Minimização irrestrita usando gradientes conjugados e regiões de confiança*

John Lenon C. Gardenghi
Profa. Dra. Sandra Augusta Santos
Departamento de Matemática Aplicada · IMECC · Unicamp

8 de maio de 2012

Resumo

Este trabalho aborda o método de gradientes conjugados para resolução do subproblema de regiões de confiança para minimização irrestrita. Nosso objetivo consiste em descrever um estudo intuitivo e detalhado sobre este método, partindo de uma introdução aos métodos de direções conjugadas, alguns pré-requisitos e ferramentas necessárias para a compreensão dos gradientes conjugados e sua integração com a estratégia de região de confiança para minimização irrestrita. A implementação computacional do método no *CAS Maxima* proporcionou a experimentação numérica, que validou o estudo e a implementação feitos e permitiu uma comparação para problemas de quadrados mínimos com o método de Levenberg-Marquardt.

Palavras-chave gradientes conjugados, regiões de confiança, minimização irrestrita, CAS Maxima.

Abstract

This work focus on the conjugate gradient method to solve the trust region subproblem for unconstrained minimization. We aim to describe an intuitive and detailed study about this method, starting from an introduction to methods of conjugate directions, some necessary requisites and tools for understanding the conjugate gradient method and its integration with the trust region strategy for unconstrained minimization. The computational implementation of the method using the CAS Maxima enabled the numerical experiments, which validated the study and the implementation done and allowed a comparison between conjugate gradient and Leverberg-Marquardt for least squares problems.

Key words conjugate gradients, trust region, unconstrained minimization, CAS Maxima.

*Este trabalho contou com o apoio do CNPq (Processos 304032/2010-7, 106731/2011-4 e 151749/2011-6) e PRONEX-Otimização.

1 Introdução

Neste trabalho, vamos considerar o problema de minimização irrestrita:

$$\begin{aligned} &\min f(x) \\ &\text{s.a. } x \in \mathbb{R}^n, \end{aligned} \tag{1}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ é suave e duas vezes continuamente diferenciável em $\mathbb{R}^n$.

Para resolver o problema (1), vamos usar o método de regiões de confiança, definindo a cada iteração o modelo quadrático $q_c(p)$ em torno do ponto corrente x_c e resolvendo, exata ou aproximadamente, o subproblema:

$$\begin{aligned} \min q_c(p) \quad &= f(x_c) + g_c^T p + \frac{1}{2} p^T H_c p \\ \text{s.a. } \|p\| &\leq \Delta_c, \end{aligned} \tag{2}$$

onde Δ_c é o raio da região de confiança corrente, $g_c \equiv \nabla f(x_c) \in \mathbb{R}^n$ é o vetor gradiente, $H_c \equiv \nabla^2 f(x_c) \in \mathbb{R}^{n \times n}$ é a matrix hessiana da função objetivo avaliada no ponto x_c e $\|.\|$ é a norma euclidiana. A solução p^* do problema (2) será o passo para obter o próximo ponto para o problema (1), $x_n = x_c + p^*$.

Neste contexto, vamos abordar o método de **gradientes conjugados** para resolução do subproblema (2). Nosso objetivo consiste em descrever um estudo intuitivo e detalhado sobre este método, partindo de uma introdução aos métodos de direções conjugadas, alguns pré-requisitos e ferramentas necessárias para a compreensão dos gradientes conjugados e sua integração com a estratégia de região de confiança para mimimização irrestrita.

2 Métodos de direções conjugadas

Para estudar o método de direções conjugadas, considere o problema de resolver um sistema linear possível e determinado

$$Ax = b, \tag{3}$$

com matriz dos coeficientes $A \in \mathbb{R}^{n \times n}$ simétrica e definida positiva, vetor dos termos independentes $b \in \mathbb{R}^n$ e vetor das incógnitas $x \in \mathbb{R}^n$. A solução x^* de (3) é o minimizador do problema quadrático:

$$\begin{aligned} \min q(x) \quad &= \frac{1}{2} x^T A x - b^T x + c \\ \text{s.a. } x &\in \mathbb{R}^n. \end{aligned} \tag{4}$$

Essa equivalência nos permite analisar o algoritmo como uma estratégia tanto para resolver um sistema linear quanto um problema de minimização quadrática, pois resolver (3) equivale a encontrar um ponto estacionário para a função q em (4). De fato, como:

$$\nabla q(x) = Ax - b,$$

então

$$\nabla q(x^*) = 0 \leftrightarrow Ax^* = b.$$

Sendo q estritamente convexa, pois A é positiva definida, condições necessárias de primeira ordem são suficientes para concluir que x^* é minimizador do problema (4).

Primeiro, vamos definir os seguintes conceitos:

Erro O erro corresponde, na i-ésima iteração, a quanto nos falta para chegar à solução x^* do problema:

$$e_i = x_i - x^*; \tag{5}$$

Resíduo O resíduo corresponde ao valor de quão longe estamos de satisfazer as equações lineares do problema (3), equivalendo ao oposto do gradiente da função q avaliado em x_i:

$$r_i = b - Ax_i = -\nabla q(x_i). \tag{6}$$

Como x^* é solução do problema (3), o erro e o resíduo se relacionam da seguinte maneira:

$$\begin{aligned} r_i &= b - Ax_i \\ r_i &= Ax^* - Ax_i = A(x^* - x_i) \\ r_i &= -Ae_i, \text{ pela Equação (5)} \end{aligned} \tag{7}$$

Com isso, suponha que nosso objetivo a cada iteração é definir

$$x_{i+1} = x_i + \alpha_i d_i, \tag{8}$$

de tal forma que a direção d_i seja esgotada, ou seja, que encontremos o passo ótimo para percorrê-la. A maneira mais direta que temos é estabelecer a ortogonalidade entre a i-ésima direção e o termo do erro da etapa $(i+1)$:

$$d_i^T e_{i+1} = 0. \tag{9}$$

De (8) e (9), podemos escrever:

$$\begin{aligned} d_i^T(e_i + \alpha_i d_i) &= 0 \\ \alpha_i = -\frac{d_i^T e_i}{d_i^T d_i}. \end{aligned} \tag{10}$$

Repare que (10) em (8) é uma projeção do erro da i-ésima etapa sobre a direção d_i. Isso significa que o próximo iterando x_{i+1} estará o mais próximo possível da solução x^* dado o quanto a direção d_i é capaz de permitir, portanto (10) determina o passo ótimo a percorrer nessa direção.

Entretanto, não conhecemos o erro, pois se este fosse conhecido o problema estava resolvido. Todavia, a relação (7) nos dá uma equivalência entre o erro e o resíduo, que por sua vez é conhecido. Este fato sugere tomarmos uma referência que envolva a matriz A de tal maneira a poder usar o resíduo. Então, ao invés de estabelecer a ortogonalidade, vamos procurar direções *conjugadas* ou *A-ortogonais* ao erro. Diz-se que os vetores de um conjunto $\{v_k\}_{k=0}^{n-1}$ são A-ortogonais ou conjugados se:

$$v_i^T A v_j = 0, \forall i \neq j.$$

Então, vamos tomar:

$$d_i^T A e_{i+1} = 0. \tag{11}$$

A relação (11) nos dá dois resultados imediatos:

1. Não se conhece o erro, entretanto se conhece uma expressão que relaciona o erro com o resíduo na i-ésima iteração, em vista disso, por (7) em (10), temos:

$$\begin{aligned} \alpha_i &= -\frac{d_i^T A e_i}{d_i^T A d_i} \\ &= \frac{d_i^T r_i}{d_i^T A d_i}. \end{aligned} \tag{12}$$

2. Essa relação equivale a fazer uma busca linear exata ao longo da direção d_i. Para isso, basta pedir que a derivada direcional de q, definida em (4), na direção d_i, avaliada em $x_{i+1} = x_i + \alpha d_i$ valha zero, ou seja:

$$\begin{aligned} \frac{d}{d\alpha} q(x_i + \alpha d_i) &= 0 \\ \nabla q(x_i + \alpha d_i)^T \frac{d}{d\alpha}(x_i + \alpha d_i) &= 0 \\ (Ax_{i+1} - b)^T d_i &= 0 \\ -r_{i+1}^T d_i &= 0, \text{ pela Equação (6)} \\ d_i^T A e_{i+1} &= 0, \text{ pela Equação (7)}, \end{aligned} \tag{13}$$

portanto a relação (11) satisfaz:

$$q(x_i + \alpha_i d_i) = \min_{\alpha \in \mathbb{R}} q(x_i + \alpha d_i). \tag{14}$$

Por fim, um dos resultados mais importantes do uso da conjugação entre as direções é que um problema n-dimensional é resolvido em, no máximo, n passos. Para mostrar este resultado, inspirados na proposta de Izmailov e Solodov [5, cap. 3], vamos primeiro apresentar um lema, mostrando que um conjunto de direções conjugadas é linearmente independente.

Lema 1. *Para toda matriz $A \in \mathbb{R}^{n\times n}$ simétrica e definida positiva, um conjunto $\{d_k\}_{k=0}^{n-1}$ de direções $d_k \in \mathbb{R}^n$ A-ortogonais ou conjugadas entre si é linearmente independente.*

Dem. Vamos fazer esta prova por absurdo. Suponha que podemos escrever d_0 como uma combinação linear dos vetores do conjunto $\{d_1, \ldots, d_{n-1}\}$. Então:

$$d_0 = \sum_{\imath=1}^{n-1} \beta_\imath d_\imath, \tag{15}$$

com $\beta_\imath \in \mathbb{R}$, $\imath = 1, \ldots, n-1$. Pré-multiplicando (15) por $d_0^T A$, dado que a matriz A é definida positiva e a direção d_0 é não nula e conjugada às direções $\{d_1, \ldots, d_{n-1}\}$, temos que:

$$0 < d_0^T A d_0 = \sum_{\imath=1}^{n-1} \beta_\imath d_0 A d_\imath = 0,$$

o que nos dá a contradição. ∎

O Lema 1 garante que é possível formar uma base para o $\mathbb{R}^n$ com o conjunto $\{d_k\}$ das n direções conjugadas. Vamos à prova de que qualquer método de direções conjugadas resolve o problema (4) em, no máximo, n iterações.

Teorema 1. *Para qualquer ponto inicial $x_0 \in \mathbb{R}^n$, a sequência $\{x_k\}$ gerada por (8), escolhendo-se qualquer conjunto $\{d_0, d_1, \ldots, d_{n-1}\}$ de direções conjugadas entre si, converge para x^*, solução global do problema (4), em, no máximo, n passos.*

Dem. Seja $x^* \in \mathbb{R}^n$ solução do problema (4). Pela convexidade do problema, x^* é solução global.

Como o conjunto $\{d_0, d_1, \ldots, d_{n-1}\}$ é linearmente independente, pelo Lema 1, podemos escrever o erro inicial como combinação linear destas direções. Temos, então:

$$e_0 = x_0 - x^* = \sum_{j=0}^{n-1} \sigma_j d_j, \tag{16}$$

com $\sigma_j \in \mathbb{R}, j = 0, 1, \ldots, n-1$. Mas por (8), podemos escrever:

$$x^* = x_0 + \sum_{j=0}^{n-1} \alpha_j d_j, \tag{17}$$

para escalares α_j definidos em (12). Essa relação sugere demonstrar que $\alpha_j = -\sigma_j$, $\forall j = 0, 1, \ldots, n-1$.

Pré-multiplicando (16) por $d_k^T A$, vem que:

$$\begin{aligned}
d_k^T A e_0 &= \sigma_k d_k^T A d_k, \text{ pela conjugação das direções} \\
\Rightarrow \sigma_k &= \frac{d_k^T A e_0}{d_k^T A d_k} \\
&= \frac{d_k^T A (e_0 + \sum_{i=0}^{k-1} \alpha_i d_i)}{d_k^T A d_k}, \text{ pela conjugação das direções} \\
&= \frac{d_k^T A e_k}{d_k^T A d_k}, \text{ pela Equação (8)} \\
&= -\frac{d_k^T r_k}{d_k^T A d_k}, \text{ pela Equação (7)} \\
&= -\alpha_k, \text{ pela Equação (12)},
\end{aligned} \tag{18}$$

o que nos dá o resultado esperado. A relação (18) pode ser vista como o processo de eliminar o i-ésimo termo do erro, a cada iteração, na base formada pelas direções conjugadas $\{d_0, d_1, \ldots, d_{n-1}\}$:

$$\begin{aligned}
e_i &= e_0 + \sum_{j=0}^{i-1} \alpha_j d_j \\
&= \sum_{j=0}^{n-1} \sigma_j d_j - \sum_{j=0}^{i-1} \sigma_j d_j, \text{ pelas Equações (16) e (18)} \\
&= \sum_{j=i}^{n-1} \alpha_j d_j.
\end{aligned} \tag{19}$$

Após n iterações, todas as componentes serão eliminadas, e o erro zerado. ■

Neste ponto, sabe-se, pelo Teorema 1, que para resolver os problemas (3) e (4) pelo esquema (8) em, no máximo, n iterações, basta conhecer um conjunto de direções conjugadas entre si $\{d_k\}_{k=0}^{n-1}$. Por isso, agora, apresentamos um método para se obter este conjunto, que é a conjugação de Gram-Schmidt, e também vamos apresentar uma relação interessante entre esse método e a eliminação Gaussiana para resolução de sistemas lineares.

2.1 Conjugação de Gram-Schmidt

Esse processo é bem semelhante ao conhecido processo de ortogonalização de Gram-Schmidt, entretanto usa informações da matriz A para fazer com que os vetores aos quais forem aplicados o método se tornem conjugados entre si.

Suponha que queremos obter um conjunto de direções conjugadas entre si $\{d_k\}_{k=0}^{n-1}$. O método consiste em partir de um conjunto de vetores linearmente independentes entre si $\mathcal{U} = \{u_0,\ u_1,\ \ldots,\ u_{n-1}\}$ e construir a direção d_i subtraindo de u_i as componentes que não forem conjugadas aos vetores $\{d_0,\ \ldots,\ d_{i-1}\}$. De uma maneira prática, toma-se $d_0 = u_0$ e, para $i = 0,\ 1,\ \ldots,\ n-1$:

$$d_i = u_i + \sum_{k=0}^{i-1} \beta_{ik} d_k, \tag{20}$$

onde $\beta_{ik} \in \mathbb{R}^n$ é definido para todo $k < i$. Para encontrar seu valor, basta pós-multiplicar o transposto de (20) por Ad_j, $j < i$:

$$\begin{aligned} d_i^T A d_j &= u_i^T A d_j + \sum_{k=0}^{i-1} \beta_{ik} d_k^T A d_j \\ 0 &= u_i^T A d_j + \beta_{ij} d_j^T A d_j \\ \Rightarrow \beta_{ij} &= -\frac{u_i^T A d_j}{d_j^T A d_j}, \quad j < i, i = 0, 1, \ldots, n-1. \end{aligned} \tag{21}$$

Com isso, temos:

$$d_0 = u_0$$

e, para $i \geq 1$:

$$\begin{aligned} d_i &= u_i - \sum_{j=0}^{i-1} \frac{u_i^T A d_j}{d_j^T A d_j} d_j \\ &= \left(I - \sum_{j=0}^{i-1} \frac{d_j d_j^T A}{d_j^T A d_j} \right) u_i. \end{aligned} \tag{22}$$

Do ponto de vista computacional, o método de Gram-Schmidt apresenta a grande desvantagem de requerer o armazenamento de todas as direções usadas até a i-ésima etapa para se obter a direção da etapa $(i+1)$.

O principal ponto a ser destacado aqui é que obter direções conjugadas equivale a obter direções ortogonais em um espaço reescalado. Ora, basta tomar $A = I$ que a conjugação se

torna ortogonalidade e o processo de *conjugação* de Gram-Schmidt é o próprio processo de *ortogonalização* de Gram-Schmidt.

Agora vamos analisar a escolha do conjunto $\{u_k\}$. Embora escolhas mais inteligentes sejam possíveis – veremos adiante que o método de gradientes conjugados é um bom exemplo disso, a escolha trivial para esse conjunto é a base canônica do $\mathbb{R}^n$, ou seja, a i-ésima coordenada do vetor u_i é 1 enquanto as demais coordenadas são 0. Aplicar o método de conjugação de Gram-Schmidt sobre uma base canônica do $\mathbb{R}^n$ equivale a aplicar a eliminação de Gauss sobre sobre uma matriz A simétrica e definida positiva [11]. É o que vamos discutir na próxima seção.

2.1.1 Uma relação interessante

Como dito anteriormente, dado o problema (3), para a matriz dos coeficientes $A \in \mathbb{R}^{n\times n}$ simétrica e definida positiva, aplicar a eliminação de Gauss equivale a usar o método de conjugação de Gram-Schmidt aplicado ao conjunto inicial de vetores canônicos do $\mathbb{R}^n$ e, portanto, a eliminação Gaussiana pode ser vista como um método de direções conjugadas. Essa relação é interessante e não trivial, e já estava presente no trabalho de Hestenes e Stiefel [4, Seção 12].

Primeiro, vamos olhar mais atentamente para o método de eliminação de Gauss. Essa estratégia consiste em transformar uma matriz arbitrária M numa matriz na forma escada, dita triangular superior, através de operações elementares, isto é, trocas de linhas (associada à estratégia de pivoteamento utilizada), multiplicação de uma linha por um escalar não nulo e substituição da i-ésima linha pela i-ésima linha mais um múltiplo da j-ésima linha. Em nosso contexto, vamos estudar essa transformação sobre matrizes quadradas, particularmente simétricas e definidas positivas, já que nosso objetivo é relacionar esse processo com conjugação de Gram-Schmidt. Por exemplo, seja a matriz A:

$$\begin{pmatrix} 8 & 2 & 1 \\ 2 & 6 & 3 \\ 1 & 3 & 4 \end{pmatrix},$$

aplicando a eliminação de Gauss, temos:

$$A = \begin{pmatrix} 8 & 2 & 1 \\ 2 & 6 & 3 \\ 1 & 3 & 4 \end{pmatrix} \xrightarrow{L_2 \leftarrow L_2 - 0.25L_1} \begin{pmatrix} 8 & 2 & 1 \\ 0 & 5.5 & 2.75 \\ 1 & 3 & 4 \end{pmatrix} \xrightarrow{L_3 \leftarrow L_3 - 0.125L_1}$$

$$\begin{pmatrix} 8 & 2 & 1 \\ 0 & 5.5 & 2.75 \\ 0 & 2.75 & 3.875 \end{pmatrix} \xrightarrow{L_3 \leftarrow L_3 - 0.5L_2} \begin{pmatrix} 8 & 2 & 1 \\ 0 & 5.5 & 2.75 \\ 0 & 0 & 2.5 \end{pmatrix} = U.$$

Neste exemplo, usamos três operações elementares e chegamos a uma matriz triangular superior. As operações elementares aplicadas sobre a matriz A podem ser expressas por matrizes $\{E_k\}_{k=1}^3$ tais que cada matriz elementar representa a operação elementar aplicada sobre a matriz identidade. Com isso, nossa matriz triangular U pode ser obtida pela expressão:

$$U = E_3E_2E_1A.$$

Repare que o processo de eliminação Gaussiana sem pivoteamento para $A \in \mathbb{R}^{n\times n}$ produz uma fatoração $A = LU$ de tal forma que:

$$L = E_1^{-1}E_2^{-1}E_3^{-1},$$

e, ainda, que desta ótica pode ser visto como um processo iterativo tal que, se denotarmos

$$\widehat{E}_j = E_m \cdots E_1, \quad j = 2, \ldots, n-1$$

onde

$$m = \left\lceil \frac{j(n-1)}{2} \right\rceil,$$

e tomarmos:

$$A^{(1)} = A$$

na k-ésima iteração, teremos:

$$A^{(k)} = \widehat{E}_k \cdots \widehat{E}_3\widehat{E}_2 A,$$

com $k = 2, \ldots, n-1$.

Como estamos considerando matrizes simétricas e definidas positivas, o processo de eliminação de Gauss é estável, seu produto final é único e a estratégia de pivoteamento é desnecessária. Por isso usamos apenas a operação elementar de eliminação por linha, em que substituimos a i-ésima linha pela i-ésima linha mais um múltiplo da j-ésima linha. Essa operação pode ser dada por:

$$A_i \leftarrow A_i - \frac{A_{i,j}}{A_{j,j}}A_j, \text{ para } i = j+1, ..., n. \tag{23}$$

A eliminação por linhas expressa em (23) consiste em zerar elementos abaixo da diagonal da matriz, das colunas da esquerda para a direita, sucessivamente, de modo a produzir uma matriz triangular superior. Neste contexto, vamos representar este processo como aplicações consecutivas sobre matrizes $\widehat{A}^{(i)} \in \mathbb{R}^{(n-i+1)\times(n-i+1)}$, com o objetivo de produzir, a cada iteração, algo da forma:

$$\widehat{A}^{(i)} = \begin{pmatrix} a_i & w_i^T \\ 0 & \widehat{A}^{(i+1)} \end{pmatrix}, \tag{24}$$

com $a_i \in \mathbb{R}$, $w_i \in \mathbb{R}^{(n-i)}$ e $\widehat{A}^{(i+1)} \in \mathbb{R}^{(n-i)\times(n-i)}$. Tradicionalmente, para a k-ésima linha, $k = 2, \ldots, n-i$, denotada por $\widehat{A}_k^{(i)}$ para a i-ésima etapa, podemos escrever:

$$\widehat{A}_k^{(i)} \leftarrow \widehat{A}_k^{(i)} - \frac{\widehat{A}_{k,1}^{(i)}}{\widehat{A}_{1,1}^{(i)}}\widehat{A}_1^{(i)}, \tag{25}$$

que é a expressão (23) aplicada à submatriz $\widehat{A}_k^{(i)}$.

Seja $Z \in \mathbb{R}^{(n-i+1)\times(n-i+1)}$ a matriz cujo vetor-coluna z_k é o k-ésimo vetor canônico do $\mathbb{R}^{(n-i+1)}$, portanto Z é a identidade de ordem $(n-i+1)$. Podemos denotar os elementos de $\widehat{A}^{(i)}$ como:

$$\widehat{A}_{i,j}^{(i)} = z_i^T \widehat{A}^{(i)} z_j. \tag{26}$$

De (26) em (25), temos:

$$(\widehat{A}_k^{(i)})^T \leftarrow \widehat{A}^{(i)} \left(z_i - \frac{z_i^T \widehat{A}^{(i)} z_1}{z_1^T \widehat{A}^{(i)} z_1} z_1 \right) = \widehat{A}^{(i)} \vartheta_i. \tag{27}$$

A semelhança entre a expressão (27) e o método de conjugação de Gram-Schmidt apresentado em (22) não é mera coincidência. Os vetores $\{\vartheta_k\}$ são, pela expressão entre parênteses, conjugação dos vetores da matriz Z com relação ao vetor z_1. Então, se definirmos a matriz

$$\widehat{T}^{(i)} = (\vartheta_1 \ \vartheta_2 \dots \vartheta_{n-i+1}), \widehat{T}^{(i)} \in \mathbb{R}^{(n-i+1)\times(n-i+1)},$$

podemos determinar uma matriz $T^{(i)}$, $i = 1, \dots, n$, tal que:

$$T^{(i)} = \prod_{k=1}^{i-1} \begin{pmatrix} I_{(k)} & 0 \\ 0 & \widehat{T}^{(k+1)} \end{pmatrix} \widehat{T}^{(1)}, T^{(i)} \in \mathbb{R}^{n\times n},$$

onde $I_{(k)}$ é a matriz identidade de ordem k. Assim, podemos aplicar:

$$A^{(i)} = T^{(i)} A,$$

e, na etapa $(n-1)$, obter a matriz triangular superior U usando $T^{(n-1)} \equiv T$:

$$U = TA. \tag{28}$$

Como o processo de eliminação Gaussiana, neste caso, produz uma fatoração $A = LU$ e de (28):

$$A = T^{-1}U,$$

podemos concluir que:

$$L = T^{-1}$$

e que T é triangular inferior. Ainda, se pós-multiplicarmos (28) por T^T, temos:

$$UT^T = TAT^T.$$

Como U e T^T são triangulares superiores, então TAT^T deve ser triangular superior. Entretanto, T é triangular inferior, por conseguinte TAT^T deve ser uma matriz diagonal. Com isso, concluímos que as colunas de T são conjugadas entre si, isto é:

$$T_i A T_j^T = 0, \forall i \neq j.$$

No Algoritmo 1, apresentamos o método para encontrar a matriz $T = T^{(n-1)}$ usando esta estratégia.

Algoritmo 1: Eliminação de Gauss e conjugação Gram-Schmidt

Dados $A \in \mathbb{R}^{n\times n}$, faça:

$T = I_n$;
$\imath = 1$;
$\widehat{A}^{(\imath)} = A$;
para $k = n, \ldots, 2$ **faça**
  $\widehat{T} = I_k$;
  para $\jmath = 2, \ldots, k$ **faça**
    $\widehat{T}_\jmath = \widehat{T}_\jmath - \dfrac{\widehat{T}_\jmath^T A^{(\imath)} \widehat{T}_1}{\widehat{T}_1^T A^{(\imath)} \widehat{T}_1} \widehat{T}_1$
  fim
  $\widehat{A}^{(\imath)} = \widehat{T}\widehat{A}^{(\imath)}$;
  $\widehat{A}^{(\imath+1)} = \widehat{A}^{(\imath)}[2:k; 2:k]$;
  se $k \neq n$ **então**
    $T = \begin{pmatrix} I_\imath & 0 \\ 0 & \widehat{T} \end{pmatrix} T$
  fim
  $\imath = \imath + 1$;
fim
retorna T;

2.2 Otimalidade do método de direções conjugadas

Métodos de direções conjugadas possuem, além da convergência apresentada no Teorema 1, uma propriedade valiosa: encontram, a cada iteração, a melhor solução possível no espaço explorado.

O espaço em questão trata-se do subespaço gerado pelas k direções conjugadas na k-ésima iteração, que são linearmente independentes como provado no Lema 1 e que denotaremos por

$$\mathcal{D}_k = \operatorname{span}\{d_0, d_1, \ldots, d_{k-1}\}. \tag{29}$$

Neste contexto, a melhor solução pode ser interpretada de dois pontos de vista diferentes, entretanto equivalentes:

- O método escolhe o valor do erro e_k na variedade afim $e_0 + \mathcal{D}_k$ que minimiza o quadrado da norma de energia, isto é, que minimiza $\|e_k\|_A^2$, para todo $k = 0, \ldots, n-1$.

  Para ver essa propriedade, retomemos a expressão (19), em que o erro era escrito como combinação linear das direções. Sabe-se que $\|e_k\|_A^2 = e_k^T A e_k$, portanto, podemos

expressar a norma da energia também em termos das direções conjugadas, ou seja:

$$\begin{aligned} \|e_k\|_A^2 &= \sum_{j=k}^{n-1}\sum_{\ell=k}^{n-1} \alpha_j \alpha_\ell d_j^T A d_\ell \\ &= \sum_{j=k}^{n-1} \alpha_j^2 d_j^T A d_j, \text{ pela conjugação das direções.} \end{aligned} \tag{30}$$

Repare que as direções com as quais a norma da energia do erro na k-ésima etapa se relaciona na expressão (30) ainda não foram usadas, ou seja, qualquer outro valor do erro tomado no espaço $e_0 + \mathcal{D}_k$ será formado por combinações dessas direções, alem das que já foram usadas. Com isso, podemos concluir que a norma do erro obtida em (30) é minima. Isto é consequência direta da expressão (19), quando concluímos que, a cada iteração, uma componente do erro era zerada e, portanto, após n iterações, o erro é zerado.

- A cada iteração k, o iterando x_k minimiza a função q definida em (4) na variedade afim $x_0 + \mathcal{D}_k$, para todo $k = 0, \ldots, n-1$.

  Se tomarmos $i = 0, \ldots, k$, temos:

$$\begin{aligned} \nabla q(x_{k+1})^T d_i &= (Ax_{k+1} - b)^T d_i \\ &= x_{k+1}^T A d_i - b^T d_i \\ &= (x_{i+1} + \sum_{j=i+1}^{k} \alpha_j d_j)^T A d_i - b^T d_i, \text{ pela relação (8)} \\ &= x_{i+1}^T A d_i - b^T d_i, \text{ pela conjugação das direções} \\ &= (Ax_{i+1} - b)^T d_i \\ &= \nabla q(x_{i+1})^T d_i. \end{aligned} \tag{31}$$

  Pela equação (13) e pela relação (14), temos que:

$$\nabla q(x_{i+1})^T d_i = \nabla q(x_i + \alpha_i d_i)^T d_i = 0. \tag{32}$$

  Por conseguinte, podemos concluir de (32) em (31) que:

$$\nabla q(x_{k+1})^T d_i = 0, \forall i = 0, \ldots, k. \tag{33}$$

  A relação (33) diz que a derivada direcional ao longo da direção d_i, $i = 0, \ldots, k$, sempre será zero no iterando x_{k+1}, obtido na k-ésima iteração. Como o problema é quadrático, vem que x_{k+1} é minimizador de q no espaço $x_0 + \mathcal{D}_k$, para todo $k = 0, \ldots, n-1$. Vale dizer que esta proposição reafirma o Teorema 1 pois, se x_k minimiza q na k-ésima iteração, então, na pior das hipóteses, quando $k = n$, $x^* = x_n$ será o minimizador de q no $\mathbb{R}^n$.

Na próxima seção, vamos apresentar o método de gradientes conjugados, suas principais propriedades teóricas e porque ele é um método de direções conjugadas que se sobressaiu aos demais.

3 Método de gradientes conjugados

O Método de Gradientes Conjugados (MGC) é um método de direções conjugadas bem interessante por suas propriedades teóricas e desempenho, especialmente em problemas de grande porte. Em linhas gerais, o MGC é uma estratégia de direções conjugadas em que os passos de busca são construídos pela conjugação dos resíduos (que são os opostos dos gradientes, como definimos em (6)). De forma mais específica, nosso conjunto de vetores linearmente independentes $\mathcal{U}$, definidos na Seção 2.1, a partir do qual o conjunto de direções conjugadas é construído, será composto pelos resíduos do sistema linear (3) nas respectivas iterações.

A escolha dos gradientes para este conjunto apresenta algumas propriedades teóricas bem úteis. Primeiramente, vamos mostrar que os gradientes formam um conjunto de vetores linearmente independentes, já que este é o primeiro requisito para definição do conjunto $\mathcal{U}$. Além disso, vamos mostrar que esse conjunto apresenta uma propriedade mais interessante: é ortogonal. Essa propriedade será o principal ingrediente para mostrar porque o método dos gradientes conjugados é o destaque entre os métodos de direções conjugadas.

Lema 2. *Seja $q : \mathbb{R}^n \to \mathbb{R}$ definida em (4), com $A \in \mathbb{R}^{n\times n}$ simétrica e definida positiva e $b \in \mathbb{R}^n$. Dado o ponto inicial $x_0 \in \mathbb{R}^n$ e a sequência $\{x_k\}$ gerada pelo metodo de gradientes conjugados, então para todo $k = 1, \ldots, n$, os gradientes $\nabla q(x_0), \nabla q(x_1), \ldots, \nabla q(x_{k-1})$ compõem um conjunto de vetores ortogonais em $\mathbb{R}^n$.*

Dem. A prova é feita por indução. Para $k = 2$ os gradientes $\nabla q(x_0)$ e $\nabla q(x_1)$ são ortogonais, pois $d_0 = -\nabla q(x_0)$ e pela relação (13):

$$\nabla q(x_1)^T d_0 = 0.$$

Agora vamos supor que esta relação seja verdadeira para $k > 2$, isto é:

$$\nabla q(x_k)^T \nabla q(x_i) = 0, \forall i = 0, 1, \ldots, k-1. \tag{34}$$

Como o nosso conjunto $\mathcal{U}$ será composto pelos gradientes, temos, de (20), para $u_i = r_i = -\nabla q(x_i)$:

$$\begin{aligned} d_i &= -\nabla q(x_i) + \sum_{\ell=0}^{i-1} \beta_{i\ell} d_\ell \\ \nabla q(x_i) &= -d_i + \sum_{\ell=0}^{i-1} \beta_{i\ell} d_\ell, \text{ para } i = 0, \ldots, k. \end{aligned} \tag{35}$$

Sabendo que (33) vale, de (34) e (35), temos:

$$\begin{aligned} \nabla q(x_{k+1})^T \nabla q(x_i) &= \nabla q(x_{k+1})^T (-d_i + \sum_{\ell=0}^{i-1} \beta_{i\ell} d_\ell) \\ &= -\nabla q(x_{k+1})^T d_i + \sum_{\ell=0}^{i-1} \beta_{i\ell} \nabla q(x_{k+1})^T d_\ell \\ &= 0, \quad \forall i = 0, 1, \ldots, k, \end{aligned}$$

que nos dá o resultado esperado. ■

Frente à propriedade de ortogonalidade do conjunto $\mathcal{U}$ apresentada no Lema 2, retomemos a relação (7) do ponto de vista do processo iterativo do MGC. Temos:

$$\begin{aligned} r_{i+1} &= -Ae_{i+1} \\ &= -A(e_i + \alpha_i d_i), \text{ pela Equação (8)} \\ &= r_i - \alpha_i A d_i. \end{aligned} \tag{36}$$

A relação (36) mostra que, além da propriedade de ortogonalidade com os demais resíduos do conjunto $\mathcal{U}$, na i-ésima iteração o resíduo r_i é uma combinação linear do resíduo r_{i-1} e do vetor Ad_{i-1}. Do ponto de vista computacional, o uso desta equação proporciona uma redução na quantidade necessária de produtos matriz-vetor para computar o i-ésimo iterando, visto que o produto Ad_{i-1} já foi usado no cálculo do coeficiente α_i em (12).

Diante disso, dado o processo de conjugação de Gram-Schmidt, de (21), com nosso conjunto $\mathcal{U}$ formado pelos resíduos, temos:

$$\beta_{ij} = -\frac{r_i^T A d_j}{d_j^T A d_j} \tag{37}$$

Pré-multiplicando (36) reescrita no índice j por r_i^T, temos:

$$\begin{aligned} r_i^T r_{j+1} &= r_i^T r_j - \alpha_j r_i^T A d_j \\ \alpha_j r_i^T A d_j &= r_i^T r_j - r_i^T r_{j+1} \end{aligned} \tag{38}$$

Usando a propriedade de ortogonalidade do conjunto dos resíduos, de (38) teremos:

- Se $i = j$, então $r_i^T r_j \neq 0$ e $r_i^T r_{j+1} = 0$;
- Se $i = j + 1$, então $r_i^T r_j = 0$ e $r_i^T r_{j+1} \neq 0$;
- Para qualquer outro valor de i e j, $r_i^T r_j = 0$ e $r_i^T r_{j+1} = 0$.

Ora, se $r_i^T r_j = 0$ e $r_i^T r_{j+1} = 0$ então $r_i^T A d_j = 0$ e, consequentemente por (37), $\beta_{ij} = 0$, sempre que $i \neq j$ e $i \neq j + 1$. Por conseguinte, o somatório da equação (20) é eliminado, exceto para o último iterando da soma, em que $j = i - 1 \Rightarrow i = j + 1$. Devido a isso, de (38) temos que:

$$r_i^T A d_j = \frac{1}{\alpha_{i-1}} r_i^T r_i,$$

resultando em um único coeficiente não-nulo que será:

$$\beta_i \equiv \beta_{i,i-1} = \frac{1}{\alpha_{i-1}} \frac{r_i^T r_i}{d_{i-1}^T A d_{i-1}}. \tag{39}$$

Eis a grande vantagem do MGC: como dentre todos os termos do somatório de (20) só restou um devido à ortogonalidade dos elementos do conjunto $\mathcal{U}$, então não é mais necessário

armazenar todas as direções $\{d_k\}_{k=0}^{i-1}$ geradas até a i-ésima iteração, reduzindo a relação (20) para simplesmente computar:

$$d_i = r_i + \beta_i d_{i-1}, \text{ com } \beta_i \text{ dado por (39).} \tag{40}$$

Como o valor de α_{i-1} é conhecido (12), podemos manipular (39) de forma que:

$$\beta_i = \frac{r_i^T r_i}{d_{i-1}^T r_{i-1}}. \tag{41}$$

No entanto, se pré-multiplicarmos (40) por r_i, vem que:

$$\begin{aligned} r_i^T d_i &= r_i^T r_i + \beta_i r_i^T d_{i-1} \\ r_i^T d_i &= r_i^T r_i, \text{ pela Equação (33).} \end{aligned} \tag{42}$$

Com isso, podemos simplificar (41) e obter:

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}. \tag{43}$$

Por fim, os elementos apresentados até agora compõem o Algoritmo 2.

Algoritmo 2: Método de Gradientes Conjugados

Dados $x_0 \in \mathbb{R}^n$ o ponto inicial, $A \in \mathbb{R}^{n\times n}$ simétrica e definida positiva, $b \in \mathbb{R}^n$ e uma tolerância $\epsilon \in \mathbb{R}$ suficientemente pequena, então:

1 $d_0 = r_0 = b - Ax_0$;
2 **para** $i = 0, \ldots, n$ **faça**
3 $\quad \alpha_i = \frac{r_i^T r_i}{d_i^T A d_i}$;
4 $\quad x_{i+1} - x_i + \alpha_i d_i$;
5 $\quad r_{i+1} = r_i - \alpha_i A d_i$;
6 $\quad$ **se** $\|r_{i+1}\| \leq \epsilon$ **então retorna** x_{i+1};
7 $\quad \beta_{i+1} = \frac{r_{i+1}^T r_{i+1}}{r_i^T r_i}$;
8 $\quad d_{i+1} = r_{i+1} + \beta_{i+1} d_i$;
fim

3.1 Sobre a convergência do MGC

Um dos princípios do MGC é que ele resolve um problema de dimensão n em, no máximo, n iterações. Todavia, erros de aproximação e arredondamento podem afetar essa propriedade. Por isso é importante a análise de convergência, para mostrar que o algoritmo é capaz de convergir à solução do problema em todos os casos. Além disso, por se destinar a resolver

problemas de grande porte, seria desejável que pudesse convergir em menos que n iterações. Nesse sentido, uma análise que permita estimar o pior caso é interessante.

A base para análise de convergência deste método vem da forma particular que o espaço gerado pelas direções conjugadas possui. Em (29), apresentamos o espaço $\mathcal{D}_i$, formado pelas direções conjugadas, no qual o MGC encontra a melhor solução possível. Todavia, como as direções são construídas a partir dos resíduos, então $\mathcal{D}_i$ também pode ser expresso como:

$$\mathcal{D}_i = \text{span}\{r_0, r_1, \ldots, r_{i-1}\}.$$

Mas, pela relação (36), temos que o resíduo, na i-ésima iteração, é gerado pela combinação linear do resíduo da iteração anterior r_{i-1} e do vetor Ad_{i-1}. Como $d_{i-1} \in \mathcal{D}_i$, então temos que o novo subespaço $\mathcal{D}_{i+1}$ será formado pelo subespaço anterior $\mathcal{D}_i$ e o subespaço $A\mathcal{D}_i$. Portanto:

$$\begin{aligned} D_i &= \text{span}\{d_0, Ad_0, A^2d_0, \ldots, A^{i-1}d_0\} \\ &= \text{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{i-1}r_0\} \\ &= \text{span}\{Ae_0, A^2e_0, A^3e_0, \ldots, A^ie_0\}, \text{ pela Equação (7).} \end{aligned} \tag{44}$$

A partir deste ponto, consideremos que o iterando (8) é escrito da seguinte forma:

$$x_{i+1} = x_0 + P_i(A)r_0, \tag{45}$$

sendo que a seleção do conjunto de coeficientes para os polinômios P_i determina a sequência $\{x_k\}$ gerada. Subtraindo x^*, de (45), sabendo que (6) vale, vem que:

$$\begin{aligned} x_{i+1} - x^* &= x_0 - x^* + P_i(A)A(x^* - x_0) \\ e_{i+1} &= [I - AP_i(A)]e_0. \end{aligned} \tag{46}$$

Com isso, o quadrado da norma de energia do erro na iteração $i+1$ pode ser expresso como:

$$\begin{aligned} \|e_{i+1}\|_A^2 &= e_{i+1}^T A e_{i+1} \\ &= \{[I - AP_i(A)]e_0\}^T A\{[I - AP_i(A)]e_0\} \\ &= e_0^T A[I - AP_i(A)]^2 e_0. \end{aligned} \tag{47}$$

Deste ponto de vista, o problema agora pode ser colocado como encontrar o polinômio P_i, entre todos os de grau i, de forma que a norma da energia do erro seja minimizada. Se expandirmos o polinômio em (45), temos:

$$x_{i+1} = x_0 + \gamma_0 r_0 + \gamma_1 Ar_0 + \ldots + \gamma_i A^i r_0, \tag{48}$$

sendo que $\{\gamma_k\}_{k=0}^i$ são coeficientes de P_i. Tomando a relação (8) recorrentemente, o vetor x_{i+1} também pode ser escrito como:

$$x_{i+1} = x_0 + \alpha_0 d_0 + \alpha_1 d_1 + \ldots + \alpha_i d_i. \tag{49}$$

Da Seção 2.2, sabe-se que na i-ésima iteração, o iterando x_i é minimizador de q na variedade afim $x_0 + D_i$. Como as relações (48) e (49) são equivalentes, e pelos espaços

equivalentes expostos em (44), então conclui-se que o MGC encontra o polinômio minimizante de grau i na i-ésima iteração. De fato, também da Seção 2.2, temos que o MGC minimiza a norma do erro na variedade afim $e_0 + \mathcal{D}_i$. Com isso, vem que o MGC é tal que:

$$\|e_{i+1}\|_A^2 = \min_{P_i} e_0^T A[I - AP_i(A)]^2 e_0. \tag{50}$$

A relação explícita entre os coeficientes $\{\alpha_k\}_{k=0}^i$, gerados pelo método, e $\{\gamma_k\}_{k=0}^i$, coeficientes do polinômio P_i, existe. Entretanto é algo relativamente complexo e impraticável, visto que o MGC encontra soluções ótimas a cada iteração sem exigir armazenamento de muitas informações. Por isso, vamos analisar o efeito da aplicação destes polinômios ao erro inicial e_0 e estimar limitantes superiores para o erro cometido. De (50), sabe-se que:

$$\|e_{i+1}\|_A^2 \leq e_0^T A[I - AP_i(A)]^2 e_0. \tag{51}$$

Como A é uma matriz simétrica e positiva definida, então ela admite uma base ortonormal de autovetores $\{v_j\}_{j=0}^{n-1}$ com n autovalores associados $\{\lambda_j\}_{j=0}^{n-1}$. Expressando o erro inicial como uma combinação linear dos autovetores, ortonormais, de A, vem que:

$$e_0 = \sum_{j=0}^{n} \xi_j v_j. \tag{52}$$

De (52) em (51), vem que:

$$\begin{aligned}
\|e_{i+1}\|_A^2 &\leq \sum_{j=0}^{n} \xi v_j^T A[I - AP_i(A)]^2 \xi v_j \\
&\leq \sum_{j=0}^{n} \xi^2 \lambda_j v_j^T v_j - \sum_{j=0}^{n} \xi^2 \lambda_j^3 P_i^2(\lambda_j) v_j^T v_j \\
&\leq \sum_{j=0}^{n} [1 - \lambda_j P_i(\lambda_j)]^2 \xi^2 \lambda_j, \text{ pela ortonormalidade de } \{v_j\}.
\end{aligned} \tag{53}$$

Na análise de pior caso, de (53), segue que:

$$\|e_{i+1}\|_A^2 \leq \max_{\lambda_j} [1 - \lambda_j P_i(\lambda_j)]^2 \sum_{j=0}^{n} \xi^2 \lambda_j, \tag{54}$$

sendo que, se manipularmos (52) temos:

$$\begin{aligned}
\|e_0\|_A^2 &= e_0^T A e_0 \\
&= \sum_{j=0}^{n} \xi^2 v_j^T A v_j \\
&= \sum_{j=0}^{n} \xi^2 \lambda_j, \text{ pela ortonormalidade de } \{v_j\},
\end{aligned}$$

que, em (54), resulta em:

$$\|e_{i+1}\|_A^2 \leq \max_{\lambda_j}[1 - \lambda_j P_i(\lambda_j)]^2 \|e_0\|_A^2. \tag{55}$$

Esta análise encerra o estudo do MGC. Na próxima seção, vamos apresentar a integração deste método com regiões de confiança, suas principais propriedades e os resultados dos testes feitos.

4 MGC e a estratégia de região de confiança

Métodos de região de confiança são estratégias iterativas que resolvem um problema de minimização por sucessivas aproximações quadráticas em torno do ponto corrente x_c. Ao problema de minimizar essas aproximações quadráticas dentro de uma região definida dá-se o nome de *subproblema de região de confiança*.

Tais métodos são essencialmente compostos por um algoritmo externo e um algoritmo interno. O algoritmo externo controla o descrescimento da função objetivo com relação ao modelo quadrático e o tamanho da região de confiança, que corresponde ao controle do passo. O algoritmo interno, por sua vez, resolve o subproblema de região de confiança a cada iteração. O algoritmo externo é, em geral, pré-definido, e não varia muito entre as várias estratégias propostas. A grande vantagem de se utilizar métodos de região de confiança está no algoritmo interno, que é geralmente personalizado para que se adeque melhor ao domínio do problema em questão. Uma referência definitiva para a estratégia de região de confiança é encontrada em Conn, Gould e Toint [1].

Neste contexto, vamos apresentar nesta seção o MGC como uma estratégia de região de confiança, isto é, um algoritmo para resolução do subproblema de região de confiança.

4.1 O problema

Vamos retomar o problema definido em (1). Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in \mathcal{C}^2$, vamos tomar o problema geral de minimização irrestrita

$$\begin{aligned} &\min f(x) \\ &\text{s.a. } x \in \mathbb{R}^n \end{aligned} \tag{56}$$

e resolvê-lo usando a estratégia de região de confiança, de forma que, para o ponto corrente x_c, a cada iteração, define-se um modelo quadrático $q_c(p)$ e resolve-se o subproblema de região de confiança, reescrevendo (2) sem perda de generalidade:

$$\begin{aligned} \min q_c(p) \quad &= \frac{1}{2} p^T H_c p + g_c^T p \\ \text{s.a } \|p\| &\leq \Delta_c, \end{aligned} \tag{57}$$

onde Δ_c é o raio da região de confiança, $g_c \equiv \nabla f(x_c) \in \mathbb{R}^n$ é o gradiente de f avaliado em x_c e $H_c \equiv \nabla^2 f(x_c) \in \mathbb{R}^{n \times n}$ é a matriz Hessiana de f avaliada em x_c.

Desta maneira, (57) será resolvido usando o MGC e a solução global p^* encontrada, minimizador de q_c na bola de raio Δ_c, será o passo para atualizar $x_n = x_c + p^*$, desde que x_n realmente provoque descrescimento em f.

4.2 Solução do subproblema usando MGC

O breve desenvolvimento do MGC feito até aqui nos serve como base para expô-lo como um método para resolução do subproblema de região de confiança (57).

O uso do MGC na resolução do subproblema objetiva definir, a cada iteração:

$$p_{i+1} = p_i + \alpha_i d_i, \tag{58}$$

de forma que o conjunto das sequências $\{d_k\}$ são encontradas a partir da conjugação dos resíduos de q, a cada iteração, e o tamanho do passo é encontrado de acordo com a estratégia do MGC apresentada na Seção 3.

Sabe-se que o MGC resolve um sistema linear possível e determinado, portanto o método poderia ser usado sem nenhuma preocupação desde que a função q em (57) fosse estritamente convexa. De fato (57) define um problema quadrático, todavia não necessariamente estritamente convexo, isto é, H_c pode definir qualquer curvatura. Por isso, é necessário que o MGC seja alterado para que possa ser uma estatégia adequada à resolução do problema (57).

Essas alterações foram propostas por Steihaug [10] e consistem em modificar os critérios de parada do algoritmo do MGC, aproveitando as caratecterísticas que a região de confiança agrega ao problema, de forma que o algoritmo pare:

1. *Se a norma do resíduo for suficiente pequena, então temos uma solução interna.*

   Este critério é o mesmo do algoritmo original. Para um dado ϵ, suficientemente pequeno, define que, se, na i-ésima iteração:

   $$\|r_{i+1}\| \leq \epsilon,$$

   então a condição necessária (e suficiente, no caso do problema em questão, que é quadrático) de otimalidade de primeira ordem é satisfeita e p_{i+1} será a solução para (57).

2. *Se a norma do passo p for maior que a região de confiança, então temos uma solução na borda.*

   Esta condição é tradicional em estratégias de região de confiança, que é justamente seu intuito: definir uma vizinhança do ponto conrrente em que o modelo seja uma representação fiel da função objetivo. Desta forma, este critério define que, se, na i-ésima iteração:

   $$\|p_{i+1}\| > \Delta_c,$$

   então basta encontrar τ de forma que:

   $$\|p_i + \tau d_i\| = \Delta_c,$$

   e, então, $p^* = p_i + \tau d_i$ será solução de (57).

3. *Se houver uma direção de curvatura negativa, então temos uma solução na borda.*

   Essa é a condição que define a validade do uso do MGC com a estratégia de região de confiança. Sabe-se que, se H_c não for definida positiva, então não é possível utilizar o MGC para resolver o problema (57). Entretanto, com o uso de região de confiança,

se H_c não for definida positiva, então há uma direção de curvatura negativa para a matriz H_c. Ora, se o algoritmo encontrar uma direção de curvatura negativa d_i, então o melhor que podemos encontrar é um passo que atinja a borda da região de confiança. Portanto, se, na i-ésima iteração:

$$d_i^T H_c d_i \leq 0,$$

então basta encontrar τ de forma que:

$$\|p_i + \tau d_i\| = \Delta_c,$$

e, com isso, $p^* = p_i + \tau d_i$ será solução de (57).

O MGC modificado, contendo os aspectos apresentados nesta seção, é definido no Algoritmo 3.

Para concluir as ideias, vamos seguir Nocedal e Wright [8, Capítulo 7] e mostrar que o i-ésimo termo da sequência $\{p_k\}$ gerada pelo Algoritmo 3 possui norma estritamente maior que seu antecessor, isto é, da iteração $i-1$, para todo $i \geq 1$. Isto é consequência direta da inicialização da sequência tomando $p_0 = 0$, e sua importância está no fato de que haverá uma quantidade finita de termos, com norma estritamente crescente e superiormente cotada pelo tamanho da região de confiança. Além disso, este teorema mostra que o algoritmo irá aproveitar ao máximo o espaço definido pela região de confiança, isto é, caso não encontre nenhuma solução interna, atingirá a borda da região de confiança.

Teorema 2. *A sequência de vetores $\{p_k\}$ gerada pelo Algoritmo 3 é tal que:*

$$0 = \|p_0\| < \ldots < \|p_k\| < \|p_{k+1}\| < \ldots < \|p^*\| \leq \Delta_c. \tag{59}$$

Dem. Temos dois casos triviais, em que p^* é obtido de forma que $\|p^*\| = \Delta_c$, que são quando (i) d_k é uma direção de curvatura negativa ou nula ($d_k^T H_c d_k \leq 0$) ou (ii) quando o passo ultrapassa a região de confiança: $\|p_{k+1}\| > \Delta_c$. Para mostrar (59) para os demais casos, precisamos provar que, para $p_{k+1} = p_k + \alpha_k d_k$, vale

$$\|p_k\| < \|p_{k+1}\|, \quad \forall k \geq 0.$$

De (58), podemos escrever:

$$\|p_{k+1}\|^2 = (p_k + \alpha_k d_k)^T (p_k + \alpha_k d_k) = \|p_k\|^2 + 2\alpha_k p_k^T d_k + \alpha_k^2 \|d_k\|^2 \tag{60}$$

De fato, α_k é positivo (veja as expressões (12) e (42), que geraram a expressão de α_i no Algoritmo 3). Então, vamos provar por indução que $p_k^T d_k > 0$. Para $k = 1$, temos:

$$\begin{aligned} p_1^T d_1 &= (p_0 + \alpha_0 d_0)^T (r_1 + \beta_0 d_0), \text{ pela Equação (58)} \\ &= \alpha_0 d_0^T r_1 + \alpha_0 \beta_0 d_0^T d_0, \quad \text{pois } p_0 = 0 \\ &= \alpha_0 \beta_0 \|d_0\|^2 > 0, \quad \text{pela Equação (33)}. \end{aligned}$$

Agora vamos supor que $p_k^T d_k > 0$ vale. Por conseguinte:

$$\begin{aligned} p_{k+1}^T d_{k+1} &= p_{k+1}^T (r_{k+1} + \beta_k d_k) \\ &= p_{k+1}^T r_{k+1} + \beta_k p_{k+1}^T d_k \\ &= p_{k+1}^T r_{k+1} + \beta_k (p_k + \alpha_k d_k)^T d_k, \text{ pela Equação (58)} \\ &= p_{k+1}^T r_{k+1} + \beta_k p_k^T d_k + \alpha_k \|d_k\|^2. \end{aligned} \tag{61}$$

Algoritmo 3: Método de Gradientes Conjugados Modificado

Dados a hessiana $H_c \in \mathbb{R}^{n\times n}$ e o gradiente $g_c \in \mathbb{R}^n$ da função f definida em (56), avaliados em um ponto corrente x_c e uma tolerância $\epsilon \in \mathbb{R}^+$ suficientemente pequena, o problema (57) pode ser resolvido pelo algoritmo:

Defina $p_0 = 0$, $r_0 = -g_c$, $d_0 = r_0$ e $\delta_0 = r_0^T r_0$;
para $i = 0, \ldots, n$ **faça**
  $h_i = H_c d_i$;
  $\eta_i = d_i^T h_i$;
  se $\eta_i \leq 0$ **então**
    Encontre τ tal que $\|p_i + \tau d_i\| = \Delta_c$;
    retorna $p^* = p_i + \tau d_i$;
  fim
  $\alpha_i = \dfrac{\delta_0}{\eta_i}$;
  $p_{i+1} = p_i + \alpha_i d_i$;
  se $\|p_{i+1}\| > \Delta_c$ **então**
    Encontre τ tal que $\|p_i + \tau d_i\| = \Delta_c$;
    retorna $p^* = p_i + \tau d_i$;
  fim
  $r_{i+1} = r_i - \alpha_i h_i$;
  $\delta_1 = r_{i+1}^T r_{i+1}$;
  se $\sqrt{\delta_1} < \epsilon$ **então retorna** p_{i+1};
  $\beta_i = \dfrac{\delta_1}{\delta_0}$;
  $d_{i+1} = r_{i+1} + \beta_i d_i$;
  $\delta_0 = \delta_1$;
fim

Como α_k, β_k (veja (43)) e $p_k^T d_k$ (pela hipótese de indução) são positivos, basta ver que:

$$\begin{aligned} p_{k+1}^T r_{k+1} &= (p_0 + \sum_{i=0}^{k} \alpha_i d_i)^T r_{k+1}, \text{ tomando (58) recursivamente} \\ &= \sum_{i=0}^{k} \alpha_i d_i^T r_{k+1}, \text{ como } p_0 = 0 \\ &= 0, \text{ pela Equação (33)} \end{aligned}$$

para constatar que (61) também é positivo. Finalmente, podemos concluir, de (60), que:

$$\|p_k\| < \|p_{k+1}\|, \quad \forall k \geq 0.$$

■

O método de gradientes conjugados como um algoritmo para resolução do subproblema de região de confiança foi estabelecido. Para constatar seu comportamento na prática, vamos descrever e analisar, na próxima seção, experimentos numéricos usando as ideias do algoritmo apresentadas até aqui.

5 Experimentos numéricos

Uma implementação do MGC modificado foi feita usando o CAS *Maxima*, um sistema algébrico computacional livre[1]. Nesta seção, vamos apresentar alguns testes numéricos feitos com o algoritmo nesta implementação, considerando problemas gerais propostos por Moré, Garbow e Hillstrom em [7].

As funções que aparecem em [7] estão na forma de quadrados mínimos, isto é, temos $F(x) = (f_1(x) \dots f_m(x))^T$, de forma que $F : \mathbb{R}^n \to \mathbb{R}^m$. Para trabalhar com problemas de minimização irrestrita, vamos tomar nossa função objetivo:

$$f(x) = \frac{1}{2}\|F(x)\|^2 = \frac{1}{2}\sum_{i=1}^{m} f_i(x)^2 \tag{62}$$

e resolver

$$\begin{aligned} &\min f(x) \\ &\text{s.a. } x \in \mathbb{R}^n. \end{aligned}$$

Os testes estão divididos em três partes:

1. Na **Parte 1**, é apresentada uma visualização da convergência do algoritmo para dois problemas bidimensionais;

2. Na **Parte 2**, são apresentados resultados de uma série de problemas resolvidos usando o algoritmo estudado neste trabalho;

3. Na **Parte 3**, é feita uma comparação entre este método de minimização irrestrita e o método de Levenberg-Marquardt para quadrados mínimos propriamente dito, estudado em [3].

5.1 Parte 1

Nesta parte, temos por objetivo uma visualização gráfica do comportamento e convergência do algoritmo, partindo do ponto inicial e convergindo para a solução. Por isso, vamos abordar apenas dois problemas, apresentados na Tabela 1.

Nos testes, siglas mnemônicas são usadas para facilitar a referência ao nome da função, acompanhadas de suas respectivas dimensões m e n, bem como dos pontos iniciais utilizados e da numeração apresentada no artigo de Moré, Garbow e Hillstrom [7]. Os resultados da Função 1 (Rosenbrock) são ilustrados na Figura 1, enquanto os resultados da Função 5 (Beale) são ilustrados na Figura 2.

[1] Acesse http://maxima.sourceforge.net/

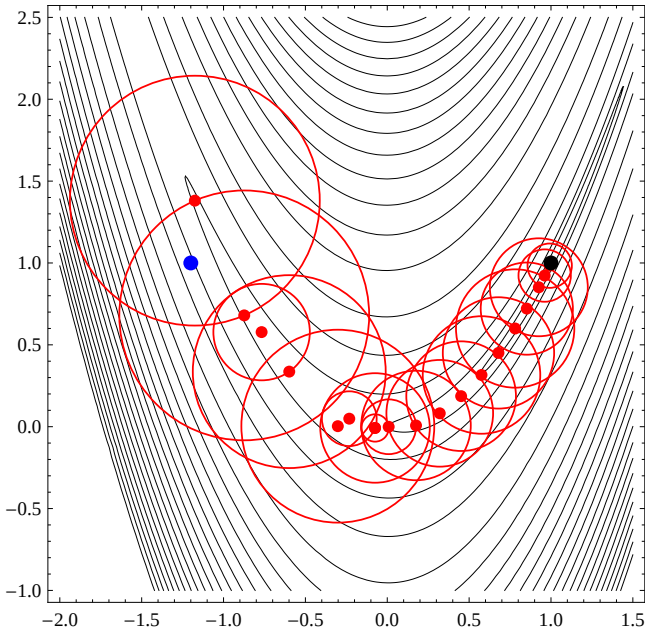


Figura 1: Comportamento da Função de Rosenbrock.

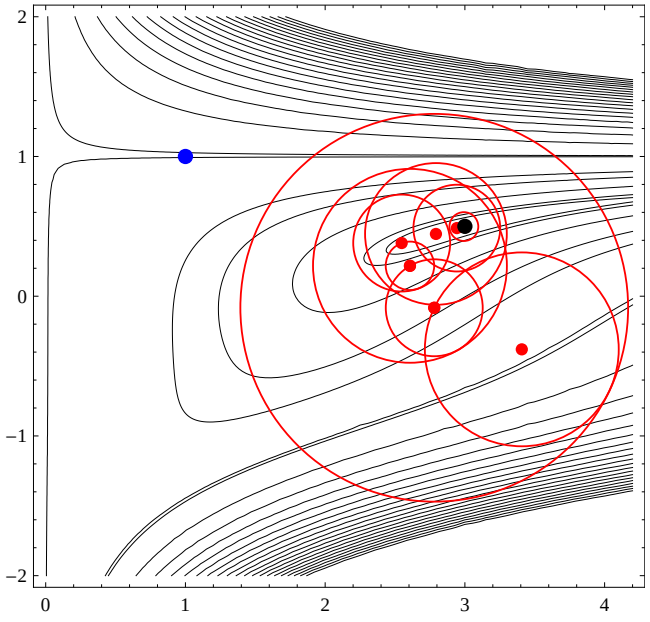


Figura 2: Comportamento da Função de Beale.

Função	Sigla	n	m	Ponto Inicial
1	ROS	2	2	$(-1.2, 1)$
5	BEA	2	3	$(1, 1)$

Tabela 1: Problemas testados na primeira parte.

5.2 Parte 2

Nesta parte, o objetivo é rodar uma bateria de funções, extraídas de Moré, Garbow e Hillstrom [7], de forma a testar a capacidade do algoritmo resolver diversos problemas de minimização irrestrita.

Para uma medida de desempenho, os critérios a serem analisados nesta parte são apresentados na Tabela 2.

Sigla	Descrição
AF	Quantidade de avaliações de função que o processo consumiu.
AG	Quantidade de avaliações do gradiente da função objetivo durante o processo.
AH	Total de avaliações da matriz Hessiana da função objetivo demandadas para a resolução do problema.
CP	Critério de parada do algoritmo.
ITEXT	Quantidade de iterações externas, isto é, do algoritmo principal.
ITINT	Quantidade de iterações internas, isto é, do MGC na resolução do subproblema.
TE	Tempo de execução (em segundos).
VF	Valor de $\|F(x^*)\|$, sendo $F(x)$ definida em (62) e x^* a solução encontrada.

Tabela 2: Descrição dos critérios de avaliação adotados.

Os critérios de parada são os seguintes:

1. *Excedeu a quantidade máxima de iterações*, identificado por -1.

   Neste critério, verifica-se se o número de iterações é maior que ou igual a uma dada tolerância `ITOL`. Nos testes, `ITOL` $= 300$.

2. *Otimalidade de primeira ordem: ponto estacionário da função*, identificado por 0.

   Neste critério, foi avaliada a norma relativa do gradiente da função, para uma dada tolerância `GTOL`, isto é:

   $$\frac{\|g_c\|}{\|g_0\|} \leq \texttt{GTOL}.$$

   Nos testes, `GTOL` $= 10^{-6}$.

3. *Houve pouca variação da norma da função $F(x)$*, identificado por 1.

   Neste critério, avaliamos a variação da norma da função F dada uma certa tolerância `VARTOL`, isto é, para um ponto corrente x_c e um novo x_n, avaliamos:

   $$\|F(x_c) - F(x_n)\| \leq \texttt{VARTOL}.$$

   Nos testes, `VARTOL` $= 10^{-12}$.

As funções testadas são apresentadas na Tabela 3, enquanto os resultados dos testes são expostos na Tabela 4.

Função	Sigla	n	m	Ponto Inicial
1	ROS	2	2	$(-1.2, 1)$
2	FRF	2	2	$(0.5, -2)$
6	JSF	2	10	$(0.3, 0.4)$
8	BARD	3	15	$(1, 1, 1)$
10	MEY	3	16	$(0.02, 4000, 250)$
12	BTD	3	10	$(0, 10, 20)$
13	POWS	4	4	$(3, -1, 0, 1)$
15	KOF	4	11	$(0.25, 0.39, 0.415, 0.39)$
16	BDF	4	20	$(25, 5, -5, -1)$
17	OS1	5	33	$(0.5, 1.5, -1, 0.01, 0.02)$
19	OS2	11	65	$(1.3, 0.65, 0.65, 0.7, 0.6, 3, 5, 7, 2, 4.5, 5.5)$
20	WAT	12	31	$(0, 0, \ldots, 0)$
27	BAL	10	10	$(0.5, 0.5, \ldots, 0.5)$
32	LPC	5	50	$(1, 1, \ldots, 1)$
33	LP1	5	50	$(1, 1, \ldots, 1)$
34	LP1Z	5	50	$(1, 1, \ldots, 1)$

Tabela 3: Problemas testados na segunda parte.

Função	CP	VF	ITEXT	ITINT	AF	AG	AH	TE
ROS	0	0.34654 E-08	22	43	23	22	22	0.02
FRF	0	0.69988 E 01	7	14	8	8	8	0
JSF	0	0.11151 E 02	8	16	9	9	9	0.01
BARD	0	0.90635 E-01	14	42	15	14	14	0.06
MEY	0	0.93779 E 01	216	826	217	199	199	1.67
BTD	0	0.92759 E-04	13	34	14	13	13	0.11
POWS	0	0.75374 E-03	12	48	13	13	13	0.02
KOF	0	0.17535 E-01	11	35	12	11	11	0.09
BDF	0	0.29295 E 03	7	33	8	8	8	0.08
OS1	0	0.73924 E-02	40	201	41	34	34	1.04
OS2	0	0.20034 E 00	22	179	23	19	19	9.66
WAT	0	0.22180 E-03	12	149	13	13	13	13.54
BAL	0	0.89312 E-05	6	13	7	7	7	0.06
LPC	0	0.67082 E 01	3	3	4	4	4	0.01
LP1	0	0.34826 E 01	2	1	3	2	2	0.14
LP1Z	0	0.36917 E 01	2	1	3	2	2	0.01

Tabela 4: Resultados dos testes da segunda parte.

Diante dos resultados, algumas observações são pertinentes:

1. O tempo de execução do algoritmo é apenas um valor simbólico, isto é, pode variar drasticamente com relação à máquina que executará o algoritmo, do sistema operacional e do uso do processador no momento da execução. Todavia, representa um bom parâmetro para ver como o algoritmo se comporta com relação às propriedades do problema. Podemos notar que os fatores mais influentes são, simultaneamente, dimensão do problema, quantidade de avaliações de função, gradiente e hessiana. Além disso, há fatores que não são tão explícitos, como a dificuldade que surge em alguns pontos do domínio para o programa realizar certos cálculos (como denominadores muito próximos de zero, gerando resulados muito grandes em valor absoluto) e erros de aproximação numérica e ponto flutuante.

2. O número de avaliações de gradientes e hessianas são sempre iguais, pois eles são avaliados juntos. Em contrapartida, diferem do valor de avaliações de função, em certos casos. Isso acontece porque a quantidade de avaliação de função é igual à quantidade de iterações externas mais um, devido ao fato de que a função é avaliada uma vez antes de começar a execução cíclica do algoritmo, enquanto que o gradiente e a hessiana são avaliados apenas quando o novo ponto provocou descréscimo satisfatório na função objetivo. Portanto, a diferença corresponde à quantidade de vezes que o ponto obtido foi descartado, isto é, não causou descréscimo na função objetivo e não foi aceito.

3. Também é interessante notar a relação entre a quantidade de iterações internas e externas e a dimensão do problema, isto é, o valor de n do problema na Tabela 3. Na grande maioria dos casos, a relação é:

$$\frac{\text{ITINT}}{\text{ITEXT}} \leq n.$$

   Isto deve-se ao fato da propriedade teórica de que o algoritmo de gradientes conjugados resolve um problema de dimensão n em, no máximo, n iterações. A única anormalidade foi no problema `MEY`. Isto deve-se a fatores não explícitos, como expostos no Item 1 acima. Todavia, a relação não é tão anômala, sendo que o valor não é muito diferente de n, que, no caso, vale 3.

4. Todos os problemas pararam com um critério satisfatório, que é a condição de otimalidade de primera ordem. Todavia, esse comportamento pode variar se mudar o valor da tolerância `GTOL`, tendo alguns parado por falta de progresso (`CP` $= 1$). Mesmo nesses casos, nota-se que o valor do gradiente no ponto está bem próximo da tolerância, e que valores mais precisos não são alcançados por questões da precisão dos cálculos na aritmética de ponto flutuante.

5.3 Parte 3

Nesta parte, o objetivo é comparar os resultados dos testes do algoritmo desenvolvido neste trabalho com o algoritmo de Levenberg-Marquardt para o problema de quadrados mínimos (LMA) [3]. A comparação será feita em duas partes: uma tabular, expondo os valores finais das funções e a quantidade de avaliações e outra gráfica, na qual exibimos alguns perfis de desempenho [2].

O quadro comparativo é apresentado na Tabela 5, enquanto os perfis de desempenho são expostos na Figura 3.

Função	MGC					LMA				
	VF	AF	AH	ITEXT	ITINT	VF	AF	AJ	ITEXT	ITINT
ROS	0.34654 E-08	23	22	22	43	0.0	19	14	18	51
FRF	0.69988 E 01	8	8	7	14	0.69988 E 01	41	28	40	95
JSF	0.11151 E 02	9	9	8	16	0.11151 E 02	20	9	19	49
BARD	0.90635 E-01	15	14	14	42	0.90635 E-01	6	6	5	5
MEY	0.93779 E 01	217	199	216	826	0.93779 E 01	144	124	143	299
BTD	0.92759 E-04	14	13	13	34	0.62754 E-10	13	12	12	34
POWS	0.75374 E-03	13	13	12	48	0.19361 E-03	9	9	8	8
KOF	0.17535 E-01	12	11	11	35	0.17536 E-01	13	11	12	43
BDF	0.29295 E 03	8	8	7	33	0.29295 E 03	41	24	40	127
OS1	0.73924 E-02	41	34	40	201	0.73924 E-02	19	16	18	35
OS2	0.20034 E 00	23	19	22	179	0.20034 E 00	15	14	14	38
WAT	0.22180 E-03	13	13	12	149	0.38213 E-04	5	5	4	6
BAL	0.89312 E-05	7	7	6	13	0.19146 E-09	15	11	14	23
LPC	0.67082 E 01	4	4	3	3	0.67082 E 01	5	5	4	7
LP1	0.34826 E 01	3	2	2	1	0.34826 E 01	3	3	2	6
LP1Z	0.36917 E 01	3	2	2	1	0.36917 E 01	2	2	1	2

Tabela 5: Quadro comparativo com os resultados da rotina LMA [3].

A análise de desempenho permite as seguintes considerações:

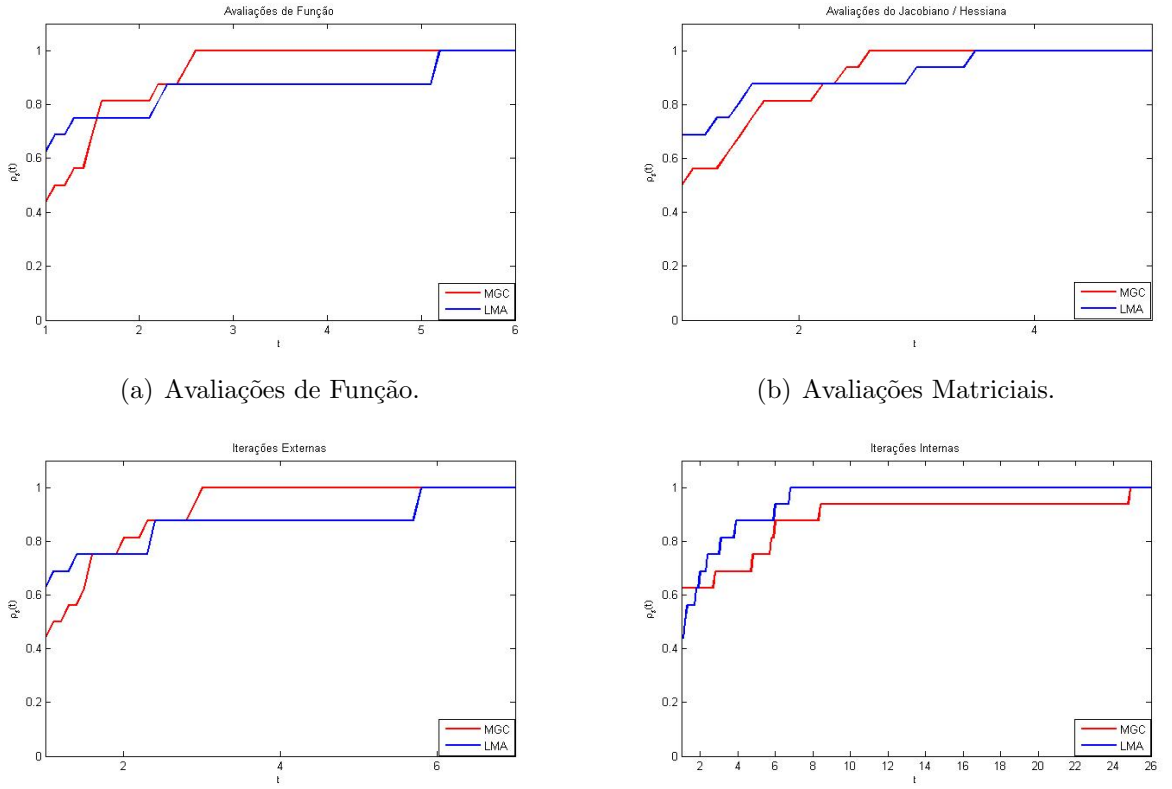
1. Consideramos avaliações de Jacobiano para LMA e Hessiana para MGC pois o algoritmo de LMA é uma estratégia de primeira ordem que trata o problema do ponto de vista matricial, aproveitando sua estrutura de quadrados mínimos, enquanto o MGC é um método de segunda ordem que encara o problema como um caso de minimização irrestrita. Estas medidas equivalem de duas maneiras: ambas se tratam de avaliações matriciais, isto é, requerem o mesmo espaço de armazenamento, e são calculadas apenas quando o novo iterando produz descrescimento suficiente na função objetivo.

2. Podemos notar que o algoritmo de LMA é mais eficiente em três casos, que são avaliações de função, de jacobiano e Hessiana e na quantidade de iterações internas. O algoritmo de MGC é mais eficiente apenas na quantidade de iterações internas dispendidas.

3. Do ponto de vista de robustez, o LMA se sobressai na quantidade de iterações internas, enquanto o algoritmo de MGC possui robustez acentuada no que diz respeito às avaliações, tanto de função quanto de Hessiana, e nas iterações externas. Isto se deve ao fato de que o algoritmo de LMA trabalha mais a cada iteração interna, isto é, aproveita melhor a estrutura do problema, resolvendo o problema em menos iterações, entretanto consumindo muito mais avaliações.

6 Conclusões

O algoritmo de gradientes conjugados é um método de direções conjugadas proposto originalmente para resolução de sistemas lineares compatíveis e determinados com matrizes



(a) Avaliações de Função.

(b) Avaliações Matriciais.

(c) Iterações Externas.

(d) Iterações Internas.

Figura 3: Perfis de desempenho dos testes da terceira parte.

de coeficientes positiva definida. Entretanto, aliado à estratégia de regiões de confiança, tornou-se uma ferramenta interessante para a resolução de problemas gerais de minimização irrestrita.

Os experimentos numéricos realizados revelaram um comportamento satisfatório do algoritmo de MGC proposto, que, implementado em uma sistema álgebrico computacional livre, o Maxima, foi capaz de chegar à solução dos problemas testados (cf. valores ótimos apresentados em [7]). Além disso, uma comparação ao LMA foi feita, com o objetivo de observar o desempenho das duas abordagens distintas para a resolução do problema de quadrados mínimos: enquanto o algoritmo LMA explora as peculiaridades e tira proveito da estrutura do problema, o MGC resolve um problema geral de minimização irrestrita. Diante disto, percebemos que o MGC é um método competitivo que apresenta robustez na maior parte dos quesitos analisados.

Referências

[1] A.R. Conn, N.I.M. Gould & P.L. Toint, *Trust-Region methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[2] E.D. Dolan & J.J. Moré, "Benchmarking optimization software with performance profiles", *Mathematical Programming*. Springer Berlin / Heidelberg, 2002. Vol. 91, pp. 201-213, jan. 2002.

[3] J.L.C. Gardenghi & S.A. Santos, "Sistemas não lineares via região de confiança: o algoritmo de Levenberg-Marquardt". Disponível em: http://www.ime.unicamp.br/rel_pesq/2011/pdf/rp03-11.pdf. Último acesso em 28 de abr. 2012.

[4] M.R. Hestenes & E. Stiefel. "Methods of Conjugate Gradients for Solving Linear Systems". *Journal of Research of the National Bureau of Standards*, v. 49, n. 6, pp. 409-436, dez. 1952.

[5] A. Izmailov & M. Solodov. *Otimização, Volume 2*: Métodos Computacionais. Rio de Janeiro: IMPA, 2007. 448 p.

[6] C.T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Philadelphia: SIAM, 1995. 166 p.

[7] J.J. Moré, B.S. Garbow & K.E. Hillstrom, "Testing unconstrained optimization software", *ACM Transactions on Mathematical Software*, Volume 7, pp. 17-41, 1981.

[8] J. Nocedal & S.J. Wright, *Numerical Optimization*. 2. ed. New York: Springer, 2006. 664 p.

[9] J.R. Shewchuk. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Edition $1\frac{1}{4}$. 1994. School of Computer Science. Carnegie Mellon University, Pittsburgh, PA, 58p. Disponível em http://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf. Último acesso em 28 de abr. 2012.

[10] T. Steihaug. "The conjugate gradient method and trust region in large scale optimization". *SIAM Journal on Numerical Analysis*, v. 20(3), pp. 626-637, 1983.

[11] J. Tenne & S.W. Armfield. "A note on the relation of Gaussian elimination to the conjugate directions algorithm", em: R. May & A. J. Roberts, *Proc. of 12th Computational Techniques and Applications Conference CTAC-2004*, Volume 46, pp. C971-C986, 2005.