data

MDPI

Article

Synthetic Data Generation for Data Envelopment Analysis

Andrey V. Lychev

College of Information Technologies and Computer Sciences, National University of Science and Technology "MISIS", 4 Leninsky Ave., Bldg. 1, 119049 Moscow, Russia; lychev@misis.ru; Tel.: +7-(495)-955-0167

Abstract: The paper is devoted to the problem of generating artificial datasets for data envelopment analysis (DEA), which can be used for testing DEA models and methods. In particular, the papers that applied DEA to big data often used synthetic data generation to obtain large-scale datasets because real datasets of large size, available in the public domain, are extremely rare. This paper proposes the algorithm which takes as input some real dataset and complements it by artificial efficient and inefficient units. The generation process extends the efficient part of the frontier by inserting artificial efficient units, keeping the original efficient frontier unchanged. For this purpose, the algorithm uses the assurance region method and consistently relaxes weight restrictions during the iterations. This approach produces synthetic datasets that are closer to real ones, compared to other algorithms that generate data from scratch. The proposed algorithm is applied to a pair of small real-life datasets. As a result, the datasets were expanded to 50K units. Computational experiments show that artificially generated DMUs preserve isotonicity and do not increase the collinearity of the original data as a whole.

Keywords: synthetic data generation; data augmentation; research data; data envelopment analysis; weight restrictions

MSC: 90B30; 90C05

JEL Classification: C61; C67

Citation: Lychev, A.V. Synthetic Data Generation for Data Envelopment Analysis. *Data* **2023**, *8*, 146. https://doi.org/10.3390/data8100146

Academic Editor: Kassim S. Mwitondi

Received: 10 July 2023
Revised: 19 September 2023
Accepted: 21 September 2023
Published: 27 September 2023

1. Introduction

Data envelopment analysis (DEA) is a nonparametric method that is used to measure the relative efficiency of a homogeneous set of decision making units (DMUs) [1]. The DEA approach is widely used in various fields, including manufacturing systems, power industry, governance, finance, supply chain, transportation, etc. [2–14]. It is assumed that each DMU consumes multiple inputs to produce multiple outputs. The estimation of efficiency scores using the DEA model is based on comparing observations with points on the frontier. The term frontier is used because it is based on best practice observations.

Some DEA papers lack testing of the proposed models on large-scale real data. In those papers, the proposed new model is illustrated by a small dataset with several dozen units and three to five variables. Such an illustrative example serves to demonstrate that the proposed method can give correct results. However, as practice shows, this is not enough to draw a conclusion about the performance of the proposed model or algorithm. A small demo example cannot reveal the computational issues that may arise on medium- and large-scale datasets.

For example, in papers by Sueyoshi and Sekitani [15–17] DEA models were proposed for the measurement of returns to scale (RTS) under a simultaneous occurrence of multiple projections and multiple supporting hyperplanes. It was proposed to use the strong complementary slackness conditions (SCSC) of linear programming [18] as constraints in order to find all vertices of a face where RTS is evaluated. The paper [19] shows that the simple example proposed by Sueyoshi and Sekitani failed to detect numerical issues with

their model. Computational experiments conducted on medium-sized datasets helped reveal the instability of the SCSC model.

Another example demonstrates how detailed testing on large-scale data allows one to benchmark DEA methods and draw conclusions about their performance. Paper [20] describes and compares some of the best-known methods for estimating returns to scale. The paper shows that well-established theoretical models may encounter numerical instability even in medium-sized datasets. In this particular example, the numerical instability is caused by the inversion of an ill-conditioned matrix during the simplex solution process. However, there are a large number of causes of algorithmic instabilities that are difficult to detect theoretically but easy to identify through numerical experiments.

One of the attempts to collect open DEA data that could be used for scientific and educational purposes is the Data Envelopment Analysis Dataset Repository [21]. Unfortunately, this DEA repository has not functioned for many years.

There exist several DEA datasets distributed with R packages (e.g., Benchmarking, rDEA, npsf) published on the Comprehensive R Archive Network (CRAN). However, these datasets are intended mainly for educational purposes and contain a small number of DMUs. Other existing open data repositories are focused on, e.g., machine learning data [22,23], or statistical data [24,25]. However, such repositories contain data that are not intended to be applied directly to DEA analysis: the variables of a DEA model are not specified; inputs and outputs may be mixed with environmental variables; datasets may contain binary, categorical, or unstructured data (audio, video, images, etc.). Therefore, the use of these repositories for DEA is very limited.

Well-known repositories for sharing scientific data [26,27] contain quite a few DEA datasets that are used in articles. However, the datasets in such repositories are difficult to find because they do not take into account DEA specifics. For example, there is no possibility to specify search details such as the number of DMUs or variables, time periods, presence of undesirable variables, etc. Moreover, many datasets do not contain descriptions, so it is impossible to recognize the variables and number of DMUs without downloading the dataset.

Largely because of those reasons, many DEA researchers, for testing purposes, use artificially generated data, usually referred to as synthetic data. Synthetic data are increasingly being used for machine learning. There are two reasons for this: a) the lack of high-quality data and b) the need for privacy protection when sensitive data are used. Recent reviews on synthetic data generation using machine learning (ML) algorithms are presented in [28,29]. Recent studies on the application of ML algorithms for the estimation of DEA technologies are given in [30,31].

On the one hand, large real datasets are rarely seen in the DEA literature, while datasets with more than 10K DMUs are extremely rare. One of the largest datasets in DEA is used in [32]; it represents real data from 30,099 power plants described by 6 variables (two inputs, one good output, and three bad outputs). As a consequence, the existing publicly available datasets for DEA are not enough to conduct comprehensive testing.

On the other hand, applications of DEA show that many inefficient units are projected on the inefficient parts of the frontier when efficiency scores are evaluated. However, this fact disagrees with the main concept of the DEA approach because the efficiency score of an inefficient unit has to be measured relative to efficient units. As a consequence, inaccurate efficiency scores may be obtained. This happens because a non-countable (continuous) production possibility set is determined on the basis of a finite number of production units.

One way to improve the frontier is to insert restrictions on the dual multipliers. A number of papers developed the DEA models, which were based on incorporating domination cones into the dual model [33–39]. Podinovski [40] proved the equivalence between weight restrictions and production trade-offs between inputs and outputs. Computational procedures with weight restrictions and production trade-offs and a discussion of their implementation can be found in [41,42].

Farrell was the first to introduce artificial units in the primal space of inputs and outputs in order to ensure the convexity of the piecewise linear isoquants. Allen and

Thanassoulis [43] elaborated further on the idea of focusing on anchor units as points of departure for formulating coordinates of artificial units. The main purpose of this was to improve the envelopment by reducing the number of inefficient units not "properly" enveloped, resulting in projections to the frontier of these units being on a weakly efficient part of the frontier.

Thanassoulis et al. [44] developed further the super-efficiency method for discovering anchor units in the VRS model [45] and proposed a method for extending the frontier with the help of anchor units. However, their method cannot be used for generating large-scale datasets for two reasons. First, the positions of artificial units are specified by a decision-maker. Second, the procedure does not guarantee full envelopment.

To the best of our knowledge, there are no studies on synthetic data generation in DEA that takes some real datasets as input and complements them with artificial efficient units. The present study attempts to address this gap and contributes by proposing the algorithm for synthetic data generation. The proposed algorithm takes a real DEA dataset as an input and complements it with artificial DMUs.

Following the ideas of Thanassoulis et al. [44], we developed the algorithm for synthetic data generation based on the principles:

(P1) Original efficient frontier should not change after adding artificial DMUs, and
(P2) Artificial efficient DMUs should extend the efficient part of the frontier.

Data generation is organized in such a way that artificial efficient units are generated in the borderline region to upsample data neighborhoods. Inefficient units are generated to follow the underlying distribution of the original data.

Based on the proposed algorithm, datasets with a large number of DMUs have been prepared that can be used for testing the numerical stability and computational performance of existing DEA methods.

The structure of the paper is as follows. Section 2 provides a review of existing approaches for generating artificial datasets in DEA and shows some limitations of current approaches. Section 3 introduces the DEA models that are used for data generation. In Section 4, the algorithm of synthetic data generation for DEA is presented and applied to real-life datasets. Section 5 concludes.

2. Literature Review

Many DEA studies [46–48] have noted that generating synthetic datasets for general multi-input/multi-output technologies is a challenging task. Existing methods for generating artificial data are based on the assumption that there is a so-called data generating process (DGP), which relates inputs and outputs, from which the artificial DMUs were generated.

One of the easiest ways to generate data is, obviously, to use a uniform distribution as a DGP. However, this method produced datasets with a very low proportion of efficient DMUs. This is not an issue, but it does not accurately reflect real data. For example, Khezrimotlagh et al. [32] reported that a dataset with 10 inputs and 10 outputs uniformly distributed on the interval [10, 20] has more than 98.8% dominated DMUs on average in the sample with 20,000 DMUs. Moreover, using a uniform distribution as a DGP gives an unrealistic input-output mix because inputs and outputs are generated independently. Therefore, the benchmarks obtained with such datasets may be biased.

Dulá [47,49] used synthetic large-scale datasets to investigate the computational performance and scale limits of DEA models. This work provides a comprehensive computational study involving DEA problems with up to 100K DMUs. Since there were no typical DEA datasets for this purpose, the author simulates DGP using a sphere as an efficient frontier. This method is also implemented in FEAR [50]. FEAR is a software package for efficiency analysis with R, a software environment for statistical computing. FEAR's function `genxy.sphere` generates n observations with m inputs and r outputs uniformly distributed on the part of a unit sphere located in the positive orthant of the output variables and in the negative orthant of the input variables. The center of the sphere is located at point

$(1, \ldots, 1, 0, \ldots, 0)$ with m ones and r zeros; that leads to the spherical frontier lying in the positive orthant. For more information on the simulation method, see Ref. [51].

The described approach is efficient, but it does not allow for generating units of different scales. In real data, there may be DMUs that differ by 5–7 orders of magnitude in some variables. The presence of a significant variation in the values may trigger certain numerical difficulties that can reveal the instability of the algorithm at large-scale problems. Therefore, for a comprehensive computational study, the dataset must contain units at multiple scales; using a sphere as a DGP is not well suited for this task.

Barr and Durchholz [46] were among the first who tested DEA models using large-scale problems. In addition to the real dataset of 8748 US banks, they also used synthetic ones because few large-scale DEA problems were available. The first proposed approach for generating such data is to draw samples from a multivariate distribution. However, as the authors warn, the variables representing inputs and outputs need to be carefully chosen since all outputs should be positively correlated with all the inputs.

The second approach used in [46] employed the Cobb–Douglas production function, which is most widely used in production economics. This function may be written as

$$y = A \prod_{i=1}^{m} x_i^{\alpha_i}, \quad x_i > 0,\ \alpha_i > 0,\ i = 1, \ldots, m, \tag{1}$$

where, y is the single output, x_i, $i = 1, \ldots, m$ are input variables, α_i is an elasticity for input i, and A is usually referred to as total factor productivity. If $\sum_{i=1}^{m} \alpha_i < 1$, then the function displays decreasing returns to scale. For $\sum_{i=1}^{m} \alpha_i = 1$ constant returns to scale exist. If $\sum_{i=1}^{m} \alpha_i > 1$, then the production function is said to be increasing returns.

For a model with a single output, inputs $x_1, \ldots, x_m$ are generated as independent and identically distributed uniform random variables, and output is directly obtained according to (1) given all $\alpha_i = 0.8/m$. For some DMUs, the A value can be multiplied by a random number between 0 and 1 to simulate inefficiency. By controlling A, we can also regulate the proportion of efficient units.

In the case of multiple outputs, Wilson [50] proposes the following approach implemented in FEAR's `genxy` command. Inputs $x_1, \ldots, x_m$ and output y are generated as for the case with a single output. Next, $(r-1)$ uniformly distributed numbers $\varphi_1, \ldots, \varphi_{q-1}$ in the interval $(0, \pi/2)$ are generated. The outputs are determined as follows:

$$\begin{aligned} y_r &= \sqrt{y^2 \Big/ \left(\sum_{j=1}^{r-1} \tan^2(\varphi_j) + 1 \right)}, \\ y_j &= y_r \tan \varphi_j, \quad j = 1, \ldots, r-1. \end{aligned} \tag{2}$$

Put simply, the outputs are randomly distributed on the part of a sphere located within the positive orthant, centered at the origin, and with radius y.

To show the difference between artificially generated data and real data, we used `genxy` function mentioned above and generated 100 DMUs with 5 inputs and 3 outputs. The pairwise scatter plots of the variables and Pearson correlation coefficients are shown in Figure 1. The histograms showing the distribution of each variable are presented along the matrix diagonal. Pearson correlation coefficients are shown in the upper triangle. The red color corresponds to a positive correlation coefficient, and the blue color represents a negative correlation. The lower triangle provides the pairwise scatter plots of the variables, where the solid line is an OLS fit.

To be economically sound, each output in the dataset should be positively correlated with all the inputs. However, the figure shows that there are negative correlation coefficients between inputs and outputs, and for some pairs, the correlation coefficients turn out to be close to zero.

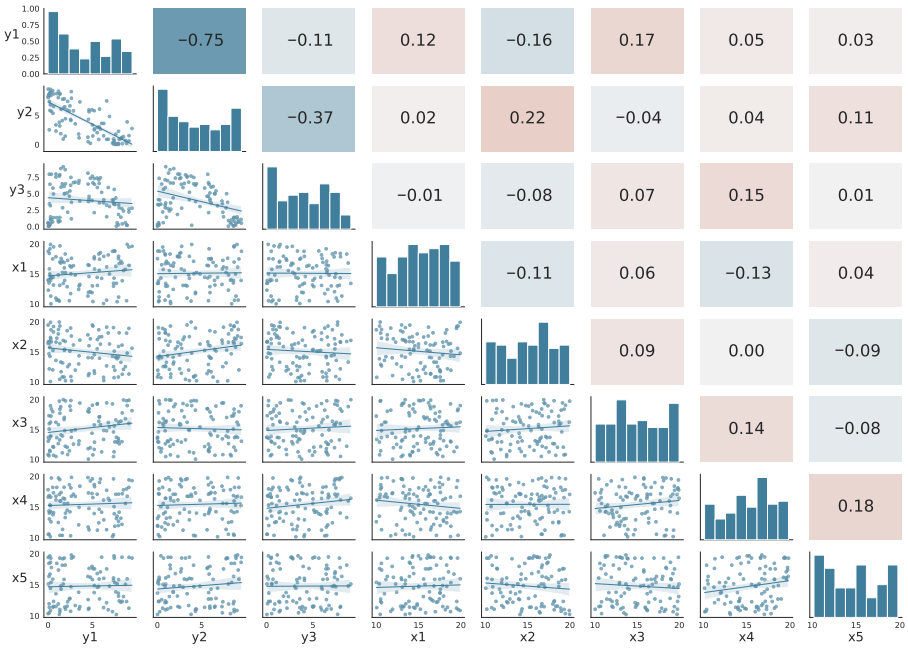


Figure 1. Correlation between variables in the synthetic dataset generated with Cobb–Douglas approach.

The same diagram is shown in Figure 2 for a real dataset taken from the R package Benchmarking, available publicly on CRAN, see [52]. The dataset is from a US federally sponsored program for providing remedial assistance to disadvantaged primary school students [53].

Figure 2 shows that all outputs possessed a significant positive correlation to all inputs, unlike artificial data. It should be emphasized that such a situation is not unique to this particular dataset but applies to most DEA datasets.

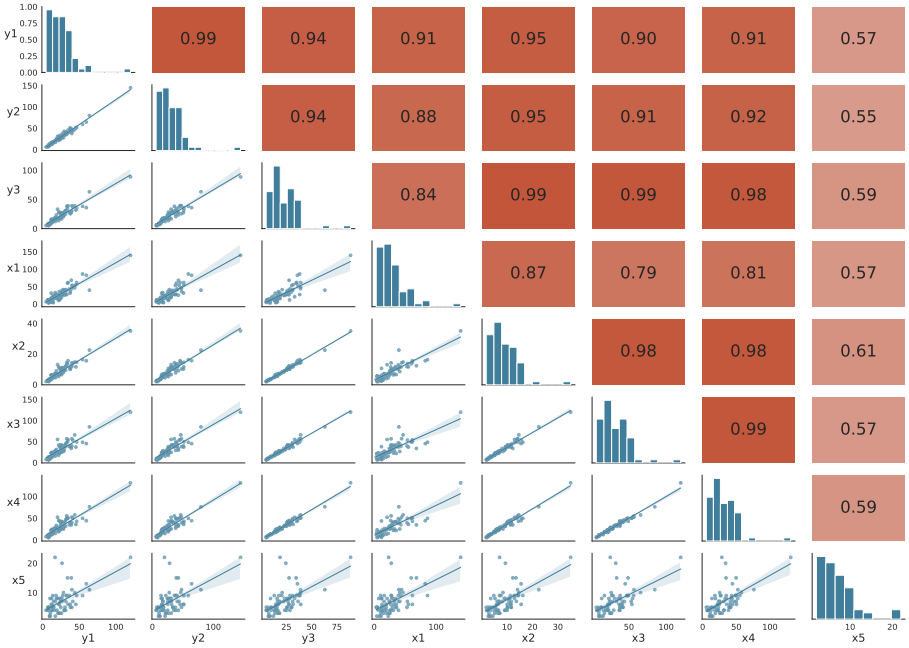


Figure 2. Correlation between variables in real dataset [53].

Paper [48] uses a modified Cobb–Douglas functional form for generating random data. This approach differs from FEAR's `genxy` in that the outputs $\tilde{y}_k$ are generated first as independent and identically distributed uniform random variables between 0.1 and 1. All

inputs except x_1 are generated using uniform distribution in the same way as the outputs. Next, the remaining input is determined by the following expression:

$$x_1 = \left(\frac{\left(\left(\sum_{k=1}^{r} \beta_k (\tilde{y}_k)^2 \right)^{1/2} \right)^{1/\gamma}}{\prod_{i=2}^{m} (x_i)^{\alpha_i}} \right)^{1/\alpha_1}. \quad (3)$$

Coefficients α_i and β_k can be chosen randomly according to the following procedure. The α_1 value is taken arbitrarily in $(0, 1)$, e.g., $\alpha_1 = 0.25$. Other coefficients are found as follows:

$$\alpha_i = \left(\tilde{\alpha}_i \Big/ \left(\sum_{l=2}^{m} \tilde{\alpha}_l \right) \right) (1 - \alpha_1), \quad i = 1, \ldots, m,$$

where parameters $\tilde{\alpha}_2, \ldots, \tilde{\alpha}_m$ are generated randomly from the uniform distribution on the interval $(0, 1)$. The β_k values are determined as $\beta_k = \tilde{\beta}_k \Big/ \left(\sum_{l=1}^{r} \tilde{\beta}_k \right)$, $k = 1, \ldots, r$, where $\tilde{\beta}_k$ are uniformly distributed over the interval $(0, 1)$. Thus, it turns out that input and output coefficients are normalized $\sum_{m=1}^{r} \beta_m = 1$, $\sum_{l=1}^{m} \alpha_l = 1$ to ensure linear homogeneity of the distance functions. Nevertheless, parameter $\gamma \in (0, 1]$ still allows us to select various returns to scale degrees.

Inefficient DMUs are generated based on maximal output vector $\tilde{y}$ by

$$y = \tilde{y} \exp(-u), \quad (4)$$

where u is a random number from a half-normal distribution, i.e., $u \sim |\mathcal{N}(0, \hat{\sigma}_u^2)|$, and $\exp(-u)$ is the measure of inefficiency.

Although this method is much more sophisticated, it has the same disadvantages as the Cobb–Douglas approach (2) described above.

In addition to the commonly utilized uniform and Cobb–Douglas approaches, another interesting method of data generation was applied in [54]. The data generating rule was $Y = 2X$, where X and Y are input and output vectors. This approach is as easy to implement as it is uniform; extra-large samples of size up to one million DMUs with high density can be easily produced. However, this approach also generates unrealistic datasets since they have a 100% correlation between outputs and inputs.

Kohl and Brunner [55] employ a Translog production function for data generation instead of the typically used Cobb–Douglas production function. However, they used just a single output in the DGP and considered only CRS settings. To emulate inefficiency, they utilized a truncated normal distribution with the specified lower bound and an upper bound of 1. The mode of the distribution was chosen below 1 to simulate the maximum point of probability density, not in 1.

Wimmer and Finger [56] used synthetic data generation to replicate the original data because it may be proprietary or confidential. They used the statistical technique proposed by Faisal et al. [57], shown below as Algorithm 1.

Algorithm 1 Generation of synthetic data according to Faisal et al. [57]

1: Take a simple random sample of x_1^{obs} and set it as x_1^{syn}.
2: **for** $i = 2, \ldots, m + r$ **do**
3: Fit model $f\left(x_i^{\text{obs}} \mid x_1^{\text{obs}}, \ldots, x_{i-1}^{\text{obs}}\right)$.
4: Draw x_i^{syn} from $f\left(x_1^{\text{syn}} \mid x_1^{\text{syn}}, \ldots, x_{i-1}^{\text{syn}}\right)$.
5: **end for**

In Algorithm 1, all variables (inputs and outputs) are designated as x for the sake of simplicity, index *obs* stands for original data, and index *syn* is for synthetic one. Different methods can be used for fitting prediction models. Wimmer and Finger utilize a non-

parametric method of classification and regression trees (CART) and a parametric method using normal linear regressions preserving the marginal distribution (NORMRANK).

3. Materials and Methods

3.1. DEA Background

Consider a set of n observed DMUs. Each DMU_j is described by a pair (X_j, Y_j), where $X_j = (x_{1j}, \ldots, x_{mj})^\mathsf{T} \geq 0$ is the input vector, and $Y_j = (y_{1j}, \ldots, y_{rj})^\mathsf{T} \geq 0$ is the output vector. At least one component of the input vector and one component of the output vector are assumed to be non-zero. The production possibility set T is the set $\{(X, Y) \mid$ *the outputs* $Y \geq 0$ *can be produced from the inputs* $X \geq 0\}$.

In DEA, a production possibility set (PPS) is constructed based on a set of axioms and using the observed DMUs. For DEA models with variable return to scale (VRS), the PPS is written in the following form:

$$T = \left\{ (X, Y) \in \mathbb{R}^{m+r} \;\middle|\; \sum_{j=1}^{n} X_j \lambda_j \leq X,\ \sum_{j=1}^{n} Y_j \lambda_j \geq Y,\ \sum_{j=1}^{n} \lambda_j = 1,\ \lambda_j \geq 0,\ j = 1, \ldots, n \right\}. \quad (5)$$

It was proved in [58] that technology (5) generalizes a wide class of DEA models. Therefore, in this paper, we consider only this type of PPS.

Based on PPS (5), an input-oriented model can be written in the form [45]:

$$\begin{aligned} \min\ & \theta - \varepsilon \left(\sum_{k=1}^{m} s_k^- + \sum_{i=1}^{r} s_i^+ \right) \\ \text{subject to } & \sum_{j=1}^{n} X_j \lambda_j + S^- = \theta X_o, \\ & \sum_{j=1}^{n} Y_j \lambda_j - S^+ = Y_o, \\ & \sum_{j=1}^{n} \lambda_j = 1, \\ & S^- = (s_1^-, \ldots, s_m^-)^\mathsf{T} \geq 0, \\ & S^+ = (s_1^+, \ldots, s_r^+)^\mathsf{T} \geq 0, \\ & \lambda_j \geq 0,\ j = 1, \ldots, n, \end{aligned} \quad (6)$$

where $S^- = (s_1^-, \ldots, s_m^-)$ and $S^+ = (s_1^+, \ldots, s_r^+)$ are slack variables, and ε is non-Archimedian value. In model (6) the optimal value θ^* describes the efficiency score of unit (X_o, Y_o), where (X_o, Y_o) is a DMU from the set of observed production units (X_j, Y_j), $j = 1, \ldots, n$.

In this input-oriented model, the possibility of proportional contraction of inputs while keeping outputs constant is sought. Solving model (6) may lead to computational inaccuracies that result in misleading solutions due to the choice of ε [59,60]. Hence, we do not use an infinitesimal constant explicitly in the DEA models since we suppose that each model is solved in two stages in order to separate efficient and weakly efficient units [1]. At the first stage, model (6) is solved by omitting the slacks by simply putting $\varepsilon = 0$. In the second stage, θ is replaced by θ^*, and the sum of the slacks is maximized. The efficiency score in model (6) is estimated as θ^*.

Definition 1 ([1]). *DMU* $(X_o, Y_o) \in T$ *is called efficient with respect to model (6) if and only if any optimal solution satisfies: (a)* $\theta^* = 1$*, (b) all slacks* $s_k^-, k = 1, \ldots, m$*,* $s_i^+, i = 1, \ldots, r$ *are zero.*

If condition (a) in Definition 1 is satisfied, then DMU (X_o, Y_o) is called input weakly efficient with respect to model (6).

In the output-oriented VRS model, the level of output is maximized, keeping levels of the inputs constant:

$$\begin{aligned} &\max \eta \\ &\text{subject to } \sum_{j=1}^{n} X_j\lambda_j + S^- = X_0 \\ &\sum_{j=1}^{n} Y_j\lambda_j - S^+ = \eta Y_0 \\ &\lambda_j \geq 0, \quad j = 1,\ldots,n \\ &S^- = \left(s_1^-,\ldots,s_m^-\right)^{\mathsf{T}} \geq 0 \\ &S^+ = \left(s_1^+,\ldots,s_r^+\right)^{\mathsf{T}} \geq 0 \end{aligned} \tag{7}$$

Definition 2 ([1]). *DMU* $(X_o, Y_o) \in T$ *is called efficient with respect to model (7) if and only if any optimal solution satisfies: (a)* $\eta^* = 1$*, (b) all slacks* $s_k^-, k = 1,\ldots,m$*,* $s_i^+, i = 1,\ldots,r$ *are zero.*

If condition (a) in Definition 2 is hold, then DMU (X_o, Y_o) is called output weakly efficient.

Definition 3 ([61]). *Efficient DMU* $(X_o, Y_o) \in T$ *is called extreme efficient in model (6) or (7) if and only if* $\lambda_o^* = 1$ *and* $\lambda_j^* = 0$*,* $j \neq o$ *for all optimal solutions.*

To test a DMU (X_o, Y_o) for efficiency, an additive model was proposed by Charnes et al. [62]. This model is written in the following form:

$$\begin{aligned} &\max \sum_{k=1}^{m} s_k^- + \sum_{i=1}^{r} s_i^+ \\ &\text{subject to } \sum_{j=1}^{n} X_j\lambda_j + S^- = X_o, \\ &\sum_{j=1}^{n} Y_j\lambda_j - S^+ = Y_o, \\ &\sum_{j=1}^{n} \lambda_j = 1, \\ &S^- = (s_1^-,\ldots,s_m^-)^{\mathsf{T}} \geq 0, \\ &S^+ = (s_1^+,\ldots,s_r^+)^{\mathsf{T}} \geq 0, \\ &\lambda_j \geq 0,\ j = 1,\ldots,n. \end{aligned} \tag{8}$$

This model provides sufficient conditions for the classification of DMUs without dealing with non-Archimedean constants.

Definition 4 ([1]). *DMU* $(X_o, Y_o) \in T$ *is efficient in model (8) if and only if the optimal value of its objective function is zero.*

Theorem 1 ([1]). *DMU* $(X_o, Y_o) \in T$ *is efficient in model (8) if and only if it is efficient in VRS model.*

The additive model has advantages over the radial VRS model (6) in finding efficient DMUs because model (6) is solved in two stages. Therefore, to separate efficient units from weakly efficient ones, two optimization problems should be solved. The additive model maximizes slack variables without projecting DMU onto the frontier first. Therefore, only one optimization problem is required to be solved.

Andersen and Petersen [63] developed a super-efficiency model for ranking efficient DMUs.

$$\begin{aligned} &\min \theta \\ &\text{subject to } \sum_{\substack{j=1 \\ j\neq o}}^{n} X_j\lambda_j \leq \theta X_o, \\ &\sum_{\substack{j=1 \\ j\neq o}}^{n} Y_j\lambda_j \geq Y_o, \\ &\sum_{\substack{j=1 \\ j\neq o}}^{n} \lambda_j = 1, \\ &\lambda_j \geq 0,\ j = 1,\ldots,n. \end{aligned} \tag{9}$$

The efficiency score is obtained by eliminating the DMU under evaluation from the PPS. This results in θ values greater than one, which are then used to rank the efficient DMUs.

Model (9) can also be used to find the radial projection of an arbitrary point onto the frontier. If (9) has an optimal solution, then $(\theta^* X_o, Y_o)$ puts point (X_o, Y_o) on the frontier. If problem (9) is infeasible, then no projection exists.

Output-oriented modes can be written as follows:

$$\begin{aligned} &\max \eta \\ &\text{subject to } \sum_{\substack{j=1 \\ j\neq o}}^{n} X_j\lambda_j \leq X_o, \\ &\sum_{\substack{j=1 \\ j\neq o}}^{n} Y_j\lambda_j \geq \eta Y_o, \\ &\sum_{\substack{j=1 \\ j\neq o}}^{n} \lambda_j = 1, \\ &\lambda_j \geq 0,\ j = 1,\ldots,n. \end{aligned} \tag{10}$$

In the output-oriented model, radial projection is obtained with $(X_o, Y_o) \rightarrow (X_o, \eta^* Y_o)$.

3.2. Assurance Region Method

Weight restrictions are used in DEA in order to incorporate implicit judgments in the dual model for modeling production trade-offs (see, e.g., [64] for a review of weight restriction approaches). For model (6), the dual input-oriented VRS model can be written in the form:

$$\begin{aligned} &\max u^{\mathsf{T}} Y_o + u_0 \\ &\text{subject to } -v^{\mathsf{T}} X_j + u^{\mathsf{T}} Y_j + u_0 \geq 0,\ j = 1,\ldots,n, \\ &v^{\mathsf{T}} X_o = 1, \\ &v \geq 0,\ u \geq 0. \end{aligned} \tag{11}$$

The dual model (11) provides another way of looking at the problem (6). Dual variables u and v are often called weights in DEA because the efficiency score of DMU (X_o, Y_o) is defined as the ratio of a virtual output (weighted sum of outputs $\sum_{i=1}^{r} u_i y_{io} + u_0$) to a virtual input (weighted sum of inputs $\sum_{k=1}^{m} v_k x_{ko}$). In the model (11), dual variables are supposed to be non-negative, and therefore they can be equal to zero in the optimal solution. This means that some variables are ignored in the efficiency evaluation. In order to overcome this situation, it is proposed to insert weight restrictions into the model.

The assurance region (AR) method proposed by Thompson et al. [35,36] extends a DEA model (11) by adding constraints for pairs of dual variables.

$$\begin{aligned} l_k \leq \frac{v_k}{v_1} \leq u_k,\ k = 2, \ldots, m, \\ L_i \leq \frac{u_i}{u_1} \leq U_i,\ i = 2, \ldots, r. \end{aligned} \tag{12}$$

The VRS-AR model is written as follows.

$$\begin{aligned} \max\ & u^{\mathsf{T}} Y_o \\ \text{subject to}\ & v^{\mathsf{T}} X_o = 1, \\ & -v^{\mathsf{T}} X_j + u^{\mathsf{T}} Y_j + u_0 \leq 0,\ j = 1, \ldots, n, \\ & v^{\mathsf{T}} P \leq 0, \\ & u^{\mathsf{T}} Q \leq 0, \\ & u \geq 0,\ v \geq 0, \end{aligned} \tag{13}$$

where

$$P = \begin{pmatrix} l_2 & -u_2 & l_3 & -u_3 & \cdots \\ -1 & 1 & 0 & 0 & \cdots \\ 0 & 0 & -1 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} L_2 & -U_2 & L_3 & -U_3 & \cdots \\ -1 & 1 & 0 & 0 & \cdots \\ 0 & 0 & -1 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix},$$

respectively.

The primal VRS-AR model can be written in the following form, which is usually easier to solve and interpret:

$$\begin{aligned} \min\ & \theta \\ \text{subject to}\ & \sum_{j=1}^{n} X_j \lambda_j - P\pi \leq \theta X_o, \\ & \sum_{j=1}^{n} Y_j \lambda_j + Q\tau \geq Y_o, \\ & \sum_{j=1}^{n} \lambda_j = 1, \\ & \lambda_j \geq 0,\ j = 1, \ldots, n, \\ & \pi \geq 0,\ \tau \geq 0, \end{aligned} \tag{14}$$

where $\pi = (\pi_1, \ldots, \pi_{2(m-1)})^{\mathsf{T}}$ and $\tau = (\tau_1, \ldots, \tau_{2(r-1)})^{\mathsf{T}}$ are extra variables that appear in the primal model as a result of the constraints (12) imposed on the dual variables.

Output-oriented VRS-AR model is written as follows:

$$\begin{aligned} \max\ & \eta \\ \text{subject to}\ & \sum_{j=1}^{n} X_j \lambda_j - P\pi \leq X_o, \\ & \sum_{j=1}^{n} Y_j \lambda_j + Q\tau \geq \eta Y_o, \\ & \sum_{j=1}^{n} \lambda_j = 1, \\ & \lambda_j \geq 0,\ j = 1, \ldots, n, \\ & \pi \geq 0,\ \tau \geq 0. \end{aligned} \tag{15}$$

4. Results

4.1. Idea of the Proposed Approach

The assurance region method puts constraints on the ratio of input and output weights in the form of lower and upper bounds. This approach is mainly used to eliminate the zero weights that often appear in solutions of DEA models. However, weight restrictions (12) have another feature. They reduce the feasible domain of multipliers while the feasible domains of inputs and outputs are expanding. Figure 3a illustrates the incorporation of weight restrictions into the VRS model. As a result, the original production possibility set T_{VRS} expands, and weakly efficient parts of the VRS frontier become inefficient in the VRS-AR model.

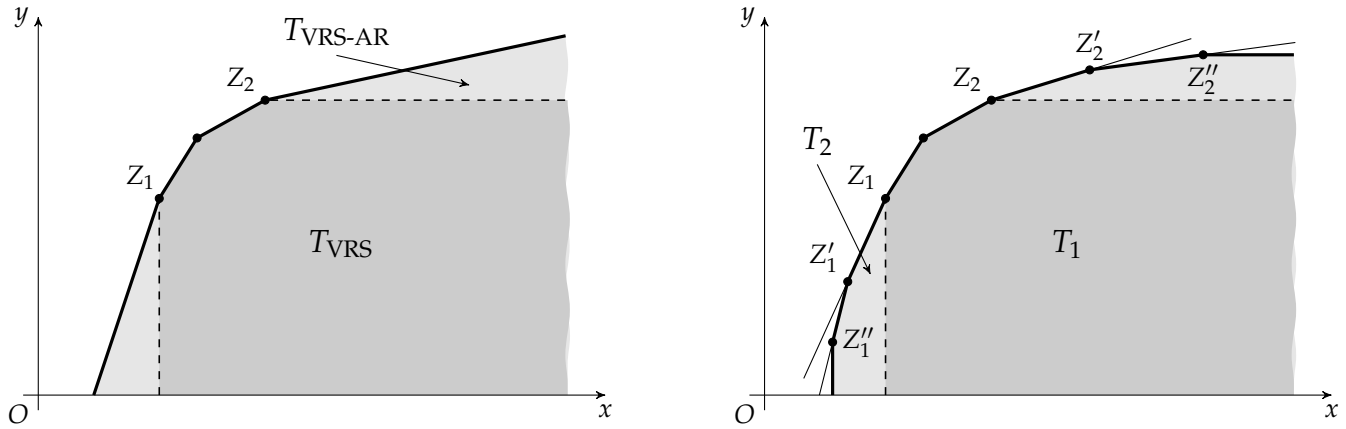


Figure 3. Generation of artificial efficient units using the assurance region method. (**a**) Expansion of a production possibility set T_{VRS} as a result of incorporation weight restrictions; (**b**) Inserting artificial DMUs obtained as projections onto the frontier of the $T_{\text{VRS-AR}}$.

The VRS-AR model is used in this paper to expand the existing data by creating artificial observations "at the edge" of the efficient frontier, i.e., where the efficient part of the frontier adjoins the inefficient part. The proposed approach is illustrated in Figure 3b.

Initially, points Z_1 and Z_2 represent units which are the end-points of two infinite edges of set T_1. These edges are marked by dashed lines. After adding weight restrictions to the model, the frontier transforms and artificial points Z_1' and Z_2' are inserted. The process repeats, reducing the current feasible domain of dual multipliers. This leads to the artificial units Z_1'' and Z_2''. Finally, a synthetic set T_2 is produced that includes DMUs from T_1 and all generated artificial DMUs. The frontier of T_2 is indicated in Figure 3b by a thick solid line. Inefficient units can be inserted into T_2 by generating a random point on the frontier and then reducing outputs or increasing inputs.

The idea of introducing artificial units into the production possibility set is not new. Already, Farrell has used artificial units to secure weights from being zero in his models [65,66]. Further, in the works of Thanassoulis and Allen [43,44,67], artificial units were used to improve the envelopment of PPS. They used the observed extreme efficient units (vertices) that lie on the boundary of the efficient part of the frontier as starting points for improving the frontier. In [67], such units are called anchor units. However, the approach of Thanassoulis and Allen has no formalized algorithm for inserting artificial DMUs.

In this paper, we use the concept of the terminal unit that is proposed in [68,69] as a point of departure for introducing artificial observations.

Definition 5. *Extreme efficient unit is terminal if an infinite edge starts at this unit.*

This definition is better than the approach of Bougnol and Dulá [70] which determines an excessive number of anchor units and the approach of Thanassoulis et al. [44], which

may not identify a sufficient number of units. The algorithm for determining terminal units in the VRS model is described in [68].

4.2. Algorithm For Synthetic Data Generation

According to the principles described in the Introduction, the algorithm for synthetic data generation should not change the original efficient frontier. Therefore, before starting the generation of artificial efficient DMUs, we need to make sure that the initial VRS-AR model does not violate (P1), i.e., that the efficient frontier is preserved after adding weight restrictions to the VRS model. The proposed algorithm for the determination of initial weight restrictions is based on the following propositions.

Proposition 1. *If all efficient units of the VRS model stay efficient in the VRS-AR model, then the efficient frontier of the VRS model belongs to the efficient frontier of the VRS-AR model.*

Proof. Let E be the set of efficient units in the VRS model. Consider a set of extreme efficient units E^*, which is a subset of E. According to Dulá and Thrall [71], the production possibility set is determined by a set of extreme efficient units. Since all units of E are efficient in the VRS-AR model, then all units from set E^* are also efficient in the VRS-AR model. It follows that any point of the original efficient frontier of the VRS model stays efficient in the VRS-AR model. Thus, the efficient frontier of the VRS model belongs to the efficient frontier of VRS-AR model. □

The following proposition asserts the existence of weight restrictions in the VRS-AR model such that all efficient units of the VRS model stay efficient in the corresponding VRS-AR model.

Proposition 2. *For a set of DMUs, there exist nonzero weight restriction coefficients $l_k > 0$, $u_k > 0$, $k = 2, \ldots, m$, and $L_i > 0$, $U_i > 0$, $i = 2, \ldots, r$ in the VRS-AR model (14) such that the set of efficient units of this model coincides with the set of efficient units of the VRS model.*

Proof. According to Lemma 4.1 in [1], there exist dual optimal solution (v^*, u^*, u_0^*) for efficient unit, such that $v^* > 0$ and $u^* > 0$. Let $v_j^* > 0$ and $u_j^* > 0$ be the optimal dual variables for jth DMU. Choose weight restriction coefficients as follows:

$$l_k = \min_{1 \le j \le n} \frac{v_{kj}^*}{v_{1j}^*} > 0,$$

$$u_k = \max_{1 \le j \le n} \frac{v_{kj}^*}{v_{1j}^*} > 0, \; k = 2, \ldots, m,$$

$$L_i = \min_{1 \le j \le n} \frac{u_{ij}^*}{u_{1j}^*} > 0,$$

$$U_i = \min_{1 \le j \le n} \frac{u_{ij}^*}{u_{1j}^*} > 0, \; i = 2, \ldots, r.$$

Then, for each efficient DMU optimal solution (v^*, u^*, u_0^*) of the VRS model satisfies the AR constraints. This means that the optimal solution of the VRS model is also optimal in the VRS-AR model (14). Hence, the efficient units of the VRS model stay efficient in the VRS-AR model. □

From Propositions 1 and 2 we obtain that there exist nonzero weight restriction coefficients in the VRS-AR model that do not change the efficient frontier of the VRS model. However, these statements do not give us an explicit expression for calculating such coefficients. The algorithm proposed below determines the maximal weight restrictions coefficients l_k, L_i, and minimal coefficients u_k, U_i, which does not change the original efficient frontier. This algorithm is close to the concept of Constrained Facet Analysis,

which was originally proposed in [72,73] and clarified in [74]. Since, for our purpose, it is not necessary to determine these weight limits precisely, we use a simple approximating algorithm. Algorithm 2 for the determination of initial weight restrictions that does not change the efficient frontier can be written as follows:

Algorithm 2 Determine initial weight restrictions.

Input: Initial dataset D; small parameter ε, parameter w.
Output: Coefficients l_k, u_k, $k = 2, \ldots, m$, and L_i, U_i, $i = 2, \ldots, s$.

1: **procedure** INITWEIGHTRESTRICTIONS
▷ *Initialize weight restrictions*
2: Set $l_k := \varepsilon$, $u_k := 1/\varepsilon$, $k = 2, \ldots, m$, $L_i := \varepsilon$, $U_i := 1/\varepsilon$, $i = 2, \ldots, r$.
▷ *Correct initial weight restrictions*
3: Determine a set of efficient units E in model (8) for dataset D.
4: Solve model (14) for dataset D. Find the set of efficient units E^{AR}.
5: **while** sets E and E^{AR} are not equal **do**
6: Set $l_k := l_k/w$, $u_k := u_k \cdot w$, $k = 2, \ldots, m$, $L_i := L_i/w$, $U_i := u_i \cdot w$, $i = 2, \ldots, r$.
7: Solve model (14) again and determine the set of efficient units E^{AR}.
8: **end while**
▷ *Adjust input weight restrictions*
9: **for each** input $k = 1, \ldots, m$ **do**
10: Increase $l_k := l_k \cdot w$ until efficient units in model (14) coincide with the set of efficient units in model (8).
11: Decrease $u_k := u_k/w$ until efficient units in model (14) coincide with the set of efficient units in model (8).
12: **end for**
▷ *Adjust output weight restrictions*
13: **for each** output $i = 1, \ldots, r$ **do**
14: Increase $L_i := L_i \cdot w$ until efficient units in model (14) coincide with the set of efficient units in model (8).
15: Decrease $U_i := U_i/w$ until efficient units in model (14) coincide with the set of efficient units in model (8).
16: **end for**
17: **end procedure**

In the algorithm, a small parameter ε is used to initialize the coefficients of the weight restrictions. Parameter w ($0 < w < 1$) characterizes the change of the weight coefficients during the iterations.

Lemma 1. *Algorithm 2 converges in a finite number of steps and does not violate (P1).*

Proof. In step 2 of Algorithm 2, coefficients l_k, u_k, L_i, and U_i are initialized with some values controlled by a small parameter ε.

Next, in steps 3–8, we check that the set of efficient units E in the VRS model and the set of efficient units E^{AR} in the VRS-AR model are equal. This guarantees that (P1) is held according to Proposition 1. If the weight restrictions chosen in step 2 violate (P1), then lower bounds l_k and L_i are decreased by factor w, while upper bounds are increased by w. These steps are repeated until coefficients that meet (P1) are found. Proposition 2 says that such coefficients exist and are positive. Thus, they can be found in a finite number of steps.

In steps 9–16, lower and upper bounds of weight restrictions are adjusted until efficient units in the VRS model coincide with the set of efficient units in the VRS-AR model, i.e., (P1) does not violate here. The adjustments lead to a consistent increase in the lower bounds and a decrease in the upper bounds. Hence, this process will be finished in a finite number of steps because lower bounds cannot be greater than upper bounds.

This completes the proof. □

Next, we describe the algorithm for generating efficient units using a three-dimensional VRS model. In Figure 4a, the PPS of the VRS model is determined by observed production units $A-G$. According to (P2), artificial efficient DMUs should extend the efficient part of the frontier. Units B and D are efficient, but they cannot be used for extending the efficient frontier because they are located in the interior part of it. Units A, C, E, F, and G are terminal since the infinite edges start at these units. These units are vertices of unbounded inefficient facets where the efficient frontier can be expanded. To explain the process of generating artificial efficient units, consider terminal unit G. Two unbounded facets, Γ_1 and Γ_2, contain this unit. In order to identify these facets, several points R_j are randomly generated in the vicinity of unit G such that $\|G - R_j\|_\infty \leq \delta$.

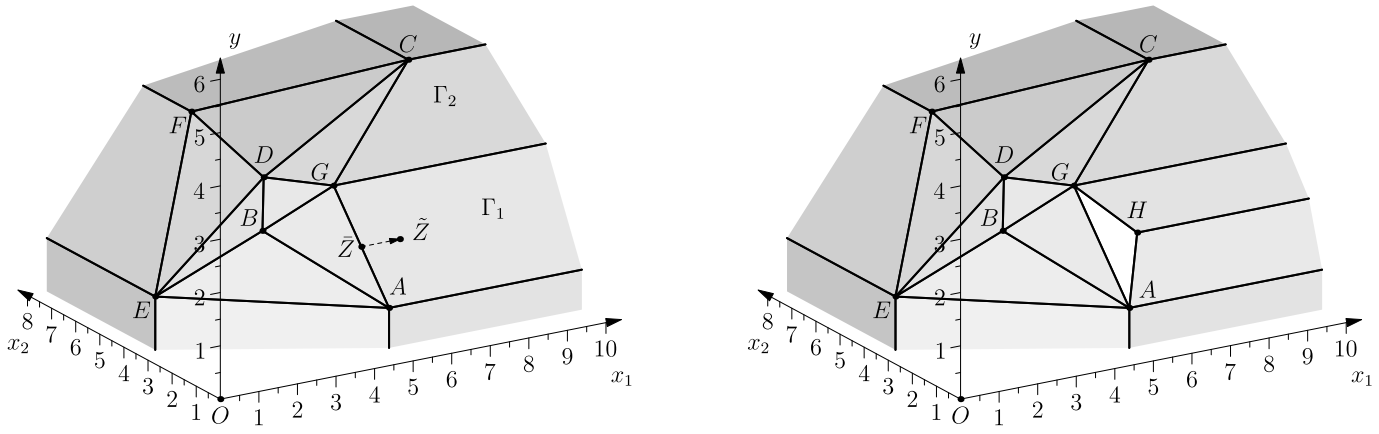


Figure 4. Illustration of generating artificial efficient units in the three-dimensional VRS model. (**a**) Production possibility set T_{VRS} before inserting artificial efficient unit, (**b**) Production possibility set T_{VRS} after inserting artificial efficient unit H.

Variables in DEA models may have various units of measurement, and their values may be of different order. In order to avoid numerical difficulties, we use a weighted infinite norm

$$\|Z\|_\infty = \max_{1\leq k\leq m+r} \beta_k |z_k|, \quad Z \in \mathbb{R}^{m+r}$$

to determine the vicinity of the terminal unit. Here, weights $\beta_k > 0$ are chosen to "normalize" each coordinate k. For the vicinity of unit $G = (g_1, \ldots, g_{m+r})$, its coordinates g_k are used for normalization, i.e., weights of the norm are obtained as $\beta_k = 1/g_k, k = 1, \ldots, m+r$.

By choosing the radius of vicinity $0 < \delta < 1$ and using uniformly distributed random values $w_{kj} \sim U[-\delta, \delta]$, $k = 1, \ldots, m+r$, $j = 1, \ldots, p$, the coordinates of random points $R_j = (r_{1j}, \ldots, r_{(m+r)j})$ are derived as follows:

$$r_{jk} = g_k(1 + w_k), \quad k = 1, \ldots, m+r, \; j = 1, \ldots, p, \tag{16}$$

where p represents the number of randomly generated points.

Next, each random point R_j is projected onto the frontier by solving models (9) and (10). If all slacks in the optimal solution are zero, then the facet is bounded and is not being used further. Otherwise, using the optimal solution of model (9) or (10), an unbounded facet that contains radial projection can be determined.

Suppose the projection lies in the facet Γ_1 and the optimal solution contains the following nonzero optimal λ-variables $\lambda_j^* > 0$, $j \in J$, and optimal slacks $s_k^{-*} > 0$, $k \in I_1$, $s_i^{+*} > 0$, $i \in I_2$, then an unbounded facet Γ_1 that contains the radial projection of some random point can be represented in the form:

$$\Gamma_1 = \left\{ Z \;\middle|\; Z = \sum_{j\in J} Z_j\lambda_j + \sum_{k\in I_1} \mu_k e_k - \sum_{i\in I_2} \rho_i e_i,\ \sum_{j\in J} \lambda_j = 1, \right.$$
$$\left. \lambda_j \geq 0,\ j \in J,\ \mu_k \geq 0,\ k \in I_1,\ \rho_i \geq 0,\ i \in I_2 \right\}, \quad (17)$$

where index set J represents the vertices of Γ_1, and index sets I_1 and I_2 correspond to the rays of Γ_1.

Since inefficient facet Γ_1 is found, it can be used for the construction of artificial efficient units. For this purpose, a point $\widetilde{Z}$ belonging to an unbounded facet is generated:

$$\widetilde{Z} = \overline{Z} + \sum_{k\in I_1} \alpha \overline{Z} e_k - \sum_{i\in I_2} \alpha \overline{Z} e_i, \quad (18)$$

where α is a parameter that determines the shift from point $\bar{Z}$ along the rays of Γ_1; $\overline{Z}$ is the centroid of all vertexes of facet Γ_1 given by

$$\overline{Z} = \left(\sum_{j\in J} Z_j \right) \Big/ |J|,$$

where J is a subset of the vertices of Γ_1. In accordance with the recommendation of Dulá [47], point $\widetilde{Z}$ is constructed with the aim of ensuring the uniformity of the distribution of artificially generated DMUs.

In Figure 4a, the face Γ_1 has two vertices A and G and one ray e_1 corresponding to the axis x_1. Therefore, point $\overline{Z}$ lies in the middle of segment AG, i.e., $\overline{Z} = 0.5A + 0.5G$. According to (18), point $\widetilde{Z}$ is obtained as $\widetilde{Z} = \overline{Z} + \alpha \overline{Z} e_1$.

After that, $\widetilde{Z}$ is projected radially onto the frontier of the VRS-AR model by solving problems (14) or (15). The projection of $\widetilde{Z}$ is the new artificial efficient unit H that is added to the PPS. In Figure 4b, we can see we can see how the PPS has changed after adding unit H. It is worthy of note that the original efficient frontier has not changed, while a new efficient facet, AGH, has appeared.

Such operations are repeated for all the facets that contain terminal units. As a result, a number of artificial efficient units are included in the PPS. Next, weight restrictions are slightly relaxed by multiplying coefficients l_k and L_i by factor w and dividing coefficients u_k and U_i by w. Thus, the iterations continue with a new PPS until the number of efficient DMUs reaches the set value.

The pseudocode of the proposed Algorithm 3 is given below. Parameter p in algorithm represents the number of random units generated for each terminal unit in order to find unbounded facets that contain this unit. The smaller value of the parameter p will result in fewer facets being detected. The higher the p value, the more test points are checked and the more facets the algorithm can potentially detect. However, too high a value of p leads to excessive computations. Therefore, we recommend choosing the p value in the range of 10 to 30. Parameter w ($0 < w < 1$) characterizes the strategy of changing the weight restrictions.

Lemma 2. *Algorithm 3 converges in a finite number of steps and does not violate (P1) and (P2).*

Proof. The algorithm stops when the number of efficient units in S^E is equal to or greater than N^E. Hence, to prove the convergence of the algorithm, it is sufficient to show that at every iteration at least one artificial efficient unit is added to the S^E.

The frontier of the VRS model always has unbounded facets due to the free disposability of inputs and outputs. Additionally, every PPS has at least one terminal unit and all terminal units can be determined in a finite number of steps using the algorithm described in [68]. For each terminal unit, by Definition 5, there exists an infinite edge starting at this unit. The production possibility set of the VRS model is a convex polyhedral set [1]. Hence, there exist a number of unbounded facets that contain the infinite edge and terminal unit itself. This means that there are points in the vicinity of the terminal unit belonging to

unbounded facets or projected onto these facets. Thus, there is a nonzero probability that the projection of the random point will belong to an unbounded facet. With a sufficiently large number of random points, there will be at least one point that is projected onto an unbounded facet. If at least one unbounded facet is found, then it is projected onto the efficient frontier of the VRS-AR model. According to Theorem 5 in [33], this projection always exists and represents an efficient artificial DMU. Thus, at least one artificial efficient unit is generated at every iteration and the algorithm converges.

At every iteration of the Algorithm 3 in Step 18, weight restrictions are being relaxed by multiplying lower bounds l_k and L_i by parameter $w < 1$ and by dividing upper bounds u_k and U_i by the same value. This ensures that the existing efficient frontier will not change after such a correction of the weight restrictions and that (P1) is not violated.

By construction, unit $\widetilde{Z}$ belongs to an unbounded facet of VRS frontier, then it will be projected onto an unbounded facet of the VRS-AR frontier. In Step 15, this projection becomes an artificial efficient unit that does not belong to the initial efficient frontier of the VRS model because unbounded facets in the VRS model are inefficient. Thus, artificial efficient DMUs extend the efficient part of the frontier, and (P2) also holds.

This completes the proof. □

Algorithm 3 Generation of artificial efficient units.

Input: Initial dataset D; number of efficient units N_E; initial weight restrictions l_k, u_k, $k = 2, \ldots, m$, and L_i, U_i, $i = 2, \ldots, r$.
Output: Synthetic dataset S^E.

1: **procedure** GENERATEEFFICIENTDMUS
2: Set $S^E := D$. ▷ *Initialization*
3: **while** number of efficient units in S^E is less than N_E **do**
4: Set $F := \emptyset$, $P := \emptyset$.
5: Find the set of terminal units E^t in dataset S using the algorithm described in [68].
6: **for each** terminal unit Z_t in E^t **do**
7: Generate random units R_j, $j = 1, \ldots, p$ in the vicinity of unit Z_t using (16).
8: Find projections of units R_j onto the boundary of PPS by solving models (9) and (10).
9: Find vertices and direction vectors of unbounded facets, where units are projected.
10: Add facets to the set F, keeping only distinct facets.
11: **end for**
12: **for each** facet f in F **do**
13: Generate artificial unit $\widetilde{Z}$ according to Equation (18).
14: Project unit $\widetilde{Z}$ onto VRS-AR frontier by solving models (14) and (15).
15: Append projected unit to the set P.
16: **end for**
17: Set $S^E := S^E \cup P$.
18: Set $l_k := l_k \cdot w$, $u_k := u_k / w$, $k = 2, \ldots, m$, $L_i := L_i \cdot w$, $U_i := u_i / w$, $i = 2, \ldots, r$.
19: **end while**
20: **if** number of efficient units in S is greater than N_E **then**
21: Remove artificial efficient units in S to make it equal to N_E.
22: **end if**
23: **return** S^E
24: **end procedure**

At the next stage, inefficient artificial DMUs are generated. In order to generate artificial units with realistic input-output mixes, we used the convex hull of DMUs from the original dataset together with previously generated efficient DMUs because the convex hull contains all possible proportions between inputs and outputs contained in the data. Uniform sampling from a convex hull of points is computationally hard due to the large number

of vertexes and large dimension of space. Hence, we chose the simpler and more computationally efficient method. First, we randomly select $m + r + 1$ different DMUs. These points form a random simplex that is contained in a convex hull. Next, a random point is chosen by uniform sampling from the derived simplex using the method proposed in [75].

After that, to generate inefficient artificial DMUs, random points from the convex hull of the dataset S^E produced by Algorithm 3 are projected onto the frontier. Inefficient artificial DMUs are generated relative to those projections by proportionally increasing inputs or proportionally contracting outputs. In the first case, inputs are increased as

$$X_k = \widetilde{X}_k / exp(-u_i),$$

where $u_i \sim |\mathcal{N}(0, \sigma_u^2)|$, i.e., u_i has a half-normal distribution that is produced by the underlying normal. In the output case, inefficient DMUs are generated according to

$$Y_k = \widetilde{Y}_k \times exp(-u_i).$$

In the proposed algorithm, both methods alternate when generating inefficient artificial DMUs.

To ensure that the generated data has a distribution of efficiency scores close to the original, σ_u^2 can be estimated from the initial dataset as the sample variance

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} u_i^2, \tag{19}$$

where $u_i = -\ln(\theta^i)$, and θ^i is the efficiency score of inefficient DMU_i from initial dataset D.

Combining the steps described above, we obtain the following Algorithm 4.

Algorithm 4 Generating inefficient DMUs.

Input: Synthetic dataset S^E; estimate $\hat{\sigma}_v^2$ for input efficiencies; estimate $\hat{\sigma}_u^2$ for output efficiencies; number of inefficient units N.
Output: Set of inefficient DMUs S^I.

procedure GENERATEINEFFICIENTDMUS
  $S^I := \emptyset$. ▷ *Initialization*
  Generate N random points (X_i^R, Y_i^R), $i = 1, \ldots, N$ in $\text{CH}(S^E)$.
  for $i := 1$ to N **do**
    if $(i \bmod 2) = 1$ **then** ▷ *Alternate between the input and output models*
      Solve model (9) on S^E for unit (X_i^R, Y_i^R).
      Find projection $(\widetilde{X}_i^R, \widetilde{Y}_i^R) := (\theta_i X_i^R, Y_i^R)$.
      Generate random $u_i \sim |\mathcal{N}(0, \hat{\sigma}_u^2)|$.
      $Z_i := (\widetilde{X}_i^R / exp(-u_i), \widetilde{Y}_i^R)$.
    else
      Solve model (10) on S^E for unit (X_i^R, Y_i^R).
      Find projection $(\widetilde{X}_i^R, \widetilde{Y}_i^R) := (X_i^R, \eta_i Y_i^R)$.
      Generate random $v_i \sim |\mathcal{N}(0, \hat{\sigma}_v^2)|$.
      $Z_i := (\widetilde{X}_i^R, \widetilde{Y}_i^R \times exp(-v_i))$.
    end if
    Append Z_i to the set S^I.
  end for
  return S^I
end procedure

The full algorithm for synthetic data generation can be summarized as Algorithm 5. In the first step, we determine restrictions on weights that are large enough but do not change the efficient frontier. Then, we generate efficient units according to Algorithm 3. Finally, we generate inefficient units with Algorithm 4 using $\hat{\sigma}_u^2$ determined from the original dataset.

The correctness of the algorithm is confirmed by the following theorem.

Theorem 2. *Algorithm 5 converges in a finite number of steps and does not violate (P1) and (P2).*

Algorithm 5 Synthetic data generation.

Input: Initial dataset D; total number of units N; number of efficient units N_E.
Output: Synthetic dataset S.
1: Find initial weight restrictions according to Algorithm 2.
2: Generate N_E efficient units using Algorithm 3.
3: Find $\hat{\sigma}_v^2$ and $\hat{\sigma}_u^2$ of dataset D according to Equation (19).
4: Generate $(N - N_E)$ inefficient units using Algorithm 4.
5: $S := S^E \cup S^I$.
6: **return** S.

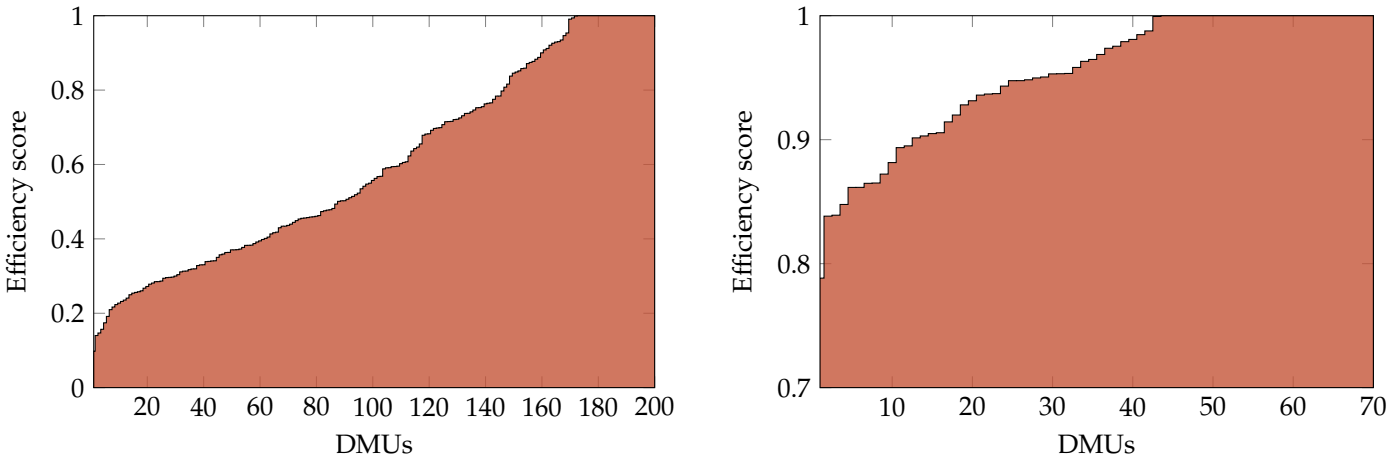
Proof. According to Lemmas 1 and 2, steps 1 and 2 of the algorithm are executed in a finite number of steps. Steps 3–6 also consist of a limited number of steps. Thus, Algorithm 5 converges.

Inefficient units generated in Algorithm 4 are located inside PPS; they cannot change the efficient frontier. Taking into account the results of Lemmas 1 and 2, it follows that Algorithm 5 does not violate (P1) and (P2).

This completes the proof. □

4.3. Computational Experiments

The work of Algorithm 5 is illustrated using two small datasets. For the first dataset, Case 1, we took the data from a Russian bank's financial accounts for 2008. The dataset contains 200 DMUs with 6 variables (3 inputs and 3 outputs), see [76]. For Case 2, we took the data from the paper of Charnes et al. [53]; this dateset is described in Section 2. The distributions of the efficiency scores in Cases 1 and 2 are presented in Figure 5.



(**a**) Case 1. (**b**) Case 2.

Figure 5. Distribution of efficiency scores in the original datasets.

The computational experiments were conducted on a PC with Intel Core i3 CPU 3.33 GHz. We use CPLEX [77] version 12.6.2 to solve optimization problems.

In Algorithm 3, we set a small parameter $\varepsilon = 10^{-3}$ to initialize the coefficients of the weight restrictions. Parameter w, which characterizes the change of the weight coefficients during the iterations, is equal to 0.5. The number of generated random points in Algorithm 3 is regulated by the parameter p, which is selected as 20.

Our computations confirmed that the algorithm works correctly, and all initially efficient DMUs remained efficient in the expanded dataset. In other words, the algorithm does not violate the established principles.

Table 1 illustrates the work of Algorithm 3 for Case 1. At the beginning of the first iteration, there were 200 DMUs, of which 28 were efficient and terminal. Then,

229 artificial DMUs are created, and the total number of DMUs becomes 429. After the second iteration, 1140 artificial DMUs were added to the dataset, and the total number of DMUs reached 1369. In the third iteration, 6452 DMUs are produced, bringing the total number of artificial DMUs to 7821. We see that the number of artificial units is increasing rapidly with each iteration, so it turned out that only three iterations were enough to obtain a sufficient number of efficient DMUs.

Table 1. Iterations of Algorithm 3.

Iterations	Number of DMUs				
	Initial	Efficient	Terminal	Generated	Total Artificial
1	200	28	28	229	229
2	429	194	194	1140	1369
3	1569	1122	1096	6452	7821

Before generating inefficient units, the variances $\hat{\sigma}_v^2$ and $\hat{\sigma}_u^2$ should be estimated from the original dataset. Applying (19) to the input and output efficiency scores of observed DMUs, the following estimates were obtained

$$\hat{\sigma}_v^2 = 1.2921, \qquad \hat{\sigma}_u^2 = 0.7548.$$

According to Algorithm 4, inefficient units were added so that the total number of DMUs became 50,000. The synthetic dataset contains 7440 efficient DMUs, so the share of efficient units is equal to 14.88%, which is approximately the same as in the original dataset.

The distribution of the efficiency score of the original dataset in comparison with artificially generated dataset for Case 1 is presented in Figure 6.

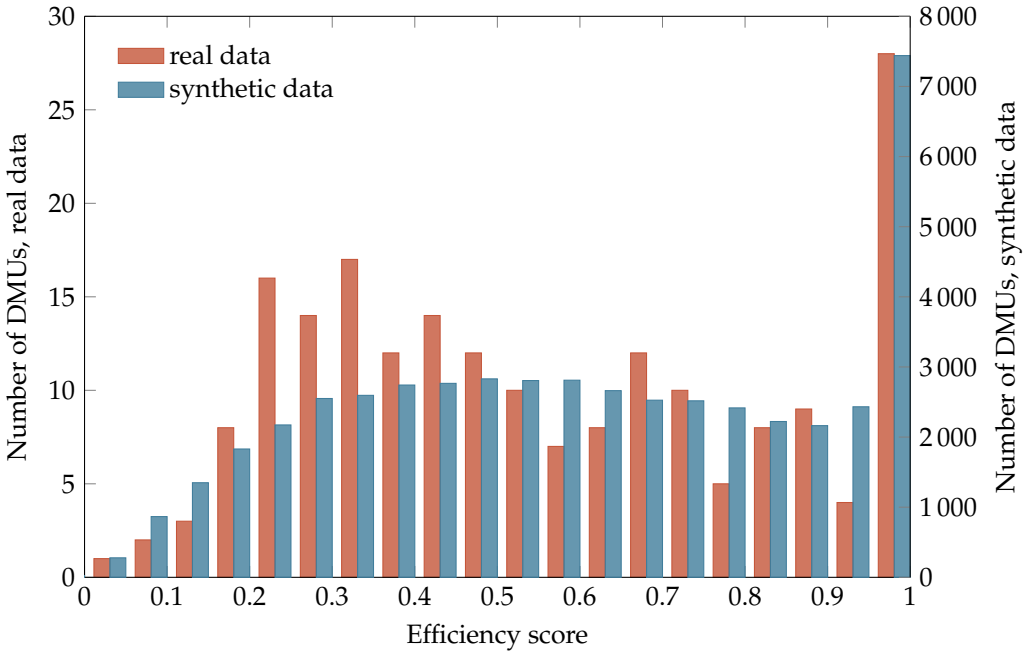
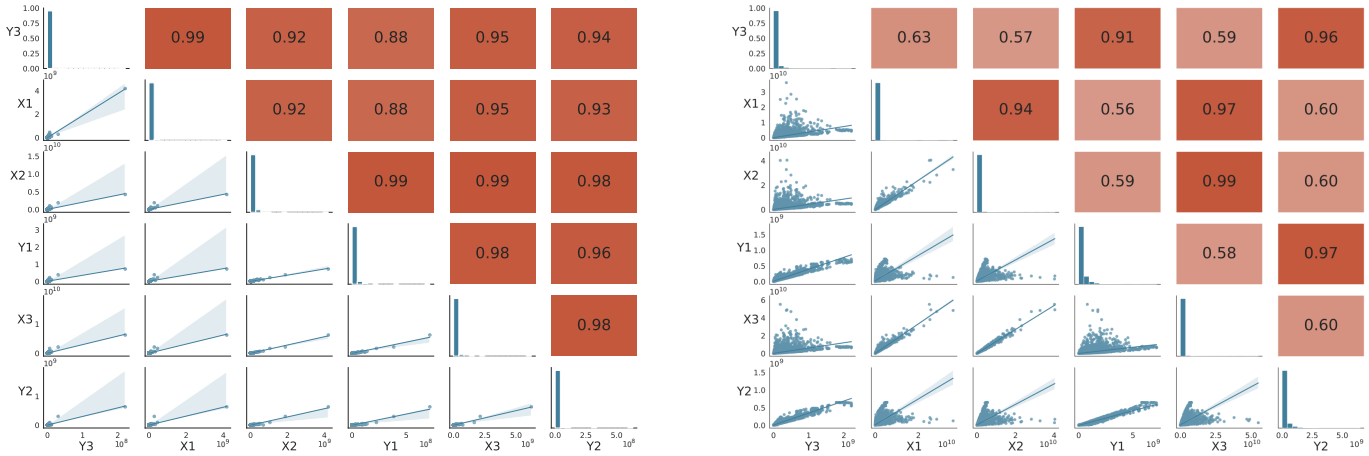


Figure 6. Distribution of efficiency scores in real and synthetic datasets in Case 1.

Figure 7 represents the correlation between variables in the original dataset and in the synthetic dataset. It can be seen from the figure that after completing the dataset with artificial DMUs, some coefficients become smaller. However, the Algorithm 5 retains the positive correlation between inputs and outputs.



(**a**) Original dataset. (**b**) Synthetic dataset.

Figure 7. Correlation between variables in Case 1.

For Case 2, the original dataset contains 70 DMUs, and 27 (or 38.57%) of those are efficient. After three iterations of Algorithm 3, the number of efficient DMUs become 19,480. The variance estimates for input and output efficiency measures in the original dataset are obtained as $\hat{\sigma}_v^2 = 0.0091$ and $\hat{\sigma}_u^2 = 0.0094$. Next, 30,477 inefficient DMUs were generated according to Algorithm 4, and the total number of DMUs in the synthetic dataset was reached 50,000. The share of efficient units in the produced dataset is equal to 38.96%.

In order to investigate the multicollinearity in synthetic datasets for Cases 1 and 2, we use the variance inflation factor (VIF). According to the test results presented in Tables 2 and 3, the proposed algorithm reduces multicollinearity among the variables as a whole. The value of VIF has increased only for variable x_5 in Case 2.

Table 2. Variance inflation factors of variables in Case 1.

Dataset	x_1	x_2	x_3	y_1	y_2	y_3
Original	127.99	323.18	397.85	54.90	40.04	100.83
Synthetic	24.22	86.09	139.04	19.16	39.14	16.48

Table 3. Variance inflation factors of variables in Case 2.

Dataset	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
Original	11.77	198.11	83.28	57.58	1.73	69.32	55.32	186.80
Synthetic	8.25	22.36	18.57	17.82	5.82	17.31	11.85	11.85

Table 4 presents the execution time for Algorithm 5 in Cases 1 and 2. Only the main steps of the algorithm are included in the table because the remaining steps are performed in a very short time, which can be neglected compared to the other steps. The computational experiments show that the total execution time for Case 2 is greater than in Case 1. This is due to the fact that the number of variables of Case 2 is greater, and the initial number of DMUs is almost three times smaller than in Case 1.

Table 4 shows that the running time of Algorithm 5 is sufficiently long. It is obvious that the proposed approach has worse time complexity than other algorithms for synthetic data generation in DEA because it requires solving a large number of LPs. However, compared to other algorithms, Algorithm 5 has certain advantages.

The basic assumption of a production theory is the monotonicity of the production process. This means that as inputs increase, outputs also increase. Using a uniform distribution as a DGP leads to input-output combinations that violate this assumption. Figure 1 shows that the Cobb–Douglas approach also produces datasets where negative correlation coefficients are presented for some pairs of inputs and outputs. At the same

time, computational experiments show that our algorithm preserves isotonicity and does not increase the collinearity of the original data as a whole.

Table 4. Execution time for Algorithm 5.

Stages of Algorithm 5	**Execution Time, s**	
	Case 1	**Case 2**
Finding initial weight restrictions (Algorithm 2)	27	9
Generating efficient DMUs (Algorithm 3)	434	994
Generating inefficient DMUs (Algorithm 4)	603	422
Total	1064	1425

Our approach uses the partial synthetic generation of the DEA datasets. This makes it possible to use the properties of real DMUs to generate artificial ones. Furthermore, the proposed algorithm has a number of useful properties. First, it does not change the existing efficient frontier. In the DEA approach, the efficient frontier contains valuable practical information, so it must be preserved when artificial units are added to the dataset. Second, the algorithm extends the efficient frontier in the borderline region, which makes the frontier more versatile.

The approach proposed in this paper differs from the method of Wimmer and Finger [56] for two reasons. First, the objective is different: they tried to replicate the original dataset, i.e., just replace the original DMUs by artificial ones, and they were not intended to generate a large dataset. Moreover, Algorithm 1 does not preserve the existing frontier since it is designed to replace the original data with artificial ones. Second, in order for the statistical models in Algorithm 1 to be sufficiently accurate, the dataset must be large enough. Our approach does not have such limitations and works well even with small datasets.

5. Conclusions

Some DEA studies need large-scale datasets to test the methods that they propose. The algorithms for applying DEA to big data are most in need because there are not enough open datasets with a large number of DMUs. Thus, these studies mainly use synthetic datasets for testing. Existing data generation algorithms in DEA produce datasets from scratch and cannot provide sufficient statistical variability that is inherent in real data. To fill this gap, a new method is proposed for generating synthetic data. This method receives a real dataset as input and complements it with artificial DMUs. Artificial efficient units are generated in the regions of input-output space that are not covered by the available data. For this reason, weight restrictions are used, which helps to adjust the properties of the efficient frontier in the data neighborhood. Inefficient DMUs are generated taking into account the statistical characteristics of the original dataset. Our computational experiments using two real datasets demonstrate that the proposed algorithm works reliably and can increase the number of DMUs up to 50 K.

The main limitation of the study is that the proposed algorithm has a worse time complexity compared to other existing algorithms in DEA. However, this disadvantage is not so critical. The implementation of parallel computations in the algorithm can significantly reduce the calculation time because many LPs in Algorithm 5 can be solved independently. Moreover, for DEA computations with large datasets, the faster algorithms can be applied [32,54]. So, the authors will address this issue in future works.

In this paper, an algorithm for generating synthetic data is presented only for the VRS model. The generalization to other DEA technologies leaves room for future research. Furthermore, the analysis of the proposed algorithm could be extended with other tests such as rules of thumb and sensitivity analysis.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/data8100146/s1.

Funding: This work was supported by the Russian Science Foundation (project No. 23-11-00197) https://rscf.ru/en/project/23-11-00197/ (accessed on 20 September 2023).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The generated synthetic datasets are available as supplementary material in the online version of the paper.

Acknowledgments: The author thanks the academic editors and anonymous reviewers for their guidance and constructive suggestions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AR	Assurance Region
CRAN	Comprehensive R Archive Network
CRS	Constant Returns to Scale
DEA	Data Envelopment Analysis
DGP	Data Generating Process
DMU	Decision Making Unit
ML	Machine Learning
LP	Linear programming
PPS	Production Possibility Set
RTS	Returns to Scale
SCSC	Strong Complementary Slackness Conditions
VRS	Variable Returns to Scale

References

1. Cooper, W.W.; Seiford, L.M.; Tone, K. *Data Envelopment Analysis. A Comprehensive Text with Models, Applications, References and DEA-Solver Software*, 2nd ed.; Springer Science and Business Media: New York, NY, USA, 2007. [CrossRef]
2. Mozaffari, M.R.; Ostovan, S. Finding projection in the two-stage supply chain in DEA-R with random data using (CRA) model. *Big Data Comput. Visions* **2021**, *1*, 146–155. [CrossRef]
3. Fallah, R.; Kouchaki Tajani, M.; Maranjory, M.; Alikhani, R. Comparison of Banks and Ranking of Bank Loans Types on Based of Efficiency with Dea in Iran. *Big Data Comput. Visions* **2021**, *1*, 36–51. [CrossRef]
4. Soltani, M.R.; Edalatpanah, S.A.; Sobhani, F.M.; Najafi, S.E. A Novel Two-Stage DEA Model in Fuzzy Environment: Application to Industrial Workshops Performance Measurement. *Int. J. Comput. Intell. Syst.* **2020**, *13*, 1134–1152. [CrossRef]
5. Ebrahimzadeh Shermeh, H.; Alavidoost, M.H.; Darvishinia, R. Evaluating the efficiency of power companies using data envelopment analysis based on SBM models: A case study in power industry of Iran. *J. Appl. Res. Ind. Eng.* **2018**, *5*, 286–295. [CrossRef]
6. Khodabakhshi, M.; Cheraghali, Z. Ranking of Iranian executive agencies using audit court budget split indexes and data envelopment analysis. *J. Appl. Res. Ind. Eng.* **2022**, *9*, 312–322. [CrossRef]
7. Montazeri, F.Z. An overview of data envelopment analysis models in fuzzy stochastic environments. *J. Fuzzy Ext. Appl.* **2020**, *1*, 272–278. [CrossRef]
8. Ucal Sari, I.; Ak, U. Machine efficiency measurement in industry 4.0 using fuzzy data envelopment analysis. *J. Fuzzy Ext. Appl.* **2022**, *3*, 177–191. [CrossRef]
9. Bazargan, A.; Najafi, S.E.; Hoseinzadeh Lotfi, F.; Fallah, M.; Edalatpanah, S.A. Presenting a productivity analysis model for Iran oil industries using Malmquist network analysis. *Decis. Mak. Appl. Manag. Eng.* **2023**, *6*, 251–292. [CrossRef]
10. Ratner, S.V.; Balashova, S.A.; Lychev, A.V. The Efficiency of National Innovation Systems in Post-Soviet Countries: DEA-Based Approach. *Mathematics* **2022**, *10*, 3615. [CrossRef]
11. Kassaei, S.; Hosseinzadeh Lotfi, F.; Amirteimoori, A.; Rostamy-Malkhalifeh, M.; Rahmani, B. Identification and evaluation of congestion in two-stage network data envelopment analysis. *Int. J. Res. Ind. Eng.* **2023**, *12*, 53–72. [CrossRef]

12. Ghomashi Langroudi, A.; Abbasi, M. A neutral DEA model for cross-efficiency evaluation. *Int. J. Res. Ind. Eng.* **2022**, *11*, 411–422. [CrossRef]
13. Sigala, M. Using Data Envelopment Analysis for Measuring and Benchmarking Productivity in the Hotel Sector. *J. Travel Tour. Mark.* **2004**, *16*, 39–60. [CrossRef]
14. Maghbouli, M.; Yekta, A.P. Undesirable Input in Production Process: A DEA-Based Approach. *J. Oper. Strateg. Anal.* **2023**, *1*, 46–54. [CrossRef]
15. Sueyoshi, T.; Sekitani, K. Measurement of returns to scale using a non-radial DEA model: A range-adjusted measure approach. *Eur. J. Oper. Res.* **2007**, *176*, 1918–1946. [CrossRef]
16. Sueyoshi, T.; Sekitani, K. The measurement of returns to scale under a simultaneous occurrence of multiple solutions in a reference set and a supporting hyperplane. *Eur. J. Oper. Res.* **2007**, *181*, 549–570. [CrossRef]
17. Sueyoshi, T.; Sekitani, K. An occurrence of multiple projections in DEA-based measurement of technical efficiency: Theoretical comparison among DEA models from desirable properties. *Eur. J. Oper. Res.* **2009**, *196*, 764–794. [CrossRef]
18. Dantzig, G.B.; Thapa, M.N. *Linear Programming 2: Theory and Extensions*; Springer: New York, NY, USA, 2003.
19. Krivonozhko, V.E.; Førsund, F.R.; Lychev, A.V. A note on imposing strong complementary slackness conditions in DEA. *Eur. J. Oper. Res.* **2012**, *220*, 716–721. [CrossRef]
20. Krivonozhko, V.E.; Afanasiev, A.P.; Førsund, F.R.; Lychev, A.V. Comparison of Different Methods for Estimation of Returns to Scale in Nonradial Data Envelopment Analysis Models. *Autom. Remote Control.* **2022**, *83*, 1136–1148. [CrossRef]
21. Anderson, T.; Rouse, P. Data Envelopment Analysis Dataset Repository. Available online: http://www.etm.pdx.edu/dea/dataset/default.htm (accessed on 21 June 2023).
22. Kaggle Datasets. Available online: https://www.kaggle.com/datasets (accessed on 21 June 2023).
23. PASCAL Network. Machine Learning Data Repository. Available online: http://mldata.org/ (accessed on 21 June 2023).
24. The World Bank Group. World Bank Open Data. Available online: https://data.worldbank.org/indicator (accessed on 21 June 2023).
25. United Nations. UNdata. Available online: http://data.un.org/ (accessed on 21 June 2023).
26. Harvard College. Harvard Dataverse Repository. Available online: https://dataverse.harvard.edu/ (accessed on 21 June 2023).
27. European Organization For Nuclear Research; OpenAIRE. Zenodo. 2013. Available online: https://doi.org/10.25495/7GXK-RD71 (accessed on 22 September 2023).
28. Figueira, A.; Vaz, B. Survey on Synthetic Data Generation, Evaluation Methods and GANs. *Mathematics* **2022**, *10*, 2733. [CrossRef]
29. Lu, Y.; Shen, M.; Wang, H.; Wei, W. Machine Learning for Synthetic Data Generation: A Review. *arXiv* **2023**, arXiv:2302.04062. Available online: https://arxiv.org/abs/2302.04062 (accessed on 21 June 2023).
30. Zhu, N.; Zhu, C.; Emrouznejad, A. A combined machine learning algorithms and DEA method for measuring and predicting the efficiency of Chinese manufacturing listed companies. *J. Manag. Sci. Eng.* **2021**, *6*, 435–448. [CrossRef]
31. Guerrero, N.M.; Aparicio, J.; Valero-Carreras, D. Combining Data Envelopment Analysis and Machine Learning. *Mathematics* **2022**, *10*, 909. [CrossRef]
32. Khezrimotlagh, D.; Zhu, J.; Cook, W.D.; Toloo, M. Data envelopment analysis and big data. *Eur. J. Oper. Res.* **2019**, *274*, 1047–1054. [CrossRef]
33. Charnes, A.; Cooper, W.W.; Wei, Q.L.; Huang, Z.M. Cone ratio data envelopment analysis and multi-objective programming. *Int. J. Syst. Sci.* **1989**, *20*, 1099–1118. [CrossRef]
34. Charnes, A.; Cooper, W.W.; Huang, Z.M.; Sun, D.B. Polyhedral Cone-Ratio DEA Models with an Illustrative Application to Large Commercial Banks. *J. Econ.* **1990**, *46*, 73–91. [CrossRef]
35. Thompson, R.G.; Singleton, F.D.; Thrall, R.M.; Smith, B.A. Comparative Site Evaluations for Locating a High-Energy Physics Lab in Texas. *Interfaces* **1986**, *16*, 35–49. [CrossRef]
36. Thompson, R.G.; Langemeier, L.N.; Lee, C.T.; Lee, E.; Thrall, R.M. The role of multiplier bounds in efficiency analysis with an application to Kansas farming. *J. Econ.* **1990**, *46*, 93–108. [CrossRef]
37. Thompson, R.G.; Dharmapala, P.; Rothenberg, L.J.; Thrall, R.M. DEA/AR efficiency and profitability of 14 major oil companies in U.S. exploration and production. *Comput. Oper. Res.* **1996**, *23*, 357–373. [CrossRef]
38. Brockett, P.L.; Charnes, A.; Cooper, W.W.; Huang, Z.M.; Sun, D.B. Data transformations in DEA cone ratio envelopment approaches for monitoring bank performance. *Eur. J. Oper. Res.* **1997**, *98*, 250–268. [CrossRef]
39. Wei, Q.; Yan, H.; Xiong, L. A bi-objective generalized data envelopment analysis model and point-to-set mapping projection. *Eur. J. Oper. Res.* **2008**, *190*, 855–876. [CrossRef]
40. Podinovski, V.V. Production trade-offs and weight restrictions in data envelopment analysis. *J. Oper. Res. Soc.* **2004**, *55*, 1311–1322. [CrossRef]
41. Podinovski, V.V. Improving data envelopment analysis by the use of production trade-offs. *J. Oper. Res. Soc.* **2007**, *58*, 1261–1270. [CrossRef]
42. Podinovski, V.V.; Bouzdine-Chameeva, T. Weight Restrictions and Free Production in Data Envelopment Analysis. *Oper. Res.* **2013**, *61*, 426–437. [CrossRef]
43. Allen, R.; Thanassoulis, E. Improving envelopment in data envelopment analysis. *Eur. J. Oper. Res.* **2004**, *154*, 363–379. [CrossRef]
44. Thanassoulis, E.; Kortelainen, M.; Allen, R. Improving envelopment in Data Envelopment Analysis under variable returns to scale. *Eur. J. Oper. Res.* **2012**, *218*, 175–185. [CrossRef]

45. Banker, R.D.; Charnes, A.; Cooper, W.W. Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Manag. Sci.* **1984**, *30*, 1078–1092. [CrossRef]
46. Barr, R.S.; Durchholz, M.L. Parallel and hierarchical decomposition approaches for solving large-scale Data Envelopment Analysis models. *Ann. Oper. Res.* **1997**, *73*, 339–372. [CrossRef]
47. Dulá, J.H. A computational study of DEA with massive data sets. *Comput. Oper. Res.* **2008**, *35*, 1191–1203. [CrossRef]
48. Zelenyuk, V. Aggregation of inputs and outputs prior to Data Envelopment Analysis under big data. *Eur. J. Oper. Res.* **2020**, *282*, 172–187. [CrossRef]
49. Dulá, J.H. An Algorithm for Data Envelopment Analysis. *INFORMS J. Comput.* **2011**, *23*, 284–296. [CrossRef]
50. Wilson, P.W. FEAR: A software package for frontier efficiency analysis with R. *Socio-Econ. Plan. Sci.* **2008**, *42*, 247–254. [CrossRef]
51. Wilson, P.W. Asymptotic Properties of Some Non-Parametric Hyperbolic Efficiency Estimators. In *Exploring Research Frontiers in Contemporary Statistics and Econometrics: A Festschrift for Léopold Simar*; Van Keilegom, I., Wilson, P.W., Eds.; Physica-Verlag HD: Berlin/Heidelberg, Germany, 2011; pp. 115–150. [CrossRef]
52. Bogetoft, P.; Otto, L. *Benchmarking with DEA, SFA, and R*; International Series in Operations Research & Management Science; Springer: New York, NY, USA, 2011. [CrossRef]
53. Charnes, A.; Cooper, W.W.; Rhodes, E. Evaluating Program and Managerial Efficiency: An Application of Data Envelopment Analysis to Program Follow Through. *Manag. Sci.* **1981**, *27*, 668–697. [CrossRef]
54. Khezrimotlagh, D.; Zhu, J. Data Envelopment Analysis and Big Data: Revisit with a Faster Method. In *Data Science and Productivity Analytics*; Charles, V., Aparicio, J., Zhu, J., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 1–34. [CrossRef]
55. Kohl, S.; Brunner, J.O. Benchmarking the benchmarks–Comparing the accuracy of Data Envelopment Analysis models in constant returns to scale settings. *Eur. J. Oper. Res.* **2020**, *285*, 1042–1057. [CrossRef]
56. Wimmer, S.; Finger, R. A note on synthetic data for replication purposes in agricultural economics. *J. Agric. Econ.* **2023**, *74*, 316–323. [CrossRef]
57. Faisal, M.; Hutson, G.; Mohammed, M. Synthetic NEWS Data. 2022. Available online: https://nhs-r-community.github.io/NHSRdatasets/articles/synthetic_news_data.html (accessed on 21 June 2023).
58. Krivonozhko, V.E.; Utkin, O.B.; Safin, M.M.; Lychev, A.V. On some generalization of the DEA models. *J. Oper. Res. Soc.* **2009**, *60*, 1518–1527. [CrossRef]
59. Ali, A.I.; Seiford, L.M. Computational Accuracy and Infinitesimals In Data Envelopment Analysis. *INFOR Inf. Syst. Oper. Res.* **1993**, *31*, 290–297. [CrossRef]
60. Podinovski, V.V.; Bouzdine-Chameeva, T. Solving DEA models in a single optimization stage: Can the non-Archimedean infinitesimal be replaced by a small finite epsilon? *Eur. J. Oper. Res.* **2017**, *257*, 412–419. [CrossRef]
61. Charnes, A.; Cooper, W.W.; Thrall, R.M. A structure for classifying and characterizing efficiency and inefficiency in Data Envelopment Analysis. *Oper. Res. Lett.* **1991**, *2*, 197–237. [CrossRef]
62. Charnes, A.; Cooper, W.; Golany, B.; Seiford, L.; Stutz, J. Foundations of data envelopment analysis for Pareto-Koopmans efficient empirical production functions. *J. Econ.* **1985**, *30*, 91–107. [CrossRef]
63. Andersen, P.; Petersen, N.C. A Procedure for Ranking Efficient Units in Data Envelopment Analysis. *Manag. Sci.* **1993**, *39*, 1261–1264. [CrossRef]
64. Førsund, F.R. Weight restrictions in DEA: Misplaced emphasis? *J. Product. Anal.* **2013**, *40*, 271–283. [CrossRef]
65. Farrell, M.J. The measurement of productive efficiency. *J. R. Stat. Soc.* **1957**, *120*, 253–281. [CrossRef]
66. Farrell, M.J.; Fieldhouse, M. Estimating efficient production functions under increasing returns to scale. *J. R. Stat. Soc.* **1962**, *125*, 252–267. [CrossRef]
67. Thanassoulis, E.; Allen, R. Simulating weight restrictions in data envelopment analysis by means of unobserved DMUs. *Manag. Sci.* **1998**, *44*, 586–594. [CrossRef]
68. Krivonozhko, V.E.; Førsund, F.R.; Lychev, A.V. Terminal units in DEA: Definition and determination. *J. Prod. Anal.* **2015**, *43*, 151–164. [CrossRef]
69. Krivonozhko, V.E.; Førsund, F.R.; Lychev, A.V. On comparison of different sets of units used for improving the frontier in DEA models. *Ann. Oper. Res.* **2017**, *250*, 5–20. [CrossRef]
70. Bougnol, M.L.; Dulá, J.H. Anchor points in DEA. *Eur. J. Oper. Res.* **2009**, *192*, 668–676. [CrossRef]
71. Dulá, J.H.; Thrall, R.M. A Computational Framework for Accelerating DEA. *J. Prod. Anal.* **2001**, *16*, 63–78. [CrossRef]
72. Bessent, A.; Bessent, W.; Elam, J.; Clark, T. Efficiency Frontier Determination by Constrained Facet Analysis. *Oper. Res.* **1988**, *36*, 785–796. [CrossRef]
73. Lang, P.; Yolalan, O.R.; Kettani, O. Controlled Envelopment by Face Extension in DEA. *J. Oper. Res. Soc.* **1995**, *46*, 473–491. [CrossRef]
74. Olesen, O.B.; Petersen, N.C. Indicators of Ill-Conditioned Data Sets and Model Misspecification in Data Envelopment Analysis: An Extended Facet Approach. *Manag. Sci.* **1996**, *42*, 205–219. [CrossRef]
75. Rubin, D.B. The Bayesian Bootstrap. *Ann. Stat.* **1981**, *9*, 130–134. [CrossRef]

76. Afanasiev, A.P.; Krivonozhko, V.E.; Lychev, A.V.; Sukhoroslov, O.V. Multidimensional frontier visualization based on optimization methods using parallel computations. *J. Glob. Optim.* **2020**, *76*, 563–574. [CrossRef]
77. Koch, T.; Berthold, T.; Pedersen, J.; Vanaret, C. Progress in mathematical programming solvers from 2001 to 2020. *EURO J. Comput. Optim.* **2022**, *10*, 100031. [CrossRef]