|  |  |  |
| --- | --- | --- |
| Operations Research Society of South Africa | Submittedfor publication inProceedings of the 2019 ORSSA Annual Conference | Operasionele Navorsingsvereniging van Suid-Afrika |

# Title of manuscript

Name 1 Surname 1[[1]](#footnote-1) Name 2 Surname 2[[2]](#footnote-2) Name 3 Surname 3[[3]](#footnote-3) Name 4 Surname 4[[4]](#footnote-4) Name 5 Surname 5[[5]](#footnote-5)

Abstract

Please use this template when preparing manuscripts for the 2019 ORSSA Conference Proceedings. Do not change paper formatting (page margins, font sizes, etc.) Abstract of the paper should not exceed 300 words, and should not include any mathematical notations or references. Full names and affiliations of authors must be included in the document. Affiliations will be removed by the proceedings editor for the review process.

Key words: Five to six key words of manuscript.

# 1 Introduction

 This paper contains examples of MS word functionality. The purpose of this paper is not to provide readers with sensible content or proper logic. The theorems and definitions used in this paper may therefore appear not to make sense. The purpose is, however, not to provide the correct theorems and definitions, but rather to provide examples of these entities. The examples demonstrate how the standard environments and elements are created within MS Word. This paper also provides examples of the preferred format in which equations, references and other standard elements are handled.

The examples provided in this paper are not exhaustive. Should an author wish to include specific or exceptional situations in a paper, the ORSSA editorial team can be contacted for guidance.

Below is an example of a table. Note that all tables should have captions, see *e.g*. Table 1.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Based on 2001 statistics** | **Based on 2011 estimates** | **% change** |
| **Premature infants requiring PDB per year** | 80 404 | 89 391 | 11.18% |
| **Bottles of PDB required per year** | 1 608 080 | 1 787 822 |

Table 1: Example of a table with a caption

# 2 Text

 To solve LPs with the help of the simplex algorithm, knowledge of linear algebra is required. Amongst other things, the concept of linearly dependent vectors is important. If $\overline{a}\_{j}$ with $j=1,2,3,…,r$ is a set of vectors, then the equation

 $k\_{1}\overline{a}\_{1}+k\_{2}\overline{a}\_{2}+…+k\_{r}\overline{a}\_{r}=\overline{0}$

will have at least one solution, which is $k\_{1}=k\_{2}=…=k\_{r}=0$. If this is the only solution, the set, $\overline{a}\_{j}$, is called linearly independent. If other solutions exist, the set is called linearly dependent. A set of column vectors, $\overline{a}\_{j}$, with $j=1,2,3,…,k$, is linear independent or linearly dependent. Assume the set of vectors is linearly dependent. Under these circumstances it is possible to reduce the number of positive variables step by step until the columns which correspond with the remaining positive variables are linearly independent. If $\overline{a}\_{j}$, with $j=1,2,3,…,n$, is linearly dependent, real numbers $α\_{j}$ (not all equal to zero) exist, so that

 $\sum\_{j=1}^{k}‍α\_{j}\overline{a}\_{j}=\overline{0}.$ (1)

 Equation (1) can be used to reduce an $x\_{j}$, say $x\_{r}$, in

 $\sum\_{j=1}^{k}‍x\_{j}\overline{a}\_{j}=\overline{b}, x\_{j}>0$ (2)

 to zero. Suppose any vector $\overline{a}\_{r}$ of the $k$ vectors in equation (1), for which $α\_{r}0$, can be expressed in terms of the remaining $k-1$ vectors. Thus

 $\overline{a}\_{r}=-\sum\_{jr}^{}‍\frac{α\_{j}}{α\_{r}}\overline{a}\_{j}.$ (3)

 Substitute equation (3) into equation (2), so that

 $\sum\_{j=1}^{p}‍\left(x\_{j}-x\_{r}\frac{α\_{j}}{α\_{r}}\right)\overline{a}\_{j}=\overline{b}.$ (4)

 This solution does not have more than $k-1$ non-zero variables. If $\overline{a}\_{r}$ is selected arbitrarily, there would be no assurance that some of the variables may not perhaps be negative. If $\overline{a}\_{r}$ is, however, selected carefully, the remaining $k-1$ variables will still be non-negative. Select $\overline{a}\_{r}$ so that

 $x\_{j}-x\_{r}\frac{α\_{j}}{α\_{r}}\geq 0 j=1,…,p.$ (5)

For any $j$, where $α\_{j}=0$, equation (5) will automatically hold. If $α\_{j}0$, then equation (5), divided by $α\_{j}$, will give one of the following inequalities:

 $\frac{x\_{j}}{α\_{j}}-\frac{x\_{r}}{α\_{r}}\geq 0 if α\_{j}>0, or $ (6)

 $\frac{x\_{j}}{α\_{j}}-\frac{x\_{r}}{α\_{r}}\leq 0 if α\_{j}<0.$ (7)

 From (6) and (7) follows that $\frac{x\_{r}}{α\_{r}}$ can not be greater than any $\frac{x\_{j}}{α\_{j}}>0$ and also not less than any $\frac{x\_{j}}{α\_{j}}<0$. If we choose the vector $\overline{a}\_{r}$ so that

 $\frac{x\_{r}}{α\_{r}}=\min\_{j}\left\{\frac{x\_{j}}{α\_{j}}, α\_{j}>0\right\},$ (8)

 then all the variables in equation (4) will be non-negative. A feasible solution with no more then $k-1$ non-zero variables has been found. If this process of reduction is continued, the remaining $x\_{j}$’s will eventually be linearly independent and the solution will be a basic feasible solution.

The theorem which summarises the theory of the solution of LPs with the simplex algorithm, is the fundamental theorem of LP.

**Theorem 2 (Fundamental theorem of linear programming (LP))** *This theorem consists of two parts:*

1. *If a feasible solution exists for an LP, a basic feasible solution exists.*
2. *If an optimal solution exists for an LP, an optimal basic feasible solution exists.*

 *Only a partial proof for Theorem 2 is presented.*

 **Proof:** Assume that a feasible solution exists with $k<n$ variables having, a positive value. Number these variables in such a way that the first $k$ variables have a positive value. The solution can then be formulated as

 $\sum\_{j=1}^{k}‍x\_{j}\overline{a}\_{j}=\overline{b}$

with $x\_{j}>0$ where $j=1,2,…,k$ and $x\_{j}=0$ for $j=k+1,…,n$. The vectors $\overline{a}\_{j}$ which are associated with the positive values can either be linearly independent or linearly dependent [6]. For a more detailed proof, see [4].

# 3 Algorithms

 The binary search algorithm starts its search in the centre of an alphabetical list of words. The algorithm compares the word in the middle of the list with the word that must be found (WRD) to test whether the middle word is, in fact, WRD. If this is not the case, it means that WRD is either after or before the word in the middle of the alphabetical list. If WRD comes alphabetically before the middle word, the word halfway between the first word and the middle word is tested to determine whether that word is WRD, while the other half (after the middle word) of the list is discarded. If WRD comes after the middle word, the word halfway between the middle and the last word is tested, while the first half of the list is discarded. This process of dividing the remaining list in half is repeated until the word is located or the list has been exhausted. This implies that only one half of the list has to be tested after the first step, one quarter of the list after the second step, one eighth of the list after the third step, etc. (The middle word is the word in the $\left⌊\frac{1+n}{2}\right⌋$-th position in the list.)

**Algorithm 3 (Binary search algorithm)** *Find a word in an alphabetical list of words by comparing the middle word with the word that must be found and by dividing the list, if required, into sublists*.

1. Let $α$ and $β$ be the first and last positions of the sublist. Initially $α=1$ and $β=n$. Thus $α\leftarrow 1$ and $β\leftarrow n$. WRD $\leftarrow $ “the word to be located".
2. Repeat this step until $α\leq β$, else go to 3.
	1. Determine $k$, the middle position (between $α$ and $β$).
	 $k\leftarrow \left⌊\frac{α+β}{2}\right⌋$
	2. If $W(k)$ = WRD, stop. Output: $k$.
	3. If WRD $>$ $W(k)$, then $β\leftarrow k-1$. Go to 2a.
	4. If WRD $<$ $W(k)$, then $α\leftarrow k+1$. Go to 2a.
3. Stop. Output: “Not in the list".

Let us examine the (worst case) complexity of the binary search algorithm. Note that step 1 and step 3 are both performed only once. In both of these steps, a constant number of calculations are performed. This means that the complexity of these steps is $O(1)$. Let $B(n)$ be the maximum number of times that step 2 must be repeated. After step 2 has been performed for the first time, the remaining list will have a maximum of $\frac{n}{2}$ elements to consider. This means that

 $B(n)=1+B\left(\left⌊\frac{n}{2}\right⌋\right).$ (9)

 We use induction to prove that

 $B(n)\leq 1+log\_{2}n$ (10)

 for each positive integer $n$. Equation (10) applies if $n=1$. Accept that this result holds for an integer, $k$. We must then show that this inequality holds for $k+1$. We therefore accept that

 $B(k)\leq 1+log\_{2}k.$ (11)

 From equation (9) we know that

 $B(k+1)=1+B\left(\left⌊\frac{k+1}{2}\right⌋\right),$

but from equation (11) it follows that

 $B\left(\left⌊\frac{k+1}{2}\right⌋\right)\leq 1+log\_{2}\left(\left⌊\frac{k+1}{2}\right⌋\right).$

We also know that

 $log\_{2}\left(\left⌊\frac{k+1}{2}\right⌋\right)\leq log\_{2}\left(\frac{k+1}{2}\right).$

This means that

 $B(k+1)\leq 2+log\_{2}\left(\frac{k+1}{2}\right)$

 $\leq 2+log\_{2}(k+1)-log\_{2}(2)$

 $\leq 1+log\_{2}(k+1),$

 which gives the desired result that $B(n)=O(log\_{2}n)$.

# 4 Matrices

 The variable, $x\_{3}$, enters the base with a value of $\frac{-20}{-1}=20$, which is smaller than or equal to the upper bound. Thus a simplex iteration can be performed. The column below $x\_{3}$ is given by

 $B\_{1}^{-1}p\_{3}=\left[\begin{matrix}-1&0&1\\0&1&0\\0&0&-1\end{matrix}\right]\left[\begin{matrix}1\\-1\\0\end{matrix}\right]=\left[\begin{matrix}-1\\-1\\0\end{matrix}\right].$

The elementary matrix is given by

 $E\_{1}=\left[\begin{matrix}\left(\frac{1}{-1}\right)&0&0\\-\left(\frac{-1}{-1}\right)&1&0\\-\left(\frac{0}{-1}\right)&0&1\end{matrix}\right].$

We can then calculate

 $B\_{2}^{-1}=E\_{1}B\_{1}^{-1}=\left[\begin{matrix}-1&0&0\\-1&1&0\\0&0&1\end{matrix}\right]\left[\begin{matrix}-1&0&1\\0&1&0\\0&0&-1\end{matrix}\right]=\left[\begin{matrix}1&0&-1\\1&1&-1\\0&0&-1\end{matrix}\right].$

After the iteration, the respective values are given by

 $\begin{matrix}A'=\left[\begin{matrix}-1&-1&1&-1&0&0\\1&0&-1&0&1&0\\-1&-1&0&0&0&1\end{matrix}\right] B\_{2}^{-1}=\left[\begin{matrix}1&0&-1\\1&1&-1\\0&0&-1\end{matrix}\right] \overline{b}=\left[\begin{matrix}-60\\30\\-80\end{matrix}\right]\\\begin{matrix}\overline{c}\_{bv}(2)&=&\left[-1,0,-1\right]& &\overline{c}\_{nbv}(2)&=&\left[-2,0,0\right]\\\overline{x}\_{bv}(2)&=&\left[x\_{3},s\_{2},\tilde{x}\_{2}\right]&&\overline{x}\_{nbv}(2)&=&\left[\tilde{x}\_{1},s\_{3},\tilde{s}\_{1}\right].\end{matrix}\end{matrix}$

The current right hand sides are given by

 $\overline{b}(2)=B\_{2}^{-1}\overline{b}=\left[\begin{matrix}1&0&-1\\1&1&-1\\0&0&-1\end{matrix}\right]\left[\begin{matrix}-60\\30\\-80\end{matrix}\right]=\left[\begin{matrix}20\\50\\80\end{matrix}\right].$

The basic variables $x\_{3}$, $s\_{2}$ and $\tilde{x}\_{2}$ have the values 20, 50 and 80 respectively. None of these variables occur at a value above their upper bounds.

Step 6:

All the right hand sides are positive, which means that an optimal solution has been found.

The optimal solution is given by

 $x\_{1}=80-\tilde{x}\_{1}=80-0=80$

 $x\_{2}=80-\tilde{x}\_{2}=80-80=0$

 $x\_{3}=20$

 $s\_{1}=0-\tilde{s}\_{1}=0-0=0$

 $s\_{2}=50$

 $s\_{3}=0.$

 The optimal objection function value is

 $ Objection function =c\_{1}x\_{1}+c\_{2}x\_{2}+c\_{3}x\_{3}$

 $=(-2)(80)+(-1)(0)+(1)(20)$

 $=-140.$

# References

[1] Chartrand G & Oellermann OR, 1993, *Applied and algorithmic graph theory*, McGraw Hill, Singapore, pp 43–46.

[2] Doob M, 1996, *TeX starting from 1*, Springer-Verlag, New York.

[3] Gass SI & Harris CM, 1996, *Encyclopedia of operations research and management science*, Kluwer Academic Publishers, Boston.

[4] Hadley G, 1962, *Linear programming*, Addison-Wesley Publishing Company Inc., Reading, Massachusetts.

[5] Lamport L, 1994, *LaTeX A document preparation system: User’s guide and reference manual*, Addison-Wesley Longman Inc., Reading, Massachusetts.

[6] Nickolson WK, 1995, *Linear algebra with applications*, 3rd Edition, PWS Publishing Company, Boston.

1. Corresponding author: Institution, Country, PO Box xxx, City Postal Code, email: xxxx@xxx.com [↑](#footnote-ref-1)
2. Institution, Country, PO Box xxx, City Postal Code, email: xxxx@xxx.com [↑](#footnote-ref-2)
3. Institution, Country, PO Box xxx, City Postal Code, email: xxxx@xxx.com [↑](#footnote-ref-3)
4. Institution, Country, PO Box xxx, City Postal Code, email: xxxx@xxx.com [↑](#footnote-ref-4)
5. Institution, Country, PO Box xxx, City Postal Code, email: xxxx@xxx.com [↑](#footnote-ref-5)