Método de Feixe aplicado em problema de Programação Linear Estocástica com Dois Estágios

PPGEE - Otimização em Engenharia Elétrica

Pedro de Mendonça Maia

I. Introdução

Este trabalho tem como objetivo apresentar o Método de Feixe (Extended Cutting Plane Method - ECPM) e mostrar sua aplicabilidade em problemas descontínuos e com incertezas.

O Método de Feixe foi originalmente desenvolvido para resolver problemas de otimização inteiros não lineares (Mixed-Integer NonLinear Programming - MINLP), porém ele também resolve problemas de otimização convexa não diferenciável, exigindo somente o cálculo da função objetivo e de seu subgradiente. Por esse motivo, este ele se tornou atrativo em situações em que calcular o valor da função objetivo e seu gradiente é caro. Este é, por exemplo, o caso da programação estocástica quando há muitos cenários para representar as incertezas.

Neste trabalho o Método de Feixe será aplicado ao planejamento de expansão de usinas termoelétricas modelado como um problema de programação linear inteira estocástica com dois estágios (Two-Stage Stochastic Linear Programming). Esse problema é descontínuo e possui cenários de incerteza, o que impede sua solução usando métodos tradicionais.

II. O Método de Feixe

O Método de Feixe pode ser entendido como uma extensão do Método de Planos Cortantes (Cutting-Plane Method), pois trabalha com aproximações lineares da função objetivo e das restrições para ir refinando o espaço de busca.

No Método de Planos Cortantes, dado um ponto inicial x^0 calcula-se o valor de função objetivo e o gradiente e cria-se uma reta com a direção de maior decrescimento da função naquele ponto. Caminha-se então nessa direção até encontrar um limite e parar no ponto x^1. Nesse ponto define-se outra reta a partir do novo ponto e repete-se a busca até encontrar um ponto de mínimo. A Figura 1 mostra o comportamento desse método em três iterações.

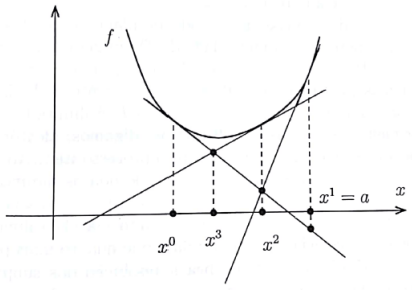

Fig. 1. Três iterações do Método de Planos Cortantes

Entretanto, o Método de Planos Cortantes possui um grave problema de convergência. Como pode ser visto na Figura 2, o ponto x^k se encontra próximo ao ponto mínimo $\overline{x}$, mas não há nada que impeça o algoritmo de se afastar desse ponto na iteração seguinte x^{k+1}.

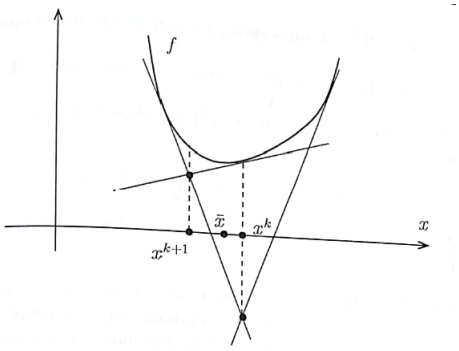

Fig. 2. Problema de convergência do Método de Planos Cortantes

O Método de Feixe apresentado a seguir armazena outras estruturas no processo de busca que o transformam em uma extensão mais robusta do Método de Planos Cortantes. Além disso, essas outras estruturas ainda permitem a aplicação do Método de Feixe em outros tipo de problemas de otimização, como será apresentado posteriormente neste trabalho.

A. O feixe

Antes de iniciar a explicação do método em si, é necessário apresentar e definir o feixe, que é a estrutura que dá nome ao método.

A partir das funções $f_i(x)$, com x podendo conter elementos inteiros, obtém-se o subgradiente $g_i(x) = \nabla f_i(x)$ dessas funções em dado ponto (x_k) da iteração k.

Uma aproximação $\hat{f}_i^k(x)$ para o valor da função é obtido

$$\hat{f}_i^k(x) = \max_{j \in \beta^k} \left\{ f_i(x^j) + \left\langle g_i^j, x - x^j \right\rangle \right\} \tag{1}$$

onde o conjunto β^k é chamado de feixe. Ele armazena o histórico de aproximações lineares dos pontos de iterações anteriores. Para que isso seja possível, é necessário que as informações $\beta^k = \left\langle x^k, f_i(x^k), g_i^k \right\rangle$ sejam armazenadas. x^k são chamados pontos de estabilidade, pois são a partir deles que as aproximações são calculadas.

Para evitar consumo excessivo de memória, o tamanho do feixe pode ser limitado e alguns critérios podem ser adotados

na seleção de qual informação será eliminada ao inserir uma nova.

Como a função objetivo e as restrições são convexas, tem-se que

$$\hat{f}_i^k(x) \leq f_i^k(x) \quad \forall i = 0, ..., p, \quad \forall x \in \mathbb{R}^n \tag{2}$$

o que é fundamental para o funcionamento do método.

B. Funcionamento do Método de Feixe

Inicialmente, o seguinte problema linear inteiro misto (Mixed-Integer Linear Programming - MILP) é resolvido:

$$\begin{aligned} \min \quad & \hat{f}_0(x) \\ s.t. \quad & \hat{f}_i(x) \leq 0 \quad i = 1, ..., p \end{aligned} \tag{3}$$

O valor ótimo da função objetivo desse problema fornece um limitante inferior f_{low} para a aproximação.

A medida que as iterações k passam, o feixe β^k ganha mais informações e a aproximação fica mais precisa, o valor do limitante f_{low} cresce, reduzindo a diferença em relação ao valor real da função.

Uma outra estrutura é usada para armazenar a evolução do algoritmo

$$H^{k+1}(f_{low}^k) = \max_{i=1,...,p} \left\{ f_0(x^{k+1}) - f_{low}^k, f_i(x^{k+1}) \right\} \tag{4}$$

O valor mínimo dessa estrutura é zero, ocorrendo quando duas condições são satisfeitas:

1) *Todas as restrições estão factíveis:* $f_i(x^{k+1}) \leq 0, \quad i = 1, ..., p$

2) *O lower bound se aproxima do valor da função:* $f_0(x^{k+1}) = f_{low}^k$

O valor mínimo dessa estrutura é chamado de residual de otimalidade e é definido por

$$h^k = \min_{j \leq k} H^j(f_{low}^k) \tag{5}$$

Como dito anteriormente, com o passar das iterações o feixe fica mais preciso e o valor de f_{low} cresce, o que faz com que h^k reduza e tenda a zero. O algoritmo termina quando $h^k \leq \delta_{tol}$, ou seja, quando a aproximação está suficientemente próxima da função real.

Nesse ponto, o valor ótimo da solução é dado por

$$x_{best}^k = \arg\min_{j \leq k} H^j(f_{low}^k) \tag{6}$$

A solução iteração seguinte x^{k+1} é encontrada a partir de

$$x^{k+1} = \arg\min \varphi(x, \hat{x}^k) \tag{7}$$

onde

$$\varphi(x, \hat{x}^k) = ||x - \hat{x}^k|| \tag{8}$$

A forma mais simples de fazer isso é resolvendo o sub-problema (3).

Com o objetivo de checar se uma iteração foi efetiva, isto é, se de fato houve ganho na função objetivo, o algoritmo introduz conteitos chamado de passo sério e passo nulo para determinar se o ponto de referência é atualizado ou não. Caso alguma das seguintes condições seja satisfeita

1) $f_0(x^{k+1}) \leq f_0(\hat{x}^k) - (1-\gamma)h^k \;\; e \;\; f_i(x^{k+1}) \leq 0, \quad i = 1, ..., p$

2) $\max\limits_{i-1,...,p} \left\{ f_i^+(x^{k+1}) \right\} \leq \max\limits_{i-1,...,p} \left\{ f_i^+(x^k) \right\} - (1-\gamma)h^k$

atualiza-se $\hat{x}^{k+1} = x^{k+1}$ e a iteração é chamada de passo sério. Caso nenhuma das duas condições seja satisfeita mantém-se $\hat{x}^{k+1} = \hat{x}^{k+1}$ e a iteração é chamada de passo nulo.

A primeira condição é satisfeita quando todas as restrições são factíveis e há um ganho considerável na função objetivo. Já a segunda é satisfeita quando existem restrições infactíveis, mas há uma redução da infactibilidade em relação a iteração anterior.

O parâmetro γ determina a tolerância do passo sério, sendo que $\gamma \in (0, 1)$. Neste trabalho considera-se $\gamma = 0.2$.

Caso um passo nulo tenha sido executado, o gradiente de x^{k+1} é adicionado ao feixe, porém o ponto de referência não se altera. Isso significa que mais um plano de corte foi encontrado, mas que caminhar em direção àquele ponto não melhora a solução.

C. Melhoria para funções objetivo pesadas

De forma opcional, pode-se criar o *Conjunto Localizador* para reduzir a quantidade de avaliações da função objetivo. Primeiramente deve-se criar um nível objetivo para o sub-problema, chamado f_{lev}, tal que

$$f_{lev}^k = f_{low} + \gamma h^k \tag{9}$$

Então *Conjunto Localizador* L^k é definido da seguinte forma

$$L^k = \left\{ x \in \mathbb{R}^n : \hat{f}_0^k(x) \leq f_{lev}^k, \hat{f}_i^k(x) \leq 0, i = 1, ..., p \right\} \tag{10}$$

Isso é realizado resolvendo-se o sub-problema (3) considerando uma restrição adicional $\hat{f}_0^k(x) \leq f_{lev}^k$.

A falta de solução para esse problema indica que a função objetivo do problema original não pode alcançar tal valor, permitindo a atualização do *lower bound* f_{low}

$$f_{low}^{k+1} = \begin{cases} f_{lev}^k, & se \;\; L^k = \emptyset \\ f_{low}^k, & caso \;\; contrario \end{cases} \tag{11}$$

Caso f_{low} seja alterado, não há necessidade de atualizar $\hat{x}$ e o problema original não precisa ser avaliado novamente, passando para a iteração seguinte apenas com a atualização desse valor.

É importante ressaltar que esse procedimento reduz a convergência do algoritmo, existindo a possibilidade de não ser prática em casos onde a avaliação da função objetivo é rápida.

D. O algoritmo do Método de feixe

Input: x_0, $f(x)$, $g(x)$, γ, δ_{tol}
$\beta \leftarrow \emptyset$;
$H^0 \leftarrow \emptyset$;
Obter $\hat{f}_i^0(x^0)$ conforme (1);
$\beta \leftarrow \beta \cup \langle x_0, f_i(x_0), g_i^0 \rangle$;
Obter x_1, f_{low}^0 resolvendo (3);
$k \leftarrow 0$;
$h^k \leftarrow \infty$;
$\hat{h}^k \leftarrow \infty$;
$\hat{x} \leftarrow x_0$;
while $h^k \geq \delta_{tol}$ **do**
  Obter H^k conforme (4);
  Obter h^k conforme (5);
  Obter x_{best} conforme (6);
  if $h^k \leq (1-\gamma)\hat{h}^k$ **then**
    $\hat{h}^k \leftarrow h^k$;
    $\hat{x} \leftarrow x_{best}$;
  end
  Obter x_{k+1}, f_{low}^k resolvendo (3);
  $\beta \leftarrow \beta \cup \langle x_{k+1}, f_i(x_{k+1}), g_i^k + 1 \rangle$;
  $k \leftarrow k + 1$;
end

Algorithm 1: Método de Feixe

III. Programação Estocástica

Um problema de programação estocástico é um problema de otimização onde a função objetivo e/ou o conjunto viável dependem de parâmetros incertos, porém, com uma distribuição de probabilidades conhecida, e independente da variável de decisão.

A partir das distribuições de probabilidade conhecidas das variáveis estocásticas, os modelos de otimização estocásticos trabalham com diversos cenários gerados aleatoriamente para tentar otimizar algo em torno do valor esperado dessas incertezas.

Outra abordagem possível é tabalhar com apenas um cenário com o pior caso, porém esse tipo de modelagem leva a resultados muito pessimistas, o que geralemente não reflete o que acontece na prática.

Um problema linear estocástico pode ser definido da seguinte forma

$$\begin{aligned} \min \quad & c^T x + \tfrac{1}{N} \sum_{i=1}^{N} d_i^T y_i \\ s.t. \quad & Ax = b \\ & Tx + Wy_i = h_i \quad i = 1, ..., N \end{aligned} \tag{12}$$

onde N é o número de cenários incertos, x e y representam as variáveis de decisão, sendo y_i as variáveis relativas aos dados incertos do cenário i. c, A e b são os parâmetros fixos do problema, d, T, W e h são os parâmetros estocásticos.

A. Programação Estocástica em Dois Estágios

Em um problema estocástico de dois estágios se resolve um problema da forma (12) em duas etapas. A primeira etapa consiste na solução de um problema que não depende das incertezas, ou seja, inicialmente apenas o problema (13) é considerado.

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & Ax = b \end{aligned} \tag{13}$$

A solução desse problema fornece uma ideia inicial da solução final. A partir disso, novas escolhas são realizadas resolvendo-se a parte incerta do problema, chamada de segundo estágio.

Na segunda etapa, alguma estratégia é adotada assumindo x fixo para resolver N problemas na forma (14), sendo um para cada cenário criado estocasticamente.

$$\begin{aligned} \min \quad & \tfrac{1}{N} \sum_{i=1}^{N} d_i^T y_i \\ s.t. \quad & Wy_i = h_i - Tx \quad i = 1, ..., N \end{aligned} \tag{14}$$

B. Método de Feixe aplicado à Programação Estocástica

Quando o Método de Feixe é aplicado em um problema estocástico de dois estágios, resolve-se inicialmente o problema (13) e o resultado é usado como valores iniciais no algoritmo. O vetor solução x é usado como ponto inicial x_0 e o valor de função objetivo $c^T x$ é usado como f_{low}^0.

A partir desses pontos iniciais, o *loop* do algoritmo é executado conforme apresentado no Algoritmo 1, sendo que a solução dos problemas estocásticos do segundo estágio são usados no cálculo da função objetivo e do gradiente.

A cada iteração, o valor inicial da função objetivo é calculado a partir de $c^T x$ o valor inicial do gradiente é dado por c. Posteriormente, N problemas duais do problema (14) são criados

$$\begin{aligned} \max \quad & (h - Tx)y \\ s.t. \quad & W^T y \leq d \end{aligned} \tag{15}$$

A partir do resultado de cada sub-problema resolvido, os valores de função objetivo e gradiente são atualizados

$$f(x) = f(x) - \frac{(h - Tx)y}{N} \tag{16}$$

$$g(x) = g(x) - \frac{T^T y}{N} \tag{17}$$

IV. Problema de Planejamento da Expansão e Operação Termoelétrica

No planejamento da operação das termoelétricas consideram-se n usinas, T estágios de operação e K patamares de operação. Em cada estágio, deve-se decidir quanto cada usina i deverá produzir de energia visando o atendimento da demanda d prevista para cada patamar, respeitando a capacidade instalada CI de cada termoelétrica.

Cada usina possui um custo operacional co que depende de diversos fatores incertos, portanto esse parâmetro é estocástico dentro do modelo, possuindo uma distribuição normal com média e desvio padrão conhecidos.

Como não há como prever de maneira exata a demanda por energia em cada estágio, considera-se que esse outro parâmetro também é estocástico, possuindo distribuição normal com média e desvio padrão conhecidos.

Além da operação, esse problema também trabalha com a expansão das usinas termoelétricas. Existe uma meta de expansão ME do parque termoelétrico ao fim do horizonte de otimização e em cada estágio cada usina possui um limite de crescimento LC e um custo de expansão ce. Assim como o custo operacional, o custo de expansão é estocástico, sendo descrito por uma distribuição normal com média e desvio padrão conhecidos.

Nesse modelo, considera-se que o primeiro estágio é conhecido, mas os estágios futuros trazem incertezas, ou seja, compõem a parte estocástica do modelo. Dessa forma, um modelo com dois estágios de operação pode ser escrito da seguinte forma:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n}[(ce_i^1)^T x_i^1 + \sum_{j=1}^{K}(co_j^1)^T y_{i,j}^1] + \\ & \frac{1}{N}\sum_{s=1}^{N}\sum_{i=1}^{n}[(ce_{i,s}^2)^T x_{i,s}^2 + \sum_{j=1}^{K}(co_{j,s}^2)^T y_{i,j,s}^2] \end{aligned} \tag{18}$$

$$s.t. \quad \sum_{i=1}^{n} y_{i,j}^1 = d_j^1 \quad j = 1, ..., K \tag{19}$$

$$\sum_{j=1}^{k} y_{i,j}^1 \leq CI_i \quad i = 1, ..., n \tag{20}$$

$$x_i^1 \leq LC_i \quad i = 1, ..., n \tag{21}$$

$$\sum_{i=1}^{n} y_{i,j,s}^2 = d_{j,s}^2 \quad j = 1, ..., K \quad s = 1, ..., N \tag{22}$$

$$\sum_{j=1}^{k} y_{i,j,s}^2 \leq CI_i + x_i^1 \quad i = 1, ..., n \quad s = 1, ..., N \tag{23}$$

$$x_{i,s}^2 \leq LC_i \quad i = 1, ..., n \quad s = 1, ..., N \tag{24}$$

$$\sum_{i=1}^{n}(x_i^1 + x_{i,s}^2) \geq ME \quad s = 1, ..., N \tag{25}$$

onde x_i^t representa a expansão da usina i no estágio t, $y_{i,j}^t$ é energia gerada pela usina i para o patamar j no período t e N é o número de cenários estocásticos.

V. Testes Computacionais

A. Problema linear inteiro estocástico de dois estágios

O primeiro problema trabalho foi um problema fictício de programação linear estocásticas de dois estágios (SLP). Esse problema é descrito conforme (12) e possui 60 variáveis exatas (sendo 15 inteiras), 30 incertas, 30 restrições exatas e 20 incertas.

Todos os testes foram realizados considerando 10 cenário estocásticos e tolerância de 10^{-5}.

1) Teste sem Conjunto Localizador: Inicialmente o algoritmo foi executado sem o *Conjunto Localizador*. O problema foi resolvido em 62 iterações e a execução demorou 72s.

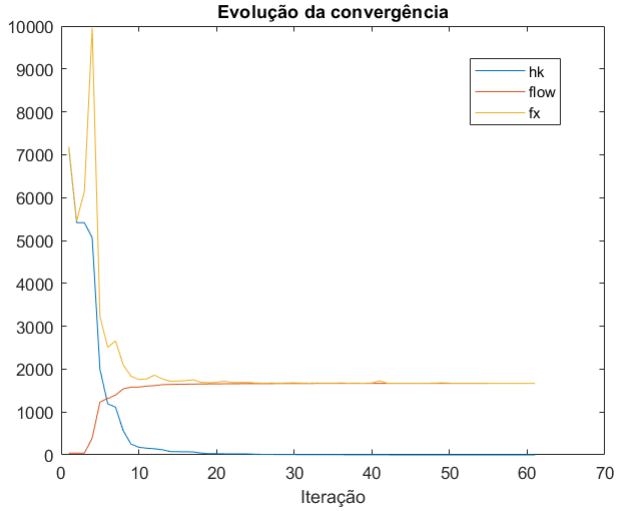

Fig. 3. Evolução do algoritmo no problema SLP

A figura 3 mostra os resultados da execução. É possível perceber que inicialmente tanto o valor da FO quanto o h_k ficam muito altos e o de f_{low} muito baixo, mas em poucas iterações os valores convergem. Por volta da decima iteração h_k já se aproxima de zero, enquanto f_{low} fica muito próximo do valor da função.

Como a execução não considerou o *Conjunto Localizador*, é realizada uma chamada da função objetivo por iteração, e as soluções encontradas em todas as iterações são adicionadas ao feixe.

2) Teste com Conjunto Localizador: O segundo teste foi realizado utilizando o *Conjunto Localizador*. O problema foi resolvido com 91 iterações e a execução demorou 47s.

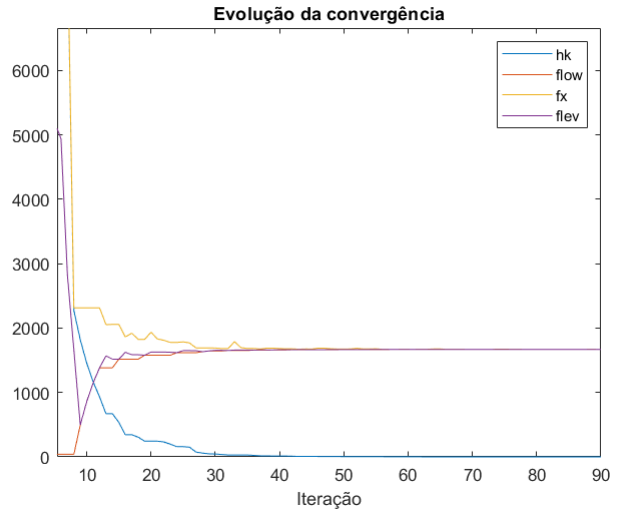

Fig. 4. Evolução do algoritmo no problema SLP

A figura 4 mostra os resultados da execução. O mesmo padrão observado anteriormente também ocorreu aqui, porém a convergência ocorreu por volta da trigésima iteração.

Outra diferença em relação ao teste anterior foi que, apesar de ter 91 iterações, só ocorreram 62 chamadas da função objetivo, o que fez com que nem todas a iterações adicionassem elementos ao feixe, como mostra a figura 5.

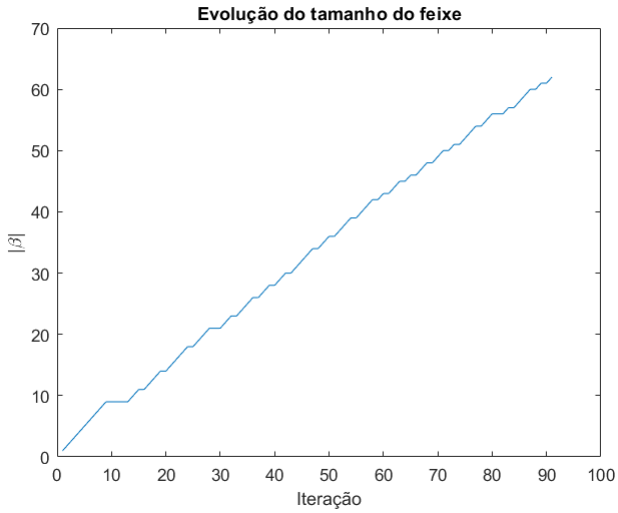

Fig. 5. Evolução da dimensão do feixe

B. Operação e Expansão Termoelétrica

1) Descrição do problema: Outro problema onde este trabalho explorou o algoritmo do Método de Feixe foi em um problema de planejamento de expansão e operação de termoelétricas. A instância testada possui dez usinas, dois estágios, três patamares e meta de expansão de 17.

As tabelas I e II mostram os parâmetros da instância estudada.

Usina	CI	LC^1	LC^2	ce^1	ce^2	co^1	co^2
1	4	4	0.5	14	$N(21, 6)$	4.8	$N(4.8, 1.2)$
2	3.8	6	0.5	9.8	$N(14.7, 4.5)$	5.4	$N(5.4, 2.1)$
3	5	1	0.5	18.2	$N(27.3, 0.3)$	3.84	$N(3.84, 0.3)$
4	6	5	0.5	8.4	$N(12.6, 6)$	6.6	$N(6.6, 1.5)$
5	4	2	2	21	$N(31.5, 6))$	4.8	$N(4.8, 1.2)$
6	1	5.5	2	22.4	$N(33.6, 6)$	4.56	$N(4.56, 1.5)$
7	3	6	4	22.4	$N(33.6, 6)$	4.68	$N(4.68, 2.7)$
8	3	4	4	22.4	$N(33.6, 6)$	4.8	$N(4.8, 1.8)$
9	1	4	4	22.4	$N(33.6, 6)$	4.56	$N(4.56, 1.2)$
10	1	4	4	22.4	$N(33.6, 6)$	4.56	$N(4.56, 0.6)$

TABLE I

PARÂMETROS DAS USINAS

Demanda	Patamar 1	Patamar 2	Patamar 3
d^1	13	10	6.5
d^2	$N(12, 6)$	$N(11, 5)$	$N(7, 3.2)$

TABLE II

VALORES DAS DEMANDAS

onde $N(a, b)$ indica uma distribuição normal com média a e desvio padrão b.

2) Resultados: Após a execução do método de feixe com e sem o *Conjunto Localizador*, foi possível obter os valores de expansão e operação de cada usina em cada patamar para o primeiro estágio.

A tabela III mostra os resultados obtidos ao otimizar sem considerar o *Conjunto Localizador*. A execução durou menos de 1s e retornou o resultado em apenas 2 iterações.

Usina	Expansão	Geração 1	Geração 2	Geração 3
1	0	0	0	4
2	6	0	3.8	0
3	0	0	5	0
4	5	3.7	0	0
5	0	0.3	1.2	2.5
6	0	1	0	0
7	0	3	0	0
8	0	3	0	0
9	0	1	0	0
10	0	1	0	0
Total	**11**	**13**	**10**	**6.5**

TABLE III

RESULTADOS OPERAÇÃO SEM CONJUNTO LOCALIZADOR

Os resultados mostram que o algoritmo optou por uma expansão de 11 unidades das 17 necessárias ao final dos dois estágios. Além disso, todas as demandas foram atendidas e capacidades respeitadas.

Já a tabela IV mostra os resultados obtidos executando o algoritmo com o *Conjunto Localizador*. A execução durou menos de 2s e retornou o resultado em 33 iterações, entretanto apenas quatro chamadas da função objetivo foram realizadas.

Usina	Expansão	Geração 1	Geração 2	Geração 3
1	0	0	4	0
2	6	0.3	1	2.5
3	0	0	5	0
4	5	3.7	0	0
5	0	4	0	0
6	0	0	0	1
7	0	0	0	3
8	0	3	0	0
9	0	1	0	0
10	0	1	0	0
Total	**11**	**13**	**10**	**6.5**

TABLE IV
RESULTADOS OPERAÇÃO COM CONJUNTO LOCALIZADOR

Os resultados obtidos foram muito similares. A expansão de 11 unidades foi mantida e os limites continuaram sendo respeitados. A única diferença foi a permutação de qual usina opera para atender qual patamar.

VI. CONCLUSÃO

Neste trabalho foi apresentado o Método de Feixe e como ele pode ser aplicado a problemas estocásticos. Essa ferramenta se mostrou robusta e eficiente para trabalhar com essa classe de problemas complicada, não abordada nessa disciplina.

A utilização do *Conjunto Localizador* traz ainda outra vantagem, a redução no número de chamadas da função objetivo. Esse mecanismo permite que o método se aproxime da solução ótima por várias iterações seguidas sem avaliar a função objetivo, o que traz uma ganho assustador para a solução de problemas onde essa avaliação é custosa.

Outra vantagem desse método é que ele trabalhou bem com problemas com variáveis inteiras. Problemas desse tipo costumam apresentar dificuldade na solução, principalmente os não lineares, por terem maior dificuldade em lidar com grande quantidade de restrições, que são geralmente impostas pelos algoritmos como o *Branch and Bound* ao longo do processo de busca. Quando o Método de Feixe transforma o problema em linear, essa dificuldade é minimizada.