A computational study of DEA with massive data sets.

J.H. Dulá

Virginia Commonwealth University

Richmond, VA 23284

FAX: (804) 828 1602

`jdula@vcu.edu`

April 2006

Abstract. Data Envelopment Analysis (DEA) is computationally intensive. This work answers conclusively questions about computational performance and scale limits of the standard LP-based procedures currently used. Examples of DEA problems with up to 15K entities are documented and it is not hard to imagine problem size increasing as new more sophisticated applications are found for DEA. This work reports on a comprehensive computational study involving DEA problems with up to 100K DMUs. We explore the impact of different LP algorithms including interior point methods as well as accelerators such as advanced basis starts and DEA specific enhancements such as "restricted basis entry" (RBE). Our results demonstrate that solution times behave close to quadratically and that massive problems can be solved efficiently. We propose ideas for extending DEA into a data mining tool.

Key Words: Data Envelopment Analysis (DEA), Linear Programming, Convex Analysis.

A computational study of DEA with massive data sets.

J.H. Dulá

Statement of Scope and Purpose. This is a comprehensive and definitive study of computations in DEA using current practices and massive data sets. The purpose is to make determinations about computational requirements for DEA analyses now and in the future. We introduce the concept of DEA as a data mining tool.

Abstract. Data Envelopment Analysis (DEA) is computationally intensive. This work answers conclusively questions about computational performance and scale limits of the standard LP-based procedures currently used. Examples of DEA problems with up to 15K entities are documented and it is not hard to imagine problem size increasing as new more sophisticated applications are found for DEA. This work reports on a comprehensive computational study involving DEA problems with up to 100K DMUs. We explore the impact of different LP algorithms including interior point methods as well as accelerators such as advanced basis starts and DEA specific enhancements such as "restricted basis entry" (RBE). Our results demonstrate that solution times behave close to quadratically and that massive problems can be solved efficiently. We propose ideas for extending DEA into a data mining tool.

Key Words: Data Envelopment Analysis (DEA), Linear Programming, Convex Analysis.

1.0 Introduction. Data Envelopment Analysis (DEA), as originally proposed by Charnes *et al.* in [5], is a non-parametric frontier estimation methodology for evaluating relative efficiencies and performance of a collection of related comparable entities (called Decision Making Units or DMUs) in transforming inputs into outputs. DEA's domain can be any group of entities characterized by the same set of multiple attributes. DEA belongs in the OR/MS toolbox because it provides a quantitative approach to obtain useful information about efficiency and performance of firms, organizations, and all sorts of functionally similar, somewhat autonomous, operating units.

The data for a DEA study consists of n entities (DMUs) each characterized by an m-dimensional vector of the attributes' magnitude. A DMU's efficiency is determined by the geometric location of its data point relative to a boundary region of the production possibility set, an m-dimensional polyhedral set defined by constrained linear operations on the data set. Determining the location of a point with respect to the boundary of a polyhedral set can be achieved with linear programming. The dimensions of the original DEA linear programs formulated by Charnes, *et al.* [5] are essentially the dimensions of the n by m matrix defined by the data. Since each data point needs to be processed once, the original DEA procedure of Charnes, *et al.* [5] requires the application of n linear programs. Although LPs can be solved in polynomial time, the repeated solution of LPs becomes computationally intensive and time consuming, especially when large data sets are involved.

There are several instances in the literature of DEA applications involving large data sets. In one of the first DEA computational studies involving large data sets, Barr and Durchholz [3] report results from banking using 8,000 DMUs. At the time, a direct implementation of an enhanced version of the standard algorithm on a 16MHz Sequent Symmetry S81B computer with 32MB of internal storage took nearly 20 hours to solve. The Siems and Barr [17] study of U.S. banks' efficiency applied DEA to 11,397 banks in a model using eight attributes. Wilson and Wheeloc [18] also worked with bank data and solved DEA problems with 15,000 DMUs and eight attributes. These works represented, at their time, DEA's computational limits.

This paper presents the results of a comprehensive DEA computational study. Simply testing and reporting limits in DEA computations using current hardware and software would be useful but the results would quickly date the paper. Our broader aim is to systematically investigate computational performance of the standard DEA algorithm and the impact of different key properties of a DEA data set. From this work we will be able to predict the computational demands and operational range of a DEA analysis based on the procedure and the characteristics of data. For the algorithmic investigation, we report on how standard DEA algorithms perform and the impact of different accelerators and enhancements. To understand how properties of the data affect performance, we investigate the role of three important data set characteristics: "cardinality" (number of DMUs), "dimension" (number of attributes: i.e., inputs plus outputs), and "density" (proportion of efficient DMUs). Many of our conclusions are independent of the technology and will allow us to plan and predict DEA analyses as the demands on this methodology become more sophisticated, as data sets grow larger, and as DEA breaks out into the realm of data mining.

2.0 Computations in DEA. DEA studies with data sets with more than a couple dimensions and just a few DMUs require a computer. This immediately suggests that there is an aspect of DEA concerned with its interactions with computers and algorithms. This is indeed an area of interest in DEA.

There is a choice of computational procedures for performing DEA. By far the most widely known and implemented is one based on a direct application of the definition of efficiency originating with the articles by Charnes, *et al.* [5], Banker, *et al.* [2], and Charnes, *et al.* [7]. In these works, efficiency is established by way of the solution of an LP formulated using the entire data set. Several factors determine the LP formulation. For example, there are four basic "returns to scale" assumptions: *constant* (CRS), *variable* (VRS), *increasing* (IRS), and *decreasing* (DRS). Each requires special linear constraints or restrictions on variables. Another determining factor is the type of benchmarking analyses required from the model. The original LP in Charnes, *et al.* [5] provides benchmarking recommendations for attaining efficiency based on either decreasing inputs or increasing outputs. Such *oriented* LPs require a decision about whether benchmarking will result from decreased inputs or increased outputs. Other objectives for a DEA LP formulation might be to arrive at a conclusive classification of the DMUs in the model as with the *additive* forms of Charnes, *et al.* [6] which has evolved in scope and utility to the slack-based generalizations of Cooper, *et al.* [19], and Tone [20]. Yet other possibilities include new "unoriented" LP formulations as in Bougnol, *et al.* [4]. As with any computations involving LPs, there is always the choice of solving one in a primal-dual pair. In DEA the primal and the dual are known as the *multiplier* and *envelopment* forms.

All LP formulations are a function of a particular DMU about which we need to determine its efficiency classification; we say that the LP "scores" this DMU. The procedure based on solving one LP for each of the DMUs using the entire data set will be referred to as *standard*. This is a formal statement of the standard procedure:

$\boxed{\text{The Standard DEA Procedure.}}$

```
For $j = 1$ to $n$ Do:
        Step 1. $j^* \leftarrow j$.
        Step 2. Solve LP to score DMU $j^*$.
        Step 3. Classify DMU $j^*$.
Next $j^*$.
```

This simple description for the standard DEA procedure can mask an onerous computational task. The size of the LPs solved in the standard procedure are essentially the size of the data matrix: $m \times n$ in the CRS envelopment LP and $n \times (m + 1)$ in the VRS multiplier LP. Every iteration requires the solution of a dense, high aspect ratio, LP. The LP is modified each iteration. The modifications depend on whether it is the primal or dual. Multiplier forms require that the objective function and, in the case of oriented formulations, one constraint be updated with the values for the DMU being scored. In the envelopment form this corresponds dually to changes in the right-hand side and, when oriented, one column of the LP. The standard procedure for DEA is widely used in practice. It is easy to implement manually for small problems and can be coded directly in a language such as Visual Basic for Applications (VBA) when the data is in a spreadsheet.

Enhancements and improvements for the standard approach are possible. Besides all that is known outside DEA for improving LP performance (e.g., multiple pricing, product forms, hot starts, etc.), there are techniques that exploit the special structure of DEA LPs. Perhaps the two best known are Reduced Basis Entry (RBE) and Early Identification of Efficient DMUs (EIE) (see Ali [1]). Both ideas are a consequence of the same result about DEA LPs; namely, that a

DEA LP optimal basis is defined only by the data points of efficient DMUs; inefficient DMUs play no role in defining the optimal solution and it makes no difference whether they are present or absent from the LP coefficient matrix. The idea of RBE is to omit inefficient DMUs from subsequent LP formulations as they become known. This idea is easy to implement as LPs are iteratively formulated and solved. The systematic application of this approach reduces the size of the LPs as the procedure progresses. EIE simply states that if a DMU's variable appears in a basis of an optimal solution of an envelopment LP, or its constraint is an equality at optimality in a multiplier form, then we have advance knowledge it is efficient. EIE appears to have much less of an impact on improving performance than RBE. This is due to the fact that the proportion of efficient DMUs is relatively small compared to the number of data points, especially in large data sets. Another factor is that a relatively small subset of the efficient DMUs seem to have a preponderant presence in optimal bases. Barr and Durchholz [3] tested these two techniques together and report a significant impact on reducing computation times. They also conclude that most of the impact on improvement is due to RBE.

2.1 A note about preprocessors. Several preprocessing ideas have been proposed for DEA. A preprocessor is effective when it conclusively determines the status of one or more DMUs without having to pay the full computational price of an LP solution. An example of a preprocessor is simple sorting in the VRS model. Unique maximum and minimum attribute values over the entire data set can correspond to efficient DMUs. As many as m different efficient DMUs in this returns to scale model can be identified this way. A sorting is equivalent to the translation of a hyperplane that is parallel to one of the axes in the attribute space. This suggests other types of preprocessors based on translating and rotating hyperplanes. Preprocessors based on translating or rotating hyperplanes involve only inner products. A more detailed discussion on sorting in DEA can be found in Dulá [22] and a more complete study about their impact on DEA will be the topic of another work.

2.2 A note about alternative procedures for DEA. For the sake of completeness, we will briefly discuss other procedures available for DEA.

- The first procedure is based on combining RBE with data partitioning schemes. The idea applies the principle that if an entity is inefficient with respect to a subset of the entities, it will be inefficient with respect to any superset. An implementation consists of partitioning the data set into uniformly sized "blocks" and independently applying standard DEA procedures to them to identify the inefficient entities within blocks. Since the procedure can be repeated with a new set of blocks composed of entities with unknown status until a final single block is processed, it becomes an effective scheme for DEA. All entities must be scored in a second phase using LPs composed of the entities which survived the culling; hopefully, a much smaller

LP than would otherwise be used in the standard approach. It may be possible, however, that more than n LPs will have to be solved. The idea originates in the paper by Barr, *et al.* [3]. They observe that the performance of such "hierarchical decomposition" schemes is affected by the size of the initial and intermediary blocks; this requires experimental tuning. Their tests indicate substantial time reductions once these parameters have been identified.

- Another idea is an adaptation of procedures used in computational geometry to solve the *frame* problem, a version of the convex hull problem. This is the problem of identifying the extreme points of a finitely generated polyhedral set. The frame problem is closely related to the problem of identifying efficient DMUs in DEA (Dulá and López [11]). The algorithm builds the production possibility set one efficient DMU at a time. As with the standard approach, exactly n LPs will be solved but the size of these LPs begins small and their final dimension is determined by the total number of efficient DMUs in the data set.

A comparison between the standard procedure, hierarchical decomposition and frame-based algorithms is the topic for a sequel. The new algorithms are not in widespread use. The scope of this study is to test the standard DEA procedure. It will report on the impact of LP algorithms, accelerators and enhancements such as advanced reoptimizers, hot starts, and RBE.

3.0 DEA Model and Problem Suite. Computational testing required decisions about the DEA LP formulation, the problem suite, and the working baselines to allow meaningful comparisons.

3.1 The DEA LP Formulation. We elected to use the variable returns (VRS), additive, envelopment, LP. The (VRS) formulation was introduced by Banker, *et al.* [2] and is perhaps the most widely used model in DEA applications. It is also interesting because, for a given DEA data set, it is the model that produces the largest set of efficient DMUs and this is a superset for all other returns to scale assumptions (see Dulá and Thrall [12]). The additive model is kin to the other "slack-based" DEA LPs which are in their own right interesting and important. The additive form offers the advantage of a conclusive determination of efficiency without the use of the "non-Archimedean" constant that originally appeared in Charnes, *et al.* [5]. Conclusive classification provides a cleaner implementation of the standard algorithm for DEA since weakly efficient DMUs have no special role and are simply classified as inefficient. When it comes to the decision between the envelopment or multiplier LPs, the decision follows the lesson taught in elementary LP courses: it is better to solve a problem with fewer rows than columns; i.e., use the envelopment LP.

The data domain for a DEA study is the set $\{a^1, \ldots, a^n\}$ of the n data points; one for each DMU. Each data point has m components composed of two types, those pertaining to m_1 inputs, $0 \neq x^j \geq 0$, and those corresponding to m_2 outputs, $0 \neq y^j \geq 0$. We organize the data in the following way:

$$a^j = \begin{bmatrix} -x^j \\ y^j \end{bmatrix}; \qquad j = 1, \ldots, n. \tag{1}$$

The DEA variable returns, additive, envelopment, LP formulation is given next:

$$\begin{array}{lcccc} \max\limits_{\lambda_j \geq 0, \mathcal{S} \geq 0} & & \langle \mathbf{e}, \mathcal{S} \rangle & & \\ \text{s.t.} & \sum\limits_{j \neq j^*} a^j \lambda_j & - \quad I_m \mathcal{S} & = & a^{j^*}, \\ & \sum\limits_{j \neq j^*} \lambda_j & & = & 1; \end{array} \tag{2}$$

where j^* is the index of one of the n DMUs being scored and where $\mathcal{S}$ is the vector of surplus variables. The LP above is a *deleted domain* formulation because the data for the DMU being scored, a^{j^*}, is not included in the coefficient matrix. An advantage of this formulation is that we are able to distinguish between two types of efficiency: "extreme" and "non-extreme efficiency" (Charnes, *et al.* [7], Dulá and Hickman [21]). This distinction provides useful geometrical insights about the production possibility set.

3.2 The Problem Suite. There is no "typical" DEA data set. The few DEA data sets available (e.g., Rousseau and Semple [16]) are useful to verify and compare results but are not suitable, and there are not enough of them, to provide a controlled set of properties on which to base a comprehensive computational study. The absence of a publicly available test bed for standardized DEA computations made it necessary to generate a specialized problem suite.

There is a choice of techniques for generating synthetic data. One is to use a Cobb-Douglas production function (Cobb and Douglas [9]). The function allows single output values to be specified as a function of multiple inputs which can be randomly generated creating points on an efficient frontier. A wide range of "shapes" are possible since this depends on the specification of continuous numerical exponents, one for each input. This approach allows the specification of cardinality, dimension, and density of a DEA data set. However, adapting the formula for multiple outputs means the loss of control of final density. This can make generating useful data sets with specific densities unpredictable and laborious.

An effective approach to produce synthetic data while controlling the three data set characteristics is to randomly generate the points' components directly. There are several ways to do this and one provides complete uniformity for the variable returns model. It is to generate points uniformly distributed on the boundary of one of the 2^m orthants of the unit spheroid in m dimensions.

(Notice that this is different from generating each point's dimension uniformly and independently, and scaling to position it on the surface of the unit spheroid since this does not achieve uniformity.) Given a specification for cardinality, dimension, and density, the procedure generates the proper number of points uniformly distributed on an orthant of the unit spheroid in $\Re^m$. Such points would be necessarily efficient for an input-output assignment determined by the orthant. The balance of the points is randomly generated in the same way but each point is scaled by a uniformly distributed random coefficient in the interval $(0, \alpha)$; $\alpha < 1$. This is the procedure used in this study.

There are several advantages to generating such uniform data besides the high degree of control. The symmetry and regularity of the data along with the minimal parametric dependence make the results a baseline for comparison within the study and, later, with future works. Another advantage for eschewing simulated production functions is that the insights from the model can apply to extensions of DEA such as the identification of geometric outliers in data mining as discussed below in Section 7. A final feature is the flexibility to reuse the data sets by re-assigning input/output designations. Although this has an impact on the final efficiency density, the 2^m different input-output designations for the same data set offer the possibility of generating many new problems. A Laplacian argument for uniformity and symmetry is that it is a reasonable assumption in view of no known typical distortions of the production possibility set or distribution of data points.

The data sets for the problem suite in this study are diverse with a focus on very large cardinalities. The five cardinalities in the sets are $n = 10,000$, $n = 25,000$, $n = 50,000$, $n = 75,000$, and $n = 100,000$. The author is not aware of any published DEA studies involving more than 15,000 DMUs. At these scales, we can learn much about DEA computations and about the tools and techniques to perform these computations. With such massive data sets we can explore the limits and performance of DEA procedures. We can also observe more clearly the relations among the different parameters of interest. Typical DEA data sets have between $m = 5$ and $m = 20$ attributes (inputs plus outputs). In this study we use data sets with $m = 5$, $m = 10$, $m = 15$, and $m = 20$ attributes. Finally, there is scant evidence, and no systematic study, about the nature of the density of efficient DMUs in DEA. In applications, it appears that densities tend to be low. For example, Barr and Durchholz [3] report a density of less than 1% on a study involving more than 8000 DMUs. We might expect in applications that density will tend to be higher in data sets with higher dimensions but lower as the cardinality increases. As we will see, density affects computations. Therefore, in order to get an accurate sense of the impact of density on computations we generated data sets with "Low" ($d \approx 1\%$), "Medium" ($d \approx 10\%$), and "High" ($d \approx 25\%$) densities.

We will see how the inherent properties of cardinality, dimension, and density affect computations in DEA. We explore the impact of generally available, controllable, factors such as the LP algorithm used, e.g., primal or dual simplex, or interior point methods; LP related accelerators such as "hot starts"; and DEA-specific enhancements such as RBE, in these computations. In the following section we formalize the objectives of the study.

4.0 Objectives of the Study. The task at hand is to understand how different factors affect DEA computations in a way that allows the analyst to predict and plan a DEA analysis now and in the future independently of the technology.

The objective of the study was to understand the impact of:

1. Intrinsic Data Set Properties. The three properties that characterize a data set are cardinality, dimension, and density. All three impact computations. A wide range of values for these parameters provide an accurate insight into their effect.

2. Massive Data Sets. The use of massive cardinalities make this work a contribution to large scale DEA. This study solves the largest DEA problems to date.

3. LP Algorithms. DEA LPs have special structure and properties. They are fully dense and tend to have an extreme aspect ratio; e.g., $5 \times 100,000$. The role of different LP algorithms that can be applied to execute DEA, primal simplex, dual simplex, and barrier methods, is a relevant question. Besides questions of speed, there is a special interest in exploring barrier methods in DEA. It has long been a question how the modern barrier methods apply to DEA. Barrier methods generate "centered" non-extreme solutions that may have special use in DEA since such solutions reduce the problem of multiple weight vectors. A solution based on an analytical center from a barrier method may provide useful information about the weights of the attributes. Our work will help analysts weigh different factors in planning computations.

4. LP accelerators. DEA LPs differ from each other only slightly throughout a full DEA implementation. We explore the impact of reoptimization; i.e., "hot starts", on DEA. Hot starts incorporate knowledge from a previous optimal solution when the LP is modified. It is clear that much can be gained from being able to use information about a previous optimal solution.

5. DEA enhancements. It has been known that "Reduced Basis Entry" (RBE) can reduce computational times substantially (Ali [1] and Barr and Durchholz [3]). What has not been known is how this benefit is affected by the properties of the data set. Also unknown is how

this enhancement interacts with other accelerators. This computational study gives a precise assessment of the impact of RBE on DEA computations.

5.0 Technology and Implementation. Conclusions of this study are intended to rely minimally on hardware and software. Implementation, however, requires specific technology. The two principal technological components are the computing platform and the LP solver.

Computations were performed on a Intel Pentium CPU running a Windows XP operating system. Unless otherwise noted, runs were executed on a P4-478 processor running at 2.53 MGz with 1024 MB of core memory.

Of the many solvers available to solve the DEA LPs we chose ILOG CPLEX [14]. Unless otherwise noted, version 8.0 was used. CPLEX is a commercial product sold by ILOG S.A. The form of CPLEX used was the *callable library*; a collection of functions that can be called from inside another program. Besides its recognized speed and stability, CPLEX offers many features that were ideally suited for this study. One of the main features relating to DEA is its ease to update and reoptimize LPs. As Bixby [23] recounts in the history of his creation:

> "It had to be a code that made it easy to handle the kinds of operations that arose in a context in which it was natural to begin with a model instantiated in one form followed by a sequence of problem modifications (such as row and column additions and deletions and variable fixings) interspersed with resolves. These needs were among the fundamental motivations behind the development of the callable-library version of CPLEX code."

CPLEX also offers a variety of LP solvers including separate primal and dual algorithms as well as a barrier method. The program that controlled the CPLEX 8.0 callable library and implemented the logic was written in Fortran 90 and compiled with *Compaq Visual Fortran 6.5.*

To take full advantage of features in CPLEX the DEA LP was recast in the following manner:

$$\begin{array}{cccccc} \max\limits_{\lambda_j \geq 0, \lambda_{j^*}=-1, \mathcal{S}\geq 0} & & & \langle \mathbf{e}, \mathcal{S} \rangle & & \\ \text{s.t.} & \sum\limits_j a^j \lambda_j & - & I_m \mathcal{S} & = & 0, \\ & \sum\limits_j \lambda_j & & & = & 0. \end{array} \qquad (3)$$

We achieve an equivalent formulation as the original deleted domain LP in (2) by iterating over $j^* = 1, \ldots, n$, and temporarily setting $\lambda_{j^*} = -1$ at each iteration.

This approach has several advantages. One is that changing the variable restriction for the DMU being scored can be done with a simple call to the `'CPXchgbds'` CPLEX function. Another advantage is that we avoid potential problems with "induced degeneracy"; the type of degeneracy

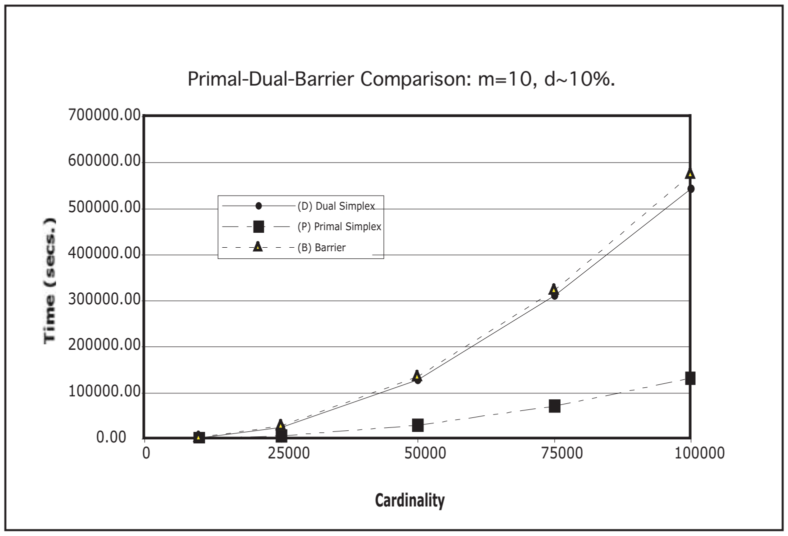

Figure 1.

commonly found in DEA due perhaps to the duplication of one of the columns in the right-hand side (see Barr and Durchholz [3] and Charnes, Rousseau and Semple [8] for more about the problem of degeneracy and cycling in DEA). Using CPLEX offered other advantages. The CPLEX operation `‘CPXdelcols’` efficiently removes an inefficient DMU's variable from the computations whenever RBE is activated. Deciding which LP algorithm to adopt reduces to selecting the proper CPLEX function: `‘CPXprimopt’`, `‘CPXdualopt’`, or `‘CPXbarropt’` for the primal simplex, dual simplex, and barrier method, algorithms. Note that the two equivalent deleted domain formulations, (2) and (3), generate infeasible LPs when a DMU is extreme-efficient. This infeasibility is conclusively identified by flags returned by the CPLEX solver.

6.0 Results and Analysis. The large number and variety, as well as the massive scale, of the DEA LPs solved for this study reveal much about DEA computations. This section reports on these findings.

It makes a difference which of the three LP algorithms – primal simplex, dual simplex and barrier methods – is used to solve DEA LPs. These three algorithms were tested to assess their impact on cardinality by fixing dimension and density. Typical results are depicted in Figure 1. This figure compares cardinality vs time for these three algorithms for $m = 10$ and “Medium” density, $d \approx 10\%$, with no enhancements; i.e., neither ‘hot starts’ nor RBE are activated.

An immediate observation from Figure 1 is that the primal simplex LP algorithm, “(P)”, outperforms both the barrier method, “(B)”, of the dual simplex algorithm, “(D)”, for DEA computations. This comparison implemented the Standard Procedure for DEA in its purest (and most naive) form solving n full-data LPs. It takes 132,580 seconds (more than 36 hours) to solve the

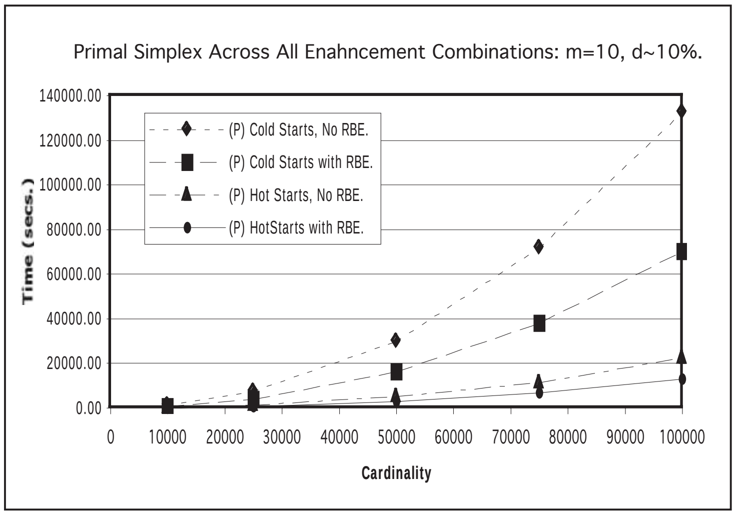

Figure 2.

largest, $n = 100$K, DEA problem with the faster primal simplex. The time values obtained give an idea of current minimum times to perform DEA studies using a pure implementation of the Standard Procedure.

This comparison resolves some issues about the role of interior point methods in DEA computations. The barrier method used is a primal-dual interior point algorithm based on a log-barrier function. The implementation disabled "crossover" (`CPX_PARAM_BARCROSSALG=-1` in CPLEX 8.0); that is, the procedure stopped at the natural analytical center optimal solution and did not proceed to identify an optimal extreme point. Disabling crossover makes the procedure faster. Barrier methods are known to perform better than simplex-based methods in certain prescribed conditions. It is indicated for "large problems, for example, those with more than 1000 rows or columns" (ILOG 2002b). Another indication, however, is that the LPs be sparse with a small number of nonzeroes per column. DEA's fully dense LPs are a contraindication for barrier methods. Another reason why barrier methods are not amenable to DEA computations is the difficulty to implement hot starts. This accelerating technique, as we are about to see, is particularly effective with simplex based methods and simply unavailable for barrier methods (Bixby [23]).

This initial experiment establishes the baseline for our next experiment. Based on the dominance of the primal simplex on DEA computations, we explore next the impact of accelerators and enhancements.

Implementing Restricted Basis Entry (RBE) requires a modification of the Standard Procedure for DEA whereby a structural variable is removed from the envelopment LP formulation whenever the corresponding DMU is found to be inefficient. The impact of this enhancement in DEA computations can be seen in Figure 2. We observe the impact of RBE when we compare the two

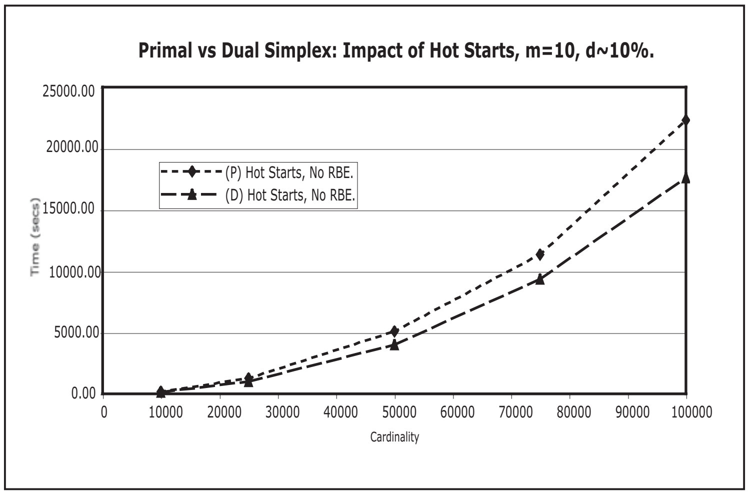

Figure 3.

curves "(P) Cold Starts, No RBE" and "(P) Cold Starts, with RBE" (highest and second highest plots). The top plot represents the performance of a "naive" implementation of the Standard Procedure for DEA. The second depicts the effect of the RBE enhancement. The impact of RBE on DEA computations is substantial reducing times by almost one half. These results square with those reported by Barr, *et al.* [3].

A new LP is solved every iteration of the Standard Procedure for DEA. These LPs, however, do not differ much from each other. In the envelopment form for the additive model, all that needs to be updated from one iteration to the next is the right hand side and, by using formulation (3), the update reduces to a simple setting of the restriction of the variable of the DMU being scored. The parameter `CPX_PARAM_ADVIND=1` in CPLEX 8.0 activates the advanced basis, or "hot starts" option for using information about a previous optimal solution in any of its simplex based procedures. The impact of this accelerator is dramatic. The plot in Figure 2 labeled "(P) Hot Starts, No RBE" (third highest of the four plots) is strong evidence of the powerful impact of reoptimization using previous optimal bases in DEA. For the case of $n = 100,000$, the impact of hot starts is a reduction of more than an 80% in computational time. RBE has an additive effect on the reduction of yet another 40% reduction for this particular problem (plot "(P) Hot Starts, with RBE") suggesting that, in combination, the two features can produce a full order of magnitude reduction in times compared to the naive Standard Procedure.

Advanced basis starts produces an unexpected result when applied using CPLEX's dual simplex. Figure 3 shows that the gain from advanced basis starts is more dramatic when implemented in the dual simplex. For 100k DMUs, when $m = 10, d = 10\%$, the reduction in time when hot starts are implemented in the primal simplex is 83.2%. This reduction for the case of the dual simplex is 96.7%. Interestingly, the impact of RBE is independent of whether the primal or the dual simplex

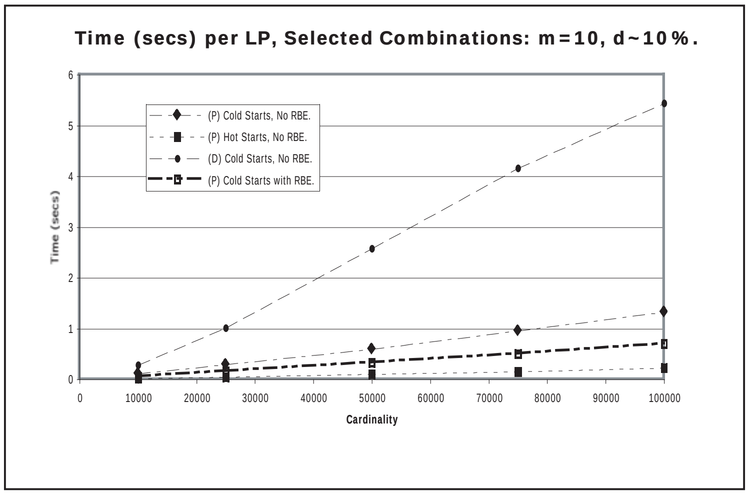

Figure 4.

is being used; in both cases there is a reduction of roughly one half. Recall that hot starts has no impact on barrier methods.

The experiment offers an opportunity to understand better the nature of the relation between cardinality, n, and computation times. We can observe a clear linear relation between time consumed per LP and cardinality for selected primal and dual simplex implementations in Figure 4. These nearly straight-line relations are typical of what was observed across trials involving different algorithms and enhancement variations. Since the procedure requires n iterations, this implies that the solution time is nearly quadratic with respect to cardinality and the actual relation between cardinality, n, and time, t, is approximated by a function of the form $t = cn^2$. The factor c will depend on inherent data attributes such as dimension, m and even density, d. This quadratic shape is verified in Figures 1, 2 & 3. Since RBE makes the procedure sensitive to the number of inefficient DMUs, density is a factor when this enhancement is implemented. Other factors affecting the value of the parameter c include the simplex algorithm being used and, clearly, the hardware and software.

These tests indicate that the quickest way to perform DEA computations using the Standard DEA Procedure is to use the primal simplex. This is good news if the solver available to the analyst is a primal simplex algorithm and accelerators and enhancements are not possible or practical. When it is possible to accelerate with advanced basis starts and enhance with RBE then substantial improvements are likely especially in massive problems. If there is a choice of simplex algorithm and access to hot starts and RBE then the fastest way to perform DEA analysis is to use the dual simplex algorithm in conjunction with the accelerator and enhancement. It is clear that the impact of these three aspects, simplex algorithm, accelerator, and enhancement, is sufficiently

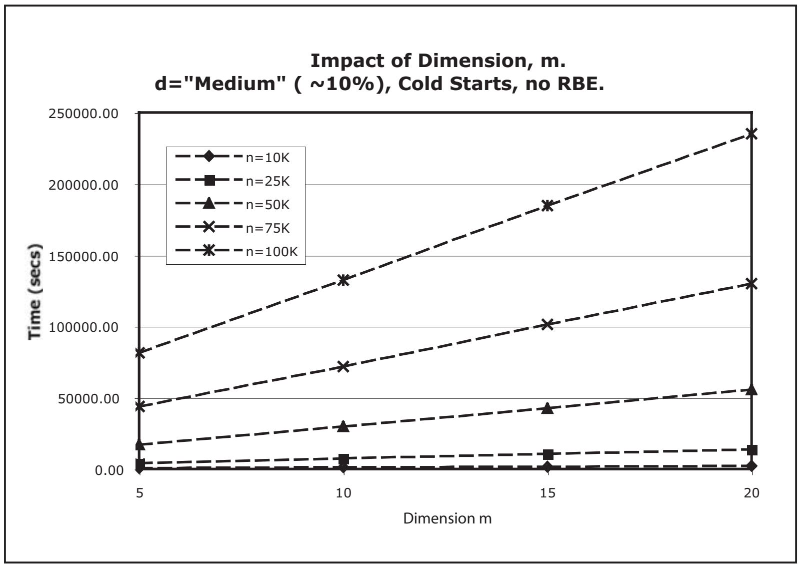

Figure 5.

powerful to justify their use in massive problems. The analyst must be prepared to pay a price for using barrier methods.

We now turn our attention to the effect of the other two basic parameters defining a DEA problem, the dimension, m, i.e., number of inputs plus outputs and the "density", d, i.e., percent of efficient DMUs. Figure 5 contains plots summarizing results of the impact of dimension and and Figure 6 that for density.

Since the early days of LP it has been empirically observed that LP iterations grow linearly with the number of constraints (see Gass [13] p. 44 and Dantzig [10] p. 160). Figure 5 is typical of our results about the relation between dimension, m, and computational time. The specific implementation represented in this chart is for the naive Standard DEA Procedure. Any one of the five plots in this figure makes the relevant point; namely, the relation between dimension and computational time is almost identically linear. The plots in Figure 5 confirm the basic extension of this well established, and always a bit surprising, relation between number of constraints and amount of work needed to solve LPs.

The impact of efficiency density on time is not as stark as with dimension. The four plots in Figure 6 seem to indicate that the density of efficient DMUs in a DEA data set does not have a dramatic impact on computation times. The heights of the four plots attest to the impact of the accelerator or enhancement used to perform the analyses and they confirm what was seen in Figure 2. An adverse effect on computation time from increased density under RBE is evident. RBE causes a reduction in the number of columns of the envelopment LPs as the number iterations increases since structural variables associated with inefficient DMUs are suppressed from

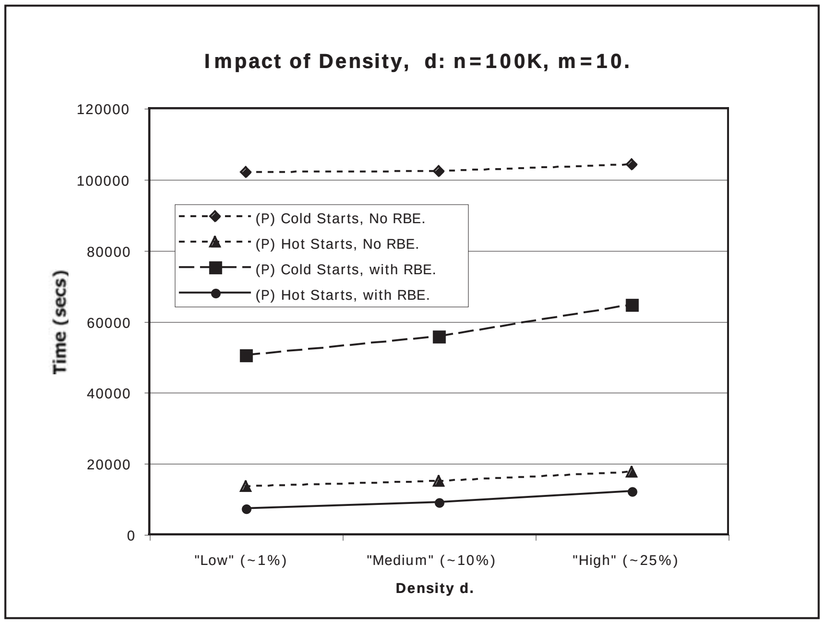

Figure 6.

subsequent operations. We can expect this benefit to be less pronounced as the density increases and this is verified in the two plots in Figure 6: "Cold Starts, with RBE" (triangles), and "Hot Starts, with RBE" (bullets)‡.

This experiment has provided data to answer several questions about the impact of cardinality, dimension and density, on DEA computational performance involving massive data sets as well as LP algorithms

7.0 DEA and Data Mining.

At what point will DEA be applied at the scales tested here, e.g., $n = 100$K, and beyond? It is not difficult to imagine large scale DEA applications in the banking industry involving many tens of thousand of units; the state-of-the-art permits this so we can expect it. A more realistic expectation for applying what we know about DEA theory and computations to truly massive data sets will come from extending the role of this analytical tool beyond its efficiency and production theory paradigm.

DEA derives from fundamental efficiency and productivity theory principles. Its basic premise is that it identifies entities in an empirical frontier that approximates a true but unavailable production frontier. DEA's production possibility set is a proxy for an actual production frontier in

‡ The numerical results reported in Figure 6 were run with CPLEX 6.6 on a P4-630 processor running at 3.00MGz with 1024MB of core memory.

the absence of any knowledge of the parameters that define it. New application domains can be claimed for DEA if it is released from its original moorings.

DEA is about extremes. A DEA study reveals which of the entities are *geometric outliers* in the sense of extreme elements or non-extreme boundary elements in one of the four production possibility sets which are nothing more than polyhedral sets generated by the data points in $\Re^m$. These geometric objects range from cones in the CRS case to combinations of convex hulls and recession directions in the VRS, IRS, and DRS cases. The geometric outliers are called "extreme-efficient" or "efficient non-extreme" DMUs in DEA. Other points are identified including points on receding faces ("weak efficient" DMUs) and interior points ("technically inefficient" DMUs).

There is a natural interest in identifying geometric outliers in multivariate point sets in contexts outside of DEA. Consider the particular federal tax return where the total in charitable contributions or employee deductions is the largest among all returns within a given category. We can imagine that an agency such as the IRS would consider such a return interesting. In the same way, a security agency may focus on the individual who has made the largest number of monetary transfers, or the largest magnitude transfer, to a problematic location on the globe. Such records in a data set are its geometric outliers in one of the dimensions in the sense that they attain extreme values there and finding them reduces to a sorting of records based on the value in that dimension.

Entities operating under strenuous circumstances push the limits in several key dimensions even though no single one attains an extreme value. Such entities may be identified by having extreme values when dimensions are combined. In the examples above, the tax return with the largest <u>sum</u> of the charitable contributions and employee deductions may prove to be interesting; or the individual whose money transfer events and total monetary value of the transfers is the largest when added up may merit closer scrutiny. The record that emerges as a geometric outlier based on this two-dimensional analysis using these simple criteria is just one of, possibly, many that can emerge if the combination of the two values are weighted differently. All such points are, in the same sense, geometric outliers and all would be interesting for different reasons.

If these simple one and two dimensional examples can persuade of a natural importance in identifying geometric outliers, then the case for identifying geometric outliers among n entities in the general case of m dimensions with wide ranging possibilities for the weights is proportionally compelling.

Using the simple sum of the attribute values to identify an entity means we place equal importance on each attribute in the identification criterion. Modifying attributes' weights results in different weighted sums and may be used to reflect different priorities or concerns. Allowing weights to be negative can be used to shift the emphasis from larger magnitudes, as in the example above, to the case where the interest is focused on smaller values. As we know from DEA, weighted sums are maximized by different entities depending on the weights. The questions that arises in this context is:

Given a specific entity, j^*, is there a set of m weights that will make its weighted sum of the attribute values the maximum from among all entities' weighted sums?

An affirmative answer means that entity j^* is "extreme" in a sense reflected by the nature of the weights. A negative answer to the question means the entity is never extreme and that other entities manage to attain larger weighted sums no matter what weights are used.

The question above can be answered by solving a linear program which is essentially the same in terms of dimensions and nature of the data, as a VRS DEA multiplier LPs (Dulá 2006). Solving it, or its dual, requires the same effort as solving DEA LPs. As with DEA, a complete implementation requires testing n points; i.e., solving that many LPs. Also equivalent is any accelerator or enhancement such as RBE and hot starts. Therefore, the lessons learned about solving large scale DEA problems apply directly to mining very large data sets for geometric outliers.

Geometric outliers as introduced here are a purely geometric concept. The same reasons that compel their identification in DEA can be expected to apply to other point sets not necessarily connected to production and efficiency analyses. It is not difficult to imagine mining several hundreds of thousands of credit-card records for potentially profitable (or disastrous) customers, fraud, theft, etc. by identifying the different geometric outliers that emerge from different models.

8.0 Concluding remarks. There has not been up to now a comprehensive study of DEA computations that looks at all the factors that make an impact. This study was performed to address this gap.

This computational study provides several insights. The LP algorithm used has an effect on solution times with the primal simplex being the fastest unless enhancements and/or accelerators are used in which case the dual simplex prevails. Our testing shows that barrier methods are not practical but that certain sifting schemes are effective. We understand better the impact of the three important DEA data set attributes on computations: cardinality (number of DMUs), dimension (inputs plus outputs), and density (proportion of efficient DMUs). The studies show that the time to perform a DEA study using the traditional method behaves nearly quadratically with respect to cardinality; that time is affected linearly by dimension; and that density also has a direct effect. Implementations of LP accelerators and DEA enhancements show that these have a dramatic effect reducing times by nearly one order of magnitude if applied simultaneously. Finally, our tests provide a better understanding of the scales at which DEA can be applied now and in the future.

Studies with DEA can be predicted to go from the traditional one-time, cross-sectional, approach to dynamic, instantaneous update, tracking of DMUs. Such applications for DEA are not hard to conceive especially when we look around and find already highly complicated financial, social, and technical systems and processes steadily growing and becoming more sophisticated. DEA can also be interpreted as a data mining tool to identify geometric outliers with applications beyond efficiency and productivity. These are new frontiers for this nonparametric frontier estimation tool and this work provides a view into the new computational demands in its future.

Acknowledgements. This work was partially funded by a VCU Summer Research Grant. I am also grateful to Ike Patterson for helping with Fortran/CPLEX interface issues.

References.

1. Ali, I. Streamlined computation for data envelopment analysis. *European Journal of Operational Research,* 1993;64: 61-67.
2. Banker, R.D., A. Charnes, and W.W. Cooper. Some models for estimating technological and scale inefficiencies in data envelopment analysis. *Management Science*, 1984;30(9): 1078–1092.
3. Barr, R.S. and M.L. Durchholz. Parallel and hierarchical decomposition approaches for solving large-scale Data Envelopment Analysis models. *Annals of Operations Research*, 1997;73: 339-372.
4. Bougnol, M.-L., J.H. Dulá, D. Retzlaff-Roberts, and N.K. Womer. Nonparametric frontier analysis with multiple constituencies. *Journal of the Operational Research Society*, 2005;56: 252-266.
5. Charnes, A., W.W. Cooper, and E. Rhodes. Measuring the efficiency of decision making units. *European Journal of Operational Research,* 1978;2(6): 429-444.
6. Charnes, A, W.W. Cooper, B. Golany, L. Seiford, and J. Stutz. Foundations of data envelopment analysis for Pareto-Koopmans efficient empirical production functions. *Journal of Econometrics* 1985;30; 91–107.
7. Charnes, A., W.W. Cooper, and R.M. Thrall. A structure for classifying and characterizing efficiency and inefficiency in data envelopment analysis. *The Journal of Productivity Analysis*, 1991;2: 197–237.
8. Charnes, A., J. Rousseau, and J. Semple. An effective non-Archimedean anti-degeneracy/ cycling linear programming method especially for data envelopment analysis and like methods. *Annals of Operations Research*, 1993;47; 271-278.
9. Cobb, C.W. and P.H. Douglas. A theory of production. *American Economic Review, Suppl.*, 1928: 139–165.
10. Dantzig, G.B. Linear Programming and Extensions, Princeton University Press, Princeton, N.J., 1963.
11. Dulá, J.H. and F.J. López. Algorithms for the frame of a finitely generated unbounded polyhedron. To appear in *INFORMS Journal on Computing.*
12. Dulá, J.H. and R.M. Thrall. A computational framework for accelerating DEA. *J. of Productivity Analysis.* 2001;16(1): 63–78.
13. Dulá, J.H. Mining Nonparametric Frontiers. Manuscript under review.

14. Gass, S. *Linear Programming*, McGraw-Hill, New York, 1958.

15. ILOG. *ILOG CPLEX 8.0 User's Manual*, ILOG S.A., Gentilly, France, 2002.

16. ILOG. *ILOG Connection*, Summer 2002, No. 20, ILOG S.A., Gentilly, France. `http://www.ilog.com/corporate/connection/summer_2002/cplex8.htm`.

17. Rousseau, J.J. and J.H. Semple. Radii of classification preservation in Data Envelopment Analysis: a case study of 'Program Follow-Through'. *Journal of the Operational Research Society*, 1995;46: 943–957.

18. Siems, T.F. and R.S. Barr. Benchmarking the productive efficiency of U.S. Banks. *Financial Industry Studies*, Federal Reserve Bank of Dallas, December 1998. `http://www.dallasfed.org/htm/pubs/fis.html`.

19. Wilson, P.W. and D.C. Wheeloc. Why do banks disappear? The determinants of US bank failures and acquisitions. *Review of Economics and Statistics*, 2000;82: 127–138.

20. Cooper, W.W., K.S. Park, and J.T. Pastor. RAM: A range adjusted measure of inefficiency for use with additive models and relations to other models and measures in DEA. *Journal of Productivity Analysis*, 1999;11(1): 5–42.

21. Tone, K. A Slacks-based Measure of Efficiency in Data Envelopment Analysis. *European Journal of Operational Research*, 2001;130: 498-509.

22. Dulá, J.H. and B. L. Hickman. Effects of excluding the column being scored from the DEA envelopment LP technology matrix. *Journal of the Operational Research Society*, 1997;48: 1001–1012.

23. Dulá, J.H. Computations in DEA. *Pesquisa Operacional*, 2002;22(2): 165-182.

24. Bixby, R.E. Solving real-world linear programs: a decade and more of progress. *Operations Research*, 2002;50(1): 3–15.