes et al. [2], is a useful method, to measuring the relative efficiency of Decision Making Units (DMUs) with multiple inputs and multiple outputs based on data oriented. One of the main objectives of DEA is to measure the efficiency score of a DMU. One of the ways for determining efficiency score of DMUs is to apply the Charnes, Cooper and Rhodes' model (CCR model) that deals with a ratio of multiple outputs and inputs.

DEA for a large dataset with many input/output variables and/or DMUs would require huge computer resources in terms of memory and CPU time and take a long time even though with a very fast computer [4]. Furthermore, in order to obtain the results, it must be solved as a separated mathematical programming problem for each DMU.

The related works of this area are as follows:

Udhayakumar et al. [6] developed a GA that employs one-point crossover and perturbation mutation operators for solving the P-model of chance constrained technique. They considered DEA problem for the banking sector in which inputs and outputs are assumed to be stochastic. In their method, the stochastic objective function and chance constraints are used and the feasibility of chance constraints is verified by simulation techniques.

Azadeh et al. [1] presented a hybrid GA-DEA for assessment and optimization of critical inputs from two different viewpoints of efficiency and cost in electricity transmission units. They used a specific measure and cost allocation super-efficiency DEA models for sensitivity analysis and to determine the critical inputs based on efficiency and cost.

In this paper, to estimate the efficiency of DMUs in large datasets, we proposed and developed the DE. This paper is organized as follows: Section 2 briefly describes

To cite this paper: Rahimian M., Mahmoodirad A., Molla-Alizadeh-Zavardehi S. 2013. Differential Evolution Algorithm for Measuring Efficiency in DEA. *Sci. J. Mech. Ind. Eng.*, 2 (3): 57-60.
Journal homepages: http://sjmie.science-line.com/

the DEA technique. The proposed DE for estimating the efficiency is explained in sections 3. The experimental design and comparisons are presented in Section 4. Finally, in section 5, conclusion is provided.

Preliminaries

Let we have *n* observations on *n* DMUs $\{DMU_j : j = 1,2,\dots,n\}$, with input and output vectors(x_j, y_j), $x_j = (x_{1j}, x_{2j}, \dots, x_{mj})^T > 0$ and $y_j = (y_{1j}, y_{2j}, \dots, y_{sj})^T > 0$ for *j*=1, 2,…,*n*. In the TDT (Thompson, Dharmapalaand Thrall) model the relative efficiency score of DMU_o is obtained by solving the following mathematical programming model:

$$Max_{(u,v)} \frac{\frac{u^T y_o}{v^T x_o}}{Max_{1\le j\le n}\left\{\frac{u^T y_j}{v^T x_j}\right\}} \quad (1)$$

s.t $u \ge 0,\ v \ge 0.$

Where $u \in R^{m\times 1}$ and $v \in R^{s\times 1}$ are the column vectors of input and output weights, respectively. If we suppose$Max_{1\le j\le n}\left\{\frac{u^T y_j}{v^T x_j}\right\} = \frac{1}{t}$, and use the Charnes and Cooper's linear transformation technique [3], so, we obtain the following linear programming problem:

$$\begin{aligned} &Max \quad \theta = u^T y_o \\ \text{s.t} \quad &v^T x_o = 1, \\ &u^T y_j - v^T x_j \le 0, \quad j = 1,2,\dots,n \\ &u \ge 0, \quad v \ge 0, \end{aligned} \quad (2)$$

Where is the CCR model to obtain the relative efficiency score of DMU_o.The model (2) is called the multiplier form of the CCR model.

Definition 1.(Cooper, et al. 2006).

1. $DMU_o$is efficient if the optimal objective function value of model (2),θ^*, turns out to be one, i.e.$\theta^* = 1$.
2. $DMU_o$is inefficient, if $\theta^* < 1$, where θ^* is the optimal objective function value of model (2).

In the evaluation of large organization (about millions) by using DEA, even if we employ a high-speed computer, many calculations are needed. Also, it may take a long time to estimate the efficiency of DMUs in these kinds of applications, and because of estimating the efficiency; a linear program must be solved for each DMU. In other words, the conventional DEA models aren't able to be solved via this number of DMUs.To get rid of this problem (relative efficiency of each number of DMUs), we proposed DE which be detailed in the following section.

The proposed differential evolution

Differential Evolution (DE) is a very simple population-based global optimization algorithm. This algorithm created by Price and Storn [7], whose main objective is functions optimization. It is one strategy based on evolutionary algorithms with some specific characteristics. The DE algorithm's main strategy is to generate new individuals by calculating vector differences between other randomly-selected individuals of the population.

DE starts with a number of populations of NP candidate solutions, so-called individuals. The DE's main strategy is to generate new individuals by calculating vector differences between other randomly selected individuals of the population. The subsequent generations in DE are denoted by $G = 0,1,\dots,G_{Max}$. It is usual to denote each individual as a D-dimensional vector $X_{i,G} = X^1_{i,G},\dots,X^D_{i,G}, i = 1,2,\dots,NP$ called a target vector. The key idea behind DE is a scheme for generating trial vectors. The main operation is founded on the differences of randomly sampled pairs of solutions in the population. The main difference between traditional evolutionary algorithms and DE is that in traditional evolutionary algorithms, mutation results in small perturbations to the genes of an individual, while in DE, the mutation is an arithmetic combination of individuals. This algorithm uses four important parameters: population size, mutation, crossover and selection operators; there are different variants.

Initial population:

Like other evolutionary algorithms, DE works with a population of individuals (candidate solutions) and this number never changes during the optimization process. Normally the initial population is randomly generated and the population will be improved by the algorithm iteratively, through the mutation, crossover and selection operators.

Mutation operator:

According to the DE, after initialization, it employs the mutation operator. The mutation in DE is a distinct innovation. It is based on the difference of different individuals (Solutions), to produce a mutant vector $V_{i,G}$ with respect to each individual $X_{i,G}$, in the current population. This main operation is founded on the differences of randomly sampled pairs of solutions in the population. For each target vector $X_{i,G}, i = 1, 2,\dots, NP$, a mutant vector $V_{i,G}$ can be made by the following mutation operators. In all types, the scale factor *F* is a positive control parameter for scaling the difference

To cite this paper: Rahimian M., Mahmoodirad A., Molla-Alizadeh-Zavardehi S. 2013. Differential Evolution Algorithm for Measuring Efficiency in DEA. *Sci. J. Mech. Ind. Eng.*, 2 (3): 57-60.
Journal homepages: http://sjmie.science-line.com/

vector. The following mutation operator proposed by Storn and Price [7]:

$$V_{i,G} = X_{r_1,G} + F(X_{r_2,G} - X_{r_3,G})$$

Crossover operator:

In order to increase the diversity of the perturbed parameter vectors, crossover is introduced after the mutation operation. Crossover operation is employed to generate a temporary or trial vector by replacing certain parameters of the target vector by the corresponding parameters of a randomly generated donor vector. To get each individual's trial vector, $U_{i,G+1}$, crossover operation is performed between each individual and its corresponding mutant vector. The following crossover operator proposed by Storn and Price [7]:

$$U_{i,j,G+1} = \begin{cases} V_{i,j,G+1} & if\ Rand(j) \le CR\ or \quad j = Rand(i) \\ X_{i,j,G+1} & if\ Rand(j) > CR\ or \quad j \ne Rand(i) \end{cases}$$

Where rand(*j*) is the *j*th evaluation of a random number uniformly distributed in the range of [0, 1], and randn(*i*) is a randomly chosen index from the set {1, 2,…, N}. $CR \in [0,1]$ is a crossover constant rate that controls the diversity of the population. The more the value of CR, the less the influence of the parent will be.

Selection operator:

To generate the new individual for the next generation, selection operation is performed between each individual and its corresponding trial vector by the following greedy selection criterion:

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & if\ f(U_{i,G+1}) < f(X_{i,G}), \\ X_{i,G} & otherwise, \end{cases}$$

Where *f* is the objective function, and $X_{i,G+1}$ is the individual of the new population.

EXPERIMENTAL DESIGN

Test problems:

In this subsection Instances generation are conducted to set the parameters and evaluate the performances of DE. First, we generated random problem instances for n = 50, 100, 150, 200, 400, 600, 800 and 1000 DMUs, respectively. After specifying the number of DMUs in a given instance, for each DMU, four problem types A, B, C, and D of inputs and outputs numbers (m, s) were generated from discrete uniform distribution [10, 50]. The problem details are shown in Table 1.

Table 1. Test problems characteristics.

Problem size	*DMUs*	Problem type (*m*, *s*)			
		A	B	C	D
1	50	(4, 4)	(4, 8)	(8, 4)	(8, 8)
2	100	(5, 5)	(5, 10)	(10, 5)	(10, 10)
3	150	(5, 5)	(5, 10)	(10, 5)	(10, 10)
4	200	(10, 10)	(10, 20)	(20, 10)	(20, 20)
5	400	(10, 10)	(10, 20)	(20, 10)	(20, 20)
6	600	(15, 15)	(15, 30)	(30, 15)	(30, 30)
7	800	(15, 15)	(15, 30)	(30, 15)	(30, 30)
8	1000	(20, 20)	(20, 40)	(40, 20)	(40, 40)

Parameter setting:

The performance of the DE is generally sensitive to the parameter tuning which affects the search ability and the convergence quality. Choosing proper parameters is time-consuming and sometimes depends on particular instances.

In the related works, to be economic, several experimental designs have been proposed to decrease the number of experiments. Among several experimental design techniques, the Taguchi experimental design method has been successfully employed for a systematic approach for optimization.

Taguchi has created a transformation of the repetition data to another value which is the measure of variation. The transformation is the signal-to-noise (S/N) ratio which explains why this type of parameter design is called robust design. Here, the term "signal" denotes the desirable value (mean response variable) and "noise" denotes the undesirable value (standard deviation). So the S/N ratio indicates the amount of variation present in the response variable. The aim is to maximize the signal-to-noise ratio. In the Taguchi method, the S/N ratio of the maximization objectives is as such [5]:

$$\text{S/N ratio} = -10\log_{10}\left(\frac{1}{\text{objective function}}\right)^2$$

The S/N ratios are averaged in each level, and its value is plotted against each control factor in Fig 1.

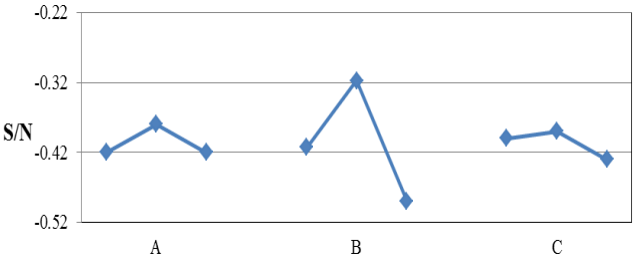

Fig 1. Mean S/N ratio plot for each level of the factors in DE

To cite this paper: Rahimian M., Mahmoodirad A., Molla-Alizadeh-Zavardehi S. 2013. Differential Evolution Algorithm for Measuring Efficiency in DEA. *Sci. J. Mech. Ind. Eng.*, 2 (3): 57-60.
Journal homepages: http://sjmie.science-line.com/

Experimental results:

A computational study was conducted to evaluate the efficiency and effectiveness of the proposed algorithm, which was coded in MATLAB and run on a PC with 2.8 GHz Intel Core 2 Duo and 4 GB of RAM memory. For this purpose, we present and compare the results of DE with the SA algorithm as an effective algorithm in the literature.

We use searching time as stopping criterion to be equal for both algorithms which is equal to 1.5 × (n + m + s) milliseconds. Therefore, CPU time is affected by all the problem characteristic *n*, *m* and *s*. The more the number of DMUs, inputs and outputs, the more the rise of CPU time increases. Each instance is run five times. The performance measure that we will be using is the Relative Percentage Deviation (RPD) is used for each instance:

$$RPD = \frac{Max_{sol} - Alg_{sol}}{Max_{sol}} \times 100$$

Where Alg_{sol} is the obtained objective value for a given instance and Max_{sol} is the maximum or the best known solution for each instance. The problems have been run ten times and the averages of RPDs for each algorithm and each problem size are showed in Fig. 2. From this figures, it is concluded that DE has a better convergence than SA on this problems.

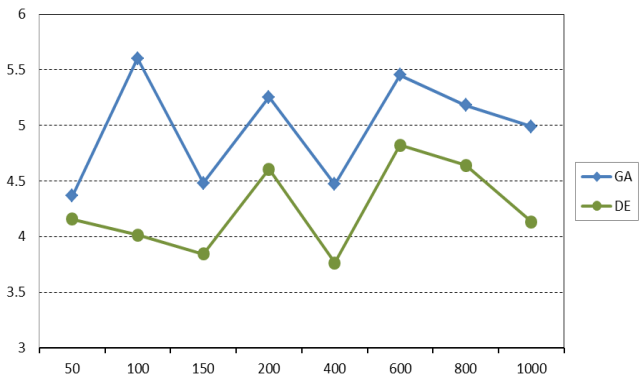

Fig 2. Interaction between DE, SA and problem size

CONCLUSION

We have considered a DEA problem with many input/output and/or many DMUs for obtain the relative efficiency DMUs. Since DEA problems with these structures needs to huge computer in terms of memory and CPU time, so, we have proposed and developed the metaheuristic algorithm, DE, to obtain the relative efficiency of DMUs in large datasets.

Acknowledgements

This study was supported under research project entitled "Using Differential Evolution for measuring efficiency of large scale dataset in DEA" by Islamic Azad University, MasjedSoleiman Branch. The authors are grateful for this financial support.

REFERENCES

1. Azadeh, A., S.M. Asadzadeh, and S. Ahmadi Movaghar, Implementation of data envelopment analysis–genetic algorithm for improved performance assessment of transmission units in power industry. International Journal of Industrial and Systems Engineering, 2011. 8(1): p. 83-103.
2. Charnes, A., W.W. Cooper, and E. Rhodes, Measuring the efficiency of decision making units. European Journal of Operation Research, 1978. 2(6): p. 429–444.
3. Charnes, A., and W.W. Cooper, Programming with linear fractional functionals, Naval Res. Logist. Quart. 1962. 9(3–4): p. 181-185.
4. Emrouznejad, A., and E. Shale, A combined neural network and DEA for measuring efficiency of large scale datasets. Computers & Industrial Engineering, 2009. 56(1): p. 249–254.
5. Molla-Alizadeh-Zavardehi, S., S. Sadi Nezhad, R. Tavakkoli-Moghaddam, and M. Yazdani. Solving a fuzzy fixed charge solid transportation problem by metaheuristics, Mathematical and Computer Modelling, 2013. 57(5–6): p. 1543–1558.
6. Udhayakumar, A., Charles, V., Kumar, M. (2011). Stochastic simulation based genetic algorithm for chance constrained data envelopment analysis problems. Omega, 39(4): p. 387-397.
7. Storn, R. Price, K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 1997. 11 (4): p. 341–359.

To cite this paper: Rahimian M., Mahmoodirad A., Molla-Alizadeh-Zavardehi S. 2013. Differential Evolution Algorithm for Measuring Efficiency in DEA. *Sci. J. Mech. Ind. Eng.*, 2 (3): 57-60.
Journal homepages: http://sjmie.science-line.com/