Métodos de Punto Interior

Oscar S. Dalmau Cedeño
dalmau@cimat.mx

26 de septiembre de 2017

Forma estándar

Forma estándar de un problema de programación lineal (PL).

$$\min c^T x, \quad s.a: \quad Ax = b, \; x \geq 0. \tag{1}$$

Condiciones de Optimalidad

Lagrangiano:

$$\mathcal{L}(x, \pi, s) = c^T x - \pi^T (Ax - b) - s^T x. \tag{2}$$

Condiciones KKT.

$$\begin{aligned} A^T \pi + s &= c, && (3)\\ Ax &= b, && (4)\\ x &\geq 0, && (5)\\ s &\geq 0, && (6)\\ x_i s_i &= 0, \ i = 1, 2, \ldots, n. && (7) \end{aligned}$$

Simplex: Algoritmo de dos Fases

- Supongamos el problema de PL

$$\begin{aligned} \min \quad & c^T x, && (8)\\ s.a: \quad & Ax = b, && (9)\\ & x \geq 0. && (10) \end{aligned}$$

donde $b \succeq 0$.

Simplex: Algoritmo de dos Fases

- Se necesita un método sistemático para buscar un punto factible.
- Consideremos el *problema artificial* con $b \succeq 0$

$$\begin{aligned} \min_{x,y} \quad & 1^T y, && (11)\\ s.a: \quad & [A, I][x; y] = b, && (12)\\ & x, y \geq 0. && (13) \end{aligned}$$

- Al vector y se le conoce como *vector de variables artificiales* y a $1^T y$ *función objetivo artificial*
- Se puede notar que el problema anterior tiene una solución básica factible trivial, ie, $[x; y] = [0; b]$

Simplex: Algoritmo de dos Fases

Proposicion

El problema original LP tiene soluciones básicas factibles si y solo si el problema artificial asociado tiene una solución optima factible con valor objetivo igual a cero

Proposition 16.1 del libro E. Chong y S. Zak.

Simplex: Algoritmo de dos Fases

- De acuerdo a la proposición anterior, si el problema original LP tiene soluciones básicas factibles entonces la solución optima factible con valor objetivo igual a cero.
- Lo anterior significa que la solución del problema artificial cumple que $y^* = 0$. Por otro lado, el óptimo correspondiente $[x^*; y^*]$ es un punto básico factible del problema artificial y como $y^* = 0$ entonces x^* es un punto básico factible en el problema original.

Simplex: Algoritmo de dos Fases

- Es decir, usando variables artificiales podemos resolver el problema de programación lineal, aplicando el *método simplex en dos fases*.
- **Fase I**: Se introducen las variables artificiales, la función objetivo artificial y se encuentra una solución básica factible (si hay solución)
- **Fase II**: Se usa la solución básica factible de la Fase I para inicializar el algoritmo simplex para resolver el problema original.

Ejemplo Matlab (I)

Ejemplo 16.4 tomado del libro E. Chong y S. Zak.

$$\begin{aligned} \text{min} \quad & 2x_1 + 3x_2 \\ \text{s.a:} \quad & 4x_1 + 2x_2 \geq 12 \\ & x_1 + 4x_2 \geq 6 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Forma estandar usando variables de exceso

$$\begin{aligned} \text{min} \quad & 2x_1 + 3x_2 \\ \text{s.a:} \quad & 4x_1 + 2x_2 - x_3 + 0x_4 = 12 \\ & x_1 + 4x_2 + 0x_3 - x_4 = 6 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

Ejemplo Matlab (II): Fase I

Problema artificial

$$\begin{aligned} \min_{x_1,x_2,x_3,x_4,y_1,y_2} \quad & y_1 + y_2 \\ \text{s.a:} \quad & 4x_1 + 2x_2 - x_3 + 0x_4 + y_1 + 0y_2 = 12 \\ & x_1 + 4x_2 + 0x_3 - x_4 + 0y_1 + 1y_2 = 6 \\ & x_1, x_2, x_3, x_4, y_1, y_2 \geq 0 \end{aligned}$$

```
% x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
f = [0,0,0,0,1,1]';
Aeq = A = [4,2,-1,0, 1,0;
     1,4,0,-1,0,1];
beq = [12,6]';
lb = [0,0,0,0,0,0]';
y0 = [[0,0,0,0]';b];
y = linprog(f,[], [], Aa,b,lb,[],y0);
```

Ejemplo Matlab (III): Fase II

Problema original en su forma estandar

$$\begin{aligned} \text{min} \quad & 2x_1 + 3x_2 \\ \text{s.a:} \quad & 4x_1 + 2x_2 - x_3 + 0x_4 = 12 \\ & x_1 + 4x_2 + 0x_3 - x_4 = 6 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

```
f = [2,3,0,0]';
Aeq = [4,2,-1,0;
    1,4,0,-1];
beq = [12,6]';
lb = [0,0,0,0]';
x0 = y(1:4); % Solución básica  factible
             % del problema artificial
x = linprog(f,[], [], Aeq,beq,lb,[],x0);
```

Métodos de Punto Interior: Historia corta

- Los métodos de punto interior fueron descubiertos por matemáticos Soviéticos en la década de 1960
- En particular el algoritmo de escalado afine (affine scaling) fue descubierto por el Matemático Ruso I. I. Dikin en 1967, y luego fue re-inventado en 1980s.
- El método elipsoide (ellipsoid method) es un método iterativo introducido Naum Z. Shor (1971). En 1972, Arkadi Nemirovski y David B. Yudin (Judin) estudiaron una aproximación para problemas convexo.

Métodos de Punto Interior: Historia corta

- Leonid Khachiyan (1979) realizó un estudio de método elipsoide para problemas de programación lineal (Algoritmo de Khachiyan) el cual tiene tiempo polinomial.
- El algoritmo de Karmarkar fue propuesto por Narendra Karmarkar en 1984 para resolver problemas de PL.
- El algoritmo de Karmarkar es un algoritmo de punto interior de tiempo polinomial y mas rapido que el método elipsoide.

Métodos de Punto Interior: Introducción

- Los métodos de Punto Interior que se estudiarán de basan en el método de newton, en particular en el empleo del método de newton en la solución de sistema de ecuaciones no lineales, Renegar (1988) Newton-based interior-point algorithm for LP.
- Cada paso de los métodos de Punto Interior puede ser computacionalmente muy costoso y puede hacer un buen progreso, al contrario del método Simplex que puede realizar muchas iteraciones que no son tan costosas.

método de Newton: Recordatorio

Sea la función $g : \mathbb{R}^n \to \mathbb{R}^n$, es decir:
$g(x) = [g_1(x), g_2(x), \ldots, g_n(x)]^T$ continuamente diferenciable y se desea determinar la solución del sistema $g(x) = 0$.

$$\begin{aligned} g_1(x) &= 0 \\ g_2(x) &= 0 \\ \ldots \quad &\ldots \quad \ldots \\ g_n(x) &= 0 \end{aligned}$$

método de Newton: Recordatorio

La solución del problema anterior se podría hallar usando algún método de optimización, considerando que $g(x)$ es el gradiente de cierta función $f : \mathbb{R}^n \to \mathbb{R}$, es decir: $g(x) = \nabla f(x)$ y aplicando algún método de descenso.

$$x^{k+1} = x^k + \alpha_k d_k$$

donde α_k garantiza suficiente descenso y d_k es una dirección de descenso.

El paso α_k se determina a partir de la función $f(x)$, pero en este caso, no se dispone de esta función, por lo que se puede tomar $\alpha_k = 1$. Si se usa el método de newton, la dirección de descenso satisface que:

$$\nabla^2 f(x_k) d_k = -\nabla f(x_k)$$

O lo que es lo mismo:

$$\nabla_x g(x_k) d_k = -g(x_k)$$

método de Newton: Recordatorio

Luego para resolver el sistema de ecuaciones, se resuelve el **sistema de newton**

$$D_x g(x_k) d_k = -g(x_k)$$

y se actualiza en forma iterativa hasta convergencia

$$x^{k+1} = x^k + d_k$$

Problemas Primal-Dual

Sea el problema (**primal**) de Programación Lineal

$$\min c^T x, \quad s.a: \ Ax = b; x \geq 0 \tag{14}$$

con $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{n \times m}$.
El **problema dual** es:

$$\max b^T \lambda, \quad s.a: \ A^T \lambda + s = c; s \geq 0 \tag{15}$$

Lagrangianos Primal-Dual

Los Lagrangianos

$$\mathcal{L}_P(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x \quad (16)$$

$$\mathcal{L}_D(\lambda, s, x, \gamma) = -b^T \lambda - x^T (A^T \lambda + s - c) - \gamma^T s \quad (17)$$

Nota: observe que $\gamma = x$, puesto que $\nabla_s \mathcal{L}_D = x - \gamma = 0$. Ambos problemas tienen las mismas **condiciones de KKT** y por tanto corresponden al mismo problema.

$$A^T\lambda + s = c \quad (18)$$
$$Ax = b \quad (19)$$
$$x_i s_i = 0 \quad (20)$$
$$x, s \geq 0 \quad (21)$$

Es un sistema no lineal de ecuaciones que puede ser resuelto por el método de newton, aunque hay que ver como tratar las restricciones de no negatividad.

KKT's Primal-Dual

Sea la función $F(x, \lambda, s) : \mathbb{R}^{(2n+m)} \to \mathbb{R}^{(2n+m)}$ que se obtiene del sistema anterior. Se tiene el siguiente sistema no lineal

$$F(x, \lambda, s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ x_1 s_1 \\ x_2 s_2 \\ \cdots \\ x_n s_n \end{bmatrix} = 0 \qquad (22)$$

Método Primal-Dual

O simplemente usemos la siguiente notación

$$F(x,\lambda,s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe \end{bmatrix} = 0 \tag{23}$$

donde $F(x,\lambda,s) : \mathbb{R}^{(2n+m)} \to \mathbb{R}^{(2n+m)}$ y

$$X = diag\{x_1, x_2, \ldots, x_n\} \tag{24}$$
$$S = diag\{s_1, s_2, \ldots, s_n\} \tag{25}$$
$$e = [1, 1, \ldots, 1]^T \tag{26}$$

Método Primal-Dual

Como X y S son matrices diagonales, se puede ver fácilmente que son válidas las siguientes igualdades.

$$s = Se, \quad x = Xe$$

$$S = S^T, \quad X = X^T, \quad SX = XS$$

$$Xs = Sx = XSe = SXe$$

Luego el problema de programación Lineal conduce a resolver el siguiente sistema no lineal

$$F(x,\lambda,s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{27}$$

$$(x,s) \geq 0. \tag{28}$$

Sistema de Newton

- El sistema anterior debe ser resuelto garantizando que $(x, s) \geq 0$.
- Los métodos primal-dual generan una secuencia $\{(x^k, \lambda^k, s^k)\}$ que satisfacen las restricciones de desigualdad de forma estricta, ie, $(x^k, s^k) > 0$.

Para resolver la ecuación $F(x, \lambda, s) = 0$ usando el método de newton se requiere:

- Determinar el tamaño de paso.
- Una medida de que tan apropiado es un punto en el espacio de búsqueda.

Medida de dualidad

Como medida de que tan apropiado es un punto se usa el promedio

$$\mu = \frac{x^T s}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i s_i$$

conocida como *medida de dualidad*

Este sistema puede ser resuelto aplicando el método de newton, basta para ello determinar la dirección de descenso $d = [\Delta x, \Delta\lambda, \Delta s]^T$ (*affine scaling direction*), que puede ser calculado usando el sistema de newton, es decir:

$$J[F](x, \lambda, s) \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = -F(x, \lambda, s) \tag{29}$$

donde $J[F](x, \lambda, s)$ es el Jacobiano de $F(x, \lambda, s)$.

Sistema de Newton

Si el punto (x, λ, s) es estrictamente factible entonces se construye el sistema

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = - \begin{bmatrix} r_c \\ r_b \\ XSe \end{bmatrix} \tag{30}$$

las ecuaciones de Newton, donde

$$r_c = A^T \lambda + s - c \tag{31}$$

$$r_b = Ax - b \tag{32}$$

Sistema de Newton

El paso completo de Newton $\alpha = 1$ podria violar las restricciones $(x, s) \geq 0$. Por lo que se realiza una busqueda en linea que no viole las restricciones

$$(x, \lambda, s) + \alpha(\Delta x, \Delta\lambda, \Delta s)$$

para $\alpha \in (0, 1]$. Muchas veces se debe tomar tamaño de paso pequeño $\alpha << 1$ para evitar violar las restricciones de no negatividad.

- La mayoria de los métodos usa una dirección menos agresiva.
- Para lo anterior se cambia ligeramente el problema de modo que el producto $x_i s_i$ correspondiente a las condiciones de complementariedad se sustituye por otro, de modo que el producto sea igual a un porciento o *factor de reduccion* de la medida de dualidad, ie,

$$x_i s_i \quad = \quad \sigma \mu$$

es decir, se pide que cada producto se reduzca a un valor menor que el promedio μ

El nuevo problema a resolver es por tanto

$$F(x,\lambda,s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe - \sigma\mu e \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{33}$$

$$(x,s) \geq 0. \tag{34}$$

donde se asume que $\mu = \frac{x^T s}{n}$ un numero fijo (este termino no se deriva respecto a x, s) pero cambia en cada iteracion.

Sistema de Newton

El nuevo sistema de Newton es

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = - \begin{bmatrix} r_c \\ r_b \\ XSe - \sigma \mu e \end{bmatrix} \tag{35}$$

las ecuaciones de Newton, donde

$$r_c = A^T \lambda + s - c \tag{36}$$

$$r_b = Ax - b \tag{37}$$

Algoritmo Primal-Dual Path Following

Dado $(x^0, \lambda^0, s^0) \in F^0$

Para $k = 1, 2, ...$

Resolver

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix}$$

Donde $\sigma_k \in [0, 1]$ y $\mu_k = (x^k)^T s^k / n$;

Poner

$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k, \lambda^k, s^k) + \alpha_k (\Delta x^k, \Delta \lambda^k, \Delta s^k)$

Seleccionando α_k, de modo que $(x^{k+1}, s^{k+1}) > 0$

Fin del ciclo

- La selección de los parametros de centrado $\sigma_k \in [0, 1]$ y el tamaño de paso $\alpha_k \in (0, 1]$ son muy importantes, y se han desarrollado varias tecnicas para controlar estos parametros
- Por otro lado, algoritmo asume que (x^0, λ^0, s^0) es estrictamente factible y que por tanto satisface: $Ax^0 = b$, $A^T\lambda^0 + s^0 = c$ y se exige que $(x^0, s^0) > 0$. Esto tiene importancia teorica.
- Determinar este punto puede ser complicado (proximas clases).

Conjuntos factibles: definicion

Definición

A los conjuntos $\mathcal{F}$ y $\mathcal{F}^0$ se les denomina *conjunto factible* y *conjunto estrictamente factible* respectivamente.

$$\mathcal{F} = \{(x, \lambda, s) | Ax = b, \quad A^T\lambda + s = c, \quad (x, s) \geq 0\} \quad (38)$$

$$\mathcal{F}^0 = \{(x, \lambda, s) | Ax = b, \quad A^T\lambda + s = c, \quad (x, s) > 0\} \quad (39)$$

Camino Central

Camino central $\mathcal{C}$ es un arco de puntos estrictamente factibles

Camino Central

El camino central $\mathcal{C}$ se puede parametrizar como sigue

$$A^T\lambda + s = c \quad (40)$$
$$Ax = b \quad (41)$$
$$x_i s_i = \tau \quad (42)$$
$$(x, s) > 0 \quad (43)$$

Camino Central

Camino Central

El camino central $\mathcal{C}$ se define como sigue

$$\mathcal{C} = \{(x_\tau, \lambda_\tau, s_\tau) | \, \tau > 0\}$$

Se puede verificar que $(x_\tau, \lambda_\tau, s_\tau)$ es unico si y solo si $\mathcal{F}^0$ es no vacío.

Barrera Logaritmica

La parametrización del camino central (40)-(42) se puede ver como las condiciones de optimalidad de la formulación de barrera logaritmica (un proximo capitulo)

$$\text{mín}\, c^T x - \tau \sum_{i=1}^{n} \log x_i;\ s.a:\ Ax = b$$

Barrera Logaritmica

El Lagrangiano es

$$\mathcal{L}(x,\lambda) \quad = \quad c^T x - \tau \sum_{i=1}^{n} \log x_i - \lambda^T (Ax - b)$$

$x > 0$ es una restricción implicita. Luego

$$D_{x_i}\mathcal{L}(x,\lambda) \quad = \quad c_i - \frac{\tau}{x_i} - A_{.i}^T \lambda$$

y recobramos la parametrización del camino central (40)-(42) definiendo $s_i := \frac{\tau}{x_i} > 0$.

Alternativa

Otra forma de definir el camino central es mediante el mapeo

$$F(x_\tau, \lambda_\tau, s_\tau) = \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix}$$

con $(x_\tau, s_\tau) > 0$

Comentarios

- Las ecuaciones (40)-(42) se acercan a las KKTs originales cuando $\tau \to 0$.
- Si $\mathcal{C}$ converge a un punto, debe ser a una solución del problema primal-dual.
- El camino central nos guia por un camino que mantiene positivas x y s.
- La mayoria de los algoritmos primal-dual toman pasos hacia puntos en $\mathcal{C}$, en lugar de tomar el paso de Newton Original.

Comentarios

- En un punto factible (x, λ, s) se cumple $r_c = r_b = 0$, luego la dirección de busqueda satisface

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ XSe - \sigma \mu e \end{bmatrix} \quad (44)$$

- $\sigma \in [0, 1]$ es el parametro de centrado.
- Si $\sigma = 1$ se obtiene la dirección de centrado, ie, es decir, un paso hacia el punto $(x_\mu, \lambda_\mu, s_\mu) \in \mathcal{C}$
- Si $\sigma = 0$ obtenemos el paso de Newton original (affine scaling) step.

- Vecindad norma-∞ de un lado:

$$\mathcal{N}_{-\infty}(\gamma) = \{(x, \lambda, s) \in F^0 | x_i s_i \geq \gamma\mu, \forall i = 1, 2, ..., n\}$$

  valor tipico de γ es $\gamma = 10^{-3}$.
- Notación:

$$\begin{aligned} (x^k(\alpha), \lambda^k(\alpha), s^k(\alpha)) &\stackrel{def}{=} (x, \lambda, s) + \alpha(\Delta x^k, \Delta\lambda^k, \Delta s^k) \\ \mu_k(\alpha) &\stackrel{def}{=} [x^k(\alpha)]^T s^k(\alpha)/n \end{aligned}$$

- El proximo algoritmo es el Algoritmo Long-Step Path-Following.
- Usa dos parametros $\sigma_{min}, \sigma_{max}$ que son cotas de parametro de centrado σ_k

- Para la dirección de descenso se resuelve tipicamente

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix}$$

- Para el tamaño de paso α_k se toma el valor mas grande en $[0, 1]$ que garantice que $(x, \lambda, s) \in \mathcal{N}_{-\infty}(\gamma)$

Algoritmo Long-Step Path-Following

Dados $\gamma, \sigma_{min}, \sigma_{max}$ con $\gamma \in (0,1)$, $0 < \sigma_{min} < \sigma_{max} < 1$ y $(x^0, \lambda^0, s^0) \in N_{-\infty}(\gamma)$;

Para $k = 1, 2, ...$

Seleccionar $\sigma_k \in [\sigma_{min}, \sigma_{max}]$;

Resolver es sistema

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix}$$

Seleccionar α_k como el valor mas grande de $\alpha \in [0,1]$ de modo que: $(x^k(\alpha), \lambda^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma)$

Poner

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k))$$

Fin de ciclo

- El algoritmo anterior tiene varias propiedades teoricas.
- Se puede probar que el numero de iteraciones requeridas para determinar un punto tal que $\mu_k < \epsilon$ es $O(n|\log \epsilon|)$ para una tolerancia dada ϵ
- Sobre el punto inicial.. proxima clase (Algoritmo Practico)