OTIMIZAÇÃO RESTRITA

BIBIANA M. PETRY, CLODOALDO DE B. LAMBIASE, ISRAEL G. DE OLIVEIRA

Disciplina de Otimização Matemática, Programa de Pós Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Sul
Av. Osvaldo Aranha 103, 90035-190, Porto Alegre, RS, Brasil
E-mails: bibianapetry@gmail.com, lambiase@gmail.com, prof.israel@gmail.com

Abstract— For most problems of minimization, that need an optimization method to give a solution, have restrictions and, generally, the optimal solution is outbound the restrictions. Every times we have limited resources, or limited combination of these resources, and an optimal numeric method that care about these restrictions of resources is needed. This work is about four implementing numerical methods to minimize a function with restrictions, this methods are: projected gradient, reduced gradient, newton with barrier and newton with penalties. For each one of these methods, we show the particularities, details about the implementing, response for different initializations and tests, graphics showing the steps and data about the solution obtained. Also we use GAMS with three differents solvers and functions from Matlab to compare with main methods implemented.

Keywords— Constrained programming, Matlab, GAMS, Numeric Methods.

Resumo— A maior demanda por busca de solução para funções não lineares é para as que contêm restrições. Problemas provenientes do mundo real, geralmente possuem limites devido ao tempo, aos recursos e ao espaço serem variáveis finitas. E métodos de otimização restrita são amplamente empregados a fim de obter a solução ótima para tais problemas. Alguns dos principais métodos são o gradiente projetado, o gradiente reduzido, penalidades e barreiras. O presente trabalhou executou o desenvolvimento desses métodos em Matlab e a sua consequente aplicação para a otimização restrita de uma função não linear proposta. A partir dos resultados coletados foram feitas algumas comparações e análises das características de cada método.

Palavras-chave— Programação Restrita, Matlab, GAMS, Métodos Numéricos.

1 Introdução

Os problemas de programação não linear podem ser divididos em dois grupos: irrestritos e restritos. No primeiro caso deseja-se obter o ponto de mínimo da função sem restrições que delimitem a solução a uma determinada área do espaço R^n. Já no segundo caso, existe a redução do espaço factível para encontrar a solução.

Os principais métodos de busca de uma solução ótima para os problemas restritos são o gradiente projetado, gradiente reduzido, barreiras e penalidades. Os dois primeiros são considerados métodos primais e, têm como característica principal, a atuação direta no gradiente. Os outros métodos baseiam-se na aplicação de penalidades a pontos fora da região factível (penalidades) ou próximos as restrições (barreiras). A aplicação de penalidades é possibilitada pela criação de uma nova função objetivo contemplando essas restrições. O presente trabalho objetiva apresentar simulações de otimização matemática restrita, a partir de algoritmos desenvolvidos para os métodos citados, além de comparar as soluções com funções pré-programadas do software Matlab e do Pacote GAMS.

2 Implementação do Problema de Otimização Restrita

2.1 Avaliação da Função Objetivo e Escolha das Restrições

O problema de otimização abordado no presente trabalho consiste em minimizar uma função (equação 1) sujeita às variáveis x1 e x2 ∈ ℝ. Tal estudo objetiva encontrar o ponto **x** tal que f(**x**) é o valor mínimo. Haja vista que se trata de um problema de otimização matemática restrita, determinaram-se oito restrições lineares de desigualdade (equações 2 a 9).

$$\min \frac{3.x_1}{x_2{}^2+1} + 2x_2 - 20x_1 \operatorname{sen}\left(\frac{x_2}{5}\right) + x_1{}^2 + 2x_2{}^2 \quad (1)$$

$$\text{s.a} \quad -x_1 - 2.x_2 \leq 16 \quad (2)$$

$$-x_1 + 0{,}5.x_2 \leq 5 \quad (3)$$

$$-x_1 + 3.x_2 \leq 10 \quad (4)$$

$$x_1 - 2.x_2 \leq 10 \quad (5)$$

$$x_1 + x_2 \leq 10 \quad (6)$$

$$x_1 - x_2 \leq 8 \quad (7)$$

$$-x_1 + x_2 \leq 4 \quad (8)$$

$$-x_2 \leq 5 \quad (9)$$

A Figura 1 demonstra o comportamento da função objetivo, bem como das restrições aplicadas ao problema de otimização. A região restrita exclui a solução ótima do problema, mas contém duas regiões com mínimos próximos ao global e ao local do problema irrestrito, conforme ilustra Figura 2.

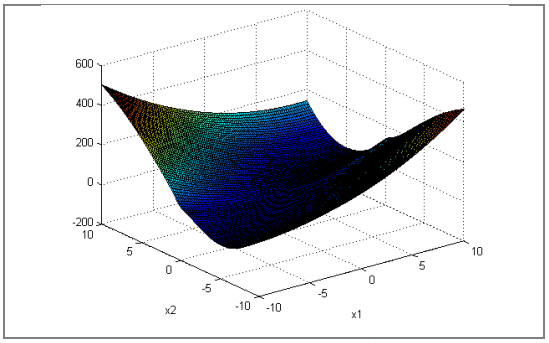

Figura 1. Ilustração do comportamento da função objetivo.

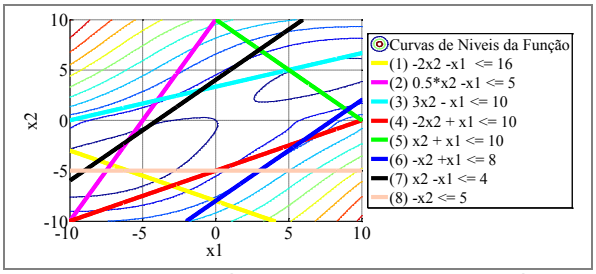

Figura 2. Ilustração da área restrita, evidenciando os mínimos restritos através das curvas de níveis da função objetivo.

Durante a solução do problema de otimização restrita, se faz necessário o uso do gradiente, uma vez que a função cresce na direção do gradiente e decresce na direção oposta (região com valor mínimo de f(x)). Os elementos do gradiente utilizados nos métodos são apresentados nas equações 10, 11 e 12.

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)^T \quad (10)$$

$$\frac{\partial f}{\partial x_1} = 2.x_1 - 20.\text{sen}\left(\frac{x_2}{5}\right) + \left(\frac{3}{{x_2}^2 + 1}\right) \quad (11)$$

$$\frac{\partial f}{\partial x_2} = 4.x_2 - 4.x_1.\cos\left(\frac{x_2}{5}\right) + 2 - \frac{6.x_1.x_2}{({x_2}^2 + 1)^2} \quad (12)$$

2.2 Definição dos Pontos Iniciais

Com o objetivo de testar o algoritmo para soluções variadas, foram escolhidos três tipos de pontos de partida: vértices (pontos de intersecção de duas restrições), pontos interiores à região restrita e pontos exteriores. Na Tabela 1 constam os pontos iniciais escolhidos e na Figura 3 é possível visualizar as três regiões que definem o tipo de ponto.

Tabela 1. Pontos de Partida Escolhidos

Vértice (V1-V2) $x_0=[x_1;x_2]^T$	Interior $x_0=[x_1;x_2]^T$	Exterior $x_0=[x_1;x_2]^T$
(2-7) $x_0=[-6;-2]^T$	$x_0=[-1;-2]^T$	$x_0=[10;1]^T$
(3-7) $x_0=[-1;-3]^T$		
(3-5) $x_0=[5;5]^T$		
(5-6) $x_0=[9;1]^T$		
(4-6) $x_0=[6;-2]^T$	$x_0=[6;0]^T$	$x_0=[0;-10]^T$
(4-8) $x_0=[0;-5]^T$		
(1-8) $x_0=[-6;-5]^T$		
(1-2) $x_0=[-7.2;-4.4]^T$		

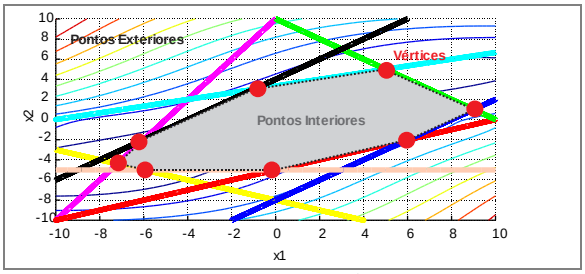

Figura 3. Ilustração das regiões que definem o tipo de ponto de partida.

2.3 Algoritmo de Otimização - Método Primal: Gradiente Projetado

O primeiro método aplicado ao problema de otimização restrita foi o gradiente projetado, segundo algoritmo de Luenberger e Ye (2008, p. 369). Basicamente, o algoritmo consiste em buscar em uma região restrita, o ponto que minimiza a função, utilizando uma estratégia de ativação de restrições para projetar a direção do passo (sempre oposta ao gradiente).

Na primeira iteração, a construção da matriz $\mathbf{A_q}$ partindo de um vértice, ativa as duas restrições de desigualdade as quais este vértice corresponde. Para os demais pontos, a matriz $\mathbf{A_q}$ inicia vazia (uma vez que todas as restrições estão inicialmente inativas). Com isso, a projeção do gradiente, realizada pela matriz $\mathbf{P_k}$, será igual à identidade caso nenhuma restrição esteja ativa.

A interpretação das componentes do vetor de direção $\mathbf{d}_k$ irá determinar se o algoritmo deve calcular um novo passo $\mathbf{x_k}$ e atualizar $\mathbf{A_q}$ ou, se deve calcular os multiplicadores de Lagrande $\boldsymbol{\lambda}$ para verificar se encontrou a solução ótima ou se necessita "relaxar" uma restrição e reprojetar o gradiente. É válido detalhar que a interpretação do vetor $\mathbf{d_k}$ consiste em determinar se alguma componente é diferente da tolerância (adotado 1e-7).

Caso seja necessário atualizar o valor de **x** para a próxima iteração, é imprescindível encontrar o melhor passo (α). Durante o desenvolvimento, este item foi sem dúvidas, o que causou maior dificuldade no entendimento do método. Surgiram questionamentos, tais como: qual a diferença de α_1 e α_2? Qual sinal adotar no α_1? Como escolher o α_1? Após entender que α_1 é o limite de busca do α_2, deduziu-se algebricamente o sinal adequado de α_1 (equações 13 a 20). Em resumo, como b - a.x ≥ 0, o α que satisfaz a equação 20 deve ser maior ou igual a zero. Por fim, a escolha do α_1 é realizada calculando o α que respeite cada uma das restrições. Dentro deste conjunto de alfas, escolhe-se como α_1 o menor positivo, para não transgredir nenhuma das restrições.

$$a.x \leq b \quad (13)$$
$$a.x - b \leq 0 \quad (14)$$
$$a.(x + \propto.d) - b \leq 0 \quad (15)$$
$$a.x + a.\propto.d - b \leq 0 \quad (16)$$
$$a.x - b + a.\propto.d \leq 0 \quad (17)$$
$$a.x - b \leq -a.\propto.d \quad (18)$$
$$\frac{a.x - b}{a.d} \leq -\propto \quad (19)$$
$$\frac{b - a.x}{a.d} \leq \propto \quad (20)$$

Por outro lado, o método de Busca Unidimensional por Segmento Áureo (S.A.) é aplicado para determinar o α_2 que ajustará o próximo passo a ser dado. Realiza-se a busca de α_2, no intervalo entre zero e α_1. A justificativa de escolha da busca por S.A., bem como o detalhamento do algoritmo desenvolvido constam na seção 2.6.

Definido o melhor ajuste de passo (α_2), atualiza-se o valor de **x**, testando-o para as oito restrições de desigualdade do problema. Se **x** atender à restrição (a.x for igual à b), significa que ativou aquela restrição. Para isso, utiliza-se uma tolerância (1e-7) a fim de garantir que se o ponto estiver extremamente próximo à restrição, entende-se que o mesmo deve deslocar-se sobre esta restrição.

Outra aplicação da tolerância é na interpretação dos elementos de $\mathbf{d_k}$, para evitar erros de truncamento no Matlab. Caso o módulo de todas as componentes de $\mathbf{d_k}$ resultarem entre 0 e 1e-7, o algoritmo interpreta que $\mathbf{d_k} = 0$ e calcula os multiplicadores de Lagrange "$\boldsymbol{\lambda}$" (que correspondem às condições de primeira ordem Karush-Kuhn-Tucker).

Por fim, o algoritmo encontra a solução ótima, caso $\boldsymbol{\lambda}_j \geq$ tolerância (adotado 1e-7).

2.4 Algoritmo de Otimização - Método Primal: Gradiente Reduzido

Este método visa separar as variáveis em básicas e não básicas. Haja vista que o problema inicial possui apenas restrições de desigualdade, são introduzidas folgas para tornas as restrições ativas (de igualdade) e o simétrico par de x. Abaixo constam os passos conforme Luenberger e Ye (2008):

1. Fazer $k = 0$ e obter um x_0 factível e determinar uma partição (I, J) inicial.
2. Calcular o gradiente da função f: $\nabla f(x_k)$.
3. Calcular o gradiente reduzido, isto é, o gradiente em relação ás variáveis independentes x_J:

$$\nabla \hat{f}(x_j) = \nabla_j f(x) - \pi(x) A^J \quad (7)$$

$$\pi(x) = \nabla_I f(x)(A^I)^{-1} \quad (8)$$

Ou seja, calcula-se

$$\rho_J = \frac{\partial \hat{f}(x_j)}{\partial x_j} = \frac{\partial f(x)}{\partial x_j} - \pi(x) A^j, \forall j \in J \quad (9)$$

4. Fazer, $\forall j \in J$, $\Delta x_j = \begin{cases} -\rho_j \text{ se } x_j > 0 \text{ } ou\ se \text{ } x_j = 0 \text{ e } \rho_j < 0 \\ 0 \quad \text{se } x_j = 0 \text{ e } \rho_j \geq 0 \end{cases}$ (10)
5. Se $\Delta x_J = 0$, para a solução atual é ótima.
6. Calcular as variações unitárias das variáveis básicas

$$\Delta x_I = -(A^I)^{-1} A^J \Delta x_J \quad (11)$$

7. Calcular $\Delta x_k = [\Delta x_I : \Delta x_J]$ (12)

$$\alpha_1 = \max\{\alpha : x_I + \alpha \Delta x_I \geq 0 \text{(factível)} \quad (13)$$

$$\alpha_2 = max\{\alpha : x_J + \alpha \Delta x_J \geq 0 \text{(factível)} \quad (14)$$

$$\alpha_L = \{\alpha_1, \alpha_2\} \quad (15)$$

$$\alpha_3 = ar\{\min f(x_k + \alpha \Delta x_k), 0 \leq \alpha \leq 0 \quad (16)$$

$$\text{Fazer} \quad x_{k+1} = \alpha_3 \Delta x_k, k = k + 1 \quad (17)$$

8. Se $\alpha_3 < \alpha_1$ então volta para o passo 2;
   Caso contrario, se $\alpha_3 = \alpha_1$ então alterar a base, ou seja, refazer a partição (I, J)
9. Retornar para o passo 2.

2.5 Algoritmo de Otimização - Métodos de Penalidades e Barreiras

O método das penalidades externas, ou simplesmente penalidades, busca um ponto de mínimo da função que respeite as restrições aplicadas. Essa adequação do ponto mínimo da função à área permitida pelas restrições é feita através de um fator de penalidade chamado μ. O método não restringe de forma definitiva que o ponto encontrado estará dentro da zona factível, e sim, atribui uma penalidade alta para pontos fora da área esperada. Assim, consequentemente os pontos acabam permanecendo dentro das restrições. As restrições são adicionadas à função objetivo original, somadas a folgas, e multiplicadas pelo termo de penalidade. A nova função objetivo pode ser solucionada como se tratasse de um problema irrestrito e a solução geral dessa nova função objetivo geralmente se encontra fora da zona factível. Mas acaba mantendo-se dentro da zona devido ao elevado custo da violação atribuído.

Já o algoritmo do método de barreiras não permite que $\mathbf{x}_k$ viole nenhuma das restrições, ou seja, o esforço necessário para romper a barreira é infinito.

2.6 Método de Busca - Determinação de Alfa

Para traçar o melhor passo em direção à solução que minimiza a função, optou-se pelo método de descida com busca unidimensional por Segmento Áureo (S.A.) para determinar o α_2. Dentre os quatro métodos estudados (Busca Dicotômica, Uniforme, Segmento Áureo e Newton), o S.A. foi escolhido, pois exige menor esforço computacional que os demais métodos de Busca Uniforme e Dicotômica (menor nº de iterações) e não necessita da derivada simbólica (que é caso do Newton). Lembrando que o esforço computacional realizado pelo Matlab para cálculos simbólicos não compensa se comparado ao S.A., no problema proposto pelo presente trabalho.

3 Análise dos Resultados Obtidos

3.1 Resultados - Método do Gradiente Projetado

Durante as iterações deste método, observou-se que ao iniciar em cima de alguma restrição ou nos vértices, os passos são dados sobre as restrições (não aplicável para pontos interiores, tão pouco exteriores/infactíveis). A Figura 4 ilustra a trajetória de dois passos distintos, saindo de um vértice, deslocando em cima das restrições e finalizando em outro. O quadrado pontilhado evidencia um ponto mínimo restrito localizado próximo ao mínimo local irrestrito. Já o círculo pontilhado exibe um ponto mínimo restrito próximo ao mínimo global. Neste método, metade dos pontos convergiram para $x^*=[-7.2;-4.4]^T$ e a outra metade para $x^*=[6.1883; 3.8117]^T$.

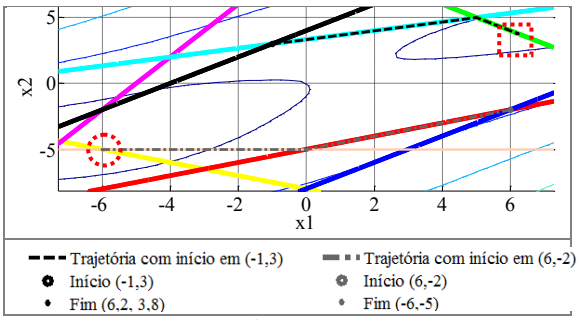

Figura 4. Ilustração de trajetórias de dois pontos que ocorrem em cima das restrições.

Observou-se também que quando distante de uma próxima restrição, o passo dado era maior (Figura 5 - flechas vermelhas) do que quando perto (Figura 5 - flechas azuis). Isso ocorre devido à limitação (α_1) da busca do melhor alfa (α_2). A quantidade de iterações até a restrição varia também varia conforme a tolerância na busca de alfa.

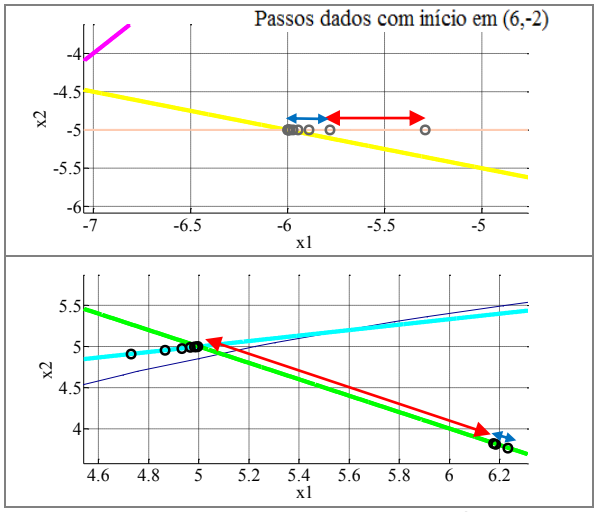

Figura 5. Ilustração de passos pontuais das trajetórias de dois pontos ao se aproximarem de restrições ou do ponto mínimo restrito.

Ao iniciar a iteração em um ponto interior, verificou-se que o gradiente não é projetado. Contudo, quando o gradiente aponta para uma restrição, o tamanho do passo é reduzido até que a restrição seja ativada. É válido ressaltar que mesmo um ponto estando próximo a uma restrição, ele só irá andar sobre a mesma, se esta estiver no caminho do gradiente projetado (conforme mostra a Figura 6). Caso contrário, o ponto segue o caminho para onde o gradiente aponta.

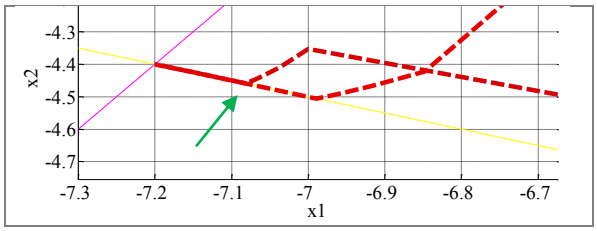

Figura 6. Ilustração de ativação de restrição.

Luenberger e Ye (2008) afirmam que neste método é necessário escolher pontos iniciais factíveis. Mas por que não podem ser escolhidos pontos exteriores à região restrita? Caso um dado ponto infactível seja escolhido, não há garantias de que a convergência ocorra para o mínimo que atenda as restrições. A Figura 7 ilustra o resultado de três simulações com pontos exteriores, convergindo ora para os mínimos locais restritos (sinalizados pela flecha verde), ora para mínimo local irrestrito (resultado infactível - sinalizado pela flecha azul).

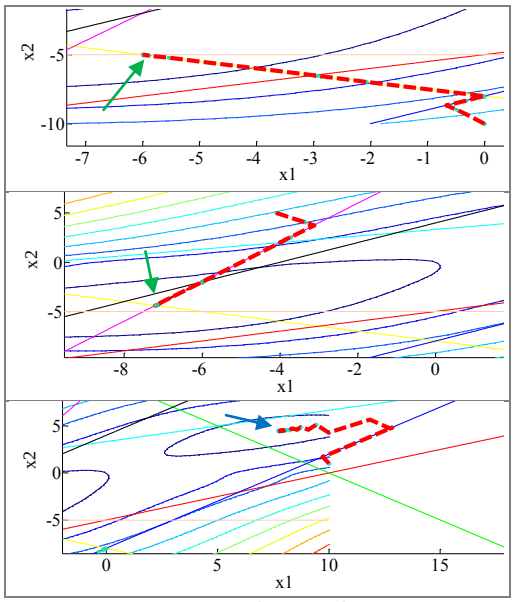

Figura 7. Ilustração de trajetórias de três pontos de partida exteriores.

Constatou-se ao longo dos testes que a mudança de $\mathbf{P_k}$ ocorre majoritariamente por causa da localidade do ponto inicial e das restrições que possivelmente estejam no caminho do gradiente projetado. Também percebeu-se que a diminuição na tolerância da busca do alfa resulta em mais iterações de busca e menos atualizações de **x** (menos passos dados). Contudo, para alguns pontos com a mesma tolerância de busca obteve-se divergência no algoritmo (tolerâncias menores → maior ocorrência de falhas).

A Figura 8 ilustra uma situação peculiar no método de gradiente projetado para o mesmo passo inicial, variando a tolerância da busca do α. Quando a tolerância é 0,05, **x** coincidiu no ponto onde o gradiente está na direção do mínimo global, reduzindo o número de passos até o mínimo restrito (reta em vermelha). Já para as outras tolerâncias, **x** passou do ponto ótimo e precisou de inúmeros passos até ativar a restrição amarela.

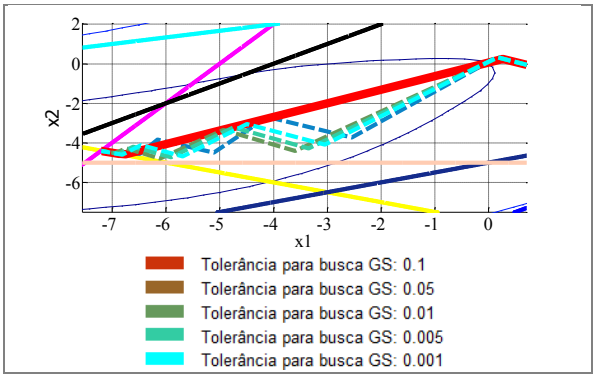

Figura 8. Ilustração de ativação de restrição.

Ao escolher pontos sobre as restrições ou muito próximos a elas (interiores ou exteriores), o algoritmo encontra diferentes mínimos restritos. A Figura 9 ilustra esta particularidade. Para diferentes pontos de partida, o ponto final pode coincidir na seta verde ou na azul. Ambas atendem às condições de primeira

ordem, mas a flecha verde apresenta menor valor para f(x). Já para alguns pontos de partida exteriores, o método converge outro mínimo infactível, fora da região restrita, conforme mostra a flecha vermelha.

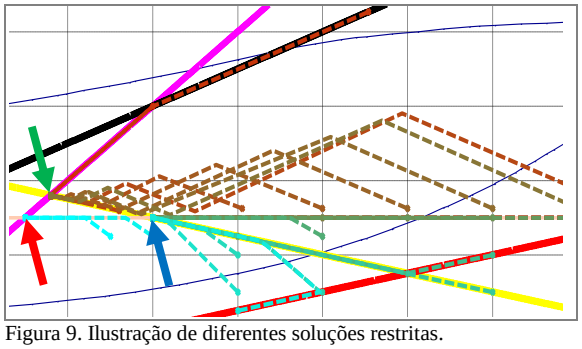

Figura 9. Ilustração de diferentes soluções restritas.

3.2 Resultados - Método do Gradiente Reduzido

Mesmo tendo uma iteração (um passo), percebe-se que as restrições são respeitadas quando se usa o gradiente reduzido. Contudo, é necessário que as bases (I, J) nas quais o gradiente é reduzido, sejam alteradas quando parte de um ponto em uma restrição. A Figura 10 apresenta as soluções encontradas neste método.

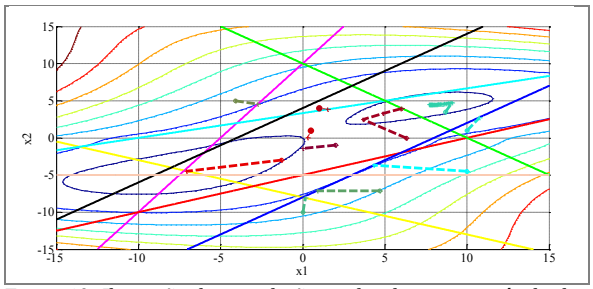

Figura 10. Ilustração das simulações realizadas para o método do gradiente reduzido.

3.3 Resultados - Métodos de Penalidades e Barreiras

Primeiramente, o problema restrito foi resolvido pelo método de penalidades. Em uma primeira tentativa com todas as oito restrições adicionadas o método não convergiu para o ponto esperado. Revisando o algoritmo implementado, a parte em negrito da equação 21 (que inclui as restrições multiplicadas pelo fator de penalidade μ) deveria tender a zero. No entanto, o resultado apresenta um comportamento crescente. O erro fica explícito, ao traçarmos a trajetória deste ponto, mostrada na Figura 11.

$$\min f(x)+\mu \sum_{i=1}^{m}[h_i(x)]^2 + \boldsymbol{\mu} \sum_{\boldsymbol{j=1}}^{\boldsymbol{p}}\left[\boldsymbol{g'}_j(\boldsymbol{x},\boldsymbol{s})\right]^{\boldsymbol{2}} \qquad (21)$$

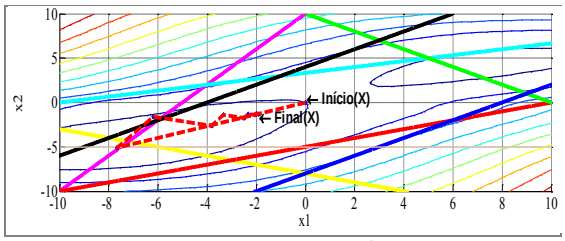

Figura 11. Ilustração de erro ocorrido no método de penalidades.

Realizando algumas correções e refazendo os testes, certos pontos apresentam a solução correta, como o exemplo mostrado na Figura 12. Em 10 iterações, o termo em negrito da equação 21 resultou em 5.4603e-05.

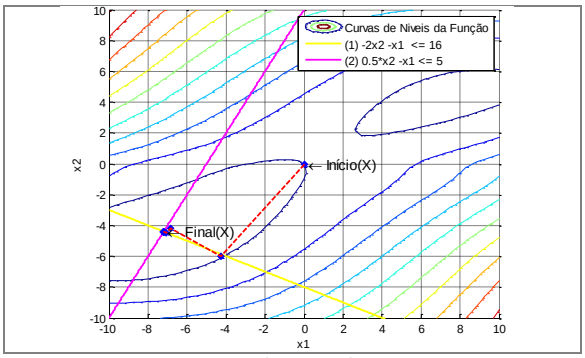

Figura 12. Ilustração de trajetórias de três pontos de partida exteriores.

Haja vista os problemas ocorridos com o método de penalidades, explorou-se o método de barreiras, o qual funcionou corretamente. A Figura13 ilustra as simulações realizadas, evidenciando que todos os pontos externos são repelidos, os pontos internos se deslocam para os mínimos restritos, os pontos em vértices ou em cima de restrições permanecem no mesmo local. Neste caso, faz-se necessário incluir um infinitésimo de x (dx), para que o mesmo entre para a região factível e encontre a solução.

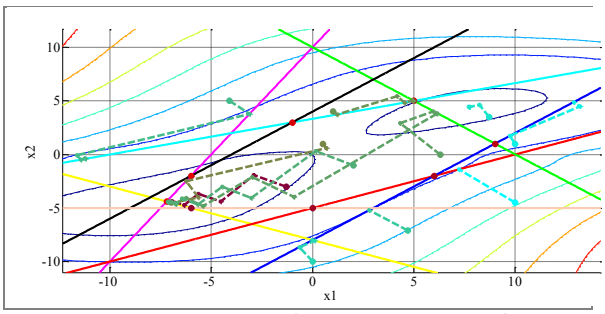

Figura 13. Ilustração das trajetórias percorridas no método de barreiras.

3.4 Resultados - Comparativo com Funções Pré-Programadas do Matlab e Pacote GAMS

A fim de comparar as soluções obtidas nos algoritmos desenvolvidos, foi sumulado o problema de otimização restrita em funções pré-programadas do Matlab e do software GAMS (*General Algebraic Modeling System*). Os pontos de partida (listados na Tabela 1) foram simulados utilizando no Matlab

através da função "fmincon" (*find minimum of constrained nonlinear multivariable function*) e nos três *solvers* do GAMS - Lindo, Conopt e Minos.

A função "fmincon", que baseia-se no método de barreiras, realizou de 1 a 9 iterações para solucionar o problema restrito, sendo que 42% convergiram para o mínimo restrito localizado próximo ao mínimo global (Figura 14). Ao simular no GAMS, verificou-se que todos os *solvers* encontraram a solução ótima restrita, no entanto, apenas o "Lindo" convergiu sempre para o mesmo ponto. Essa propriedade consta em seu manual, o qual declara que este *solver* é capaz de encontrar o ponto de mínimo de funções sem derivada de 1º ordem definida. Característica esta, que os *solvers* Conopt e Minos não possuem, por exemplo. Em contrapartida, esta propriedade gera um esforço computacional muito maior. Enquanto os *solvers* Minos e Conopt apresentam entre 1 e 8 iterações em seus relatórios de compilação/execução, solver Lindo realizou até 613 iterações para que o mínimo restrito fosse encontrado (dados detalhados na Tabela 2). Constatou-se ainda, que ao iniciar no vértice x_0=[-7.2;-4.4] que coincide com a solução ótima restrita, apenas o *solver* Minos realizou uma iteração, os outros *solvers* efetuaram mais iterações.~

Comparando com os scripts implementados, o gradiente projetado executou de 1 a 8 iterações para obter uma solução ótima. O que demonstra uma similaridade grande com os solver e funções prontas.

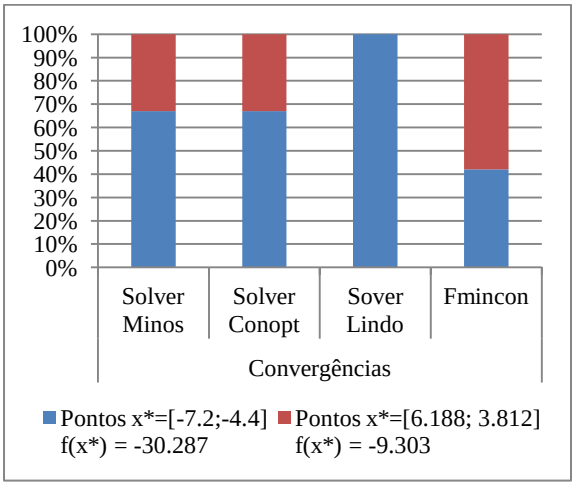

Figura 14. Percentual de convergências das funções pré-programadas para as duas soluções.

Tabela 2. Pontos de Partida Escolhidos

Parâmetro	Funções Pré Programadas	Pontos	
		x*=[-7.2;-4.4] f(x*) = -30.287	x*=[6.18; 3.81] f(x*) = -9.303
Mínimo de Iterações	Solver Minos	1	4
	Solver Conopt	4	7
	Sover Lindo	606	0
	Fmincon	1	4
Máximo de Iterações	Solver Minos	4	3
	Solver Conopt	8	8
	Sover Lindo	613	0
	Fmincon	5	9

4 Conclusão

Os métodos de otimização restrita em sua maioria obtiveram êxito na busca por uma solução ótima dentro dos limites impostos. Foi possível observar na prática as características de cada método no que tange o caminho percorrido, o número de iterações, o atendimento (ou violação) das restrições e o comportamento diante de diferentes pontos de início.

As soluções ótimas encontradas pelos métodos se concentraram em dois pontos que foram x*=[-7.2;-4.4] e x*=[6.18; 3.81]. A tendência para uma ou outra solução dependia da direção do gradiente, no caso dos métodos gradiente projetado e reduzido, ou da influência das penalidades, no caso do método de barreiras e penalidades.

Todos os métodos garantem que para um ponto inicial dentro da região factível será encontrada uma solução ótima. Entretanto, para pontos fora da região factível apenas o método de penalidades pode apresentar uma solução ótima. Os demais métodos não foram projetados para oferecer tal propriedade.

Um dos *solvers* disponibilizados pelo pacote GAMS encontrou a solução ótima em 100 das simulações, assim como a função pré-programada fmincon do Matlab, inclusive para pontos externos. O que indica a utilização do método de penalidades externas.

As dificuldades encontradas concentram-se principalmente no desenvolvimento prático do gradiente reduzido devido a dificuldade de automatizar tal processo complexo. E no caso do método de penalidades não foi possível obter resultados satisfatórios com mais de duas restrições incluídas.

Referências Bibliográficas

AFFONSO, Carolina de Matos. **Mini Curso de Programação Não Linear**. Belém: UFPA, 2005.

GAMS - General Algebric Modeling System -Versão 23.7.

HAFFNER, Sérgio Luis. **Otimização Irrestrita.** Disciplina Introdução à Otimização Matemática. Porto Alegre: UFRGS, 2012.

LUENBERGER, David G. e Ye, Yinyu. **Introduction to linear and nonlinear programming**. Addison -Wesley Publising Company Inc., 2008.

MARTÍNEZ, J. M.; SANTOS, A. S. **Métodos computacionais de otimização**. Campinas: Unicamp, 1995.

MATLAB, The Language of Technical Computing. Versão R2013a.