Proyecto II: Dualidad y Análisis de sensibilidad

Andrés Ángeles C.U. 131749
Mauricio Trejo C.U. 138886

18 de Noviembre 2018

1. Fase II y análisis de sensibilidad

El objetivo de esta sección es implementar una función en Matlab que resuelva problemas de la forma

$$\begin{aligned} \text{maximizar} \quad & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{sujeto a} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0},\, \boldsymbol{b} \geq \boldsymbol{0} \end{aligned} \tag{1}$$

mediante la Fase II del Método Simplex dos fases para maximización y que realice un análisis de sensibilidad para determinar las cotas superiores e inferiores de las perturbaciones que podemos realizar a las restricciones y la función objetivo sin modificar la base asociada a la solución óptima del problema original.

Nota: el problema (1) siempre es factible porque el vector de ceros, $\boldsymbol{0}$, es tal que

1. $\boldsymbol{A}\boldsymbol{0} = \boldsymbol{0} \leq \boldsymbol{b}$
2. $\boldsymbol{0} \geq \boldsymbol{0}$

por lo que $\boldsymbol{0} \in \boldsymbol{C}_F(1)$ y el parámetro `ban` de nuestra función `mSimplexMax` siempre será distinto de cero.

1.1. Fase II para problemas de maximización

Consideremos el siguiente problema[1] y verifiquemos que nuestro programa lo resuelve correctamente:

$$\begin{aligned} \text{maximizar} \quad & 3x_1 + x_2 + 3x_3 \\ \text{sujeto a} \quad & 2x_1 + x_2 + x_3 \leq 2 \\ & x_1 + 2x_2 + 3x_3 \leq 5 \\ & 2x_1 + 2x_2 + x_3 \leq 6 \\ & x_1, x_2, x_3 \geq 0 \end{aligned} \tag{2}$$

Si multiplicamos la función objetivo por -1 y agregamos variables de holgura x_4, x_5, x_6, entonces obtenemos el siguiente problema de minimización en forma estándar

$$\begin{aligned} \text{minimizar} \quad & -3x_1 - x_2 - 3x_3 \\ \text{sujeto a} \quad & 2x_1 + x_2 + x_3 + x_4 = 2 \\ & x_1 + 2x_2 + 3x_3 + x_5 = 5 \\ & 2x_1 + 2x_2 + x_3 + x_6 = 6 \\ & x \geq \boldsymbol{0} \end{aligned} \tag{3}$$

[1] Este problema lo utiliza David Luenberger en su libro *Linear and Nonlinear Programming* para ejemplificar el uso del tableau en el capítulo tres, *El Método Simplex*, p.48.

El tableau asociado al problema (3) es

Cuadro 1: Tableau inicial asociado al problema (3).

	x_1	x_2	x_3	x_4	x_5	x_6	LD
x_4	2	1	1	1			2
x_5	1	2	3		1		5
x_6	2	2	1			1	6
	3	1	3				

Usamos la Regla de Bland para escoger las variables de entrada y de salida.

Cuadro 2: Tableau en la primera iteración.

	x_1	x_2	x_3	x_4	x_5	x_6	LD
x_1	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$			1
x_5		$\frac{3}{2}$	$\frac{5}{2}$	$-\frac{1}{2}$	1		4
x_6		1		-1		1	4
		$-\frac{1}{2}$	$\frac{3}{2}$	$-\frac{3}{2}$			-3

Cuadro 3: Tableau final del problema (3).

	x_1	x_2	x_3	x_4	x_5	x_6	LD
x_1	1	$\frac{1}{5}$		$\frac{3}{5}$	$-\frac{1}{5}$		$\frac{1}{5}$
x_3		$\frac{3}{5}$	1	$-\frac{1}{5}$	$\frac{2}{5}$		$\frac{8}{5}$
x_6		1		-1		1	4
		$-\frac{7}{5}$		$-\frac{6}{5}$	$-\frac{3}{5}$		$-\frac{27}{5}$

La solución del problema (3) es $x^* = (x_1, x_2, x_3, x_4, x_5, x_6)^T = (\frac{1}{5}, 0, \frac{8}{5}, 0, 0, 4)^T$ y el valor óptimo es $z^* = -\frac{27}{5}$, por lo que la solución y el valor óptimo del problema original son

$$x^* = (x_1,\ x_2,\ x_3)^T = \left(\frac{1}{5},\ 0,\ \frac{8}{5}\right)^T, \quad z^* = \frac{27}{5}$$

Aquí incluimos el resultado de la consola de Matlab al ejecutar nuestro programa[2] para resolver el problema (2).

```
>> A = [2 1 1; 1 2 3; 2 2 1]; b = [2; 5; 6]; c = [3; 1; 3];
>> [x0, z0, ban, iter, ~] = mSimplexMax(A, b, c, false)

x0 =

       1/5
       0
       8/5
```

[2]El código de nuestra implementación de la Fase II del Método Simplex para maximización se encuentra al final del documento en el anexo junto con todos los programas que generan los resultados y las gráficas que se presentan en este documento.

```
z0 =

      27/5


ban =

       0


iter =

       2
```

1.2. Análisis de sensibilidad

Sea $x_{\boldsymbol{B}}$ una SBF óptima de un problema de la forma (1) asociada a la base $\boldsymbol{B} \subset \{1, \ldots, n\}$. El análisis de sensibilidad busca responder dos preguntas principales:

1. ¿Qué tanto podemos modificar los valores del vector de restricciones, $\boldsymbol{b}$, de tal forma que la base $\boldsymbol{B}$ siga siendo factible?

2. ¿Qué tanto podemos modificar los valores del vector de coeficientes de la función objetivo, $\boldsymbol{c}$, de tal forma que la SBF $x_{\boldsymbol{B}}$ siga siendo óptima?

1.2.1. Cambios en las restricciones

Sea $\bar{\beta}_k \in \mathbb{R}^m$ un vector con $\beta_k \in \mathbb{R}$ en la k-ésima entrada y ceros en todas las demás y denotemos por $[\boldsymbol{A}_{\boldsymbol{B}}^{-1}]_k$ a la k-ésima columna de $\boldsymbol{A}_{\boldsymbol{B}}^{-1}$.

Sea $\tilde{\boldsymbol{b}} = \boldsymbol{b} + \bar{\beta}_k$ para alguna $k \in \boldsymbol{B}$, entonces la base $\boldsymbol{B}$ seguirá siendo factible sí y sólo si

$$
\begin{aligned}
& \tilde{\boldsymbol{h}} = \boldsymbol{A}_{\boldsymbol{B}}^{-1}\tilde{\boldsymbol{b}} = \boldsymbol{A}_{\boldsymbol{B}}^{-1}\boldsymbol{b} + \boldsymbol{A}_{\boldsymbol{B}}^{-1}\bar{\beta}_k = \boldsymbol{h} + \beta_k[\boldsymbol{A}_{\boldsymbol{B}}^{-1}]_k \geq \boldsymbol{0} \\
\iff \quad & \beta_k[\boldsymbol{A}_{\boldsymbol{B}}^{-1}]_k \geq -\boldsymbol{h} \\
\iff \quad & \beta_k a_{sk} \geq -h_s \quad \text{para cada} \quad s \in \boldsymbol{B}
\end{aligned}
$$

donde a_{sk} es el elemento sk de la matriz $\boldsymbol{A}_{\boldsymbol{B}}^{-1}$.

Notemos que $h_s \geq 0$ para toda $s \in \boldsymbol{B}$ porque partimos del supuesto que $x_{\boldsymbol{B}}$ es factible. Así pues, el signo de la desigualdad depende enteramente del signo de a_{sk} y tenemos dos casos:

$$
\begin{cases}
\beta_k \geq -\dfrac{h_s}{a_{sk}} & \forall\, s \in \boldsymbol{B}\,\big(a_{sk} > 0\big) \\[2ex]
\beta_k \leq -\dfrac{h_s}{a_{sk}} & \forall\, s \in \boldsymbol{B}\,\big(a_{sk} < 0\big)
\end{cases}
\iff
\begin{cases}
\beta_k \geq \max\limits_{s \in \boldsymbol{B}} \Big\{ -\dfrac{h_s}{a_{sk}} \;\Big|\; a_{sk} > 0 \Big\} \\[2ex]
\beta_k \leq \min\limits_{s \in \boldsymbol{B}} \Big\{ -\dfrac{h_s}{a_{sk}} \;\Big|\; a_{sk} < 0 \Big\}
\end{cases}
$$

Por lo tanto, el intervalo máximo de sensibilidad para la k-ésima entrada del vector de restricciones $\boldsymbol{b}$ queda dado por

$$\boldsymbol{S}_k(\boldsymbol{b}) = \left[\max_{s\in\boldsymbol{B}}\left\{ -\frac{h_s}{a_{sk}} \;\Big|\; a_{sk} > 0\right\},\ \min_{s\in\boldsymbol{B}}\left\{ -\frac{h_s}{a_{sk}} \;\Big|\; a_{sk} < 0\right\}\right] \tag{4}$$

1.2.2. Cambios en los coeficientes de la función objetivo

Sea $\bar{\Gamma}_k$ un vector con $\gamma_k \in \mathbb{R}$ en la k-ésima entrada y ceros en todas las demás.

Antes de comenzar con el desarrollo, recordemos la definición del vector de costos relativos:

$$\boldsymbol{r}_{\boldsymbol{N}}^\top = \boldsymbol{c}_{\boldsymbol{B}}^\top \boldsymbol{H} - \boldsymbol{c}_{\boldsymbol{N}}^\top \tag{5}$$

El vector $\boldsymbol{c}$ pertenece a $\mathbb{R}^n$ y contiene valores que están asociados tanto a variables básicas como a variables no-básicas. Esto implica que, si perturbamos la k-ésima entrada de $\boldsymbol{c}$ y k está en $\boldsymbol{B}$, entonces la perturbación únicamente afecta al primer término del lado derecho de la ecuación (5), pues es el único lugar en el que aparecerá el coeficiente relacionado a la variable básica que estamos analizando. En cambio, si k pertenece a $\boldsymbol{N}$, sólo cambiara el segundo término de (5). Consecuentemente, lo más sencillo es dividir el análisis en dos casos:

1. **Cambios en los coeficientes básicos**

   Sean $j \in \boldsymbol{B}$, $\tilde{\Gamma}_j \in \mathbb{R}^m$ y $\tilde{\boldsymbol{c}}_{\boldsymbol{B}} = \boldsymbol{c}_{\boldsymbol{B}} + \bar{\Gamma}_j$, entonces $\boldsymbol{x}_{\boldsymbol{B}}$ seguirá siendo solución óptima si y sólo si

$$\begin{aligned}
&\tilde{\boldsymbol{r}}_{\boldsymbol{N}}^\top = \tilde{\boldsymbol{c}}_{\boldsymbol{B}}^\top \boldsymbol{H} - \boldsymbol{c}_{\boldsymbol{N}}^\top = \boldsymbol{c}_{\boldsymbol{B}}^\top \boldsymbol{H} + \bar{\Gamma}_j^\top \boldsymbol{H} - \boldsymbol{c}_{\boldsymbol{N}}^\top = \boldsymbol{\lambda}^\top \boldsymbol{A}_{\boldsymbol{N}} + \bar{\Gamma}_j^\top \boldsymbol{H} - \boldsymbol{c}_{\boldsymbol{N}}^\top = \bar{\Gamma}_j^\top \boldsymbol{H} + \boldsymbol{r}_{\boldsymbol{N}}^\top \geq \boldsymbol{0} \\
\iff\ & \bar{\Gamma}_j \boldsymbol{H} \geq -\boldsymbol{r}_{\boldsymbol{N}}^\top \\
\iff\ & \gamma_j \boldsymbol{H}_{j*} \geq -\boldsymbol{r}_{\boldsymbol{N}}^\top \\
\iff\ & h_{js}\gamma_j \geq -[\boldsymbol{r}_{\boldsymbol{N}}]_s \quad \text{para cada } s \in \boldsymbol{N}
\end{aligned}$$

   donde $\boldsymbol{H}_{j*}$ y h_{sj} son la j-ésima fila y el elemento sj de $\boldsymbol{H}$, respectivamente.

   Notemos que $\boldsymbol{r}_{\boldsymbol{N}} \geq \boldsymbol{0}$ porque partimos del supuesto que $\boldsymbol{x}_{\boldsymbol{B}}$ es solución óptima de (1), por lo que el signo de la desigualdad depende únicamente del signo de h_{js} y tenemos dos casos:

$$\begin{cases} \gamma_j \geq -\dfrac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{sj}} & \forall\, s \in \boldsymbol{N}\ \big(h_{js} > 0\big) \\[2ex] \gamma_j \leq -\dfrac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{js}} & \forall\, s \in \boldsymbol{N}\ \big(h_{js} < 0\big) \end{cases} \iff \begin{cases} \gamma_j \geq \max\limits_{s\in\boldsymbol{N}}\left\{ -\dfrac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{js}} \;\Big|\; h_{js} > 0\right\} \\[2ex] \gamma_j \leq \min\limits_{s\in\boldsymbol{N}}\left\{ -\dfrac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{js}} \;\Big|\; h_{js} < 0\right\} \end{cases}$$

   Por lo tanto, el intervalo máximo de sensibilidad para la j-ésima entrada básica del vector de coeficientes de la función objetivo $\boldsymbol{c}$ queda dado por

$$\boldsymbol{S}_k(\boldsymbol{c}_{\boldsymbol{B}}) = \left[\max_{s\in\boldsymbol{N}}\left\{ -\frac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{js}} \;\Big|\; h_{js} > 0\right\},\ \min_{s\in\boldsymbol{N}}\left\{ -\frac{[\boldsymbol{r}_{\boldsymbol{N}}]_s}{h_{js}} \;\Big|\; h_{js} < 0\right\}\right] \tag{6}$$

2. **Cambios en los coeficientes no-básicos**

   Sean $j \in \boldsymbol{N}$, $\tilde{\Gamma}_j \in \mathbb{R}^m$ y $\tilde{\boldsymbol{c}}_{\boldsymbol{N}} = \boldsymbol{c}_{\boldsymbol{N}} + \bar{\Gamma}_j$, entonces $\boldsymbol{x}_{\boldsymbol{B}}$ seguirá siendo solución óptima si y sólo si

$$
\begin{aligned}
&\tilde{\boldsymbol{r}}_{\boldsymbol{N}}^{\top} = \boldsymbol{c}_{\boldsymbol{B}}^{\top}\boldsymbol{H} - \tilde{\boldsymbol{c}}_{\boldsymbol{N}}^{\top} = \boldsymbol{c}_{\boldsymbol{B}}^{\top}\boldsymbol{H} - \boldsymbol{c}_{\boldsymbol{N}}^{\top} - \bar{\Gamma}_j^{\top} = \boldsymbol{\lambda}^{\top}\boldsymbol{A}_{\boldsymbol{N}} - \boldsymbol{c}_{\boldsymbol{N}}^{\top} - \bar{\Gamma}_j^{\top} = \boldsymbol{r}_{\boldsymbol{N}}^{\top} - \bar{\Gamma}_j^{\top} \geq \boldsymbol{0} \\
\iff \quad &\boldsymbol{r}_{\boldsymbol{N}}^{\top} \geq \bar{\Gamma}_j^{\top} \\
\iff \quad &[\boldsymbol{r}_{\boldsymbol{N}}]_j \geq \gamma_j
\end{aligned}
$$

Por lo tanto, el intervalo máximo de sensibilidad para la j-ésima entrada no-básica del vector de coeficientes de la función objetivo $\boldsymbol{c}$ queda dado por

$$
\boldsymbol{S}_k(\boldsymbol{c}_{\boldsymbol{N}}) = \Big(-\infty,\, [\boldsymbol{r}_{\boldsymbol{N}}]_j \Big] \tag{7}
$$

1.3. El problema de los relojes

El siguiente cuadro muestra un resumen de la información importante del problema de los relojes El planteamiento del modelo correspondiente a este problema de maximización es sencillo:

Cuadro 4: Resumen de la información importante del problema de los relojes. El número de horas de cada tarea y el tiempo disponible se miden por semana.

Problema de los relojes				
Encargado	Tarea	Reloj de pedestal	Reloj de pared	Tiempo disponible
David	Ensamblaje	6 horas	4 horas	40 horas
Diana	Lijado	8 horas	4 horas	40 horas
Lidia	Envíos	3 horas	3 horas	20 horas
	Precio de venta	300 pesos	200 pesos	

1. **Función objetivo y variables**

   La fórmula más sencilla para modelar las ganancias brutas de una empresa está dada por

   $$\text{GANANCIA} = \text{PRECIO} \times \text{CANTIDAD}$$

   Las variables serán x_1 y x_2 y corresponden a la cantidad de relojes de pedestal y de relojes de pared que la empresa produce por semana, respectivamente. Las entradas del vector $\boldsymbol{c}$ serán los precios del reloj de pedestal y de pared, por lo que la función objetivo será

   $$f(x_1, x_2) = 300x_1 + 200x_2$$

2. **Restricciones**

   Las variables x_1 y x_2 deben ser no-negativas porque los valores negativos no tendrían interpretación en este modelo.

   Dado que la cantidad de relojes de pedestal y de pared que podemos construir está limitado por la cantidad de tiempo que David, Diana y Lidia pueden invertir en sus respectivas tareas por semana y la cantidad de tiempo que conlleva cada tarea para cada tipo de reloj, tenemos que

   $$\boldsymbol{A} = \begin{bmatrix} 6 & 4 \\ 8 & 4 \\ 3 & 3 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} 40 \\ 40 \\ 20 \end{bmatrix}$$

y las restricciones serán de la forma $\boldsymbol{Ax} \leq \boldsymbol{b}$ donde $\boldsymbol{x} = (x_1, x_2)^\top$.

El modelo asociado a este problema será

$$\begin{aligned} \text{maximizar} \quad & 300x_1 + 200x_2 \\ \text{sujeto a} \quad & 6x_1 + 4x_2 \leq 40 \\ & 8x_1 + 4x_2 \leq 40 \\ & 3x_1 + 3x_2 \leq 20 \\ & x_1, x_2 \geq 0 \end{aligned}$$

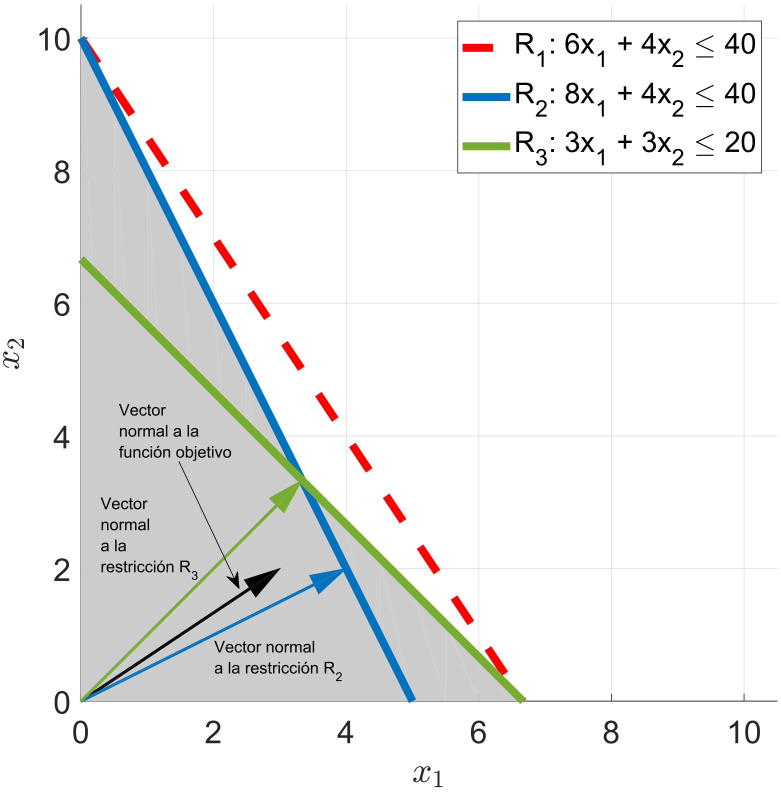

Figura 1: Gráfica del conjunto factible definido por las restricciones del problema de los relojes[3].

Resolviendo el problema con nuestro programa, tenemos que el tableau final es

Cuadro 5: Tableau final del problema de los relojes.

	x_1	x_2	x_3	x_4	x_5	LD
x_3			1	$-\frac{1}{2}$	$-\frac{2}{3}$	$\frac{20}{3}$
x_1	1			$\frac{1}{4}$	$-\frac{1}{3}$	$\frac{10}{3}$
x_2		1		$-\frac{1}{4}$	$\frac{2}{3}$	$\frac{10}{3}$
				25	$\frac{100}{3}$	$\frac{5000}{3}$

y la solución del problema original es $\boldsymbol{x} = (x_1, x_2) = (\frac{10}{3}, \frac{10}{3})$ con $z^* = \frac{5000}{3}$.

[3]El script de Matlab que genera esta gráfica utiliza la función `arrow` de Erik Johnson, la cual no incluimos en este documento ni en las carpetas del proyecto. Sin embargo, la función puede ser descargada en la página `https://la.mathworks.com/matlabcentral/fileexchange/278-arrow`.

Contestemos las siguientes preguntas con la función `mSimplexMax`.

```
>> A = [6 4; 8 4; 3 3]; b = [40; 40; 20]; c = [300; 200];
>> [x0, z0, ban, iter, sensinfo] = mSimplexMax(A, b, c, false);
```

1. Determinar si la solución óptima cambia cuando la estimación de la ganancia unitaria por reloj de pedestal incrementa de 300 a 375 pesos. Después determine si la solución óptima cambia cuando, además del cambio anterior, la estimación de la ganancia unitaria por reloj de pared disminuye de 200 a 175 pesos.

```
>> c = [375; 200];
>> [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

x0 =

      10/3
      10/3


z0 =

    5750/3

>> c = [375; 175];
>> [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

x0 =

       5
       0


z0 =

    1875
```

La solución óptima no cambia cuando la ganancia unitaria de los relojes de pedestal aumenta a 375, pero si cambia cuando la ganancia de los relojes de pared disminuye a 175.

2. ¿Cuánto puede variar la ganancia unitaria de cada tipo de reloj antes de que cambie la solución óptima?

```
>> sensinfo.gammas

ans =
```

```
-100                100
 -50                100
```

La ganancia unitaria de los relojes de pedestal puede aumentar o disminuir hasta 100 pesos antes de que cambie la solución óptima.

La ganancia unitaria de los relojes de pedestal puede disminuir hasta 50 pesos o puede aumentar hasta 100 pesos sin cambiar la solución óptima.

3. Determine el efecto sobre la solución óptima y la ganancia total si cada uno de los socios aumenta de forma independiente en 5 el máximo número de horas disponible por semana para trabajar.

```
>> b = [45; 40; 20];
>> [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

x0 =

      10/3
      10/3


z0 =

    5000/3

>> b = [40; 45; 20];
>> [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

x0 =

      55/12
      25/12


z0 =

    5375/3

>> b = [40; 40; 25];
>> [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

x0 =

       5/3
      20/3
```

```
z0 =

     5500/3
```

El aumento en la disponibilidad de horas de David no afecta la solución óptima. Esto tiene sentido, pues vimos en la Figura (1) que la restricción correspondiente a su labor estaba inactiva, por lo que sus horas de disponibilidad iniciales no limitaban la producción de relojes.

El aumento de horas de Diana y de Lidia cambian la solución óptima y aumentan las ganancias totales.

4. Un socio puede trabajar menos sin afectar la solución óptima. ¿Quién es?¿Cuál es el rango permisible de cambio en su número de horas de disponibilidad semanal que no alteran la solución óptima?

```
>> sensinfo.betas(1, :)

ans =

     -20/3            1/0
```

El intervalo de sensibilidad de las horas de disponibilidad de David es

$$\boldsymbol{S}_1(\boldsymbol{b}) = \Big[-\tfrac{20}{3}, \infty \Big)$$

y nos dice que él puede disminuir su máximo número de horas de trabajo semanal hasta en $\frac{20}{3} = 6.6\bar{6}$ horas y puede aumentarlo infinitamente sin afectar la solución óptima.

5. Calcule de forma sistemática la solución óptima y la ganancia total cuando el único cambio es que el número de horas de disponibilidad por semana de David toma los siguientes valores: 35, 37, 39, 41, 43, 45.

```
>> for i = 35:2:45
    b(1) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de David: %d \n', i);
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

Horas de disponibilidad de David: 35
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3
```

```
Horas de disponibilidad de David: 37
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3

Horas de disponibilidad de David: 39
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3

Horas de disponibilidad de David: 41
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3

Horas de disponibilidad de David: 43
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3

Horas de disponibilidad de David: 45
La solucion optima es x0 = (10/3, 10/3)
La ganancia optima es 5000/3
```

6. Repita el inciso interior, pero ahora con el número de horas de disponibilidad de Diana.

```
>> for i = 35:2:45
    b(2) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de Diana: %d \n', i);
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

Horas de disponibilidad de Diana: 35
La solucion optima es x0 = (25/12, 55/12)
La ganancia optima es 4625/3

Horas de disponibilidad de Diana: 37
La solucion optima es x0 = (31/12, 49/12)
La ganancia optima es 4775/3

Horas de disponibilidad de Diana: 39
La solucion optima es x0 = (37/12, 43/12)
La ganancia optima es 4925/3

Horas de disponibilidad de Diana: 41
La solucion optima es x0 = (43/12, 37/12)
La ganancia optima es 5075/3
```

```
Horas de disponibilidad de Diana: 43
La solucion optima es x0 = (49/12, 31/12)
La ganancia optima es 5225/3

Horas de disponibilidad de Diana: 45
La solucion optima es x0 = (55/12, 25/12)
La ganancia optima es 5375/3
```

7. Calcule de forma sistemática la solución óptima y la ganancia total cuando el único cambio es que el número de horas de disponibilidad por semana de Lidia toma los siguientes valores: 15, 17, 19, 21, 23, 25.

```
>> for i = 15:2:25
    b(3) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de Lidia: %d \n', i);
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

Horas de disponibilidad de Lidia: 15
La solucion optima es x0 = (5, 0)
La ganancia optima es 1500

Horas de disponibilidad de Lidia: 17
La solucion optima es x0 = (13/3, 4/3)
La ganancia optima es 4700/3

Horas de disponibilidad de Lidia: 19
La solucion optima es x0 = (11/3, 8/3)
La ganancia optima es 4900/3

Horas de disponibilidad de Lidia: 21
La solucion optima es x0 = (3, 4)
La ganancia optima es 1700

Horas de disponibilidad de Lidia: 23
La solucion optima es x0 = (7/3, 16/3)
La ganancia optima es 5300/3

Horas de disponibilidad de Lidia: 25
La solucion optima es x0 = (5/3, 20/3)
La ganancia optima es 5500/3
```

8. ¿Es válido usar los precios sombra que se obtuvieron en la pregunta 3 para determinar el efecto sobre la ganancia si Lidia cambiara su número de horas de disponibilidad por semana de 20 a 25? Si es así, calcule el incremento de la ganancia total. ¿Y si - adicionalmente - David reduce sus horas de disponibilidad por semana de 40 a 35?

   Sí es válido utilizar los precios sombra siempre y cuándo el problema que resolvimos no sea degenerado. A continuación mostramos que este es el caso para el problema de los relojes usando nuestra función. Para empezar, calculamos el valor de la multiplicación del aumento de horas de disponibilidad de Lidia por el precio sombra correspondiente. Luego, resolvemos el problema de los relojes antes y después del cambio en $\boldsymbol{b}$ y guardamos el valor de las ganancias totales óptimas en cada caso. Por último, mostramos que el cambio en el valor óptimo de las ganancias totales es igual que el valor de la multiplicación que calculamos al principio.

```
>> sensinfo.lambda(3)

ans =

     100/3

>> sensinfo.lambda(3)*5

ans =

     500/3

>> [~, z_20, ~, ~, ~] = mSimplexMax(A, b, c, false)

z_20 =

    5000/3

>> b(3) = 25;
>> [~, z_25, ~, ~, ~] = mSimplexMax(A, b, c, false)

z_25 =

    5500/3

>> z_25 - z_20

ans =

     500/3
```

   Aplicamos el mismo procedimiento de antes para el cambio de horas simultáneo de David y de Lidia.

```
>> sensinfo.lambda([1 3])

ans =

       0            100/3

>> dot(sensinfo.lambda([1 3]), [-5 5])

ans =

     500/3

>> [~, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

z0 =

    5000/3

>> b(1) = b(1) - 5; b(3) = b(3) + 5;
>> [~, z1, ~, ~, ~] = mSimplexMax(A, b, c, false)

z1 =

    1750

>> z1 - z0

ans =

     250/3
```

Podemos ver que el valor que calculamos usando los precios sombra de David y Lidia no coincide con el valor real del cambio en la ganancia total que calculamos con la función Simplex.

2. El Método Simplex Dual

2.1. Teoría

Supongamos que tenemos el siguiente problema primal

$$\begin{aligned} \text{minimizar} \quad & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{sujeto a} \quad & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b} \\ & \boldsymbol{x}, \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \tag{8}$$

cuya forma estándar es

$$\begin{aligned} \text{minimizar} \quad & \boldsymbol{c}^\top \boldsymbol{x} + \boldsymbol{0}^\top \boldsymbol{y} \\ \text{sujeto a} \quad & \boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{y} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \tag{9}$$

entonces

1. Si $\boldsymbol{x}$ pertenece al conjunto factible de (8), $\boldsymbol{C}_F(8)$, entonces existe un vector $\boldsymbol{y}$ tal que $(\boldsymbol{x}, \boldsymbol{y})$ pertenece al conjunto factible de (9), $\boldsymbol{C}_F(9)$.

   Demostración. Sea $\boldsymbol{x} \in \boldsymbol{C}_F(8)$, entonces $\boldsymbol{x} \geq \boldsymbol{0}$ y $\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}$

   $$\iff \quad \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \geq \boldsymbol{0}$$

   Definimos $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$, entonces $y \geq \boldsymbol{0}$ y

   $$\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} - (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{b}$$

   Por lo tanto, $(\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{C}_F(9)$ □

2. Si $(\boldsymbol{x}, \boldsymbol{y})$ pertenece a $\boldsymbol{C}_F(9)$, entonces $\boldsymbol{x}$ también pertenece a $\boldsymbol{C}_F(8)$.

   Demostración. Sea $(\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{C}_F(9)$, entonces

   $$\begin{aligned} & \boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} = \boldsymbol{b} \quad \text{y} \quad \boldsymbol{x}, \boldsymbol{y} \geq \boldsymbol{0} \\ \implies & \quad \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{y} \\ \implies & \quad \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \geq \boldsymbol{0} \end{aligned}$$

   Por lo tanto, $\boldsymbol{x} \in \boldsymbol{C}_F(8)$. □

3. El problema dual asociado a (8) es equivalente al problema dual asociado a (9).

   Demostración. Primero, encontramos el dual del problema (8).

   $$\begin{aligned} \text{minimizar} \quad & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{sujeto a} \quad & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b} \\ & \boldsymbol{x},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \quad \iff \quad \begin{aligned} -\text{maximizar} \quad & -\boldsymbol{c}^\top \boldsymbol{x} \\ \text{sujeto a} \quad & -\boldsymbol{A}\boldsymbol{x} \leq -\boldsymbol{b} \\ & \boldsymbol{x},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned}$$

   Aplicamos la definición del problema dual y tenemos

   $$\begin{aligned} -\text{minimizar} \quad & -\boldsymbol{b}^\top \boldsymbol{\lambda} \\ \text{sujeto a} \quad & -\boldsymbol{A}^\top \boldsymbol{\lambda} \geq -\boldsymbol{c} \\ & \boldsymbol{\lambda} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \quad \iff \quad \begin{aligned} \text{maximizar} \quad & \boldsymbol{b}^\top \boldsymbol{\lambda} \\ \text{sujeto a} \quad & \boldsymbol{A}^\top \boldsymbol{\lambda} \leq \boldsymbol{c} \\ & \boldsymbol{\lambda} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned}$$

   El dual del problema (9) lo conseguimos utilizando la tabla de Tucker:

	Primal	Dual
Tipo	Minimizar	Maximizar
Variables	$\boldsymbol{x}$, $\boldsymbol{y}$	$\boldsymbol{\lambda}$
Matriz	$\begin{bmatrix} \boldsymbol{A} & -\boldsymbol{I} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{A}^\top \\ -\boldsymbol{I} \end{bmatrix}$
Lado derecho	$\boldsymbol{b}$	$\begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{0} \end{bmatrix}$
Función objetivo	$\boldsymbol{c}^\top \boldsymbol{x}$	$\boldsymbol{b}^\top \boldsymbol{\lambda}$
Restricción $\leftrightarrow$ Variable	$\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} = b$	$\boldsymbol{\lambda} \in \mathbb{R}^m$
Variable $\leftrightarrow$ Restricción	$\boldsymbol{x}, \boldsymbol{y} \geq \boldsymbol{0}$	$\boldsymbol{A}^\top \lambda \leq \boldsymbol{c}$

entonces tenemos

$$\begin{aligned} &\text{maximizar} && \boldsymbol{b}^\top \boldsymbol{\lambda} \\ &\text{sujeto a} && \boldsymbol{A}^\top \boldsymbol{\lambda} \leq \boldsymbol{c} \\ & && -\boldsymbol{I}\boldsymbol{\lambda} \leq \boldsymbol{0} \\ & && \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \iff \begin{aligned} &\text{maximizar} && \boldsymbol{b}^\top \boldsymbol{\lambda} \\ &\text{sujeto a} && \boldsymbol{A}^\top \boldsymbol{\lambda} \leq \boldsymbol{c} \\ & && \boldsymbol{\lambda} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned}$$

Por lo tanto, los duales de los primales (8) y (9) coinciden. □

2.2. Fase II del Método Simplex Dual y el problema de los relojes

Consideremos el problema dual siguiente

$$\begin{aligned} &\text{minimizar} && \boldsymbol{c}^\top \boldsymbol{y} \\ &\text{sujeto a} && \boldsymbol{A}\boldsymbol{y} \geq \boldsymbol{b} \\ & && \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned} \tag{10}$$

En esta sección mostramos los resultados que obtuvimos al resolver el problema de los relojes con nuestra implementación de la Fase II del Método Simplex Dual para minimización en Matlab.

Nota: Una observación importante es que el problema primal asociado a (10) será

$$\begin{aligned} &\text{maximizar} && \boldsymbol{b}^\top \boldsymbol{x} \\ &\text{sujeto a} && \boldsymbol{A}^\top \boldsymbol{x} \leq \boldsymbol{c} \\ & && \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{c} \geq \boldsymbol{0} \end{aligned}$$

En la primera sección de este reporte mostramos que los conjuntos factibles de los problemas que tienen esta forma son no vacíos. Esto y el Teorema de dualidad implican que el problema (10) siempre tiene una cota inferior y el resultado `ban` que devuelve nuestra función siempre será distinto de uno.

El modelo del problema de los relojes en forma estándar es

$$\begin{aligned} \text{maximizar} \quad & 300x_1 + 200x_2 \\ \text{sujeto a} \quad & 6x_1 + 4x_2 + x_3 = 40 \\ & 8x_1 + 4x_2 + x_4 = 40 \\ & 3x_1 + 3x_2 + x_5 = 20 \\ & \boldsymbol{x} \geq \mathbf{0} \end{aligned} \tag{11}$$

y este será nuestro problema primal.

El problema dual de (11) es

$$\begin{aligned} \text{minimizar} \quad & 40\lambda_1 + 40\lambda_2 + 20\lambda_3 \\ \text{sujeto a} \quad & 6\lambda_1 + 8\lambda_2 + 3\lambda_3 \geq 300 \\ & 4\lambda_1 + 4\lambda_2 + 3\lambda_3 \geq 200 \end{aligned} \tag{12}$$

Utilizamos nuestra función `mSimplexDual` para resolver el problema (12) y el resultado de la consola de Matlab se muestra a continuación.

```
>> A = [6 8 3; 4 4 3]; b = [300; 200]; c = [40; 40; 20];
>> [x0, z0, ban, iter, lambda0] = mSimplexDual(A, b, c, false)

x0 =

       0
      25
     100/3


z0 =

    5000/3


ban =

       0


iter =

       2


lambda0 =

      10/3
      10/3
```

La solución óptima del dual (12) es $\boldsymbol{\lambda}^* = (0, 25, \frac{100}{3})^\top$ con valor objetivo óptimo asociado $z^* = \frac{5000}{3}$, el cual coincide con el valor objetivo óptimo que obtuvimos con la función `mSimplexMax` en la sección 1.3.

Las condiciones de optimalidad están dadas por el Teorema de complementaridad asimétrica que vimos en clase y el cual enunciamos a continuación.

Teorema (Complementaridad asimétrica): *Consideremos el problema primal en forma estándar **(P)***

$$\begin{array}{ll} \text{minimizar} & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{sujeto a} & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{array}$$

*con dual asociado **(D)***

$$\begin{array}{ll} \text{maximizar} & \boldsymbol{b}^\top \boldsymbol{\lambda} \\ \text{sujeto a} & \boldsymbol{A}^\top \boldsymbol{\lambda} \leq \boldsymbol{c} \end{array}$$

*y supongamos que $\boldsymbol{x}$ y $\boldsymbol{\lambda}$ son soluciones factibles de **(P)** y **(D)**, respectivamente. Las soluciones $\boldsymbol{x}$ y $\boldsymbol{\lambda}$ son óptimas si y sólo si*

1. *para cualquier índice $j \in \{1, \ldots, n\}$, si $x_j \geq 0$, entonces $c_j = \lambda^\top \boldsymbol{A}_j$*
2. *para cualquier índice $j \in \{1, \ldots, n\}$, si $c_j \geq \lambda^\top \boldsymbol{A}_j$, entonces $x_j = 0$*

donde $\boldsymbol{A}_j$ es la j-ésima columna de $\boldsymbol{A}$.

La SBF del primal es $\boldsymbol{x}^* = (\frac{10}{3}, \frac{10}{3}, \frac{20}{3})^\top$ y la SBF del dual es $\boldsymbol{\lambda}^* = (0, 25, \frac{100}{3})^\top$, entonces todas las entradas de $\boldsymbol{x}^*$ son mayores o iguales que 0 y $\boldsymbol{\lambda}^*$ es tal que

$$\boldsymbol{A}^\top \boldsymbol{\lambda} = \begin{bmatrix} 6 & 8 & 3 \\ 4 & 4 & 3 \end{bmatrix} \begin{bmatrix} 0 \\ 25 \\ \frac{100}{3} \end{bmatrix} = \begin{bmatrix} 8 \times 25 + 3 \times \frac{100}{3} \\ 4 \times 25 + 3 \times \frac{100}{3} \end{bmatrix} = \begin{bmatrix} 300 \\ 200 \end{bmatrix} = \boldsymbol{c}$$

por lo que sí satisfacen las condiciones de optimalidad.

Anexos

A. Fase II y análisis de sensibilidad

A.1. Fase II para problemas de maximización

MATLAB/mSimplexMax.m

```
function [x0, z0, ban, iter, sensinfo] = mSimplexMax(A, b, c,
    imprimirPasos)

%Esta funcion realiza la Fase II del Metodo Simplex para problemas
%que tienen la siguiente forma
%
%                maximizar    c'x
%                sujeto a     Ax <= b , x >= 0 , b >= 0
%
%In :   A ... mxn matrix
%       b ... column vector with as many rows as A
%       c ... column vector with as many entries as one row of A
%       imprimirPasos ... boolean variable which indicates whether
%       or not to print the step-by-step solution of the given
    problem.
%       Setting this parameter equal to true forces the program to
    use
%       a less efficient version of the Revised Simplex Method.
%
%Out:   xo ..... SBF optima del problema
%       zo ..... valor optimo del problema
%       ban .... indica casos:
%           -1 ... si el conjunto factible es vacio
%           0 .... si se encontro una solucion optima
%           1 .... si la funcion objectivo no es acotada.
%       iter ... numero de iteraciones (cambios de variables basicas)
%       que hizo el metodo

    format rat; %MATLAB imprime fracciones en vez de decimales

    iter = 0;
    c = -c;

    %1 Definicion de las variables iniciales

    [m, n] = size(A); %m variables basicas
    ban = 0;
    [N, B, c, A] = deal( 1:n, (n+1):(m+n), [c' zeros(1,m)], [A eye(m)
        ] );
```

```
[lambda, h] = deal( c(B), b );
rN = lambda*A(:, N) - c(N);

%Si el conjunto factible es vacio, el Metodo Simplex no
%tendra opcion mas que escoger un punto que no cumpla
%la restriccion de no-negatividad por lo que algun valor
%de b es negativo. En teoria, no deberiamos tener este
%caso porque la documentacion pide b >= 0.
if any(b < 0)

    if imprimirPasos
        fprintf("\n\nConjunto factible vacio\n");
    end

    ban = -1;

end

if imprimirPasos
   imprimirTableau(A(:, B), A(:, N), c(B), rN, h, B, N, iter);
end

%2 Probamos la condicion de optimalidad
while any(rN > 0) && ban == 0

    %2.1 Seleccionamos la variable de entrada mediante
    %la Regla de Bland
    e = find(rN > 0, 1);

    colEntrada = A(:, B)\A(:, N(e));

    %3 Probamos si el problema es acotado
    if any(colEntrada > 0)

        %3.1 Seleccion de la variable de salida mediante
        %la Regla de Bland

        %Buscamos los indices de los denominadores no-positivos
        noPositivos = colEntrada <= 0;

        %Calculamos los cocientes y asignamos infinito a los
        %cocientes que tienen denominador menor o igual a cero
        cocientes = h./colEntrada;
        cocientes(noPositivos) = inf;

        %s es el indice que corresponde al minimo de cocientes
        [~, s] = min(cocientes);
```

```
        if imprimirPasos
            fprintf(['La variable de entrada es X%d y la '...
                'variable de salida es X%d\n'],[N(e), B(s)]);
        end

        %4 Redefinimos los conjuntos B y N
        [B(s), N(e)] = deal( N(e), B(s) );
        iter = iter + 1;

        %4.1 Calculamos los nuevos costos relativos
        lambda = A(:, B)'\c(B)';
        lambda = lambda';
        rN = lambda*A(:, N) - c(N);
        h = A(:, B)\b;

        if imprimirPasos
            imprimirTableau(A(:, B), A(:, N), c(B), rN, h, B, N,
                iter);
        end

    else

        %El problema es no-acotado.
        ban = 1;
        if imprimirPasos
            fprintf("\n\nProblema no-acotado\n");
        end

    end

end

if ban == 0
    x0 = zeros(m+n, 1);
    x0(B) = h;
    z0 = -c*x0;

    %Solo devolvemos los valores de las variables originales
    x0 = x0(1:n);

    %Calculamos los precios sombra y los intervalos de
    %variacion para el vector c y el vector b
    sensinfo.lambda = -lambda;
    sensinfo.gammas = calcularGammas(n, A(:, B)\A(:, N), rN, B, N
        );
    sensinfo.betas = calcularBetas(h, A(:, B));
```

```
    else
        [x0, z0, sensinfo] = deal( [], [], [] );
    end

    return;

end

function [gammas] = calcularGammas(n, H, r, B, N)

    [~, hn] = size(H);
    gammas = Inf(n, 2);
    gammas(:,1) = -gammas(:,1);

    for var = 1:n

        if(ismember(var, B))

            %La variable es basica.

            %Determinamos la posicion en la que se encuentra la
                variable.
            %Esto nos dice la fila de H que tendremos que recorrer
            i = (B == var);

            for j = 1:hn %Recorremos las columnas de H

                if H(i, j) ~= 0

                    if H(i, j) > 0
                        %En este caso el cociente es negativo
                        gammas(var,1) = max( gammas(var,1), r(j)/H(i,
                            j) );
                    else
                        %En este caso el cociente es positivo
                        gammas(var,2) = min( gammas(var,2), r(j)/H(i,
                            j) );
                    end

                end

            end

        else

            %La variable es no-basica, asi que solo necesitamos
```

```
            % que gamma sea menor o igual que la entrada
            % correspondiente del vector r
            i = (N == var);
            gammas(var, 2) = r(i);

        end

    end

    return;

end

function [betas] = calcularBetas(h, AB)

    m = length(AB); % Numero de filas de AB

    betas = Inf(m, 2);
    betas(:,1) = -betas(:,1);

    % Invertimos la matriz AB con Factorizacion LU
    InvAB = AB\eye(m);

    for j = 1:m % Recorremos las columnas de InvAB

        for i = 1:m % Recorremos las filas de InvAB

            if InvAB(i, j) ~= 0

                if InvAB(i, j) > 0

                    betas(j, 1) = max( betas(j, 1), -h(i)/InvAB(i, j)
                        );

                else

                    betas(j, 2) = min( betas(j, 2), -h(i)/InvAB(i, j)
                        );

                end

            end

        end

    end
```

```
end

function imprimirTableau(AB, AN, cB, rN, h, B, N, iter)

% Este metodo imprime el tableau asociado a la base B
% usando las matrices AB, AN y los vectores rN y h

    m = length(B);
    n = length(N);

    T = zeros(m+1,m+n+1);

    T(1:m, B) = eye(m);
    T(1:m, N) = AB\AN;
    T(1:m, m+n+1) = h;
    T(m+1, N) = -rN;
    T(m+1, m+n+1) = -cB*h;

    fprintf("\nVariables basicas:    ");
    disp(B);
    fprintf("Variables no-basicas: ");
    disp(N);
    fprintf("\nIteracion %d\n\n", iter);
    disp(T);

    return;

end
```

A.2. El problema de los relojes

MATLAB/GraficaProblemaRelojes.m

```
% Este script grafica el conjunto factible del problema de los
    relojes

x = 0:0.5:12;
R1 = 10 - (6/4)*x;
R2 = 10 - 2*x;
R3 = 20/3 - x;
Z = [3; 2];

figure1 = figure;

axes1 = axes('Parent', figure1);
hold(axes1,'on');
axis square;
```

```
% Coloreamos el area factible
A1 = area(x', R2', 'LineStyle', 'none');
A2 = area(x', R3', 'LineStyle', 'none');
A1.FaceColor = [0.8, 0.8, 0.8];
A2.FaceColor = [0.8, 0.8, 0.8];

% Dibujamos las rectas de las restricciones
pR1 = plot(x, R1, 'color', 'r', 'linestyle', '--', 'linewidth', 5,...
    'Parent', axes1);
pR2 = plot(x, R2, 'color', '[0, 0.4470, 0.7410]', 'LineWidth', 5,...
    'Parent', axes1);
pR3 = plot(x, R3, 'color', '[0.4660, 0.6740, 0.1880]', 'LineWidth',
   5,...
    'Parent', axes1);

% Ajustes de los ejes
xlim(axes1,[0 10.5]);
ylim(axes1,[0 10.5]);
grid(axes1,'on');
axis(axes1,'square');
xlabel('$x_1$', 'Interpreter', 'latex');
ylabel('$x_2$', 'Interpreter', 'latex');
set(axes1,'FontSize',20);
[LEGH,OBJH,OUTH,OUTM] = legend([pR1, pR2, pR3],...
    {'R_1: 6x_1 + 4x_2 \leq 40','R_2: 8x_1 + 4x_2 \leq 40',...
    'R_3: 3x_1 + 3x_2 \leq 20'});

% Dibujamos vectores normales
vectorNormalZ = arrow([0 0], [3 2], 'Width', 1, ...
    'NormalDir', [3,2]);
vR2 = arrow([0 0], [4 2], 'Width', 1);
vR3 = arrow([0 0], [10/3 10/3], 'Width', 1);
set(vR2, 'Edgecolor', '[0, 0.4470, 0.7410]',...
    'FaceColor', '[0, 0.4470, 0.7410]');
set(vR3, 'Edgecolor', '[0.4660, 0.6740, 0.1880]',...
    'FaceColor', '[0.4660, 0.6740, 0.1880]');

% Agregamos anotaciones

% textbox
annotation(figure1,'textbox',...
    [0.413132756489831 0.193525284331669....
    0.0705451577801945 0.0836713995943199],...
    'String',{'Vector normal','a la restricción','R_2'},...
    'LineStyle','none',...
    'FitBoxToText','off');
```

```
% textarrow
annotation(figure1,'textarrow',[0.389009802426333
    0.413492927094668],...
     [0.447793338842508 0.29924650161464],...
     'String',{'Vector','normal a la','función objetivo'},...
     'HorizontalAlignment','left');

% textbox
annotation(figure1,'textbox',...
     [0.338323175842277 0.320775036076372...
     0.0664646354733408 0.107633892798424],...
     'String',{'Vector normal','a la restricción R_3'},...
     'LineStyle','none',...
     'FitBoxToText','off');

hold(axes1, 'off');
clearvars
```

MATLAB/SolsProblemaRelojes.m

```
% Este script resuelve las preguntas de las actividades 1 y 3 del
% proyecto usando las funciones mSimplexMax y mSimplexDual

fprintf('\n\nActividad 1\n\n')

% Declaramos el problema y resolvemos mostrando el tableau
A = [6 4; 8 4; 3 3]; b = [40; 40; 20]; c = [300; 200];
[x0, z0, ban, iter, sensinfo] = mSimplexMax(A, b, c, false)
fprintf("\nsensinfo.lambda =\n\n");
disp(sensinfo.lambda);
fprintf("\nsensinfo.gammas =\n\n");
disp(sensinfo.gammas);
fprintf("\nsensinfo.betas =\n\n");
disp(sensinfo.betas);

% P1
fprintf("\n\nPregunta 1\n");
c = [375; 200]
[x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)
c = [375; 175]
[x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

c = [300; 200];

% P2. Ya tenemos guardados los intervalos de sensibilidad para
% el vector c de antes, asi que solo los imprimimos
```

```
fprintf("\n\nPregunta 2\n");

% Reloj de pedestal
fprintf(['\nEl intervalo de sensibilidad ',...
    'para relojes de pedestal es [%s , %s].\n'],...
    strtrim(rats(sensinfo.gammas(1, 1))),...
    strtrim(rats(sensinfo.gammas(1, 2))));
% Reloj de pared
fprintf(['\nEl intervalo de sensibilidad ',...
    'para relojes de pared es [%s, %s]\n'],...
    strtrim(rats(sensinfo.gammas(2, 1))),...
    strtrim(rats(sensinfo.gammas(2, 2))));

% P3
fprintf("\n\nPregunta 3\n");

% David
b = [45; 40; 20]
[x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Diana
b = [40; 45; 20]
[x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Lidia
b = [40; 40; 25]
[x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

% P4. David puede trabajar menos sin afectar la solucion optima.
% Ya tenemos guardados los intervalos de sensibilidad para b de
% antes, asi que solo los volvemos a imprimir
fprintf("\n\nPregunta 4\n");

fprintf(['\nEl intervalo de sensibilidad de las horas de ',...
    'David es [%s, %s).\n'],...
    strtrim(rats(sensinfo.betas(1, 1))),...
    strtrim(rats(sensinfo.betas(1, 2))));

b = [40; 40; 20];

% P5
fprintf("\n\nPregunta 5\n");

for i = 35:2:45
    b(1) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de David: %d \n', i);
```

```
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

b = [40; 40; 20];

% P6
fprintf("\n\nPregunta 6\n");

for i = 35:2:45
    b(2) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de Diana: %d \n', i);
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

b = [40; 40; 20];

% P7
fprintf("\n\nPregunta 7\n");

for i = 15:2:25
    b(3) = i;
    [x0, z0, ~, ~, ~] = mSimplexMax(A, b, c, false);
    fprintf('\nHoras de disponibilidad de Lidia: %d \n', i);
    fprintf('La solucion optima es x0 = (%s, %s)\n',...
        strtrim(rats(x0(1))), strtrim(rats(x0(2))));
    fprintf('La ganancia optima es %s\n', strtrim(rats(z0)));
end

b = [40; 40; 20];

% P8. Si es valido utilizar los precios sombra si solo cambian las
% horas de disponibilidad de Lidia. Lo demostramos a continuacion
fprintf("\n\nPregunta 8\n");

% Precio sombra de Lidia y cambio en las ganancias por el aumento
% en sus horas de disponibilidad semanal
fprintf('\nEl precio sombra de Lidia es %s.\n',...
    strtrim(rats(sensinfo.lambda(3))));
fprintf('\n(Precio sombra de Lidia)*5 = %s.\n',...
    strtrim(rats(sensinfo.lambda(3)*5)));

% Ganancias cuando Lidia trabaja hasta 20 horas semanales
```

```
[~, z_20, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Ganancias cuando Lidia trabaja hasta 25 horas semanales
b(3) = 25;
[~, z_25, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Cambio en las ganancias totales optimas
fprintf(['\nEl cambio en la ganancia total es',...
    ' z_25 - z_20 = %s.\n'], strtrim(rats(z_25 - z_20)));

% No es valido utilizar los precios sombra si cambian las horas
% de disponibilidad de ambos porque altera la solucion optima

% Precios sombra de David y Lidia
fprintf('\nEl precio sombra de David es %s.\n',...
    strtrim(rats(sensinfo.lambda(1))));
fprintf('\nEl precio sombra de Lidia es %s.\n',...
    strtrim(rats(sensinfo.lambda(3))));

% Como el precio sombra de David es cero, la multiplicacion
% del cambio de horas y sus precios sombra es igual a
% la multiplicacion del precio sombra de Lidia por su
% cambio de horas
fprintf(['\nLa multiplicación de los precios sombra de David ', ...
    'y de Lidia \npor el cambio en sus horas es igual a %s.\n'],...
    strtrim(rats(sensinfo.lambda(3)*5)));

% Ganancias con b = [40, 40, 20]
[~, z0, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Ganancias despues del cambio en las horas de David y Lidia
b(1) = b(1) - 5; b(3) = b(3) + 5;
[~, z1, ~, ~, ~] = mSimplexMax(A, b, c, false)

% Cambio en las ganancias totales optimas
fprintf(['\nEl cambio en la ganancia total es',...
    ' z1 - z0 = %s.\n\n'], strtrim(rats(z1 - z0)));

fprintf('\nActividad 3\n\n');

A = [6 8 3; 4 4 3]; b = [300; 200]; c = [40; 40; 20];
[x0, z0, ban, iter, lambda0] = mSimplexDual(A, b, c, false)
```

B. El Método Simplex Dual

B.1. Fase II del Método Simplex Dual y el problema de los relojes

MATLAB/mSimplexDual.m

```
function [x0, z0, ban, iter, lambda0] = mSimplexDual(A, b, c,
    imprimirPasos)

%Esta funcion realiza el Metodo Simplex Dual para problemas (duales)
%que tienen la siguiente forma:
%
%                 minimizar     c'x
%                 sujeto a      Ax >= b , x >= 0 , c >= 0
%
%In :   A ... m x n matrix
%       b ... column vector with as many rows as A
%       c ... column vector with as many columns as A
%
%Out:   x0 ... SFB optima del problema
%       z0 ... valor optimo del problema
%       ban ... indica casos:
%           -1 ... si el conjunto factible es vacio
%            0 ... si se encontro una solucion optima
%            1 ... si la funcion objectivo no es acotada.
%       iter ... numero de iteraciones que hizo el metodo
%       lambda0 ... Solucion del problema dual

    format rat; %MATLAB imprime fracciones en vez de decimales

    ban = 0;
    iter = 0;
    [x0, z0, lambda0] = deal( [], [], [] );

    %1 Definicion de las variables iniciales

    [m, n] = size(A); %m variables basicas
    [N, B, c, A] = deal( 1:n, (n+1):(m+n), [c' zeros(1,m)], [-A eye(m
        )] );
    [lambda, h] = deal( c(B), -b );
    rN = lambda*A(:, N) - c(N);
    Id = eye(m); %matriz identidad

    %Revisamos si el problema es no acotado
    if any(c < 0)

        %Si c < 0, entonces el primal de este problema será
        %no-factible.

        if imprimirPasos
            fprintf("\n\nConjunto factible vacio\n");
```

```
    end

    ban = -1;

end

if imprimirPasos
  imprimirTableau(A(:, B), A(:, N), c(B), rN, h, B, N, iter);
end

% 2 Probamos la condicion de optimalidad
while any(h < 0) && ban == 0

    % 2.1 Seleccionamos la variable de salida mediante
    % la Regla de Bland

    s = find(h < 0, 1);

    filaSalida = (Id(s, :)/A(:, B))*A(:, N);

    % 3 Probamos si el problema es acotado
    if any(filaSalida < 0)

        % 3.1 Seleccion de la variable de entrada mediante
        % la Regla de Bland

        % Buscamos los indices de los denominadores no-negativos
        noNegativos = filaSalida >= 0;

        % Calculamos los cocientes y asignamos infinito a los
        % cocientes que tienen denominador mayor o igual a cero
        cocientes = rN./filaSalida;
        cocientes(noNegativos) = inf;

        % s es el indice que corresponde al minimo de cocientes
        [~,e] = min(cocientes);

        if imprimirPasos
            fprintf(['La variable de entrada es X%d y la '...
                'variable de salida es X%d\n'],[N(e), B(s)]);
        end

        % 4 Redefinimos los conjuntos B y N
        [B(s), N(e)] = deal( N(e), B(s) );
        iter = iter + 1;

        % 4.1 Calculamos los nuevos costos relativos
```

```
            lambda = A(:, B)'\c(B)';
            lambda = lambda';
            rN = lambda*A(:, N) - c(N);
            h = -A(:, B)\b;

            if imprimirPasos
                imprimirTableau(A(:, B), A(:, N), c(B), rN, h, B, N,
                    iter);
            end

        else

            % El problema es no-acotado.
            ban = 1;
            if imprimirPasos
                fprintf("\n\nProblema no-acotado\n");
            end

        end

    end

    if ban == 0

        x0 = zeros(n + m, 1);
        x0(B) = h;
        lambda0 = -lambda';
        z0 = c*x0;

        % Solo devolvemos los valores de las variables originales
        x0 = x0(1:n);

    end

    return;

end

function imprimirTableau(AB, AN, cB, rN, h, B, N, iter)

% Este metodo imprime el tableau asociado a la base B
% usando las matrices AB, AN y los vectores rN y h

    m = length(B);
    n = length(N);

    T = zeros(m+1,m+n+1);
```

```
    T(1:m, B) = eye(m);
    T(1:m, N) = AB\AN;
    T(1:m, m+n+1) = h;
    T(m+1, N) = rN;
    T(m+1, m+n+1) = cB*h;

    fprintf("\nVariables basicas:     ");
    disp(B);
    fprintf("Variables no-basicas: ");
    disp(N);
    fprintf("\nIteracion %d\n\n", iter);
    disp(T);

end
```