Fundamentos de Programación Entera

A. Revisión de fundamentos

Carlos Testuri – Fernando Islas

Departamento de Investigación Operativa – Instituto de Computación
Facultad de Ingeniería – Universidad de la República

2012–2025

Contenido

1. Revisión: algebra lineal, convexidad y programación lineal
   - Algebra lineal y geometría
   - Conjunto convexo, combinación convexa y casco convexo
   - Programación lineal, resolución, dualidad
   - Algoritmos

Producto interno

El producto interno de los vectores $v, w \in \mathbb{R}^n$ tiene como resultado el escalar

$$v_1 w_1 + \ldots + v_n w_n = \sum_{i=1}^{n} v_i w_i = v^\tau w.$$

Para el caso del espacio Euclidiano, el producto interno de los vectores es

$$v^\tau w = \|v\| \|w\| \cos(\theta)$$

donde θ es el ángulo entre v y w.

1. ¿Cuándo vale cero el producto interno de dos vectores?
2. ¿Cuándo es máximo el producto interno de dos vectores?
3. ¿Cuándo es mínimo el producto interno de dos vectores?

Producto interno y ecuaciones (1/2)

Dado el vector constante $a \in \mathbb{R}^n$.
¿Qué región del espacio describe el vector variable $x \in \mathbb{R}^n$ tal que

1. $a^\tau x = 0$?

2. $a^\tau x \geq 0$?

3. $a^\tau x \leq 0$?

Producto interno y ecuaciones (2/2)

Dados los vectores constantes $a, d \in \mathbb{R}^n$.
¿Qué región del espacio describe el vector variable $x \in \mathbb{R}^n$ tal que

$$a^\tau (x + d) = 0 \, ?$$

Equivalentemente:

$$a^\tau x = -a^\tau d$$

Sea $b := -a^\tau d$,

$$a^\tau x = b$$

Hiperplano y semiespacio

Dados el vector $a \in \mathbb{R}^n$, el escalar $b \in \mathbb{R}$ y el vector variable $x \in \mathbb{R}^n$ se define el *hiperplano afín*

$$a^\tau x = \sum_{i=1}^n a_i x_i = b,$$

y los *semiespacios afines*

$$a^\tau x = \sum_{i=1}^n a_i x_i \leq b,$$
$$a^\tau x = \sum_{i=1}^n a_i x_i \geq b.$$

Conjunto convexo

Definición:
Sea C un subconjunto de $\mathbb{R}^n$. Se dice que C es convexo si para todo $x, y \in C$, y para todo $\lambda \in [0, 1]$, se tiene que

$$\lambda x + (1 - \lambda)y \in C.$$

Combinación convexa y Casco convexo

Sean $x_1, ..., x_k$ vectores de $\mathbb{R}^n$, y $\lambda_1, ..., \lambda_k$ escalares no negativos cuya suma es uno,

- se denomina *combinación convexa* de $x_1, ..., x_k$ al vector $\sum_{i=1}^{k} \lambda_i x_i$
- se denomina *casco convexo* de los vectores de $x_1, ..., x_k$, al conjunto de todas sus combinaciones convexas. Dicho conjunto se denota $\text{conv}(\{x_1, ..., x_k\})$.

Convexidad: propiedades

- Un hiperplano es un conjunto convexo
- Un semiespacio es un conjunto convexo
- La intersección de conjuntos convexos es convexa.
- El conjunto $P := \{x \in \mathbb{R}^n : Ax \leq b\}$, denominado *poliedro*, es un conjunto convexo.
- La combinación convexa de un número finito de elementos de un conjunto convexo pertenece al conjunto.
- El casco convexo de un número finito de vectores es un conjunto convexo.

Función convexa

Sea C subconjunto convexo de $\mathbb{R}^n$. Se dice que la función $f : C \to \mathbb{R}$ es convexa si para todo $x, y \in C$, y para todo $\lambda \in [0, 1]$, se tiene que

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y).$$

Programación Lineal

Dados los parámetros: matriz $A \in \mathbb{R}^{m \times n}$ y los vectores columnas $b \in \mathbb{R}^m$ y $c \in \mathbb{R}^n$; y dada la variable de decisión: vector columna $x \in \mathbb{R}^n$, se define el problema de programación lineal (estándar) como

$$\begin{array}{ll} \min & c^\tau x \\ \text{s.a} & Ax = b \\ & x \geq 0. \end{array}$$

Se dice que x^* es $\begin{cases} \textit{solución factible,} & \text{si } Ax^* = b \text{ y } x^* \geq 0 \\ \textit{solución óptima,} & \text{si es factible y } c^\tau x^* \leq c^\tau x, \ \forall x \text{ factible.} \end{cases}$

Dada una instancia del problema, definida según valores de los parámetros A, b y c, se dice que la misma es

- *factible*: cuando su conjunto factible no es vacío,
- *no factible*: cuando su conjunto factible es vacío,
- *no acotada*: cuando el valor del óptimo es no acotado.

Programación Lineal: Equivalencias en formulación

Equivalencias

Dada la fila i-ésima de la matriz A, a_i,
1. la restricción de igualdad $a_i^\tau x = b_i$ es equivalente a la conjunción de las restricciones $a_i^\tau x \leq b_i$ y $a_i^\tau x \geq b_i$,
2. la restricción $a_i^\tau x \leq b_i$ es equivalente a $-a_i^\tau x \geq -b_i$.
3. el objetivo: $\min c^\tau x$, es equivalente a objetivo: $-\max -c^\tau x$.

Reducciones

1. *eliminación de variable libre*: la variable x_j no restringida en signo se reemplaza por la diferencia de dos variables no negativas, $x_j^+ - x_j^-$.
2. *eliminación de restricción de desigualdad*: dada $\sum_{j=1}^n a_{ij}x_j \leq b_i$, se agrega una nueva variable (de *holgura*), $s_i \geq 0$, tal que $\sum_{j=1}^n a_{ij}x_j + s_i = b_i$.
Similarmente, para la desigualdad opuesta, se agrega una variable de *exceso* que se resta a la restricción.

Programación Lineal: Bases y condiciones de factibilidad y optimalidad

¿Cómo resolver el problema de programación lineal?

$$\begin{array}{ll} \min & c^\tau x \\ \text{s.a} & Ax = b \\ & x \geq 0. \end{array}$$

En el poliedro $P = \{x : Ax = b, x \geq 0\}$ las soluciones están determinadas por los puntos extremos y rayos extremos.

Particionando $A = [B, N]$, donde B es *base*; además, $x = [x_B, x_N]$ y $c = [c_B, c_N]$ se tiene la solución básica $x_B = B^{-1}b, x_N = 0$, con valor $c_B^\tau B^{-1}b$.

Condición de *factibilidad*, dada por $x_B = B^{-1}b \geq 0$.

Condición de *optimalidad*, dada por $c_N^\tau - c_B^\tau B^{-1}N \geq 0$.

Programación Lineal: Algoritmo simplex

Iteración de algoritmo simplex

1. Obtener la base $B := A_{B(1)}, ..., A_{B(m)}$, calcular su inversa B^{-1} y su solución básica $x_B := B^{-1}b$.
2. Calcular el valor dual $p^\tau := c_B^\tau B^{-1}$ y los costos reducidos $\widehat{c}_j := c_j - p^\tau A_j$, para todo j. Si todo $\widehat{c}_j \geq 0$, la solución es óptima, *Parar*; sino, seleccionar algún j para el que $\widehat{c}_j < 0$.
3. Calcular $u := B^{-1}A_j$. Si ningún componente de u es positivo, el valor óptimo es $-\infty$, el problema es no acotado, *Parar*.
4. Si alguno de los componentes de u es positivo, calcular $\theta^* := \min_{\{i=1,...,m:u_i>0\}} x_{B(i)}/u_i$.
5. Sea k el índice donde $\theta^* = x_{B(k)}/u_k$. Construir la nueva base reemplazando $A_{B(k)}$ con A_j. Ir al paso 1.

Si no hay base de partida se utiliza una variación que implica obtenerla previamente (método de dos fases).

Programación Lineal: Algoritmo simplex resumen

El método itera de un punto extremo a otro reduciendo el valor de la función objetivo hasta que llega a la solución óptima.

La codificación computacional del método pone especial énfasis en el cálculo de la inversa de la base y en evitar quedar atrapado en secuencias de iteración con cambio de base sin reducción de valor en la función objetivo, debido a la presencia de *degeneración*.

La complejidad computacional (peor caso) es exponencial en el número de variables y restricciones.
En la práctica es muy eficiente, en promedio la complejidad es proporcional al número de restricciones.

Programación Lineal: Dualidad

Propiedad algebraica que establece para un problema, denominado primal, otro problema equivalente, denominado dual, en términos estructurales. Las variables de un problema están asociadas a las restricciones del otro problema.

Primal		*Dual*	
min	$c^\tau x$	max	$p^\tau b$
s.a	$Ax = b$	s.a	$p^\tau A \leq c^\tau$.
	$x \geq 0$.		

En la formulación dual las variables p pueden interpretarse como el beneficio marginal en el valor del óptimo del problema primal que se obtendría al aumentar los términos independientes de las restricciones correspondientes del problema primal, b. Viceversa, en la formulación primal sobre las variables x.

Formulación general del problema dual

Sean $\mathbf{a}_i$ la fila i-ésima y $\mathbf{A}_j$ la columna j-ésima de la matriz $\mathbf{A}$.
La conversión estructural del problema primal de minimización sobre variable $\mathbf{x}$ al problema dual de maximización sobre variable $\mathbf{p}$ queda establecida en términos de restricciones y variables por

Primal					*Dual*			
min	$\mathbf{c}^\tau \mathbf{x}$				max	$\mathbf{p}^\tau \mathbf{b}$		
s.a	$\mathbf{a}_i^\tau \mathbf{x} \geq b_i,$	$i \in M_1$			s.a	$p_i \geq 0,$	$i \in M_1$	
	$\mathbf{a}_i^\tau \mathbf{x} \leq b_i,$	$i \in M_2$		$\rightarrow$		$p_i \leq 0,$	$i \in M_2$	
	$\mathbf{a}_i^\tau \mathbf{x} = b_i,$	$i \in M_3$				p_i libre,	$i \in M_3$	
	$x_j \geq 0,$	$j \in N_1$				$\mathbf{p}^\tau \mathbf{A}_j \leq c_j,$	$j \in N_1$	
	$x_j \leq 0,$	$j \in N_2$				$\mathbf{p}^\tau \mathbf{A}_j \geq c_j,$	$j \in N_2$	
	x_j libre,	$j \in N_3$				$\mathbf{p}^\tau \mathbf{A}_j = c_j,$	$j \in N_3.$	

Por otra parte se cumple que $\max(\cdot) \equiv -\min -(\cdot)$.

Programación Lineal: Propiedades de dualidad

Dados los problemas primal y dual anteriores:

- *Dualidad débil*
  Si $\hat{x}$ es una solución primal factible y $\hat{p}$ es una solución dual factible entonces
  $$c^\tau \hat{x} \geq \hat{p}^\tau b.$$

- *Dualidad fuerte*
  Existe x^* solución óptima $\Leftrightarrow$ existe p^* solución óptima, tales que
  $$c^\tau x^* = (p^*)^\tau b.$$

- *Holgura complementaria*
  Sean $\hat{\mathbf{x}}$ solución primal factible y $\hat{\mathbf{p}}$ solución dual factible, $\hat{\mathbf{x}}$ y $\hat{\mathbf{p}}$ son óptimos si y solo si
  $$\hat{p}_i(\mathbf{a}_i\hat{\mathbf{x}} - b_i) = 0, \quad i = 1, ..., m,$$
  $$(c_j - \hat{\mathbf{p}}^\tau \mathbf{A}_j)\hat{x}_j = 0, \quad j = 1, ..., n.$$

Algoritmos

Los problemas de optimización se resuelven mediante algoritmos, que al iterar obtienen una secuencia de soluciones factibles que mejora el valor objetivo hasta que se detecta una que es óptima.
Los algoritmos son diseñados para los problemas, pero estos se aplican para resolver instancias de los problemas.

Un *algoritmo* es una secuencia finita de instrucciones (de aritmética, lógica, lectura, escritura, etc.) que resuelve una función. A partir de un estado inicial, la ejecución de las instrucciones genera una secuencia finita de estados de información.

Los algoritmos utilizan los recursos tiempo y memoria en su ejecución. Las cantidades de estos dependen del tamaño de las instancias de datos y la complejidad del problema.
En general hay un compromiso de uso entre ambos recursos.

Algoritmos: comparación en uso de recursos

El *conteo de instrucciones* es un mecanismo usado para comparar algoritmos en términos normalizados con respecto al *tiempo*.

Para el conteo hay dos modelos, dependiendo de la unidad de base:
- *modelo aritmético*: cada instrucción se cuenta en forma unitaria,
- *modelo de bit*: cada instrucción se descompone en instrucciones elementales del sistema binario.

Algoritmos: conteo de operaciones

Ejemplo: el producto escalar de dos vectores de dimensión n,

$$\sum_{i=1}^{n} c_i x_i,$$

implica $2n - 1$ operaciones aritméticas.

Para problemas complejos una cuenta exacta no es viable; por lo que basta con tener un estimado de la cantidad de operaciones como función del tamaño de la instancia.
En el ejemplo se dice que la cantidad de operaciones crece linealmente con respecto a n.

Este esquema se generaliza mediante la *notación de orden* de crecimiento asintótico.

Algoritmos: notación de orden asintótico $O(\cdot)$

Permite clasificar los algoritmos mediante una cota superior del crecimiento asintótico para el peor caso de instancia según el tamaño de la instancia.

Sea la función $f(n)$ el peor tiempo de ejecución de un algoritmo para una instancia de tamaño n.
Dada la función $g(n)$, se define que $f(n)$ es de orden $g(n)$, formalmente $f(n) =: O(g(n))$, si existen las constantes positivas n_0 y K tales que $f(n) \leq K \cdot g(n)$ para todo $n \geq n_0$.
Se dice que f esta asintóticamente acotada por g.

Ejemplo: Dada $f(n) = 20 \log n + 10n + 5n^2 + 2n^3$,
se tiene que $f(n) =: O(n^3)$

¿Porqué utilizar el peor caso en vez del caso promedio?
El peor caso es muy pesimista; en la práctica una medida promedio es más adecuada (eg. método simplex), pero la medida promedio es mucho más difícil de obtener que el peor caso.