# Implementação do Método Simplex

Bruno Sesso 8536002
Gustavo Estrela de Matos 8536051

11 de Junho de 2015

# 1 Introdução

## 1.1 Apresentação do problema

Os problemas de Programação Linear (PL) são casos específicos de otimização combinatória em que a função objetivo e as restrições são ambos lineares. Portanto a função objetivo é da forma $c^T x$ e as restrições são da forma $a_i^T x \geq b_i$ ou $a_i^T x \leq b_i$, com $c, x, a_i \in \mathbb{R}^n$ e $b_i \in \mathbb{R}$.

Multiplicando por $-1$ todas as restrições da forma $a_i^T x \geq b_i$, podemos escrever qualquer PL como:

$$\begin{array}{rl} \text{minimizar} & c^T x \\ \text{sujeito a} & Ax \leq b, \\ & A \in R^{m \times n} \text{ e } b \in \mathbb{R}^m. \end{array}$$

Também é possível mostrar que qualquer PL pode ser escrito na forma:

$$\begin{array}{rl} \text{minimizar} & c^T x \\ \text{sujeito a} & Ax = b, \\ & x \geq 0 \text{ [1].} \end{array}$$

Se for escrito dessa maneira, dizemos que o problema está no formato padrão. Adotaremos esse formato durante todo o trabalho.

Se vale que $Ax^1 = b$ e $x^1 \geq 0$ dizemos que $x^1$ é um ponto viável. O conjunto $P = \{x | Ax = b, x \geq 0\}$ de todos os pontos viáveis é chamado conjunto viável.

Uma solução ótima do problema é um ponto $x^1 \in P$ que minimiza [1] a função objetivo $c$. Se $x^1$ existe, dizemos que o custo ótimo é $c^T x$. Se $x^1$ não existe, ou não existem pontos viáveis ($P = \emptyset$), ou podemos diminuir o custo o quanto quisermos e dizemos que o custo ótimo é $-\infty$.

## 1.2 Objetivos do trabalho

Neste trabalho, temos o objetivo de desenvolver, na linguagem Octave, o algoritmo simplex para resolver problemas de Programação Linear.

[1] Se o interesse for maximizar $c^T x$, podemos simplismente conseguir um problema equivalente em que o objetivo seja minimizar $-c^T x$.

# 2 Conceitos fundamentais

Antes de introduzirmos o funcionamento do nosso algoritmo, precisamos definir alguns conceitos que são fundamentais para garantir sua corretude.

Seja o nosso problema de Programação Linear o seguinte:

$$\begin{aligned} \text{minimizar} \quad & c^T x \\ \text{sujeito a} \quad & Ax = b \\ & x \geq 0 \\ \text{com} \quad & c, x \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n} \text{ e } b \in \mathbb{R}^m. \end{aligned}$$

Além disso, vamos usar a notação $a_i$ para a i-ésima linha de A e $A_i$ para a i-ésima coluna de A.

## 2.1 Restrições e degenerecência

Uma restrição $a_i^T x \geq b_i$ (ou $a_i^T x \leq b_i$), com $a_i \in \mathbb{R}^n$ e $b_i \in \mathbb{R}$, é uma *restrição ativa* em um ponto $x^1 \in \mathbb{R}^n$ se $a_i^T x^1 = b_i$. Uma restrição de igualdade é sempre ativa. Um conjunto de restrições será dito LI se os vetores $a_i$ correspondentes forem LI.

Diremos que $x^1$ uma solução viavel básica é *degenerada* se existem mais de $n$ restrições ativas LI nesse ponto. Como as $m$ restrições de igualdade são sempre cumpridas, temos que as soluções básicas degeneradas possuem mais do que $n - m$ componentes nulas, enquanto que as não degeneradas possuem exatamente $n - m$.

## 2.2 Soluções Viáveis Básicas

Dizemos que um ponto $x \in \mathbb{R}^n$ do conjunto viável $P$ é uma *solução viável básica*, se existem $n$ restrições ativas em $x$ que são LI. Note que para problemas no formato padrão, existem sempre $m$ restrições ativas LI vindas de $Ax = b$, e as outras $n - m$ vem, necessariamente de $x \geq 0$. Portanto, uma solução viável básica possui ao menos $n-m$ componentes nulas.

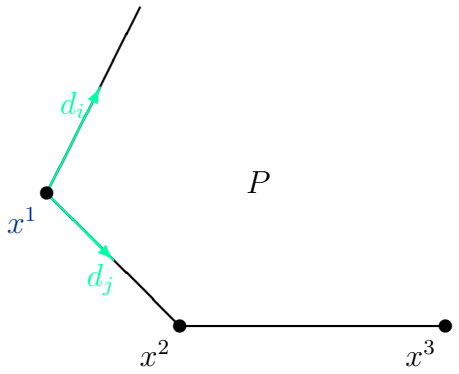
Se $x^1$ é uma solução básica não degenerada e seja $B(1), ..., B(m)$ os índices das componentes não nulas de $x$. A matriz $B = \left[A_{B(1)}, ..., A_{B(m)}\right]$ é chamada *matriz básica* associada a $x^1$.

Se o conjunto $P$ tem uma solução viável básica, então ou o custo ótimo é $-\inf$ ou existe $x^1 \in P$ solução viável básica que é ótimo, ou seja, o custo de qualquer ponto do conjunto viável é maior ou igual do que o custo de $x^1$. Portanto, na solução de um PL com ao menos uma solução viável básica, podemos limitar a esses elementos a nossa busca por um ponto de custo ótimo [1].

## 2.3 Direções básicas

Se $x^1$ é um solução viável básica de $P$, com índices básicos $B(1), ..., B(m)$. Dizemos que $d \in \mathbb{R}^n$, tal que $d_j = 1$, $Ad = 0$ $(A(x + \theta d) = b)$ e $d_i = 0$ para todo $i \notin \{B(1), ..., B(m)\}$, é a j-ésima direção básica partindo de $x^1$. Seja $d_B = \left[d_{B(1)}, ..., d_{B(m)}\right]$, como $A(x+\theta d) = b$, temos que $d_B = -B^{-1}A_j$. Usaremos $u = -d_B = B^{-1}A_j$ por facilidade de notação, durante o trabalho.

A figura 2.3 dá um exemplo de direções básicas, $di$ e $dj$ a partir de uma solução viável básica $x^1$. Note que o poliedro pode ou não limitar um $\theta$ tal que o ponto $y = x^1 + \theta * d$ ($d$ direção básica) seja viável, e como veremos em 2.5 isso pode implicar em custo $-\infty$ se nessa direção o custo diminui.




## 2.4 Custos reduzidos

Seja $x^1$ uma solução viável básica, $B$ a matriz básica associada e $c_B = \left[c_{B(1)}, ..., c_{B(m)}\right]$. Definimos, para cada $j \in \{1, ..., n\}$ o custo reduzido:

$$\overline{c}_j = c_j - c_B^T B^{-1} A_j.$$

Seja $x^1$ uma solução viável básica e $\overline{c}$ o vetor de custos reduzidos correspondente. Sabemos que se $\overline{c} \geq 0$, então $x^1$ é ótimo. Além disso, se $x^1$ for ótimo e não degenerado, então $\overline{c} \geq 0$ [1]. Portanto, se estivermos em uma solução viável básica e $\overline{c} \geq 0$, então estamos em um ponto ótimo.

Note que ao escolhermos uma direção viável básica, o custo de um ponto $y = x^1 + \theta d_j$ é $c^T(x^1 + \theta d_j)$

## 2.5 Soluções Viáveis Básicas adjacentes

Seja $x^1$ uma solução viável básica com índices básicos $B(1), ..., B(m)$. Uma solução viável básica é *adjacente* a $x^1$ se compartilha $m-1$ índices com $x^1$. Para achar uma solução viável básica adjacente, vamos usar as direções básicas, pois elas forçam o crescimento de uma variável $j$ não-básica, mantendo $Ax = b$ e $x \geq 0$. Veremos que para um $\theta \geq 0$, o ponto $x^1 + \theta d_j$ é solução viável básica adjacente a $x^1$, com $d_j$ como foi definido em 2.4.

Vamos tomar $\theta = \min_{i=1,...,m|u_i>0} \{x_{B(i)}/u_i\}$ e ver que $x^2 = x^1 + \theta d_j$ é de fato uma solução viável básica adjacente a $x^1$. Caso todas as componentes de $u_i$ sejam menores ou igual a zero e o custo reduzido na direção $j$ menor do que zero teremos que o problema tem custo ótimo $-\infty$, como será explicado a seguir.

Se $\theta$ definido acima não existe, temos que todas as componentes de $u_i$ são menores ou igual a zero ($d \geq 0$), logo qualquer ponto $x^2 = x^1 + \theta d$ é viável com $\theta \geq 0$, pois a restrição $Ax^2 = b$ é verificada (por construção), e $x_j^2 = x_j^1 + \theta \geq x_j^1 \geq 0$, e para $i$ básico $x_j^2 = x_j^1 + \theta d_j \geq x_j^1 \geq 0$. Se ainda tivermos que o custo diminui nessa direção, poderemos diminuir o custo o quanto quisermos e a solução do problema será $-\infty$.

Se $\theta \in \mathbb{R}$, como $d_i = 0 \ \forall i \in \{B(1), ..., B(m)\}, i \neq j$, temos que para essas mesmas componentes $x^2$ é nulo. Logo, temos $n-1$ restrições ativas LI em $x^2$. Suponha que para $l \in \{1, .., m\}$ vale que $\theta = x_{B(l)}/u_l$, então $x^2_{B(l)} = x^1_{B(l)} + (-x^1_B(l)/d_{B(l)}) * d_{B(l)} = 0$ (diremos que $B(l)$ sai da base), logo existem $n$ restrições ativas LI em $x^2$. Além disso, por construção, vale que $Ax = b$ e $x \geq 0$ para variáveis não básicas e para $x_B(l)$. Para $B(k)$ básico diferente de $B(l)$, temos que $x^2_B(k) \geq x^1_{B(k)} + (-x^1_B(k)/d_{B(k)}) * d_{B(k)} = 0$. Portanto $x^2$ é solução viável básica adjacente a $x^1$ e, como a base de $x^2$ é $\{B(1), ..., B(l-1), j, B(l+1), ..., B(m)\}$, $x^2$ é adjacente a $x^1$.

# 3 O algoritmo

## 3.1 Ideia do algoritmo

A ultima seção apresenta ideias essenciais para a construção da fase 2 do algoritmo simplex. Dentre elas, as mais importantes são: podemos reduzir nosso espaço de busca as soluções viaveis básicas; se $\overline{c} \geq 0$ e estamos em uma solução viável básica, então esse ponto é ótimo.

Portanto, utilizamos uma dinâmica que percorre as soluções viáveis básicas, com auxilio das direções básicas, sempre diminuindo a função custo, até que não seja mais possível sair de um ponto sem aumentar ou manter o custo, ou até encontrar uma direção que podemos diminuir o custo sem limitações.

## 3.2 Algoritmo

**function** $simplex(A, b, c, m, n, x)$
  calcula indices basicos ($Ib$) e não básicos ($In$)
  $B \leftarrow A_{Ib(i)}$, i = 1, ..., m
  $invB \leftarrow B^{-1}$
  $imin \leftarrow 0$
  **if** $\nexists j$ t.q. $\overline{c_j} < 0$ **then**
    $\overline{c_j} \leftarrow 0$
  **else**
    $\overline{c_j} \leftarrow c_j - c_B^T B^{-1} A_j$, algum $j \in In$ t.q. $\overline{c_j} < 0$
    $u \leftarrow invB * A_j$
  **end if**
  **while** $\overline{c_j} < 0$ **do**
    **if** $u_l < 0, l = 1, ..., m$ **then**
      **return** $-1, d(u, j)$
    **end if**
    $\theta \leftarrow \min_{u_l >= 0}\{\frac{x_{Ib(l)}}{u_l}\}, l = 1, ..., m$
    $x \leftarrow x + \theta * d(u, j)$
    $x_{Ib(l)}$ sai da base ▷ Atualiza $In$
    $x_j$ entra na base ▷ Atualiza $Ib$
    Atualiza $invB$
    $\overline{c_j} \leftarrow c_j - c_B^T B^{-1} A_j$, algum $j \in In$ t.q. $\overline{c_j} < 0$

$u \leftarrow invB * A_j$
**end while**
**return** $0, x$
**end function**

# 4 Condições do problema

Durante a elaboração do algoritmo foram consideradas duas condições: existe ao menos uma solução viável básica e qualquer solução viável básica é não degenerada. Essas condições foram importantes para implementações de detalhes do código e sem elas o algoritmo não será correto.

A existência de ao menos uma solução viável básica implica, como discutido na subseção 2.2, que ou o custo ótimo é $-\infty$ ou existe uma solução viável básica com custo ótimo. Isso nos permite limitar nosso espaço de busca às soluções viáveis básicas, somente.

A condição de que todas as soluções viáveis básicas são não-degeneradas tem outras aplicações. Com essa condição, é possível determinar a base da solução inicial dada. Além disso, ela garante que em todo passo em que calculamos um novo $\theta$, o mesmo será maior do que zero, pois a não degenerecência implica que $x_B(i) > 0$, evitando o problema de passar uma interação do algoritmo sem sair do ponto anterior, o que pode criar um ciclo sem fim no algoritmo. Além disso, houve uma condição não citada no enunciado que é importante para a solução do problema: o posto completo da matriz $A$. Se $posto(A) = k < n$ precisaríamos construir uma matriz $A^1$ com posto completo, para garantir sabemos escolher $k$ colunas LI de $A^1$ que formam bases para soluções viáveis básicas.

# Referências

[1] Dimitris Bertsimas, John N. Tsitsiklis. Introduction to Linear Optimization. 1997.