

Modelos y Optimización I

María Inés Parnisari

7 de enero de 2012

Índice

1. Definiciones	2
2. Método Simplex	2
3. Problemas	4
4. Modelización	10
5. Heurísticas	10

1 Definiciones

Investigación operativa: aplicación de la ciencia moderna a problemas complejos que aparecen en la administración de sistemas formados por personas, materiales, equipos y dinero. Su característica es la elaboración de modelos científicos que mediante la incorporación de factores de riesgo e incertidumbre permitan evaluar decisiones.

1.1 Supuestos básicos de la programación lineal

1. **Proporcionalidad:** el beneficio y el uso de recursos son directamente proporcionales al nivel de actividad.
2. **Aditividad:** no existen interacciones entre las actividades que cambien la medida total de la efectividad o el uso total de algún recurso.
3. **Divisibilidad:** pueden permitirse valores fraccionarios.
4. **Certeza:** todos los parámetros del modelo son constantes conocidas.

1.2 Elementos de un modelo

- **Análisis del problema**
- **Objetivo:** ¿qué hacer? ¿cuándo? ¿para qué?
- **Hipótesis:** supuestos aceptados como verdaderos
- **Actividad:** procesos que se realizan en el sistema, caracterizados por consumir recursos y generar resultados económicos
- **Variables:** indican el estado de una actividad
 - Continuas: miden
 - Enteras: indican

2 Método Simplex

Algoritmo 1 Método simplex

1. Transformar las desigualdades en igualdades, agregando variables “slacks” o variables artificiales
 2. Mover las constantes a los 2dos miembros
 3. Armar la tabla
 4. Mientras hay algún $z_j - c_j$ negativo (z_{max}) o positivo (z_{min}):
 - a) Elegir la variable que entra:
 - 1) Menor $z_j - c_j$ (z_{max})
 - 2) Mayor $z_j - c_j$ (z_{min})
 - b) Elegir la variable que sale:
 - 1) Menor θ positiva (o $\theta = 0$ si $a_{ij} > 0$)
 - c) Armar nueva tabla:
 - 1) Dividir la fila del pivote por el pivote
 - 2) Aplicar el método del rectángulo para las demás filas
 - 3) $z_{nuevo} = z_{viejo} - \theta_{min} \cdot (z_j - c_j)$ variable que entra a la base
-

2.1 Teoremas

Teorema 1. El conjunto de todas las soluciones factibles a un problema de programación lineal es un poliedro convexo.

Teorema. El funcional alcanza su mínimo (o máximo) en un punto extremo del poliedro convexo.

Teorema 2. El número de vértices del conjunto factible del problema es:

$$\binom{m+n}{m} = \frac{(m+n)!}{m!(m+n-m)!}$$

donde n es la cantidad de variables reales y m es la cantidad de ecuaciones.

2.2 Análisis de variables

C_j	Coefficiente en el funcional de cada X_j
X_j	Variable que forma parte de la base (nivel de actividad j)
B_j	Valor de X_j en la base (las que no están valen 0)
a_{ij}	Cuánto disminuye el valor de X_i [fila] por cada unidad de X_j [columna] que decida fabricarse o que sobre
$z_j - c_j$	Costo de oportunidad si X_j está en la base; Valor marginal de un X_j "slack"
Y_j	Valor marginal del recurso j

Recurso saturado: recurso con cero sobrante. Equivale a tener una variable slack $x_j = 0$.

Lucro cesante: $\sum_{\forall i} \text{UsorRecurso}_i \cdot \text{ValorMarginalRecurso}_i$

	Valor marginal	Costo de oportunidad
Variable real		Cuánto va a empeorar el funcional si debemos fabricar una unidad de ese producto
Variable slack	Cuánto va a mejorar el funcional si se tiene una unidad más de ese recurso	

2.3 Análisis de sensibilidad

El **análisis de sensibilidad** se efectúa en todo estudio de programación lineal porque la mayoría de los parámetros utilizados en el modelo suelen ser solamente estimaciones de condiciones futuras.

1. Variación de los c_j

El rango para que la solución óptima siga siéndolo es c_j tal que $z_j - c_j \geq 0$ (z_{max})

2. Variación de los b_j

El análisis se hace tomando la tabla del problema dual y analizando los c_j

3. Variación simultánea de varios b_j

El análisis se hace tomando la tabla del problema dual y analizando los b_j

4. Variación simultánea de varios c_j y varios b_j

Primero analizar los cambios de b_j en el problema dual, y luego hacer el dual del dual para analizar los cambios de c_j

5. Curva de oferta

Es un gráfico de la variación de cada X_j en función de C_j

2.4 Dualidad

Primal	Dual
Maximizar: $z = \sum_{j=1}^n C_j \cdot X_j$	Minimizar: $W = \sum_{i=1}^m B_j \cdot Y_i$
Sujeto a: $X_j \geq 0 \forall j$ $\sum_{j=1}^n a_{ij} \cdot X_j \leq B_j \forall i$	Sujeto a: $Y_i \geq 0 \forall i$ $\sum_{i=1}^m a_{ij} \cdot Y_i \geq C_j \forall j$

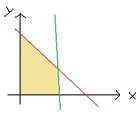
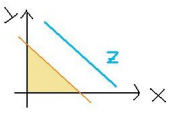
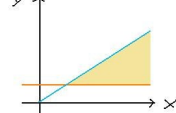
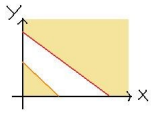
2.4.1 Teorema fundamental de la dualidad

1. X tiene solución óptima finita $\iff Y$ tiene solución óptima finita
2. X es poliedro abierto $\iff Y$ es incompatible
3. X es incompatible $\iff (Y \text{ es incompatible}) \vee (Y \text{ es poliedro abierto})$

2.4.2 Modificaciones al problema original

1. Agregar un producto:
 - a) La columna del producto en la tabla óptima será: $y = A \cdot x$ donde A es la matriz inversa óptima (tiene las columnas de la tabla óptima que corresponden a la matriz identidad del 1er paso, ordenados¹), y x es el vector de coeficientes del producto.
 - b) Con la nueva tabla formada, calcular el $z_j - c_j$ y proseguir normalmente.
2. Cambios en los C_j
Reemplazar en la tabla final en todos los lugares que aparezcan y recalcular los $z_j - c_j$ involucrados.
3. Cambios en los B_j
Realizar (2) tomando la tabla del dual.
4. Agregar una restricción
 - a) Verificar si la nueva restricción no afecta a la solución actual.
 - b) Si afecta, realizar (1) tomando la tabla del dual. Recordar que si había un coeficiente negativo en el funcional del primal, habrá un coeficiente de x negativo.

2.5 Tipos de soluciones

	Punto degenerado	Solución alternativa	Poliedro abierto	Incompatible
Cuándo	Acumulación de vértices en un punto	Hay otro vértice del poliedro con el mismo z	No existe solución óptima	No existe solución factible
En Simplex	Tabla anterior: empate de θ mínimos. Tabla actual: $B_k = 0$ para una variable de la base	$z_j - c_j = 0$ de una variable que no está en la base	Hay una variable que quiere entrar a la base pero ninguna puede salir	En la tabla óptima hay una variable artificial en la base con $B_k \neq 0$
Gráfico				

3 Problemas

3.1 Problema de análisis de actividad

Datos:

- n actividades
- m recursos

¹Si la matriz identidad del 1er paso incluye a variables artificiales, no utilizar éstas, sino las reales cambiadas de signo.

- B_i =disponibilidad del recurso i
- a_{ij} =cantidad de recurso i que consume la actividad j
- c_i =ganancia unitaria de la actividad i

Problema: encontrar la intensidad de trabajo de las actividades, para maximizar la ganancia total.

Definición de variables:

- x_i =intensidad de la actividad i

Funcional:

- $Z_{max} = \sum_{i=1}^n c_i \cdot x_i$

Modelización:

1. La cantidad de recursos utilizados no puede superar la disponibilidad del recurso

$$\sum_{j=1}^n a_{ij} \cdot x_j \leq B_i \text{ para todo } i$$
2. Condición de no negatividad

$$x_i \geq 0 \text{ para todo } i$$

3.2 Problema de centros de producción

Problema: determinar cuánto producir y cuánto insumo consumir en cada centro de producción.

Modelización:

1. Relación E/S:

$$MERMA \cdot \sum Entrada_i C_k = \sum Salida_i C_k$$
2. Mezcla a la entrada:

$$\%dDE_i EN_k \cdot \sum Entrada_i C_k = Entrada_i C_k$$
3. Mezcla a la salida:

$$\%DE_i EN_k \cdot \sum Salida_i C_k = Salida_i C_k$$
4. Capacidad productiva:

$$HS.PORkG.C_k \cdot \sum Entrada_i C_k \leq Cap.Hs.C_k$$

3.3 Problema del viajante

Datos:

- n ciudades
- C_{ij} =costo de ir de la ciudad i a la ciudad j
- Simetría ($C_{ij} = C_{ji}$) o asimetría ($C_{ij} \neq C_{ji}$) del problema

Problema: encontrar el orden en el cual un viajante visita las ciudades para minimizar la distancia total recorrida. Cada ciudad debe ser visitada solo una vez, y al terminar el recorrido debe volverse a la ciudad de origen.

Definición de variables:

- $Y_{ij} = \begin{cases} 1 & \text{si va de la ciudad } i \text{ a la ciudad } j \text{ directamente} \\ 0 & \text{si no} \end{cases}$
- U_i =orden de visita de la ciudad i , $\{1, 2, \dots, n\}$

Funcional:

- $Z_{min} = \sum_{i=0}^n \sum_{j=0}^n C_{ij} \cdot Y_{ij}$

Modelización:

1. Hay solo una ciudad después de i
 $\sum_{j=0}^n Y_{ij} = 1, \forall i, i \neq j$
2. Hay solo una ciudad antes de j
 $\sum_{i=0}^n Y_{ij} = 1, \forall j, i \neq j$
3. Evitamos subtours
 $U_i - U_j + n \cdot Y_{ij} \leq n - 1, \forall i, j, i \neq j$

3.4 Problema de distribución

Datos:

- m Orígenes / Suministros
- n Demandas / Destinos
- C_{ij} = costo de enviar el producto desde el origen i al destino j
- S_i = cantidad de producto disponible en el suministro i
- D_i = cantidad de producto demandada por el destino i

Hipótesis:

- Cualquier origen puede enviar producto a cualquier destino
- Producto homogéneo
- Costos lineales

Problema: determinar qué origen le envía a cuál destino para minimizar los costos de transporte.

Definición de variables:

- X_{ij} = cantidad de productos enviados desde el origen i al destino j

Funcional:

- $Z_{min} = \sum_{i=1}^m \sum_{j=1}^n C_{ij} \cdot X_{ij}$

Modelización:

1. La cantidad enviada desde S_i debe ser menor o igual a su disponibilidad
 $\sum_{j=1}^n X_{ij} \leq S_i$ para cada i
2. La cantidad que llega a D_i debe ser mayor o igual a su demanda
 $\sum_{i=1}^m X_{ij} \geq D_j$ para cada j
3. Condición de no negatividad
 $X_{ij} \geq 0$ para cada i, j

3.5 Problema de transbordo

Datos:

- o Orígenes
- d Destinos
- t Transbordos
- CO_iT_j = costo de enviar un producto desde el origen i al transbordo j
- CT_iD_j = costo de enviar un producto desde el transbordo i al destino j

Hipótesis:

- Los transbordos tienen capacidad de stock ilimitada
- Costos lineales

Problema: determinar qué origen le envía a cuál transbordo, y qué transbordo le envía a cuál destino, para minimizar los costos de transporte.

Definición de variables:

- XO_iT_j = cantidad enviada desde el origen i al transbordo j
- XT_iD_j = cantidad enviada desde el transbordo i al destino j

Funcional:

- $Z_{min} = \sum_{i=1}^o \sum_{j=1}^t CO_iT_j \cdot XO_iT_j + \sum_{i=1}^t \sum_{j=1}^d CT_iD_j \cdot XT_iD_j$

Modelización:

1. La cantidad enviada desde O_i debe ser menor o igual a su disponibilidad
$$\sum_{j=1}^t XO_iT_j \leq O_i \text{ para cada } i$$
2. La cantidad que llega a D_i debe ser mayor o igual a su demanda
$$\sum_{i=1}^t XT_iD_j \geq D_j \text{ para cada } j$$
3. Todo lo que entra a un transbordo, es enviado a un destino
$$\sum_{i=1}^o XO_iT_j = \sum_{k=1}^d XT_jD_k \text{ para cada } j$$
4. Condición de no negatividad
$$XO_iT_j \geq 0 \text{ para cada } i, j$$
$$XT_iD_j \geq 0 \text{ para cada } i, j$$

3.6 Problema de la dieta

Datos:

- m Comidas
- n Nutrientes
- R_i = requerimiento mínimo diario del nutriente i
- P_i = precio por unidad de la comida i
- c_{ij} = cantidad de nutriente j en la comida i

Problema: determinar una dieta que satisfaga los requisitos alimenticios a un costo mínimo.

Definición de variables:

- Y_i = cantidad de comida i a comprar por día

Funcional:

- $Z_{min} = \sum_{i=1}^m P_i \cdot Y_i$

Modelización:

1. Se deben satisfacer los requerimientos alimenticios
 $\sum_{i=1}^m c_{ij} \cdot Y_i \geq R_j$ para cada j
2. Condición de no negatividad
 $Y_i \geq 0$ para cada i

3.7 Problema del secuenciamiento de máquinas (Job Shop Scheduling)

Datos:

- n trabajos
- m máquinas

Problema: determinar en qué orden y en qué máquinas se procesarán todos los trabajos para minimizar el tiempo que lleva terminar todos los trabajos.

3.8 Problema de asignación

Datos:

- p personas
- t trabajos
- a_{ij} = ganancia que se obtiene cuando la persona i está con el trabajo j

Problema: determinar qué personas realizarán qué trabajos.

Definición de variables:

- $x_{ij} = \begin{cases} 1 & \text{si la persona } i \text{ se asigna al trabajo } j \\ 0 & \text{si no} \end{cases}$

Funcional:

- $Z_{max} = \sum_{i=1}^p \sum_{j=1}^t a_{ij} \cdot x_{ij}$

Modelización:

1. Un trabajo por persona
 $\sum_{j=1}^t x_{ij} = 1$ para cada i
2. Una persona por trabajo
 $\sum_{i=1}^p x_{ij} = 1$ para cada j

3.9 Problema de asignación cuadrática

Datos:

- m fábricas
- m ciudades
- d_{ij} = distancia de la ciudad i a la ciudad j
- c_{ij} = capacidad de transporte de la fábrica i a la fábrica j

Problema: determinar qué fábricas se instalarán en qué ciudades para minimizar la suma de las distancias multiplicado por la capacidad de transporte.

Definición de variables:

- $c_{ijkl} = c_{ij} \cdot d_{kl}$
- $x_{ij} = \begin{cases} 1 & \text{si la fábrica } i \text{ se instala en la ciudad } j \\ 0 & \text{si no} \end{cases}$
- $Y_{ijkl} = \begin{cases} 1 & \text{si } x_{ij} = x_{kl} = 1 \\ 0 & \text{si no} \end{cases}$

Funcional:

- $Z_{min} = \sum_{i=1}^m \sum_{j=1}^m \sum_{k=1}^m \sum_{l=1}^m Y_{ijkl} \cdot c_{ijkl}$ para todo $i \neq k$ y para todo $j \neq l$

Modelización:

1. Valor para Y_{ijkl}
2. $2 \cdot Y_{ijkl} \leq x_{ij} + x_{kl} \leq Y_{ijkl} + 1$

3.10 Problema de la mochila

Datos:

- k objetos
- p_k = peso del objeto k
- v_k = valor económico del objeto k
- P = peso máximo de la mochila

Problema: determinar qué objetos meter dentro de la mochila para satisfacer la restricción de peso, y maximizar el valor de los objetos guardados.

Definición de variables:

- $x_i = \begin{cases} 1 & \text{si el objeto } i \text{ se guarda en la mochila, } i = \{1, \dots, k\} \\ 0 & \text{si no} \end{cases}$

Funcional:

- $Z_{max} = \sum_{i=1}^k v_i \cdot x_i$

Modelo:

1. El peso de los objetos de la mochila no debe superar el peso máximo permitido:
$$\sum_{i=1}^k p_i \cdot x_i \leq P$$

4 Modelización

$Y_{AND} = x_1 \wedge x_2 \wedge \dots \wedge x_n$	$n \cdot Y_{AND} \leq \sum_{i=1}^n x_i \leq Y_{AND} + n - 1$
$Y_{OR} = x_1 \vee x_2 \vee \dots \vee x_n$	$Y_{OR} \leq \sum_{i=1}^n x_i \leq n \cdot Y_{OR}$
XOR	$\sum_{i=1}^n x_i \leq 1$
$x > 0$	$m \cdot Y_{NN} \leq x \leq M \cdot Y_{NN}$
$MAX = \max(x_1, x_2, \dots, x_n)$	$x_i \leq MAX \leq x_i + M \cdot Y_i \quad i = \{1, \dots, n\}$ $\sum_{i=1}^n Y_i = n - 1,$ Y_i es bivalente
$MIN = \min(x_1, x_2, \dots, x_n)$	$x_i - M \cdot Y_i \leq MIN \leq x_i \quad i = \{1, \dots, n\}$ $\sum_{i=1}^n Y_i = n - 1,$ Y_i es bivalente
$Y \in [c_1, c_2]$ o $Y \in [c_3, c_4]$... o $Y \in [c_n, c_{n+1}]$	$c_1 \cdot x_1 + c_3 \cdot x_3 + \dots + c_n \cdot x_n \leq Y \leq c_2 \cdot x_1 + c_4 \cdot x_3 + \dots + c_{n+1} \cdot x_n$ $\sum_{i=1}^n x_i = 1,$ x_i es bivalente
$Y = \text{round}(x)$	$Y - (0,5 - m) \leq x \leq Y + 0,5$
$x \leq 0, x \neq 0$	$-m \cdot (1 - Y_{POS}) + m \cdot Y_{POS} \leq x \leq M \cdot Y_{POS}$
Restricciones financieras	$\$INI + INGRESOS - EGRESOS - \$FIN = EXCESO - DEFECTO$
Costos dependientes de la cantidad de producto	$Q = \text{cantidad de producto (u.)}$ $C_i = \text{costo } i \text{ (\$)}, i = \{1, \dots, n\}, n \text{ intervalos}$ $Q = Q_1 + Q_2 + \dots + Q_n$ $COSTO_Q = \$C_1 \cdot Q_1 + \$C_2 \cdot Q_2 + \dots + \$C_n \cdot Q_n$ $\$min_1 \cdot x_1 \leq Q_1 \leq \$max_1 \cdot x_1$ $\$min_2 \cdot x_2 \leq Q_2 \leq \$max_2 \cdot x_2$ \vdots $\$min_n \cdot x_n \leq Q_n \leq \$max_n \cdot x_n$ $\sum_{i=1}^n x_i = 1$
Continuidad de inventarios	$stock_{inicial} + produccion = demanda + stock_{sobrante}$

5 Heurísticas

Heurística: técnica que aumenta la eficiencia de un proceso de búsqueda, posiblemente sacrificando demandas de completitud.

- **Heurísticas de construcción:** se utilizan para encontrar una solución al problema
- **Heurísticas de mejoramiento:** parten de una solución conocida y tratan de mejorarla

Función heurística: correspondencia entre las descripciones de estados del problema hacia alguna medida de deseabilidad, generalmente representada por números.

$$HEUR \leq K \cdot OPTIMO$$

donde:

- $HEUR$ = valor óptimo encontrado por la heurística
- K = constante de calidad
- $OPTIMO$ = valor óptimo real

Si para una función heurística puede encontrarse un valor de K que permanezca constante para todos los problemas de un mismo tipo, esta heurística tendrá **garantía de calidad**.

5.1 Análisis de características de un problema para elegir el mejor método

1. ¿Puede descomponerse el problema?

2. ¿Pueden deshacerse o ignorarse pasos hacia una solución?
3. ¿El universo es predecible?
4. Una solución adecuada, ¿es absoluta o relativa?
5. La solución, ¿es un estado o una ruta?
6. ¿Cuál es el papel del conocimiento?
7. ¿La tarea necesita interactuar con la persona?

5.2 Ejemplos de heurísticas

5.2.1 Para el problema del viajante

De construcción:

1. Heurística del vecino más próximo: consiste en seleccionar en cada paso la ciudad que se encuentre más cerca de la actual. Este tipo de algoritmos también se conocen como *greedy*, porque en cada paso hacen lo que sea mejor, sin tener en cuenta estados anteriores o posteriores. Es muy fácil de implementar, pero puede arrojar soluciones muy lejanas a la óptima.
2. Heurística del emparchamiento más cercano.
3. Heurística de la inserción más cercana.
4. Heurística de barrido: se rota una semirrecta alrededor del centro del plano y se determina el tour a medida que la recta vaya tocando ciudades.
5. Heurística con curvas de llenado.
6. Heurística con algoritmos genéticos.

De mejoramiento:

1. Heurística de los K-intercambios: se toma una solución factible y se trata de mejorarla cambiando K ejes de la solución. La complejidad de este algoritmo es de n^k .
2. Heurística de recocido simulado: se evalúan algunos vecinos del estado actual y probabilísticamente se decide entre efectuar una transición a un nuevo estado o quedarse en el estado actual.

5.2.2 Para el problema de secuenciamiento

Técnicas de despacho:

1. SPT (*Shortest processing time*)
2. FCFS (*First come, first served*)
3. MWKR (*Most work remaining*)
4. LWKR (*Least work remaining*)
5. MOPNR (*Most operation remaining*)

5.2.3 Para el problema de distribución

1. Regla del Noroeste
2. Costos mínimos progresivos
3. Método "VAM"

5.3 Metaheurísticas

Metaheurística: método de solución general que proporciona una estructura general y criterios estratégicos para desarrollar un método heurístico específico que se ajuste a un tipo particular de problema.

Ejemplos:

- *GRASP: Greedy Randomized Adaptative Search Procedure.* Para resolver problemas combinatorios. En cada iteración hay dos fases: construcción y búsqueda local. En la fase de construcción se construye una solución factible, y en la búsqueda local se examinan las soluciones vecinas hasta encontrar un mínimo local.
- Algoritmos genéticos
- *Ant Colony Optimization*
- *Simulated Annealing*
- *Taboo Search*