Formación Universitaria
Vol. 11(6), 29-40 (2018)
http://dx.doi.org/10.4067/S0718-50062018000600029

Un Sistema de Apoyo para la Enseñanza del Método Simplex y su Implementación en Computadora

Luis A. Moncayo-Martínez y David Fernando Muñoz
Departamento de Ingeniería Industrial y Operaciones, ITAM, Rio Hondo 1, Ciudad de México 01080, México
(e-mail: luis.moncayo@itam.mx; davidm@itam.mx)

Recibido Mar. 9, 2018; Aceptado May. 16, 2018; Versión final Jun. 22, 2018, Publicado Dic. 2018

Resumen

El objetivo de este trabajo es desarrollar una aplicación computacional para la implementación del algoritmo simplex utilizando el método de las dos fases y la descomposición LU (del inglés Lower-Upper). Debido a los resultados obtenidos por las empresas en la aplicación de modelos de programación lineal, el algoritmo simplex es un tema obligatorio que se imparte en programas de pre y post grado en ingeniería y negocios. Sin embargo, la literatura utilizada para su enseñanza presenta el método tableau que es el menos eficiente para su implementación computacional. Para resolver este problema, programamos una aplicación computacional en Visual Basic estructurada en tres módulos: Declaraciones, Entradas/Salidas y Procedimiento. Para resolver un problema el usuario lo introduce en su forma estándar y realiza las iteraciones dando clic en el botón indicado en la aplicación. Al utilizar nuestra aplicación en el salón de clases, los alumnos obtienen mejores calificaciones en la evaluación de esta parte del curso. Podemos concluir que el alumno entiende y puede desarrollar el algoritmo y no solo obtener la solución como con el software comercial.

Palabras clave: algoritmo simplex; eficiencia computacional; descomposición LU; Visual Basic

A Computational Application for Teaching the Simplex Algorithm and its Computational Implementation

Abstract

The objective of this work is to develop a computational application for the implementation of the simplex algorithm using two-phase method and the Lower-Upper decomposition. Since companies have applied linear models with excellent results, the simplex algorithm is a mandatory topic in undergraduate and graduate programs in engineering and business. However, the available literature to teach it introduces the tableau method that is the least efficient for its computational implementation. To solve this problem, we programmed a computational application in Visual Basic with three modules: Declarations, Outputs/Inputs, and Procedure. To solve a problem, the user enters it in its standard form and performs the iterations by clicking on the button indicated in the application. By using our application in the classroom, students get better grades in the evaluation of this part of the course. We can conclude that student understand and can develop the algorithm and not only obtain the solution as done with commercial software.

Keywords: simplex algorithm; computational efficiency; LU decomposition; Visual Basic

INTRODUCCIÓN

El método simplex, propuesto por Dantzig en la década de los 40's (Gass, 2002), es un algoritmo que resuelve problemas cuando son representados como modelos de programación lineal (PL), es decir, el método simplex es un procedimiento para determinar la solución óptima de un problema (e. las máximas ganancias o los mínimos costos) cuando es modelado por relaciones lineales. Matemáticamente, el método simplex es una técnica para optimizar una función objetivo lineal sujeto a restricciones lineales de igualdad (=) y/o desigualdad (≤, ≥). Los modelos de PL se pueden representar en su forma canónica de la siguiente manera:

$$\text{Maximizar } z = cx^T \text{ sujeto a } Ax \leq b \text{ con } x \geq 0 \tag{1}$$

En la ecuación 1, $x^T = (x_1, x_2, \dots)^T$ es un vector de variables denominadas "variables de decisión" que son la solución del problema, c y b son dos vectores de coeficientes conocidos y A es la matriz de coeficientes. La ecuación cx^T es la función objetivo (z) que se maximiza o minimiza sujeta a las restricciones $Ax \leq b$ y $x \geq 0$.

En las primeras aplicaciones del método simplex, la implementación computacional no era un aspecto que impidiera su aplicación. El primer problema a "gran escala" que se resolvió fue el problema de la dieta que tenía 48 restricciones y 71 variables (Dantzig, 1963). Desde entonces, se han propuesto muchas mejoras para acelerar la solución de problemas reales; es decir, a gran escala. Propuestas como el algoritmo de barrera y el algoritmo dual simplex han sido posibles gracias al desarrollo de computadoras más potentes (Bixby, 2002). De esta manera, los modelos de PL se han aplicado a muchas áreas prácticas como a la producción/manufactura (Harjunkoski et al., 2014), al servicio de salud (Abe et al., 2016), a la agricultura (Brandt et al., 2017), a la administración de redes de distribución de agua (Creaco y Pezzinga, 2015) y a la logística (Agrawal et al., 2015).

Por otro lado, muchas empresas han reportado ahorros sustanciales cuando utilización modelos de PL para administrar sus procesos, por ejemplo Procter & Gamble lo hizo rediseñando el sistema de producción y distribución (Camm et al., 1997), Samsung Electronics redujo los tiempos de producción y de inventarios (Leachman et al., 2002), Federal Express planeó los envíos en su sistema logístico (Mason et al., 1997) y Continental Airlines reasignado la tripulación cuando hay interrupciones (Yu et al., 2003). Por otro lado, cuando se quieren resolver problemas reales o de gran escala, como los mencionados anteriormente, se utiliza software especializado ya sea libre (GLPK, LP_SOLVE o SoPlex) o software comercial (CPLEX, Gurubi o GAMS) (Meindl y Templ, 2013). Sin importar que software se utiliza, el algoritmo mayormente implementado es el método simplex revisado debido a su eficiencia computacional (Williams, 2013).

De acuerdo con los resultados mencionados anteriormente, la programación lineal es un tema que se incluye en los programas de pre y post grado en las áreas de ingeniería y negocios. Sin embargo, en la literatura se han reportado algunas dificultades que enfrentan los alumnos cuando aplican esta técnica. Dos problemas identificados son: la modelación del problema como un modelo de PL y la interpretación de la solución (Williams, et al., 2016). Yoder y Kurz (2015) desarrollaron un procedimiento para probar que los alumnos de ingeniería tienen mejores resultados en la modelación y los de negocios en la evaluación de los resultados. Además, los autores mencionan que en los programas de negocios una parte fundamental del curso es que los alumnos formulen los modelos de PL y los resuelvan usando hojas de cálculo. Para los ingenieros, el objetivo es que los formulen y los resuelvan con algoritmos de optimización, específicamente el método simplex.

Para la enseñanza de este método existe una amplia bibliografía utilizada en cursos de investigación de operaciones. Los libros de texto que se utilizan para introducir al alumno al método simplex son los de Hillier y Lieberman (2010), Taha (2016), Winston (2004) y Hillier y Hillier (2015). Estos libros introducen al alumno al método tableu y sobre las tablas se aplican las operaciones de reducción de Gauss-Jordan para cambiar la base del problema de una iteración a otra. Si bien los principios básicos del método simplex se pueden abordar siguiendo este procedimiento (ampliamente difundido en los libros de texto), es sabido que el método simplex con tablas no es el método más eficiente para implementar en computadora, siendo más bien el método simplex de las dos fases el método que utiliza la mayoría del software comercial disponible en el mercado para resolver problemas de PL, debido a su eficiencia computacional. Si bien es cierto que el método simplex de las dos fases sí se describe en los textos más difundidos para la enseñanza de la Investigación de Operaciones, éste se aborda en capítulos posteriores al que describe el método simplex con tablas, reconociendo el hecho de que el método simplex revisado es computacionalmente más eficiente, y mencionando inclusive que una implantación eficiente del método simplex revisado en computadora utilizaría la descomposición LU, que es el método reconocido como el más eficiente para resolver por computadora un sistema de ecuaciones lineales (Press et al., 2007). Por otro lado, los libros de texto disponibles dan muy poca información para que el profesor de la materia pueda enseñar a sus estudiantes cómo podrían implantar eficientemente el método simplex revisado en una computadora.

La novedad de este trabajo tiene dos aspectos. Primero, algunos trabajos han concluido que los problemas que presentan los estudiantes durante un curso de modelado y optimización son la formulación, utilizar software para resolver el modelo y la interpretación de resultados (Williams et al., 2018) pero no se analiza la implementación computacional que es una característica primordial para resolver problemas reales. A diferencia de otras aplicaciones como PHP Simplex (https://goo.gl/tjvxEL) no sabemos que procedimiento matemático se utiliza por lo tanto no sabes si es eficiente para resolver problemas a gran escala, además el código no está disponible para que el alumno pueda comprender el algoritmo. El método grafico incluido en esta aplicación no es de ayuda para resolver problemas de más de dos variables, es decir es impráctico. Segundo, se propone una herramienta computación basada en el método dual simplex y descomposición LU con los objetivos de apoyar la enseñanza del método simplex, así como implementar el algoritmo que ha demostrado ser computacionalmente más eficiente en términos de tiempo CPU. En la bibliografía que se utiliza para la enseñanza del método simplex no se reporta ninguna aplicación computacional utilizando descomposición LU, ni tampoco el código para implementarlo.

EL MÉTODO SIMPLEX

En esta sección se presentan los conceptos fundamentales del método simplex y sus dos condiciones fundamentales: la de factibilidad y la condición de optimalidad. Así mismo, se ilustra el método de las dos fases.

La forma estándar del problema

De forma general el problema de PL en la ecuación 1 tiene *i* restricciones ($i = 1,2,\ldots,m$) y *j* variables ($j = 1,2,\ldots,n$). La forma estándar de un modelo de PL es transformar las restricciones del tipo $\leq$ y/o $\geq$ a ecuaciones con signo =. Esto se hace agregando una variable de holgura con signo positivo a las restricciones del tipo $\leq$ (por ejemplo $+ x_j$) y se agrega una variable de exceso con signo negativo a las restricciones del tipo $\geq$ (por ejmplo $- x_j$). A las restricciones que tienen sigo = no necesitan ningún cambio. Las variables de holgura y de exceso son agregadas a la función objetivo con coeficiente cero (por ejemplo $+ 0x_j - x_j$).

La tabla simplex y variables básicas.

Para ejemplificar el método simplex utilizaremos el siguiente ejemplo: Max $z = c_1x_1 + c_2x_2$ sujeto a $a_{11}x_1 + c_{12}x_2 \leq b_1$; $a_{21}x_1 + c_{22}x_2 \leq b_2$; $x_1, x_2 \geq 0$. La forma estándar del problema es: Max $z = c_1x_1 + c_2x_2 + 0x_3 + 0x_4$ sujeto a $a_{11}x_1 + a_{12}x_2 + x_3 = b_1$; $a_{21}x_1 + a_{22}x_2 + x_4 = b_2$; $x_1, x_2, x_3, x_4 \geq 0$. Para encontrar el valor máximo de *z* sujeto a las restricciones tenemos que definir una base de tamaño *m*, es decir tenemos que resolver el sistema de ecuaciones para encontrar el valor de las *m* variables básicas (v.b.), al resto de las variables ($n - m$) se les llama no básicas y su valor es 0. En nuestro ejemplo existen $m = 2$ variables básicas y $n - m = 4 - 2 = 2$ variables no básica. La Tabla 1 es un ejemplo de una tabla simplex con base formada por x_3 y x_4, es decir, estas variables son básicas y las variables no básicas son x_1 y x_2. Note que la función objetivo es escrita como $z - c_1x_1 - c_2x_2 = 0$.

Tabla 1: Ejemplo de una tabla simplex.

v.b. \ z	x_1	x_2	x_3	x_4	solución
	$-c_1$	$-c_2$	0	0	0
x_3	a_{11}	a_{12}	1	0	b_1
x_4	a_{21}	a_{22}	0	1	b_2

Cada columna de la tabla simplex la denotamos por P_j; por ejemplo, en la Tabla 1 $P_1 = \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}$, $P_2 = \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$, $P_3 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ y $P_4 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Las variables básicas son $x_B = \begin{pmatrix} x_3 \\ x_4 \end{pmatrix}$, las variables no básicas son $x_N = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, el vector de costos de las variables básicas es $c_B = (c_3, c_4) = (0,0)$ y la base es $B = (P_3, P_4) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

La base es una matriz cuadrada de tamaño *m* y es necesario que los vectores P_j en B sean independientes, los vectores P_j en B forman una base si su determinante no es cero, $\det(B) \neq 0$. Las matrices del modelo de PL representado en la ecuación 1 son $c = (-c_1, -c_2, 0, 0)$, la matriz de coeficientes $A = (P_1, P_2, P_3, P_4)$, el vector $b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$ y el problema representado en la Tabla 1 tiene seis bases $B = (P_1, P_2)$, $B = (P_1, P_3)$, $B = (P_1, P_4)$, $B = (P_2, P_3)$, $B = (P_2, P_4)$, $B = (P_3, P_4)$.

La idea fundamental del método simplex es cambiar una de las variables básicas $x_j \in x_B$ por una variable no básica $x_j \in x_N$ y formar la nueva B. Con esta nueva B se calcula la tabla simplex haciendo las operaciones entre las matrices como se muestra en la Tabla 2.

Tabla 2: Tabla simplex para una base B.

v.b. \ *z*	x_j ($j = 1,\ldots,n$)	solución
	$c_B B^{-1} P_j - c_j$	$c_B B^{-1} b$
x_B	$B^{-1} P_j$	$B^{-1} b$

El valor de la función objetivo es $z = c_B B^{-1} b$, el valor de las variables de decisión (o básicas) es $x_B = B^{-1} b$ y el de las variables no básicas es $x_N = 0$; sin embargo, si algún elemento del vector de variables básicas x_B es negativo la solución se llama, solución básica no factible porque que la restricción $x \geq 0$ (ver ecuación 1) no se cumple. Por otro lado, cuando $x_B \geq 0$ la solución se llama solución básica factible.

El método simplex

La solución óptima del problema está en por lo menos una solución básica factible, y el método simplex es un algoritmo que en cada iteración cambia de base en forma sistemática hasta llegar al óptimo. En este método una iteración implica hacer todas las operaciones de la Tabla 2 con la base x_B. El número de iteraciones no es necesariamente el número de bases factibles, cuando el método simplex llega al óptimo no hay que iterar más.
Si l es el número de iteración y la base de esa iteración es B^l, la base inicial ($l = 0$) del ejemplo en la Tabla 1 es $B^0 = (P_3, P_4) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. En la siguiente iteración $l = 1$ sacamos una variable básica del vector $x_{B^0} = \begin{pmatrix} x_3 \\ x_4 \end{pmatrix}$ y la cambiamos por una del vector de variables no básicas del vector $x_{N^0} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, por lo tanto, una base en la iteración $l = 1$ pudiera ser $x_{B^1} = \begin{pmatrix} x_2 \\ x_4 \end{pmatrix}$; es decir, sacamos de la base la variable básica x_3 y metemos la variable no básica x_2, de manera que $B^1 = (P_2, P_4) = \begin{pmatrix} a_{12} & 0 \\ a_{22} & 1 \end{pmatrix}$.
Antes de describir el criterio simplex para agregar y eliminar una variable básica de la base, es necesario definir el concepto de costo reducido que corresponde al renglón de z en la Tabla 1 y 2. El costo reducido de una variable, calculado con la ecuación 2, se puede definir como la cantidad que se tienen que pagar para introducir una unidad de esa variable en la solución.

$$z_j - c_j = c_B B^{-1} P_j - c_j \qquad (2)$$

Condición de optimización: el objetivo de esta condición es seleccionar una variable no básica para que sea parte de la base en la siguiente iteración; es decir, seleccionar la variable básica que será no básica. Esta condición depende de si se quiere minimizar o maximizar la función objetivo y la idea es que el incremento de una variable no básica puede aumentar el valor de z (en caso de maximización) con respecto a su valor actual solo si $z_j - c_j < 0$, de manera que la variable no básica que va a formar parte de la base es la que tenga el valor $z_j - c_j$ (ver ecuación 2) más negativo. En caso de minimizar la función objetivo entra la variable no básica con el coeficiente $z_j - c_j$ más positiva. El algoritmo se detiene si el valor $z_j - c_j$ para todas las variables no básicas es $z_j - c_j \leq 0$ en caso de maximizar y $z_j - c_j \geq 0$ en caso de minimizar.

Condición de factibilidad: el objetivo es seleccionar la variable básica que será no básica en la siguiente iteración. Una vez determinada la variable entrante x_j y su vector P_j, la variable básica que será no básica en la siguiente iteración será la que tenga el cociente mínimo de la ecuación 3. Ésta implica calcular la relación del vector solución $B^{-1}b$ con los coeficientes de restricciones estrictamente positivos bajo la variable de entrada x_j. Esta condición no depende del tipo de función objetivo (maximizar o minimizar). En ambas condiciones los empates se rompen arbitrariamente.

$$x_j = \min_i \left\{ \frac{(B^{-1}b)_i}{(B^{-1}P_j)_i} \middle| (B^{-1}P_j)_i > 0 \right\} \qquad (3)$$

Una vez que se aplicó la condición de optimización y factibilidad se tienen la nueva base B, aplican las operaciones Gaussianas sobre una tabla como la Tabla 1. El pivote es la intersección de la columna de la variable que entra a la base y el renglón de la variable que sale. En la Figura 1 se muestra el algoritmo simplex haciendo las aperciones de la Tabla 2 y el método tableu (Tabla 1).

Este algoritmo tiene tres partes fundamentales: 1) evaluar la condición de optimización, si se no se cumple, se pasa a la segunda parte, de lo contrario se termina el algoritmo; 2) evaluar la condición de factibilidad para determinar la variable que saldrá de la base; y 3) se hacen las operaciones entre matrices descritas en la Tabla 2 o se hacen las operaciones gaussianas sobre una tabla como la Tabla 1.

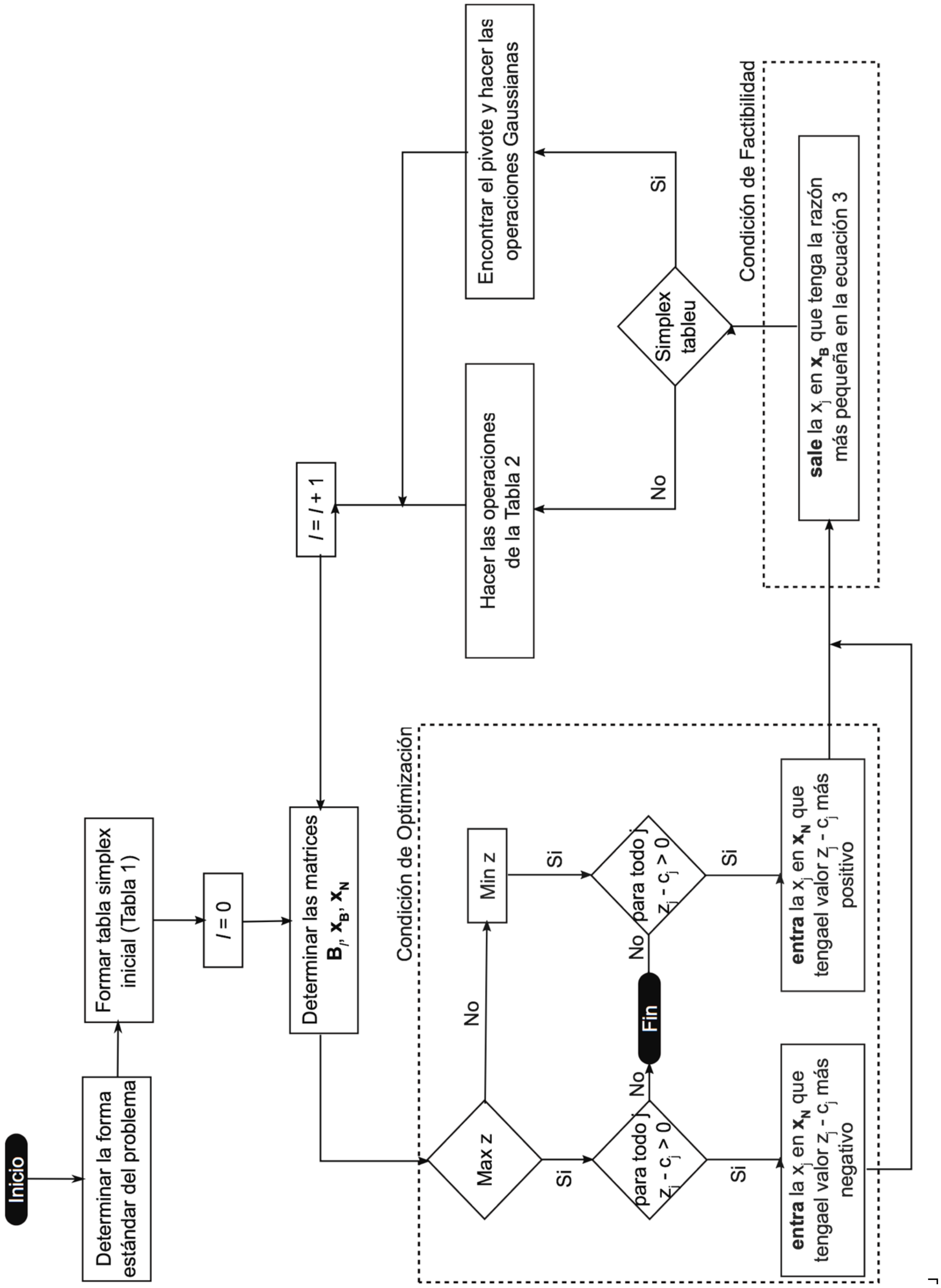


Fig. 1: Algoritmo del Método Simplex

Básicamente cuando se genera una nueva base se debe satisfacer la condición de optimización. Si se cumple esta condición el algoritmo se termina. Recuerde que esta condición depende de si la función objetivo es de maximizar o minimizar. Con esta condición se determina la variable básica que sale de la base. El agoritmo descrito en la Figura 1 sirve cuando el problema de PL tiene una función objetivo que se quiere maximizar, todas las restricciones son del tipo ≤ y el problema tiene solución factible. Los casos sin solución o solución degenerada no son ilustrados, aunque la aplicación computacional si resuelve ese tipo de problemas.

El método de las dos fases

Cuando se requiere minimizar una función objetivo y tiene restricciones del tipo ≥ y/o = es necesario utilizar otro método, ya sea el de la gran M o el de las dos fases. El método de la gran M es de los métodos más antiguos; sin embargo, nunca se utiliza en código comercial debido a su inherente error de redondeo. En su lugar siempre se prefiere utilizar el método de las dos fases (Taha, 2016). Para ejemplificar el método este método utilicemos el siguiente ejemplo: Minimizar $z = c_1x_1 + c_2x_2$ sujeto a $a_{11}x_1 + a_{12}x_2 \geq b_1$; $a_{21}x_1 + a_{22}x_2 \leq b_2$; $a_{31}x_1 + a_{32}x_2 = b_3$; $x_1,x_2,x_3 \geq 0$. El primer paso es expresar el problema en su forma estándar, además se agregan variables artificiales a las restricciones del tipo ≥ e =. Las variables artificiales son denotadas por R_i, por ejemplo R_1 es la variable artificial agregada a la restricción 1. La forma estándar de ejemplo anterior es: Minimizar $z = c_1x_1 + c_2x_2 – x_3 + x_4 + R_1 + R_3$ sujeto a $a_{11}x_1 + a_{12}x_2 – x_3 + R_1 = b_1$; $a_{21}x_1 + a_{22}x_2 + x_4 = b_2$; $a_{31}x_1 + a_{32}x_2 + R_3 = b_3$; $x_1,x_2,x_3,x_4,x_5,R_1,R_2 \geq 0$. A continuación se describen las dos fases:

Fase I: se minimiza el valor de las variables artificiales; es decir, Min $r = R_1 + R_2$ sujeto a las restricciones del problema. Para generar la tabla inicial se suma la función objetivo anterior ($r – R_1 - R_2 = 0$) a las restricciones que tengan una variable artificial R, como se ejemplifica en la Tabla 3. Note que en nuestro ejemplo las restricciones 1 y 3 tienen una variable artificial.

Tabla 3: Suma de las restricciones con variables artificiales a la función objetivo.

	x_1	x_2	x_3	x_4	R_1	R_2		
r	0	0	0	0	-1	-1	=	0
Restr. 1	a_{11}	a_{12}	-1	0	1	0	=	b_1
Restr. 3	a_{31}	a_{32}	0	0	0	-1	=	b_3
z	$a_{11} + a_{31}$	$a_{12} + a_{32}$	-1	0	0	0	=	$b_1+ b_3$

Una vez que se determina el renglón de la función objetivo se puede proceder a determinar la tabla inicial (Tabla 5) para aplicar el algoritmo simplex (Figura 1). Si al final del algoritmo la solución en el renglón de la función objetivo es cero, el problema tiene solución y podemos proceder la Fase II. De lo contrario, el problema no tiene solución.

Tabla 4: Tabla inicial en la Fase I

z / v.b.	x_1	x_2	x_3	x_4	R_1	R_2	solución
	$a_{11} + a_{31}$	$a_{12} + a_{32}$	-1	0	0	0	$b_1+ b_3$
R_1	a_{11}	a_{12}	1	0	-1	0	b_1
x_4	a_{21}	a_{22}	0	1	0	0	b_2
R_2	a_{31}	a_{32}	0	0	0	-1	b_3

Fase II: La ultima tabla de la Fase I es la solución básica con la que empieza esta fase, con dos modificaciones: a) se borran las columnas de las variables artificiales como se muestra en la Tabla 5 y b) el renglón de la función objetivo (z) se modifica de acuerdo con la Tabla 6 (Taha, 2016). Note que en la Fase I la función objetivo siempre es de minimizar, mientras que en la Fase II la función objetivo se maximiza o minimiza de acuerdo con el problema original.

Tabla 5: La tabla inicial de la Fase II sin las columnas de las variables artificiales.

z / v.b.	x_1	x_2	x_3	x_4	solución
	$-c_1$	$-c_2$	0	0	0
x_1	1	0	p_1	0	k_1
x_2	0	1	p_2	0	k_2
x_4	0	0	p_3	1	k_3

Tabla 6: Tabla inicial en la Fase II y se le aplica el algoritmo de la Figura 1.

z / v.b.	x_1	x_2	x_3	x_4	solución
	$-c_1+(c_1*1+c_2*0)$	$-c_2+(c_1*0+c_2*1)$	$0+(c_1*p_1+c_2*p_2)$	$0+(c_1*0+c_2*0)$	$0+(c_1*k_1+c_2*k_2)$
x_1	1	0	p_1	0	k_1
x_2	0	1	p_2	0	k_2
x_4	0	0	p_3	1	k_3

SISTEMA DE APOYO Y SU IMPLEMENTACIÓN

En esta sección se describe la estructura y el uso del sistema de apoyo, además se presenta un ejemplo. El sistema de apoyo está dividido en tres módulos como se muestra en la Figura 2: a) *EntradasSalidas*: sirve para preparar la hoja de cálculo y leer los datos del problema a resolver. El módulo necesita el tamaño del problema; es decir, el número de variables del problema en su forma estándar y el número de restricciones. Si es necesario borra todos los datos que se han introducido anteriormente; b) *Procedimientos*: En esta parte se desarrolla el método de las dos fases.

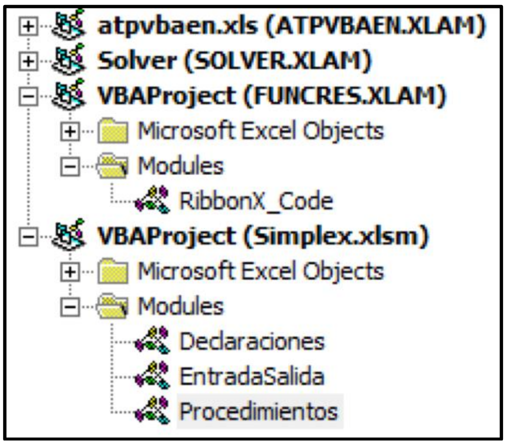


Fig. 2: Estructura del sistema de apoyo

Una vez que el módulo anterior ha preparado los datos, la aplicación tiene un botón llamado "Siguiente Iteración" el cual enviará un mensaje cuando haya terminado la primera fase y paso a paso mostrará la tabla simplex de las siguientes iteraciones. c) *Declaraciones:* En este módulo se declara la función *LESolve* para realizar los cálculos. La declaración se escribe a continuación:

Declare PtrSafe Function LESolve Lib "GenVarAlea.dll" (n As Long, A As Double, b As

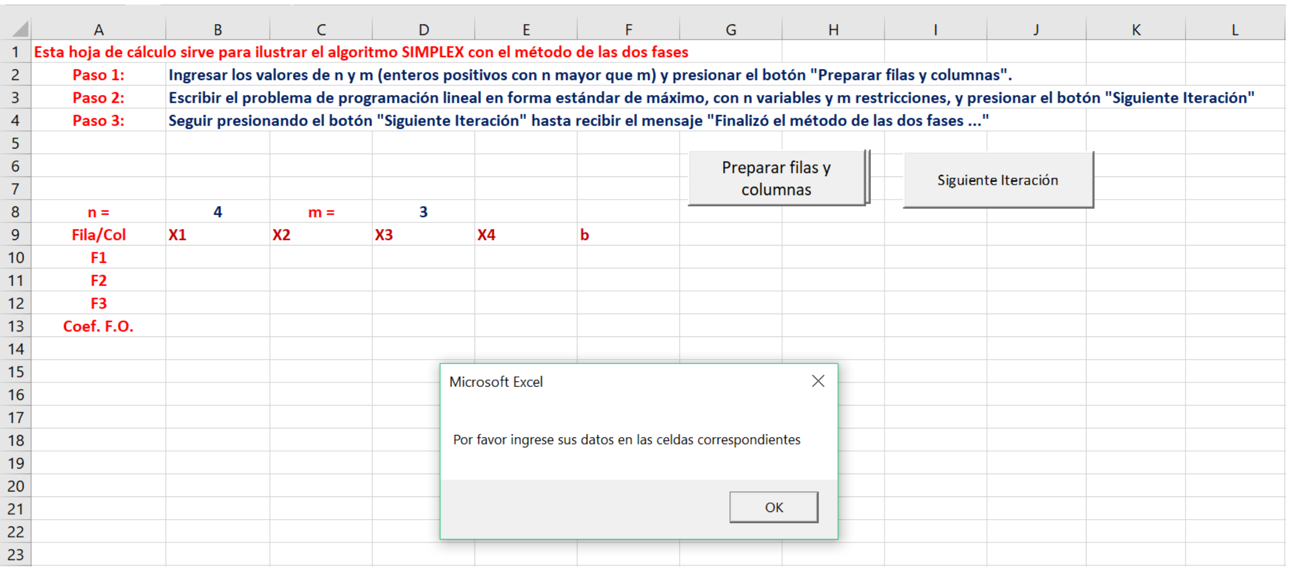
Double, Sol As Double) As Double

La librería "GenVarAlea.dll" está disponible en la página web del autor (https://goo.gl/NvYyYU) en la ruta Software -> Random Number Generators.

Para ilustrar el funcionamiento del sistema de apoyo se resolverá paso a paso el siguiente ejemplo: Max $z = 3x_1 + 4x_2$ sujeto a $x_1 + 4x_2 = 20$; $2x_1 + x_2 \geq 40$; $x_1 + x_2 \geq 15$; $x_1, x_2 \geq 0$. La forma estándar del problema es Max $z = 3x_1 + 4x_2 - x_3 - x_4$ sujeto a $x_1 + 4x_2 = 20$; $2x_1 + x_2 - x_3 = 40$; $x_1 + x_2 + x_2 - x_4 = 15$; $x_1,x_2 \geq 0$.

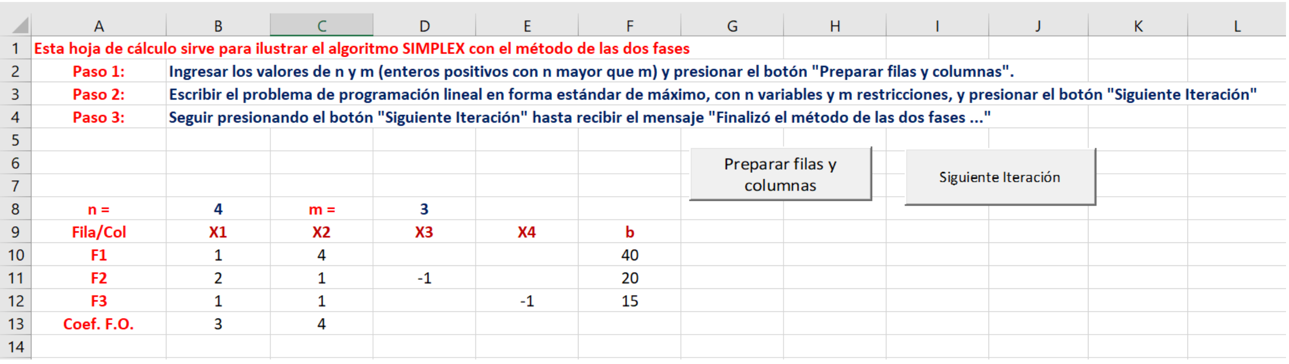
Paso 1: escribir el número de variables en la forma estándar del problema y el número de restricciones en la página de datos inicial como en la Figura 3. En nuestro ejemplo, el número de variables es n = 4 variables, dos variables (x_1 y x_2) en la forma canónica del problema y dos más (x_3 y x_4) para transformar el problema a su forma estándar. Una vez que ingrese el valor de n y m, deben hacer clic sobre el botón "Preparar filas y columnas". El sistema le enviará un mensaje para asegurar que usted comenzará un nuevo modelo. Si usted lo quiere hacer, el mensaje "Por favor ingrese sus datos en las celdas correspondientes". En la Figura 4 se muestra el ingreso de los datos de nuestro ejemplo.

Paso 2: una vez que ha ingresado todos los datos, presione el botón "Siguiente iteración" hasta que el sistema termine la Fase I del algoritmo. De acuerdo con la programación del sistema solo acepta funciones objetivo de maximizar, de manera que si el problema es de minimizar solo tienen que multiplicar la función objetivo por -1. Cada vez que le indica al sistema que haga una iteración, el sistema le señala en el último renglón las variables que pertenecerán a la siguiente base (renglón 20 en nuestro ejemplo, ver Figura 5) y también le muestra la base con la que se llevó acabo la iteración actual (renglón 15 en nuestro ejemplo, ver Figura 5). Para nuestro ejemplo, el sistema nos indica que la base está formada por las variables x_1, x_2 y x_3 con el valor de TRUE debajo de la columna de la variable. Además, aparecen las operaciones hechas en la Tabla 2, resolviendo el sistema por descomposición LU. Cada que el sistema termina una iteración, éste manda un mensaje especificando que la iteración terminó exitosamente. Específicamente, para la iteración cero el mensaje sería "Finalizó la iteración 0 con éxito". El siguiente mensaje es para hacerle saber que la Fase I termino con éxito y el número de iteraciones que hizo para terminarla (ver Figura 5).



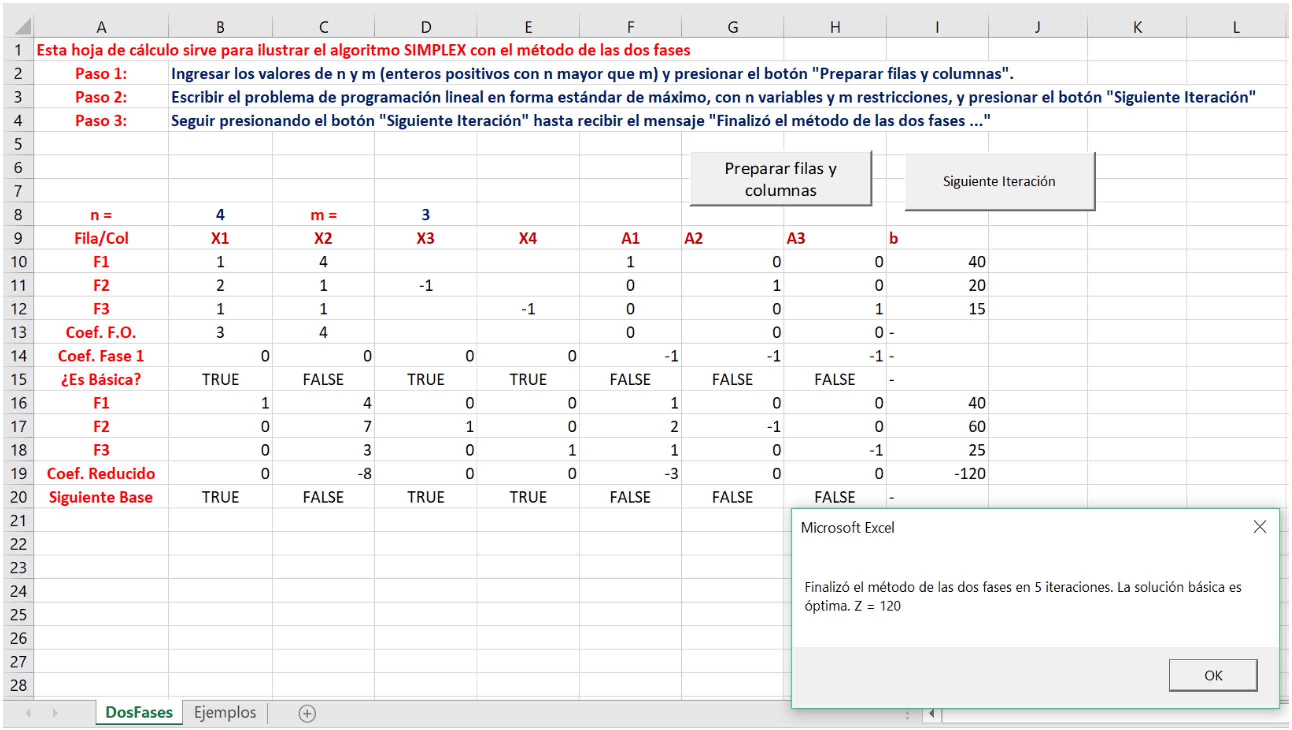
	A	B	C	D	E	F
1	Esta hoja de cálculo sirve para ilustrar el algoritmo SIMPLEX con el método de las dos fases					
2	Paso 1:	Ingresar los valores de n y m (enteros positivos con n mayor que m) y presionar el botón "Preparar filas y columnas".				
3	Paso 2:	Escribir el problema de programación lineal en forma estándar de máximo, con n variables y m restricciones, y presionar el botón "Siguiente Iteración"				
4	Paso 3:	Seguir presionando el botón "Siguiente Iteración" hasta recibir el mensaje "Finalizó el método de las dos fases ..."				
8	n =	4	m =	3		
9	Fila/Col	X1	X2	X3	X4	b
10	F1					
11	F2					
12	F3					
13	Coef. F.O.					

Fig. 3: Preparación del sistema para ingresar los datos



	A	B	C	D	E	F
1	Esta hoja de cálculo sirve para ilustrar el algoritmo SIMPLEX con el método de las dos fases					
2	Paso 1:	Ingresar los valores de n y m (enteros positivos con n mayor que m) y presionar el botón "Preparar filas y columnas".				
3	Paso 2:	Escribir el problema de programación lineal en forma estándar de máximo, con n variables y m restricciones, y presionar el botón "Siguiente Iteración"				
4	Paso 3:	Seguir presionando el botón "Siguiente Iteración" hasta recibir el mensaje "Finalizó el método de las dos fases ..."				
8	n =	4	m =	3		
9	Fila/Col	X1	X2	X3	X4	b
10	F1	1	4			40
11	F2	2	1	-1		20
12	F3	1	1		-1	15
13	Coef. F.O.	3	4			

Fig. 4: Preparación del sistema para ingresar los datos

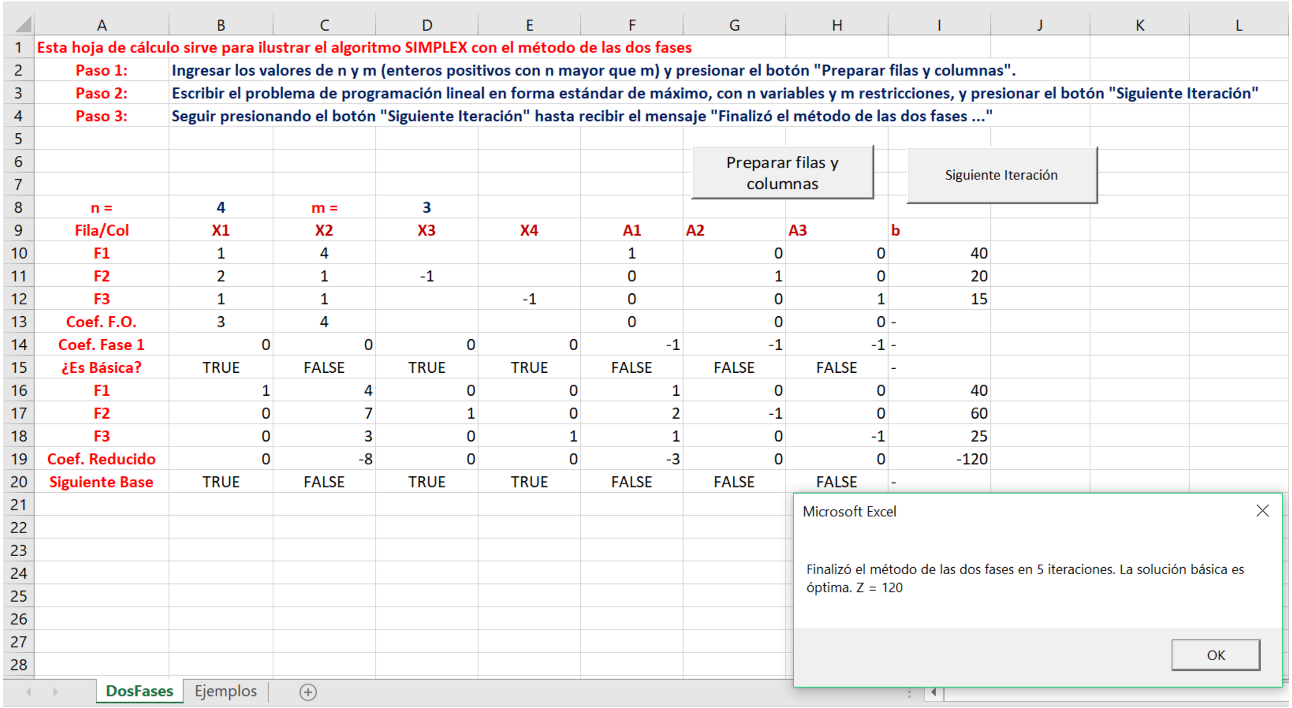


	A	B	C	D	E	F	G	H	I
1	Esta hoja de cálculo sirve para ilustrar el algoritmo SIMPLEX con el método de las dos fases								
2	Paso 1:	Ingresar los valores de n y m (enteros positivos con n mayor que m) y presionar el botón "Preparar filas y columnas".							
3	Paso 2:	Escribir el problema de programación lineal en forma estándar de máximo, con n variables y m restricciones, y presionar el botón "Siguiente Iteración"							
4	Paso 3:	Seguir presionando el botón "Siguiente Iteración" hasta recibir el mensaje "Finalizó el método de las dos fases ..."							
8	n =	4	m =	3					
9	Fila/Col	X1	X2	X3	X4	A1	A2	A3	b
10	F1	1	4			1	0	0	40
11	F2	2	1	-1		0	1	0	20
12	F3	1	1		-1	0	0	1	15
13	Coef. F.O.	3	4			0	0	0	-
14	Coef. Fase 1	0	0	0	0	-1	-1	-1	-
15	¿Es Básica?	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	-
16	F1	1	4	0	0	1	0	0	40
17	F2	0	7	1	0	2	-1	0	60
18	F3	0	3	0	1	1	0	-1	25
19	Coef. Reducido	0	-8	0	0	-3	0	0	-120
20	Siguiente Base	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	-

Fig. 5: Mensaje enviado por el sistema cuando ha terminado la Fase I

Paso 3: hacer clic sobre el botón “Siguiente iteración” hasta el sistema termina la Fase II del algoritmo, ver la

Figura 6. El usuario se pude dar cuenta que llego a la solución óptima cuando el sistema le indica que ya ha finalizado el método de las dos fases indicándole el número de iteraciones requeridas. El mensaje final del sistema le reporta el valor de la solución final Z = 120 y el número de iteraciones en que se alcanzó el máximo en este caso. Note los valores en el renglón 20, en este renglón se especifica si la variable en la columna respectiva pertenece a la base. En nuestro ejemplo la variable x1, x3 y x4 forman la base que $\mathbf{B}^1 = (\mathbf{P}_1, \mathbf{P}_3, \mathbf{P}_4)$. La solución del problema original es Z = 120, $x_1 = 40$, $x_2 = 0$. Además, en cada iteración aparece en el renglón 14 la base actual y en el renglón 15 la base para la siguiente iteración. El cambio de base corresponde a la aplicación de la condición de optimalidad y de factibilidad. En cada una de las iteraciones el sistema **Ax** = **b** es resuelto por la factorización LU como sigue **Ly** = **B** y **Ux** = **y**. La solución del sistema de ecuaciones en cada iteración por medio de la factorización LU es la novedad de este trabajo.



	A	B	C	D	E	F	G	H	I
1	Esta hoja de cálculo sirve para ilustrar el algoritmo SIMPLEX con el método de las dos fases								
2	Paso 1:	Ingresar los valores de n y m (enteros positivos con n mayor que m) y presionar el botón "Preparar filas y columnas".							
3	Paso 2:	Escribir el problema de programación lineal en forma estándar de máximo, con n variables y m restricciones, y presionar el botón "Siguiente Iteración"							
4	Paso 3:	Seguir presionando el botón "Siguiente Iteración" hasta recibir el mensaje "Finalizó el método de las dos fases ..."							
5									
6							Preparar filas y columnas		Siguiente Iteración
7									
8	n =	4	m =	3					
9	Fila/Col	X1	X2	X3	X4	A1	A2	A3	b
10	F1	1	4			1	0	0	40
11	F2	2	1	-1		0	1	0	20
12	F3	1	1		-1	0	0	1	15
13	Coef. F.O.	3	4			0	0	0	-
14	Coef. Fase 1	0	0	0	0	-1	-1	-1	-
15	¿Es Básica?	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	-
16	F1	1	4	0	0	1	0	0	40
17	F2	0	7	1	0	2	-1	0	60
18	F3	0	3	0	1	1	0	-1	25
19	Coef. Reducido	0	-8	0	0	-3	0	0	-120
20	Siguiente Base	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	-

Fig. 6: El sistema llega a la solución final

RESULTADOS PRELIMINARES

El estudio hecho por Andujar et al. (2008) es similar al nuestro en el sentido que propone una herramienta con técnicas audiovisuales para un montaje de un circuito eléctrico. Para medir el impacto del uso de la herramienta propuesta, ésta fue utilizada para enseñar el método simplex, la última vez que los autores explicaron este tema a estudiantes de las carreras de ingeniería industrial y de negocios en la clase de modelado y optimización. No es objetivo de una clase de optimización que el alumno meta datos y vea resultados, deber ser capaz de describir el fundamento matemático del concepto y por tanto su implementación para mejorar la solución del problema en términos computacionales (Williams et al., 2016).

Las columnas de la Tabla 7 representan los diferentes semestres en que se impartió la clase de optimización, el método simplex es un tema obligado. Las observaciones (renglones) son los puntos que obtuvieron los alumnos solo en el problema del examen que mide la habilidad del alumno de desarrollar el método simplex sin el uso de la computadora; es decir, el alumno tiene que hacer el examen solo con el uso de una calculadora. La parte de implementación de la descomposición LU se califica con un proyecto en dónde el alumno programa el algoritmo simplex y la descomposición LU en algún lenguaje de programación (Java, C, C++, Python, etc.), de manera que estos puntos obtenidos no están incluidos en la Tabla 7. La aplicación propuesta se utilizó con los estudiantes del Periodo 1, el resto de los semestres sirven de comparación para ver si existe alguna mejora con el uso de la aplicación.

El problema en que se mide la habilidad del alumno para desarrollar el método simplex tiene una valoración de 0 a 100 puntos y los puntos mínimos para acreditar el problema son 60. Como el alumno solo cuenta con una calculadora, una parte importante de la evaluación es que el alumno explique el fundamento matemático en cada iteración del cambio de base, es decir, debe explicar cómo aplico la condición de optimización y de factibilidad para llegar a la solución, por tal razón el evaluador se asegura que el alumno no solo es un espectador de la aplicación si no que entendió los conceptos fundamentales del algoritmo simplex.

De acuerdo con los resultados de la Tabla 7 y figura 7 cuando se utilizó la aplicación en el grupo ninguno de los alumnos obtuvo menos de 60 puntos, todos acreditaron el problema. Por otro lado, el promedio de los alumnos del Periodo 1 es el mayor de todos (80.87) y hubo alumnos que obtuvieron el valor máximo (100), hubo por lo menos un alumno que pudo desarrollar perfectamente el algoritmo, a diferencia de los Periodos 2, 3 y 4 en donde el valor máximo no alcanzo el 100. De acuerdo con esto se puede inferir que el uso del sistema de apoyo ayudo a incrementar la media del grupo. Por otro lado, la mediana en el Periodo 1 es menor (80) que la del Periodo 2 (85). Esto sugiere que el 50% de los alumnos del Periodo 2 obtuvieron una calificación igual o mayor a 85 y los del Periodo 1 una calificación igual o mayor a 80; los alumnos donde no se utilizó el sistema de apoyo obtuvieron más puntos. Sin embargo, en la Figura 7 se pude observar que el 50% de los alumnos del periodo 2 obtienen de 50 a 85 puntos; es decir, la dispersión en este periodo es mayor. De acuerdo con los resultados, se puede observar que cuando se utiliza el sistema de apoyo el número de estudiantes que obtienen menos de 60 puntos se reduce, pero la mediana es mayor cuando no se utilizó el sistema de apoyo; es decir, con éste se redujo el número de alumnos que no obtienen los puntos mínimos para acreditar el problema, pero el 50% de los alumnos obtuve de 80 a 100 puntos menor al del Periodo 2.

Para los Periodos 1 y 2 los resultados no son mejores que para los Periodos 3 y 4, por eso el análisis se reduce a que se obtuvieron mejores resultados en el Periodo 2 sobre el 3 y 4 por efecto de una mejor técnica de enseñanza del profesor; sin embargo, cuando se introdujo el sistema de apoyo el número de alumnos que no pudo obtener por lo menos 60 puntos se redujo. Cabe mencionar que los datos de la Tabla 7 se obtuvieron aplicando el mismo problema y el mismo profesor impartió el curso; así mismo, la calificación final del alumno en el examen compre no solo la habilidad para desarrollar el método simplex si no de la habilidad de modelar un problema.

Tabla 7: Puntos obtenidos en el problema del examen en donde los alumnos tienen que desarrollar el método simplex

Observación	*Semestre A*	*Semestre B*	*Semestre C*	*Semestre D*
1	100	98	55	90
2	70	50	60	75
3	80	93	70	70
4	80	78	90	75
5	85	70	90	70
6	90	86	55	75
7	100	88	95	90
8	80	55	85	50
9	80	55	80	85
10	100	83	75	75
11	70	95	85	90
12	60	75	95	75
13	70	75	55	80
14	80	76	75	50
15	85	88	55	
16	90	95	60	
17	80	91	95	
18	60	90	85	
19	100	85	60	
20	60	89	90	
21	90	60	60	
22	70	60		
23	80	96		
Media	80.87	79.61	74.76	75.00
Moda	80	88	55	75
Mediana	80	85	75	75
Mínimo	60	50	55	50
Máximo	100	98	95	90

El problema en todos los semestres es del mismo grado de complejidad para que la comparación sea justa y valida. Al final del curso, el alumno es capaz de implementar el método simplex en una computadora para resolver problemas a gran escala con diferentes métodos de reducción como la factorización LU. Por tal motivo, cuando el alumno se enfrenta a un problema con muchas variables y muchas restricciones podrá implementar la versión del método simplex que sea adecuada para resolver el problema de acuerdo con su complejidad. Solis et al. (2014) presenta algunos resultados similares al de la figura 7 para alumnos de ingeniería.

CONCLUSIONES

En este trabajo se presentó una aplicación computacional para apoyar la enseñanza del método simplex y se implementó el uso de la descomposición LU para la solución del modelo en cada iteración. El sistema le permite al usuario ver iteración por iteración la aplicación de las condiciones de optimalidad y factibilidad; es decir, el cambio de base en cada una de ellas. El sistema desarrolla el algoritmo que se programa en los softwares comerciales (el de las dos fases) que junto con la descomposición LU podemos asegurar que nuestra aplicación es eficiente computacionalmente de acuerdo con la revisión bibliográfica. Por otro lado, en los primeros resultados se puede asegurar que el número de alumnos que tienen menos de 60 puntos en la evaluación del método simplex es menor que cuando no se utilizó el sistema de apoyo.

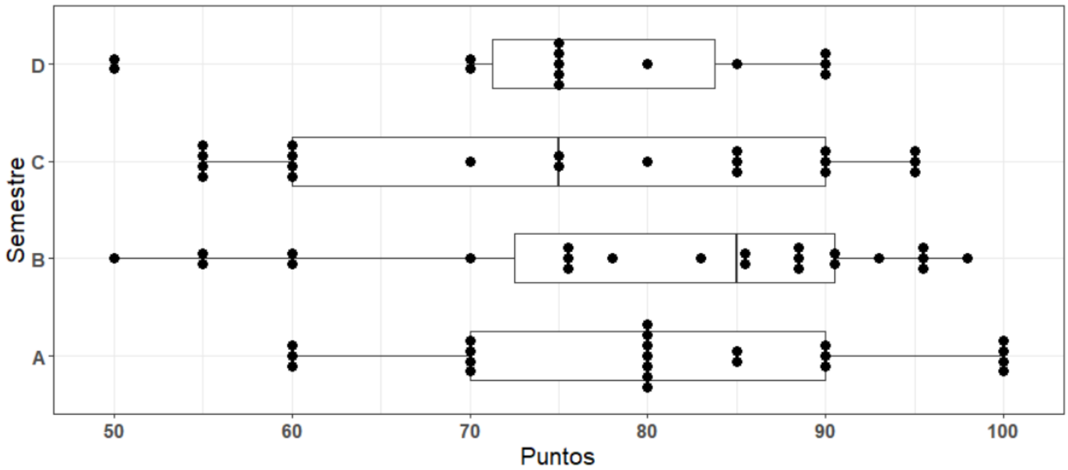


Fig. 7: Distribución de los puntos obtenidos en diferentes periodos; en el periodo 1 se utilizó la aplicación propuesta

AGRADECIMIENTOS

Se agradece el apoyo financiero de la Asociación Mexicana de Cultura, A.C. y al Consejo Nacional de Ciencia y Tecnología (CONACyT) de México.

REFERENCIAS

Abe, T.K., B.M. Beamon, R. L. Storch y J. Agus, *Operations Research Applications in Hospital Operations: Part I*, doi: 10.1080/19488300.2015.1134727, IIE Transactions on Healthcare Systems Engineering, 6(1), 42–54 (2016)

Agrawal, S., R.K. Singh y Q. Murtaza, *A Literature Review And Perspectives in Reverse Logistics*, doi: 10.1016/j.resconrec.2015.02.009, Resources, Conservation and Recycling, 97, 76–92 (2015)

Andujar, J.M, C. García, M. J. Redondo y J. Aroba, *Innovación Educativa en la Enseñanza de la Electrónica*, doi: 10.4067/S0718-50062008000400005, Formación Universitaria, 1(4), 29-34 (2008)

Bixby, R.E., *Solving Real-World Linear Programs: A Decade and More of Progress*, doi: 10.1287/opre.50.1.3.17780, Operations Research, 50(1), 3–15 (2002)

Brandt, P., M. Kvakić, K. Butterbach-Bahl y M.C. Rufino, *How to Target Climate-Smart Agriculture? Concept And Application of the Consensus-Driven Decision Support Framework "targetCSA"*, doi: 10.1016/j.agsy.2015.12.011, Agricultural Systems, 151, 234–245 (2017)

Camm, J. D., T. E. Chorman y otros cuatro autores, *Blending OR/MS, Judgment, and GIS: Restructuring P&G's Supply Chain*, doi: 10.1287/inte.27.1.128, Interfaces, 27(1), 128–142 (1997)

Creaco, E. y G. Pezzinga, *Embedding Linear Programming in Multi Objective Genetic Algorithms for Reducing the Size of the Search Space With Application to Leakage Minimization in Water Distribution Networks*, doi: 10.1016/j.envsoft.2014.10.013, Environmental Modelling & Software, 69, 308–318 (2015)

Dantzig, G. B., *Linear Programming and Extensions*, 1ª Ed., 1-627, Princeton Univ. Press, New Jersey, USA (1963)

Gass, S. I., *The First Linear-Programming Shoppe*, doi: 10.1287/opre.50.1.61.17781, Operations Research, 50(1), 61–68 (2002)

Harjunkoski, I., C. T. Maravelias y otros cinco autores, *Scope for Industrial Applications of Production Scheduling Models and Solution Methods*, doi:10.1016/j.compchemeng.2013.12.001, Computers & Chemical Engineering, 62, 161–193 (2014)

Hillier, F.S. y M.S. Hillier, *Introduction to Management Science : A Modeling and Case Studies Approach with Spreadsheets*, 5ª Ed., 1-720, McGraw-Hill, Stanford, USA (2015)

Hillier, F. S. y G. J. Lieberman, *Introduction to Operations Research*, 10ª Ed., 1-888, McGraw-Hill, Stanford, USA (2010)

Leachman, R. C., J. Kang y V. Lin, *SLIM: Short Cycle Time and Low Inventory in Manufacturing at Samsung Electronics*, doi: 10.1287/inte.32.1.61.15, Interfaces, 32(1), 61–77 (2002)

Mason, R.O., J.L. McKenney, W. Carlson y D. Copeland, *Absolutely, Positively Operations Research: The Federal Express Story*, doi: 10.1287/inte.27.2.17, Interfaces, 27(2), 17–36 (1997)

Meindl, B. y M. Templ, *Analysis of Commercial and Free and Open Source Solvers for the Cell Suppression Problem*, ISSN: 1888-5063, Trans. Data Privacy, 6(2), 147-159 (2013)

Press, W. H., S. A. Teukolsky, W. T. Vetterling y B. P. Flannery, Numerical Recipes : The Art of Scientific Computing, 3ª Ed., Cambridge, Cambridge University Press, Reino Unido (2007)

Solís, L. E., J. R. Baeza, D. A. Mena y M. D. Rodríguez, *Estudio Comparativo de dos Modelos Educativos basado en los Resultados del Rendimiento Académico de los Alumnos de Licenciatura en Ingeniería*, doi: 10.4067/S0718-50062015000300006, Formación Universitaria, 8(3), 47-56 (2015)

Taha, H. A., *Operations Research: An Introduction*, 10ª Ed., 1-832, New Jersey, Pearson, USA (2016)

Williams, H. P., *Model Building In Mathematical Programming,* 4ª Ed., 1-370, New Jersey, Wiley, USA (2013)

Williams, J.A.S., R. Reid, K. Gallamore y M. Rankin, *Introducing Troubleshooting for Model Formulation, Spreadsheet Development, and Memo Communication with Feedforward*, doi: 10.1287/ited.2017.0179, INFORMS Transactions on Education, 18(2), 102–115 (2018)

Williams, J., M. Rankin, K. Gallamore y R. Reid, *Beyond Model Formulation: Assessment of Novices Graphing, Interpreting, and Writing About their Model and Solution*, doi: 10.1287/ited.2016.0161, INFORMS Transactions on Education, 17(1), 13–19 (2016)

Winston, W. L., *Operations Research: Applications and Algorithms*, 4ª Ed., 1-1440, Boston, Cengage, USA (2004)

Yoder, S.E. y M. E. Kurz, *Linear Programming Across the Curriculum*, doi: 10.1080/08832323.2014.968516, Journal of Education for Business, 90(1), 18–23 (2015)

Yu, G., M. Argüello, G. Song, S. M. McCowan y A. White, *A New Era for Crew Recovery at Continental Airlines*, doi: 10.1287/inte.33.1.5.12720, Interfaces, 33(1), 5–22 (2003)