Evaluating Popular Statistical Properties of Incomplete Block Designs: A MATLAB Program Approach

Emmanuel Ikechukwu Mba *, Polycarp Emeka Chigbu and Eugene Chijindu Ukaegbu

Abstract: Evaluating the statistical properties of a semi-Latin square, and in general, an incomplete block design, is vital in determining the usefulness of the design for experimentation. Improving the procedures for obtaining these statistical properties has been the subject of some research studies and software developments. Many available statistical software that evaluate incomplete block designs do so at the level of analysis of variance but not for the popular A-, D-, E-, and MV-efficiency properties of these designs to determine their adequacy for experimentation. This study presents a program written in the MATLAB environment using MATLAB codes and syntaxes which is capable of computing the A-, D-, E-, and MV-efficiency properties of any \((n \times n)/k\) semi-Latin square and any incomplete block design via their incidence matrices, where \(N\) is the number of rows and columns and \(k\) is the number of plots. The only input required for the program to compute the four efficiency criteria is the incidence matrix of the incomplete block design. The incidence matrix is the binary representation of an incomplete block design. The program automatically generates the efficiency values of the design once the incidence matrix has been provided, as shown in the examples.

Keywords: canonical efficiency factors; incidence matrix; information matrix; MATLAB codes and syntaxes; simple contrasts

1. Introduction

An \((n \times n)/k\) semi-Latin square, according to Bedford and Whitaker [1], Bailey and Royle [2] and Soicher [3], has \(N^2\) cells (called blocks) in a square array, where each block has \(k\) plots, and there are \(Nk\) treatments which are allocated to plots such that each treatment occurs once in each row and once in each column (see also References [2,3]). Bailey [4] listed some methods for the construction of semi-Latin squares, such as the inflation method, interleaving method, Trojan method, superposition method, etc. Software such as the nauty package by McKay [5], GRAPE package by Soicher [6] and Traces package by McKay and Piperno [7] aid in the enumeration of semi-Latin squares up to isomorphism classification by exploring the graphical properties of the semi-Latin squares. Bailey and Royle [2] pointed out that results from graph theory can be applied to problems of incomplete block designs and the quotient block design of a semi-Latin square is said to be a regular-graph design.

A semi-Latin square could be referred to as an incomplete block design with the usual parameters, \(t, b, r, k\) and \(\lambda\), where \(b\) is the number of blocks, \(r\) is the treatment replication while \(\lambda\) is the number of times each pair of treatments appears together in the design [2]. The most important statistical property of a block design is the efficiency, and there are many measures of efficiency of a block design, including the popular A-, D-, E-, and MV-efficiencies. According to John [8], incomplete block designs with equal block sizes often differ in the precision with which treatment effects are estimated, and efficiency factors of the incomplete block designs provide useful measures of these differences.

The efficiency factors, in particular, the pairwise and canonical efficiency factors, of incomplete block designs have been the subject of extensive research in the area of statistical
The nauty package is not easily accessible and runs only on a number of systems. The C product of the eigenvalues of \( A \) obtains the isomorphism and automorphism properties of the block design, \( \Sigma \). The nauty package uses the graph properties of semi-Latin squares to obtain the isomorphism and automorphism classifications but does not evaluate the statistical properties of the semi-Latin squares, such as the optimality and efficiency properties. The nauty package is not easily accessible and runs only on a number of systems. The Bailey and Royle program was designed for only \((6 \times 6)\)-2 semi-Latin squares and also uses the orthogonal one-factorizations of the regular graph designs for the construction of the semi-Latin squares on SparcStation ELC, and each graph takes about one minute of CPU time. GRAPE software was developed by Soicher \[6\] in 1993, while the shipped version was developed in 2019. The GRAPE software runs partly on GAP environment and is primarily designed for constructing and analyzing graphs related to finite geometry, groups and designs. The other component runs on McKay's nauty package. According to Soicher \[12\], the GRAPE algorithm performs more slowly and the computer time is not exact. The MATLAB program presented here utilizes simple matrix algebra, which is already part of the MATLAB codes and syntaxes, to obtain the four popular statistical properties, \( A\), \( D\), \( E\), and \( MV\)-efficiencies, of any incomplete block design in about one second of computer time.

The MATLAB R2019a software has an in-built statistics toolbox which is a collection of MATLAB codes and syntaxes for statistical data analysis. The statistics toolbox could analyze incomplete block designs up to analysis of variance tests of hypothesis but does not contain the required codes to obtain the \( A\), \( D\), \( E\), and \( MV\)-efficiencies of incomplete block designs. In this study, we provide a MATLAB program for the computation of the \( A\), \( D\), \( E\), and \( MV\)-efficiencies of incomplete block designs from their canonical and pairwise efficiency factors. The program, presented in the Appendix A, is capable of accurately computing the \( A\), \( D\), \( E\), and \( MV\)-efficiencies of any semi-Latin square, and in general, all incomplete block designs. The program requires only input of the binary equireplicate incidence matrix of the block.

2. Materials and Methods
2.1. Algebraic Basis for the Program

Given a semi-Latin square, \( \Xi \), with \( t = nk \) treatments arranged in \( b = n^2 \) blocks, the incidence matrix, \( N \), emanating from \( \Xi \) or that of any other \( t\)-treatment incomplete block design, \( \Sigma \), in \( b \) blocks, is a \( t \times b \) matrix of \( 1\)'s and \( 0\)'s. The \( t \times t \) concurrence matrix of the incomplete block design is a product of \( N \) and its transpose, \( N' \). The concurrence matrix, \( NN' \), is then transformed to the information, \( C = R - \left( \frac{1}{t} NN' \right) \), of order \( t \), where \( R = rI \), \( r \) is the treatment replication in \( N \) and \( I \) is an identity matrix, also of order \( t \). The product of the eigenvalues of \( C \) and the reciprocal of \( r \) gives the \( t-1 \) canonical efficiency factors, \( \omega_i \), \( i = 1, 2, \ldots, t-1 \), of \( \Xi \) or \( \Sigma \), with one of the values of \( \omega_i \) being equal to zero. Conversely, multiply \( C \) by the reciprocal of \( r \) to normalize the information matrix to obtain \( C^* = I - \left( \frac{1}{r_n} NN' \right) \). Then, the canonical efficiency factors, \( \omega_i \), are the \( t-1 \) non-zero eigenvalues of \( C^* \).

The \( A\)-efficiency measure of \( \Xi \) or \( \Sigma \) is the harmonic mean of the canonical efficiency factors and is given by \( A = (t-1) \left( \frac{t-1}{\sum_{i=1}^{t-1} \frac{1}{\omega_i}} \right)^{-1} \). The \( D\)-efficiency measure is the geometric
mean of the canonical efficiency factors and is given by
\[ D = \left( \prod_{i=1}^{t-1} \omega_i \right)^{1/1-t} \],
while the E-efficiency is the minimum of the canonical efficiency factors, given by
\[ E = \min(\omega_i) \].
The MV-efficiency measure is obtained from the components of the generalized inverse of \( C \). Let \( C_{inv} \) be the Moore–Penrose generalized inverse of \( C \) and \( \delta_i, i = 1, 2, \ldots, t \), the \( i \)-th treatment effect, with \( \delta \) the least squares estimator of \( \delta \). Then, the treatment contrast is \( (\hat{\delta}_i - \delta_i) \), \( i \neq j \), while the variance from intra-block analysis of the incomplete block design is
\[ \text{var}(\hat{\delta}_i - \delta_i) = \left( \xi_{ii} + \xi_{ij} - 2\xi_{ij} \right) \sigma^2, \]
where \( \xi_{ij} \) is the \((i,j)\)-th element of \( C_{inv} \). The MV-efficiency criterion is given as
\[ \max \{ \text{var}(\hat{\delta}_i - \delta_i) \} \]: see, for example, Reference [11].

2.2. Theoretical Computation of the Efficiency Criteria

Given an incomplete block design, \( \Sigma(t, b, r, k, \lambda) \), the \( t \times b \) incidence matrix, \( N \), has its \((i, j)\) entry as the number of times treatment \( i \) occurs in block \( j \). The column sums of \( N \) are all equal to the block size, \( k \), and the sum of the \( i \)-th row, \( r_i \), is the \( i \)-th treatment replication (that is, the number of times treatment \( i \) occurs overall). If \( r_i \) is the same for all the treatments, then \( \Sigma(t, b, r, k, \lambda) \) is called an equireplicate design. The matrix, \( N \), is made binary by identifying treatment positions in each block by ‘1’ and non-treatment positions by ‘0’. Let \( \Xi \in \Sigma(t, b, r, k, \lambda) \) be the \( (6 \times 6) / 2 \) semi-Latin square, \( \Delta_0 \), of Bailey and Royle [2].

The design, \( \Delta_0 \), is displayed in Figure 1. The input matrix required by this MATLAB program to compute the efficiency criteria of \( \Delta_0 \) is the \( 12 \times 36 \) incidence matrix of \( \Delta_0 \), given by \( N = \)

\[
\begin{array}{cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc}
which is also of order 12. Dividing $C$ by $r$, the normalized information matrix is obtained as $C^r =

\begin{align*}
0.5 & \quad 0 & -0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 \\
0 & 0.5 & 0 & -0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 \\
-0.0833 & 0 & 0.5 & 0 & -0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 \\
-0.0833 & -0.0833 & 0 & 0.5 & 0 & -0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 \\
-0.0833 & 0 & -0.0833 & 0 & 0.5 & 0 & -0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 \\
0 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & 0.5 & 0 & -0.0833 & 0 & -0.0833 \\
-0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & 0.5 & 0 & -0.0833 & 0 & -0.0833 \\
-0.0833 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & -0.0833 & 0 & 0.5 & 0 & -0.0833 \\
0 & 0 & 0 & 0 & -0.0833 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{align*}

The variances of the elementary treatment contrasts for $i \neq j$ are presented as $ttt =

\begin{align*}
0.4394 & \quad 0.0379 & -0.0417 & -0.3712 & -0.3712 & 0.1061 & 0.1061 & -0.4508 & 0.1061 & 0.1061 & -0.4508 & 0.1061 \\
0.4394 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.4508 & 0.1061 \\
0.0379 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & 0.3712 & -0.1061 & -0.4508 & 0.1061 & -0.4508 & 0.1061 \\
0.0379 & 0.0379 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & 0.3712 & -0.4508 & 0.1061 & -0.4508 & 0.1061 \\
0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 \\
0.0379 & 0.0379 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & 0.3712 & -0.0417 & 0.379 & -0.4394 & 1.8674 \\
0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 \\
0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 \\
-0.4394 & 0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 \\
\end{align*}

The canonical efficiency factors of $\Sigma \in \Sigma(t, b, r, k, \lambda)$ are the non-zero eigenvalues of $C^r$, given by $\omega = 1.6340, 1.6340, 3.0, 3.0, 3.36, 3.36, 4.0, 4.0, 4.0, 4.0, 4.0$. Therefore, $A = 0.4909$, $D = 0.5210$, and $E = 0.2723$. The $MV$ is obtained from the contrasts of the Moore–Penrose inverse of $C$, given by $M =$

\begin{align*}
1.8674 & \quad -0.4394 & 0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.0379 & -0.4394 \\
-0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.0379 \\
0.0379 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & 0.3712 & -0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 \\
-0.0417 & 0.0379 & -0.4394 & 1.8674 & -0.4394 & 0.0379 & -0.0417 & 0.3712 & -0.4508 & 0.1061 & -0.3712 & 0.0417 \\
-0.3712 & -0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 & -0.0417 & 0.3712 & -0.4508 & 0.1061 & -0.3712 \\
0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 & -0.0417 & 0.3712 & -0.4508 & 0.1061 \\
-0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 & -0.0417 & 0.3712 & -0.4508 \\
0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 & -0.0417 & 0.3712 \\
0.1061 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 \\
-0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 & 0.379 \\
-0.0379 & -0.0417 & -0.3712 & 0.1061 & -0.4508 & 0.1061 & -0.3712 & 0.0417 & 0.379 & -0.4394 & 1.8674 & -0.4394 \\
\end{align*}
The minimum of the variances of the elementary contrasts for each treatment is 0.4314. Hence, $MV$ is the maximum of the minimum contrasts, $MV = 0.4314$.

2.3. Description of the Program

The MATLAB software by Mathworks is designed to run on C, C++, and MATLAB programming languages. The MATLAB program presented in this work requires inputting only the incidence matrix of the incomplete block design as a matrix to obtain the results of the four efficiency criteria as output. Paste the program on the MATLAB Command Window from where the program will request for the input of the incidence matrix through the command prompt, $'N = '$. Paste the incidence matrix of the desired incomplete block design at the command prompt and the results of the $A$, $D$, $E$, and $MV$-efficiencies would be displayed immediately on the Command Window. Alternatively, input the incidence matrix row-by-row, with rows separated by semi-colons in a bracket, then press the ‘Enter’ key on the computer keyboard to display the results. Another method of running the program is through the MATLAB M-file console. Copy and paste the program on the M-file console for easy editing. After inputting the incidence matrix either as a matrix or row-by-row, press the ‘Run’ button on the M-file toolbar to display the results of the four efficiency values on the Command Window.

Generally, by providing input of the incidence matrix, the program automatically estimates the basic parameters of an incomplete block design: the number of treatments, $t$, number of blocks, $b$, block size, $k$, and number of treatment replications, $r$. These parameters are vital in the computation of the information matrix, normalized information matrix, etc. Like the Command Window of the MATLAB software, the MATLAB program presented here is interactive. By interactive, we mean the ability of the program to accept input of the incidence matrix of an incomplete block design using the computer keyboard by typing the incidence matrix row-by-row and separating the rows with semi-colons or importing already prepared incidence matrix from other environments, such as Excel spreadsheet, Notepad, etc., and pasting the matrix on the program’s input command prompt on the Command Window or the M-file console.

The program computes the $A$-efficiency using the command, $A_{\text{efficiency}} = (t - 1)^{\ast}(1/\text{sum1})$. The $D$-efficiency is computed using the command, $D_{\text{efficiency}} = (\text{sum2}^\ast(1/(t - 1)))$, while the $E$-efficiency is computed with the command, $E_{\text{1_Efficiency}} = \min(\text{caneffifactor}(2:t,:))$. The $MV$-efficiency criterion is computed using the program command, $MV_{\text{efficiency}} = \max(\text{mve}(1,:))$.

We now highlight some of the advantages of the MATLAB program in the evaluation of incomplete block designs. The program utilizes computer time of about one second and occupies a small disc space of 911 bytes. These are obvious advantages when compared to Bailey and Royle’s [2] program for evaluating only $(6 \times 6)/2$ semi-Latin squares, which takes about one minute of computer time and could only be executed on SparcStation ELC and the GRAPE package, which runs more slowly without definite computer time. The program is also designed to accept only the incidence matrix as its input (instead of multiple inputs), from which the parameters of the incomplete block designs are estimated by the program. The program computes the exact values of the four efficiency measures for any incomplete block design and does not give any approximate values for any of the efficiency measures. For instance, in the first illustrative example in Section 3.1, the program computed the exact values of some of those efficiency measures of the seven semi-Latin squares provided by Bailey and Royle [2] as approximate values.

3. Results

The MATLAB program, presented in the Appendix A, for the computation of the $A$, $D$, $E$, and $MV$-efficiencies of incomplete block designs, including semi-Latin squares, completes the evaluation of any semi-Latin square or any other incomplete block design from the incidence matrix in less than one second of computing time on an HP Intel (R) Core
(TM)i5–4200U CPU with 4.0 GB RAM in a 64-bit Operating System running on Windows. The program size is 911 bytes.

In the subsequent subsection, we provide some illustrative examples of the performances of the MATLAB program in evaluating semi-Latin squares.

3.1. Evaluation of \((6 \times 6) / 2\) semi-Latin Squares (Bailey and Royle)

The first example is the computation of the \(A_r, D_r, E_r\), and \(MV\)-efficiency values of the seven \((6 \times 6) / 2\) semi-Latin squares, \(\Delta_0, \Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5,\) and \(\Delta_6\) constructed and evaluated by Bailey and Royle [2] using the graphical properties of the semi-Latin squares as quotient block designs. The semi-Latin squares are presented in Figures 1–7 respectively, for \(\Delta_0, \Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5,\) and \(\Delta_6\). The corresponding efficiency values are presented in Table 1.

**Table 1.**

|       | 0 | 1 | 2  | 3  | 4  | 5  | 6  | 7  |
|-------|---|---|----|----|----|----|----|----|
| \(\Delta_0\) | 9 | 2 | 11 | 4  | 1  | 6  | 3  | 8  | 5  | 10 | 7  |
| \(\Delta_1\) | 2 | 4 | 7  | 6  | 8  | 11 | 1  | 10 | 0  | 3  | 5  |
| \(\Delta_2\) | 5 | 7 | 4  | 9  | 11 | 8  | 10 | 1  | 3  | 0  | 2  |
| \(\Delta_3\) | 3 | 8 | 5  | 10 | 7  | 9  | 2  | 11 | 4  | 1  | 6  |
| \(\Delta_4\) | 1 | 10 | 3 | 0 | 5 | 2 | 7 | 4 | 9 | 6 | 11 | 8 |
| \(\Delta_5\) | 6 | 11 | 8 | 1 | 10 | 3 | 0 | 5 | 2 | 7 | 4 | 9 |

**Figure 1.** Semi-Latin square design, \(\Delta_0\).

|       | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-------|---|---|---|---|---|---|---|---|
| \(\Delta_1\) | 8 | A | 4 | 5 | 3 | 6 | 2 | B |
| \(\Delta_2\) | 5 | 0 | 2 | 7 | A | 1 | 8 | 6 |
| \(\Delta_3\) | 8 | 5 | 9 | 0 | 3 | 7 | B | 1 |
| \(\Delta_4\) | 9 | B | 1 | 7 | 2 | 6 | 0 | 4 |
| \(\Delta_5\) | 2 | 7 | 4 | 6 | 8 | B | 9 | A |
| \(\Delta_6\) | 4 | A | 3 | B | 1 | 9 | 2 | 5 |

**Figure 2.** Semi-Latin square design, \(\Delta_1\).

|       | 0 | 1 | 2  | 3  | 4  | 5  | 6  | 7  |
|-------|---|---|----|----|----|----|----|----|
| \(\Delta_2\) | 8 | A | 4 | 5 | 3 | 6 | 2 | B |
| \(\Delta_3\) | 5 | 0 | 2 | 7 | A | 1 | 8 | 6 |
| \(\Delta_4\) | 8 | 5 | 9 | 0 | 3 | 7 | B | 1 |
| \(\Delta_5\) | 9 | B | 1 | 7 | 2 | 6 | 0 | 4 |
| \(\Delta_6\) | 2 | 7 | 4 | 6 | 8 | B | 9 | A |
| \(\Delta_7\) | 4 | A | 3 | B | 1 | 9 | 2 | 5 |

**Figure 3.** Semi-Latin square design, \(\Delta_2\).

|       | 0 | 1 | 2  | 3  | 4  | 5  | 6  | 7  |
|-------|---|---|----|----|----|----|----|----|
| \(\Delta_3\) | 5 | A | 2 | 7 | 4 | 9 | 6 | B |
| \(\Delta_4\) | 4 | B | 2 | 3 | 1 | 9 | 8 | A |
| \(\Delta_5\) | 3 | 6 | 0 | 8 | 4 | 5 | 1 | B |
| \(\Delta_6\) | 5 | 8 | 9 | B | 0 | A | 6 | 7 |
| \(\Delta_7\) | 7 | A | 4 | 6 | 3 | B | 0 | 2 |
| \(\Delta_8\) | 2 | 9 | 1 | 7 | 6 | 8 | 3 | 5 |

**Figure 4.** Semi-Latin square design, \(\Delta_3\).
From Table 1, the results of the MATLAB program show very high levels of accuracy in the computations of the \( A \), \( D \), \( E \), and MV-efficiencies of the \( 6 \times 6 \)/2 semi-Latin squares. That is, the values of the four efficiency measures in Bailey and Royle [2] and from the MATLAB program presented here are exactly the same for most of the seven \( 6 \times 6 \)/2 semi-Latin squares, except the approximate values. Additionally, the MATLAB program computed the exact values of the efficiency measures for some of the \( 6 \times 6 \)/2 semi-Latin squares for which Bailey and Royle [2] could provide only the approximate values.

### 3.2. Evaluation of \((4 \times 4)/4\) semi-Latin Square (Chigbu)

The second illustrative example for the use of the MATLAB program is in the evaluation of the three optimal \((4 \times 4)/4\) semi-Latin squares whose statistical properties were studied by Chigbu [13]. The MATLAB program was used to obtain the values of the \( A \), \( D \), \( E \), and MV-efficiency measures of the three optimal \((4 \times 4)/4\) semi-Latin squares. The three semi-Latin squares are \( \Omega_1 \), \( \Omega_2 \), and \( \Omega_3 \), and are presented in Figures 8–10, while the results of their \( A \), \( D \), \( E \), and MV-efficiency measures are displayed in Table 2.
Table 1. Comparison of statistical efficiency values of the seven \((6 \times 6)/2\) designs from our MATLAB program with the efficiency values in Bailey and Rolye [2].

| S/N | The Design | Efficiency Values by Bailey and Royle [2] | Efficiency Values from Our MATLAB Program |
|-----|------------|------------------------------------------|------------------------------------------|
| 1   | \(\Delta_0\) | \(A\)-efficiency = 0.4909 \(D\)-efficiency = 0.5210 \(E\)-efficiency = 0.2723 \(MV\)-efficiency = 0.4314 \(A\)-efficiency = 0.5133 | \(A\)-efficiency = 0.4909 \(D\)-efficiency = 0.5210 \(E\)-efficiency = 0.2723 \(MV\)-efficiency = 0.4314 \(A\)-efficiency = 0.5133 |
| 2   | \(\Delta_1\) | \(D\)-efficiency = 0.5286 \(E\)-efficiency = 0.3333 \(MV\)-efficiency \(\approx\) 0.4444 \(A\)-efficiency \(\approx\) 0.5129 | \(D\)-efficiency = 0.5286 \(E\)-efficiency = 0.3333 \(MV\)-efficiency \(\approx\) 0.4444 \(A\)-efficiency \(\approx\) 0.5129 |
| 3   | \(\Delta_2\) | \(E\)-efficiency \(\approx\) 0.3759 \(MV\)-efficiency \(\approx\) 0.4619 \(A\)-efficiency \(\approx\) 0.5127 | \(E\)-efficiency \(\approx\) 0.3759 \(MV\)-efficiency \(\approx\) 0.4619 \(A\)-efficiency \(\approx\) 0.5127 |
| 4   | \(\Delta_3\) | \(A\)-efficiency \(\approx\) 0.5127 \(D\)-efficiency \(\approx\) 0.5283 \(E\)-efficiency \(\approx\) 0.3970 | \(A\)-efficiency \(\approx\) 0.5127 \(D\)-efficiency \(\approx\) 0.5283 \(E\)-efficiency \(\approx\) 0.3970 |
| 5   | \(\Delta_4\) | \(D\)-efficiency \(\approx\) 0.5281 \(E\)-efficiency \(\approx\) 0.3970 | \(D\)-efficiency \(\approx\) 0.5281 \(E\)-efficiency \(\approx\) 0.3970 |
| 6   | \(\Delta_5\) | \(E\)-efficiency \(\approx\) 0.3970 \(MV\)-efficiency \(\approx\) 0.4688 \(A\)-efficiency \(\approx\) 0.5115 | \(E\)-efficiency \(\approx\) 0.3970 \(MV\)-efficiency \(\approx\) 0.4688 \(A\)-efficiency \(\approx\) 0.5115 |
| 7   | \(\Delta_6\) | \(D\)-efficiency \(\approx\) 0.5277 \(E\)-efficiency \(\approx\) 0.3865 \(MV\)-efficiency \(\approx\) 0.4675 | \(D\)-efficiency \(\approx\) 0.5277 \(E\)-efficiency \(\approx\) 0.3865 \(MV\)-efficiency \(\approx\) 0.4675 |

Figure 8. Semi-Latin square, \(\Omega_1\).

| A | \(\alpha\) | B | \(\beta\) | C | \(\gamma\) | D | \(\delta\) |
|---|------------|---|------------|---|------------|---|------------|
| a | 1          | b | 2          | c | 3          | d | 4          |
| B | \(\gamma\) | A | \(\delta\) | D | \(\alpha\)| C | \(\beta\)  |
| d | 2          | c | 1          | b | 4          | a | 3          |
| C | \(\delta\) | D | \(\gamma\) | A | \(\beta\)| B | \(\alpha\) |
| b | 4          | a | 3          | d | 2          | c | 1          |

Figure 9. Semi-Latin Square, \(\Omega_2\).
Table 2. Comparison of the statistical efficiency values of the three \((4 \times 4) / 4\) semi-Latin squares from our MATLAB program with their efficiency values in Chigbu [13].

| Design | Efficiency Criteria | Efficiency Values by Chigbu [13] | Efficiency Values from MATLAB Program |
|--------|---------------------|----------------------------------|--------------------------------------|
| \(\Omega_1\) | A                  | 0.7500                           | 0.7506                               |
|        | D                  | 0.7759                           | 0.7761                               |
|        | E                  | 0.5000                           | 0.5000                               |
|        | MV                 | -                                | 0.6667                               |
| \(\Omega_2\) | A                  | 0.7500                           | 0.7500                               |
|        | D                  | 0.7759                           | 0.7759                               |
|        | E                  | 0.5000                           | 0.5000                               |
|        | MV                 | -                                | 0.7500                               |
| \(\Omega_3\) | A                  | 0.7500                           | 0.7500                               |
|        | D                  | 0.7759                           | 0.7759                               |
|        | E                  | 0.5000                           | 0.5000                               |
|        | MV                 | -                                | 0.6667                               |

Table 2 reveals that the \(A\)-, \(D\)- and \(E\)-efficiency values of \(\Omega_2\) and \(\Omega_3\) are exactly the same, while the three \((4 \times 4) / 4\) semi-Latin squares have exactly the same \(E\)-efficiency values. Slight differences exist between the values of \(A\)- and \(D\)-efficiencies of \(\Omega_1\) and \(\Omega_2\), which are attributable to what Soicher [3] referred to as mixture of manual and machine computations.

### 4. Discussion

In this work, the MATLAB program was presented for the evaluation of some statistical efficiency characteristics of any \((n \times n) / k\) semi-Latin squares, and in general, any incomplete block design using the design’s incidence matrix. To demonstrate the high level of accuracy of the program in computing the \(A\)-, \(D\)-, \(E\)-, and \(MV\)-efficiency measures of incomplete block designs, two illustrative examples were presented. The illustrative examples are the evaluations of the seven \((6 \times 6) / 2\) semi-Latin squares in [2] and the three optimal \((4 \times 4) / 4\) semi-Latin squares in [12]. Apart from giving exactly the same values for the four efficiency measures of the seven \((6 \times 6) / 2\) semi-Latin squares, the MATLAB program also provided exact efficiency values for some of the \((6 \times 6) / 2\) semi-Latin squares for which Bailey and Royle [2] could only provide approximate efficiency values. Specifically, the exact values of the \(MV\)-efficiency for \(\Delta_1\), \(A\)-, \(D\)-, \(E\)-, and \(MV\)-efficiencies for \(\Delta_2\) and \(MV\)-efficiency for \(\Delta_4\) were computed.

Furthermore, the MATLAB program dictated slight differences between the \(A\)- and \(D\)-efficiency values of the semi-Latin square, \(\Omega_1\), and those of \(\Omega_2\) and \(\Omega_3\) in [12]. Additionally, the \(MV\)-efficiency measures of the three optimal \((4 \times 4) / 4\) semi-Latin squares which were not previously obtained by [12] were computed by the program. Using the minimum variance of the simple contrasts of the three optimal \((4 \times 4) / 4\) semi-Latin squares, Chigbu [13] concluded that \(\Omega_2\) is preferable to \(\Omega_1\) and \(\Omega_3\). The results of the \(MV\)-efficiencies show that the value for \(\Omega_2\) is higher than those for \(\Omega_1\), and \(\Omega_3\) reinforces that claim.
In general, the MATLAB program uses less than one second of computing time to compute and display the $A_\gamma$, $D_\gamma$, $E_\gamma$, and $MV_\gamma$-efficiencies of any incomplete block design with a very high level of accuracy, as indicated in the illustrative examples.

**Author Contributions:** Conceptualization, E.I.M. and P.E.C.; methodology, E.I.M. and E.C.U.; software, E.I.M.; validation, E.I.M., P.E.C., and E.C.U.; formal analysis, E.I.M. and E.C.U.; investigation, E.I.M.; resources, E.I.M. and P.C.; data curation, E.I.M. and P.E.C.; writing—original draft preparation, E.I.M. and E.C.U.; writing—review and editing, E.I.M., P.E.C., and E.C.U.; visualization, E.I.M. and E.C.U.; supervision, P.E.C.; project administration, E.I.M. and P.E.C.; funding acquisition, E.I.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**Appendix A**

% A MATLAB program for calculating the $A_\gamma$, $D_\gamma$, $E_\gamma$, and $MV_\gamma$-efficiency of %Incomplete Block Designs using only the incidence matrix as an input
clc

% This MATLAB program can compute the $A_\gamma$, $D_\gamma$, $E_\gamma$, and $MV_\gamma$-efficiency of any incomplete block design using only the incidence matrix as an input

% The incidence matrix is a required input for the program irrespective of the type of design entered an input above
N = input('N=');

% parameters required for any incomplete block design with the above
% Incidence matrix will be calculated in the next five lines

% .
s = size(N)
t = s(1,1);
b = s(1,2);
k = sum(N(:,1:1));
r = sum(N(1:1,:));

% The concurrence matrix will be calculated as the product of the incidence % matrix and its transpose
NNP = N*N';

% The diagonal matrices of the treatment replications and the block sizes % will be given below
R = r*eye(t);
K = k*eye(b);

% The information matrix is calculated using the formula C=R-N*inv(K)*N'
GC = C+ones(t,t);

% The generalized information matrix will be obtained below
GCinv = inv(GC);

% The eigen values of the information matrix will then be obtained below
cef = eig(C);
caneffifactor = (1/r)*cef;
canonical efficiency factor = caneffifactor'

% The canonical efficiency factor is calculated as the non-zero eigen values of the normalized information matrix.

non-zero eigen value = cef(2:t,:);
% The loop below is required particularly for the calculation of A-Efficiency
sum = 0;
for j = 2:t;
    sum = sum+k/(r*k-cef(j));
end
effactor = (t-1)/(r*sum);
l = r*(k-1)/(t-1);
ef = (l*t)/(r*k);
sum1 = 0;
for j = 2:t;
    sum1 = (sum1+1/caneffifactor(j));
end
% The generalized inverse of the information matrix is used to obtain all the variances of the simple contrasts, as given in the next twelve lines below.
F = (1/r)*C;
P = (1/t)*ones(t);
gg = F+P;
M = inv(gg)-P;
for i = 1:t-1;
    for j = (i+1):t;
        ff = (M(i,i)+M(j,j)-M(i,j)-M(j,i));
        tt(i,j) = 2/ff;
    end
end
ttt = tt
% The matrices of the variances of all elementary contrasts are given in % matrix form below
Var_of_all_elementary_contrasts = tt(1:t-1,:)

% The Minimum variance of the columns of the matrix of all elementary % contrast are given by mve = min(tt(1:t-1,2:t))

mve = min(tt(1:t-1,2:t))
% The A-Efficiency is given by A_efficiency = (t-1)*(1/sum1)
A_efficiency = (t-1)*(1/sum1)
% The product below is required particularly for the calculation of % D-Efficiency
prod1 = prod(caneffifactor(2:t,:));
% The D-Efficiency is calculated as D_efficiency = (prod1)^((1/(t-1))
D_efficiency = (prod1)^((1/(t-1))
% The E-Efficiency is calculated as E_Efficiency = min(caneffifactor(2:t,:));
E1_Efficiency = min(caneffifactor(2:t,:))
% The MV-Efficiency is calculated as MV_efficiency = max(mve(1,:))
MV_efficiency = max(mve(1,:))
References
1. Bedford, D.; Whitaker, R. A New Construction for efficient semi-Latin squares. J. Stat. Plan. Inf. 2001, 9, 278–292. [CrossRef]
2. Bailey, R.A.; Royle, G. Optimal semi-Latin squares with side six and block size two. Proc. Roy. Soc. 1997, 453, 1904–1914. [CrossRef]
3. Soicher, L.H. Optimal and efficient semi-Latin squares. J. Stat. Plan. Inf. 2013, 143, 573–582. [CrossRef]
4. Bailey, R.A. Semi-Latin squares. J. Stat. Plan. Inf. 1988, 18, 299–312. [CrossRef]
5. McKay, B.D. Practical graph isomorphism. Cong. Numer. 1981, 30, 45–87. [CrossRef]
6. Soicher, L.H. GRAPE: A system for computing with graphs and groups. In Groups and Computation; Larry, F., William, M.K., Eds.; American Mathematical Society: Providence, RI, USA, 1993; Volume 11, pp. 287–291.
7. McKay, B.D.; Piperno, A. Practical Graph Isomorphism II. J. Sym. Comp. 2014, 60, 94–112. [CrossRef]
8. John, J.A. Cyclic Designs; Springer-Science+Business Media, B.V.: Berlin/Heildeberg, Germany, 1987.
9. John, J.A.; Williams, E.R. Cyclic and Computer Generated Designs; Springer-Science+Business Media, B.V.: Berlin/Heildeberg, Germany, 1995.
10. Hinkelmann, K.; Kempthorne, O. Design and Analysis of Experiments; John Wiley and Sons Inc.: Hoboken, NJ, USA, 2005; Volume 2.
11. Morgan, J.P. Blocking with independent responses. In Handbook of Design and Analysis of Experiments; Dean, A., Morris, M., Stufken, J., Bingham, D., Eds.; Chapman and Hall/CRC Press: Boca Raton, FL, USA, 2015.
12. Soicher, L.H. The GRAPE Package for GAP. 2019. Available online: https://www.gap-system.org/Manuals/pkg/grape-4.8.3/doc/manual.pdf (accessed on 19 May 2021).
13. Chigbu, P.E. The “Best” of the three optimal (4 × 4)/4 semi-Latin squares. Sankhya Indian J. Stat. 2003, 65, 641–648.