UM MÉTODO ITERATIVO PARA MINIMIZAÇÃO DE QUADRÁTICAS EM CAIXAS

Lucas Garcia Pedroso Maria Aparecida Diniz–Ehrhardt

DMA – IMECC – UNICAMP

Resumo

Neste trabalho trataremos de uma variação do método dos gradientes conjugados para minimização de quadráticas gerais em caixas. Esta abordagem é a base do método implementado em QUACAN, um software desenvolvido por A. Friedlander, J.M. Martinez e S.A. Santos, do DMA – IMECC (UNICAMP). Nosso objetivo é a inserção de precondicionadores do tipo diagonal ao método, na tentativa de acelerar sua convergência. Experimentos numéricos seguirão a devida análise teórica do assunto.

Abstract

In this work we focus our attention on a variation of the conjugate gradient method for bound–constrained quadratic minimization. This approach is implemented in the subroutine QUACAN, a software developed by A. Friedlander, J.M. Martinez and S.A. Santos, from DMA – IMECC (UNICAMP). Our aim is to insert preconditioners to the method, trying to accelerate its convergence. Numerical experiments are presented.

1 Introdução

A minimização de quadráticas é, sem dúvida, essencial para se resolver um enorme conjunto de problemas em otimização. Isto se deve ao fato de que as quadráticas são de simples manipulação, de modo que são freqüentemente utilizadas em subproblemas auxiliares de algoritmos para resolver problemas mais complexos.

O QUACAN, desenvolvido no DMA - IMECC (UNICAMP), é um algoritmo que faz uso do método dos gradientes conjugados para resolver problemas de minimização de quadráticas com restrições de desigualdade do tipo caixa, ou seja, problemas do tipo

$$\begin{array}{c} \text{Minimizar } q(x) \\ \text{s.a. } l \leq x \leq u \end{array}$$

com $q : I\!R^n \to I\!R$ função quadrática e l e u vetores n-dimensionais de componentes finitas.

No caso em que a função $q(x)$ tem Hessiana mal condicionada a convergência do QUACAN pode ser lenta, pois assim é a convergência do método dos gradientes conjugados nesta situação. Nosso intuito é utilizar o conceito de precondicionamento no algoritmo QUACAN e tentar melhorar sua convergência para tais problemas.

Após feitas as devidas modificações no algoritmo, realizamos experimentos numéricos tais como estes foram propostos em [10], onde é sugerida uma maneira de se gerar funções quadráticas aleatórias que tenham, entre outros fatores, um número de condição escolhido pelo usuário.

2 Método dos Gradientes Conjugados

Como problema inicial, estudaremos a minimização de quadráticas sem restrições, ou seja, o problema:

$$\text{Minimizar } q(x) = \tfrac{1}{2} x^t A x - b^t x + c,$$

com $A \in I\!R^{nxn}$ simétrica, $b, c, x \in I\!R^n$. Calculando o gradiente e a Hessiana do problema, temos [4, 8]:

$$\nabla q(x) = Ax - b, \nabla^2 q(x) = A.$$

Desta forma, se a matriz A for definida positiva, temos a garantia que a única solução x^* do problema de minimização é a solução do sistema linear

$$Ax^* = b,$$

pois, nesse caso, o ponto que anula o gradiente de $q(x)$ é necessariamente o minimizador da quadrática [4, 8]. Para encontrarmos a solução x^* deste sistema, podemos usar métodos diretos e métodos iterativos. Métodos diretos são processos através dos quais encontramos a solução exata para o problema $Ax = b$. O mais comum é o método do escalonamento de Gauss [6]. Já métodos iterativos são aqueles em que, dada uma aproximação inicial, em cada passo do processo encontramos aproximações melhores para a solução. Os mais simples são o método da máxima descida [1] e o método dos gradientes conjugados, alvo desta seção.

Os métodos de máxima descida e dos gradientes conjugados baseiam-se em direções de descida para a função objetivo. Uma direção de descida a partir de um ponto x é definida por um vetor d para o qual existe um $\bar{\alpha} > 0$ tal que $q(x + \alpha d) < q(x)$, $\forall \alpha \in (0, \bar{\alpha}]$ [4].

No método dos gradientes conjugados usamos como direção de descida da i-ésima iteração vetores que sejam A-conjugados em relação às direções usadas nas iterações anteriores, ou seja,

$$d_i^t A d_j = 0, \forall j = 0, 1, \ldots, i-1.$$

Dado um chute inicial x_0, em cada passo do método é feita uma atualização para a aproximação da solução da forma $x_{k+1} = x_k + \tau_k d_k$. Para calcular o passo τ_k, basta minimizar a quadrática na direção d_k, e é fácil obter $\tau_k = -d_k^t g(x_k)/d_k^t A d^k$, onde empregamos a notação $g(x_k) = \nabla q(x_k)$ [4]. Utilizando adequado processo para a atualização de d_k (de modo a criar direções A-conjugadas) e do gradiente g_k da função no ponto x_k (sem ter que calcular $g_k = Ax_k - b$), obtemos o seguinte algoritmo [9, 1]:

Dado $x_0 \in \mathbb{R}^n$, seja $g_0 = Ax_0 - b$ e $d_0 = -g_0$. Para $k = 0, 1, 2, \ldots$

$\tau_k = g_k^t g_k / d_k^t A d_k$
$x_{k+1} = x_k + \tau_k d_k$
$g_{k+1} = g_k + \tau_k A d_k$
$\beta_k = g_{k+1}^t g_{k+1} / g_k^t g_k$
$d_{k+1} = -g_{k+1} + \beta_k d_k$

Em cada iteração do algoritmo acima, o ponto x_k encontrado é o mínimo da quadrática restrita à variedade linear $x_0 + K_k$, onde K_k é o conjunto dado por

$$K_k = \text{span}\{g_0, Ag_0, \ldots, A^{k-1} g_0\},$$

conhecido como k-ésimo subespaço de Krylov da matriz A. É importante observar que a dimensão de tal subespaço é dada pelo número de autovalores distintos da matriz A, o que indica que o método dos gradientes conjugados encontra a solução exata para o problema

de minimização em, no máximo, p iterações, onde p é o número de autovalores distintos de A [9, 7].

Apesar deste resultado, o método dos gradientes conjugados pode se tornar extremamente lento para problemas de grande porte em duas situações: quando os autovalores distintos de A são muitos e quando o número de condição da matriz é grande [1]. Nestas situações, que são extremamente comuns em problemas reais advindos de aplicações em diversas áreas, é comum a construção de um *problema precondicionado*. Em tais problemas, dada uma matriz $C = EE^t$ de fácil manipulação, fazemos $y = E^t x$ e encontramos o problema

$$\tilde{q}(y) = \frac{1}{2} y^t E^{-1} A E^{-t} y - b^t E{-t} * y + c.$$

Seria interessante que a matriz $\tilde{A} = E^{-1} A E^{-t}$ fosse de alguma maneira próxima da matriz identidade. Como $\tilde{A}$ é semelhante à matriz $C^{-1}A$, uma vez que

$$E^{-t} \tilde{A} E^t = E^{-t} E^{-1} A E^{-t} E^t = (EE^t)^{-1} A (EE^{-1})^t = C^{-1} A,$$

concluímos que $\tilde{A}$ tem os mesmos autovalores de $C^{-1}A$. Se esta matriz tiver menos autovalores distintos que A e/ou número de condição menor, o problema acima pode convergir mais rapidamente pelo método dos gradientes conjugados que o problema original. Fazendo uso destas idéias, podemos construir um algoritmo de gradientes conjugados precondicionado, que possivelmente encontrará a solução do problema original em menos iterações:

Dado $x_0 \in I\!R^n$, sejam $g_0 = Ax_0 - b$, $h_0 = C^{-1} g_0$ e $d_0 = -h_0$. Para $k = 0, 1, 2, \ldots$

$\tau_k = g_k^t h_k / d_k^t A d_k$
$x_{k+1} = x_k + \tau_k d_k$
$g_{k+1} = g_k + \tau_k A d_k$
$h_{k+1} = C^{-1} g_{k+1}$
$\beta_k = g_{k+1}^t h_{k+1} / g_k^t h_k$
$d_{k+1} = -h_{k+1} + \beta_k d_k$

O produto por C^{-1} no algoritmo acima deve ser interpretado como a resolução de um sistema linear com matriz dos coeficientes igual a C. Para ser um bom precondicionador do problema, a matriz C deve ser suficientemente próxima de A, pois assim $C^{-1}A$ é próxima da matriz identidade. É também desejável que C seja tal que a resolução dos sistemas lineares em cada iteração do algoritmo anterior seja muito simples.

3 Minimização de Quadráticas em Caixas

De volta ao problema:

$$\begin{array}{c} \text{Minimizar } q(x) = \frac{1}{2}x^t Ax - b^t x + c \\ \text{s.a.} \quad l \leq x \leq u \end{array} \tag{1}$$

com $l, u, x, c, b \in I\!R^n$ e $A \in I\!R^{nxn}$.

Restrições de desigualdades lineares implicam em algumas alterações nos algoritmos de minimização, e, geralmente, no uso dos multiplicadores de Lagrange, uma vez que as condições de otimalidade envolvem também a condição do minimizador ser factível, ou seja, pertencer ao conjunto $\Omega = \{x \in R^n \mid l \leq x \leq u\}$ [4, 8]. No entanto, como neste caso as restrições são do tipo caixa, é mais conveniente simplesmente verificar, a cada iteração, se o passo ótimo em uma determinada direção não ultrapassará os limites da caixa.

Mais adiante, nos será útil a notação $P[x, S]$ para a projeção ortogonal de x no conjunto S.

Dado um vetor $x \in \Omega$, dizemos que a restrição i está ativa se $x_i = l_i$ ou $x_i = u_i$. Desta forma, montamos o subconjunto $I(x)$ de $\{1, 2, \ldots, 2n\}$, de maneira que

$$i \in I(x) \Leftrightarrow x_i = l_i, n + i \in I(x) \Leftrightarrow x_i = u_i.$$

De posse do conjunto $I(x)$, definimos uma face aberta F_I como

$$F_I = \{x \in [l, u] \mid x_i = l_i \text{ se } i \in I,\ x_i = u_i \text{ se } n + 1 \in I,\ l_i < x_i < u_i \text{ nos outros casos}\}$$

Definimos também os vetores gradiente projetado negativo $g_p(x)$, gradiente interno negativo $g_i(x)$ e gradiente chopado negativo $g_c(x)$ como:

$$g_p(x)_i = \begin{cases} 0 & \text{se } x_i = l_i \text{ e } [\nabla q(x)\_i] > 0 \\ 0 & \text{se } x_i = u_i \text{ e } [\nabla q(x)\_i] < 0 \\ -[\nabla q(x)]_i & \text{nos outros casos} \end{cases}$$

$$g_i(x)_i = \begin{cases} 0 & \text{se } i \in I \text{ ou } n + i \in I \\ -[\nabla q(x)]_i & \text{nos outros casos} \end{cases}$$

$$g_c(x)_i = \begin{cases} 0 & \text{se } i \notin I \text{ e } n + i \notin I \\ 0 & \text{se } i \in I \text{ e } [\nabla q(x)]_i > 0 \\ 0 & \text{se } n + i \in I \text{ e } [\nabla q(x)]_i \\ -[\nabla q(x)]_i & \text{nos outros casos} \end{cases}$$

É fácil ver que os vetores $g_i(x)$ e $g_c(x)$ são ortogonais e que $g_p(x) = g_i(x) + g_c(x)$. O algoritmo QUACAN faz uso destes vetores para encontrar a solução do problema (1), minimizando parcialmente a quadrática nas diferentes faces visitadas [3, 9, 5]. Durante o processo de minimização, quando é interessante abandonar uma face, o algoritmo utiliza a direção do gradiente chopado. Por outro lado, nas situações em que é conveniente explorar mais uma face antes de abandoná-la, um algoritmo interno faz uso de direções de gradientes conjugados apenas com as componentes livres de x (usando como direção inicial $g_i(x)$), como podemos ver no algoritmo abaixo:

Seja $\eta \in (0,1)$ independente de k, dado um $l \leq x_0 \leq u$ e o conjunto $I(x_0)$. Começando com $d = 0$ e $\nu = 0$, os passos do algoritmo são:

1. (Critério de parada) Se $\|g_p(x_k)\| = 0$, parar.

2. (Testar se a face atual deve ou não ser deixada) Se

$$\|g_i(x_k)\| > \eta \|g_p(x_k)\|$$

   então $d_\nu = g_c(x_k)$, ir para o passo 4.

3. (A direção d_ν é obtida por gradientes conjugados)
   Se $\nu = 0$ então
   $d_\nu = g_i(x_k)$,
   caso contrário
   obter β pelo algoritmo dos gradientes conjugados, $d_\nu = g_i(x_k) + \beta d_{\nu-1}$.

4. (Busca no caminho $P[x_k + \lambda d_\nu, \Omega]$, $\lambda \geq 0$)
   Obter $\bar{\lambda}$ tal que $q(P[x_k + \bar{\lambda} d_\nu, \Omega] < q(x_k)$.

5. (Preparar para a próxima iteração)
   $x_{k+1} = P[x_k + \bar{\lambda} d_\nu, \Omega]$
   Se $I = I(x_{k+1})$
   $\nu = 0$
   Caso contrário $\nu = \nu + 1$.

6. (Atualizações) Fazer $k = k + 1$, atualizar $g_p(x_k)$, $g_i(x_k)$, $g_c(x_k)$ e voltar para o passo 1.

É no algoritmo interno do passo 3 que nos interessa implementar precondicionadores, como serão descritos na próxima seção. Para detalhes sobre a convergência do algoritmo acima, ver [5].

4 Experimentos Numéricos

Para realizar experimentos numéricos, fizemos uso dos problemas testes propostos em [10]. Neste trabalho, os autores sugerem uma maneira de gerar funções quadráticas $q(x) = \frac{1}{2}x^tAx - b^tx$, limites das restrições l e u, ponto inicial x_0 e solução x^*, tais que o usuário possa controlar as seguintes características do problema:

- número de condição de A,
- número de elementos de $I(x_0)$,
- número de elementos de $I(x^*)$,
- degenerecência de x^*,
- número de autovalores negativos de A.

O número de condição da matriz A é a propriedade que nos interessa. Esta é dada pelo parâmetro *ncond* do algoritmo que gera tais problemas. Temos que as matrizes A geradas satisfazem

$$\text{cond}(A) = e^{ncond}.$$

Os testes foram programados em FORTRAN 77 e rodados em microcomputador PC. Dois precondicionadores diagonais foram adaptados ao algoritmo QUACAN. O primeiro, conforme sugerido em [2], é dado por:

$$C_{ii} = \max\{\varepsilon, A_{ii}\}$$

com $\varepsilon = 10^{-15}$. O segundo precondicionador foi elaborado pelos autores e é da forma:

$$\hat{C}_{ii} = \max\{1, |Aii|\}.$$

A diferença principal entre os precondicionadores acima é que C substitui um elemento negativo da diagonal de A (por maior que seja seu módulo) por um valor ε pequeno, ao passo que $\hat{C}$ leva em consideração o valor absoluto de tais elementos.

Do grande número de experimentos realizados, selecionamos alguns da seguinte maneira: cada conjunto de testes representa uma dimensão n do problema ($n = 500, 1000$ e 2000). Dois parâmetros variaram em cada teste, o valor de *ncond* e de *negeig*, que representa (aproximadamente) o número de autovalores negativos de A. Escolhemos ncond= $4, 8$ e 10 e fizemos ora negeig = 0, ora negeig não nulo. Os demais parâmetros do algoritmo que gera problemas aleatórios, por não serem de interesse para a análise que almejamos fazer, foram mantidos constantes. Seguem as tabelas com o número de iterações que o QUACAN realizou em cada caso (Nas tabelas abaixo, QUACAN se refere

ncond	negeig	QUACAN	C	$\hat{C}$
4	0	63	16	16
4	200	68	16	16
8	0	422	23	23
8	200	92	13	14
10	0	1030	33	34
10	200	122	15	17

Tabela 1: n = 500

ncond	negeig	QUACAN	C	$\hat{C}$
4	0	66	24	19
4	400	71	12	17
8	0	444	51	51
8	400	146	15	17
10	0	1194	51	51
10	400	154	13	14

Tabela 2: n =1000

ncond	negeig	QUACAN	C	$\hat{C}$
4	0	64	10	10
4	700	70	17	15
8	0	474	75	63
8	700	297	12	15
10	0	1300	10	10
10	700	457	14	17

Tabela 3: n = 2000

ao algoritmo original sem precondicionadores):

As figuras 1 e 2 mostram o número de iterações do QUACAN variando em função de ncond. Os desempenhos do QUACAN não-precondicionado, precondicionador C e precondicionador $\hat{C}$ são indicados, respectivamente, pelas linhas tracejada, pontilhada e contínua. O valor de n usado foi 2500.

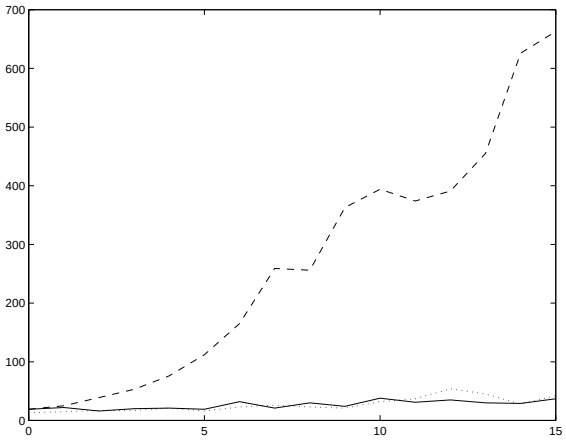

Figura 1: Desempenho do QUACAN, C e $\hat{C}$

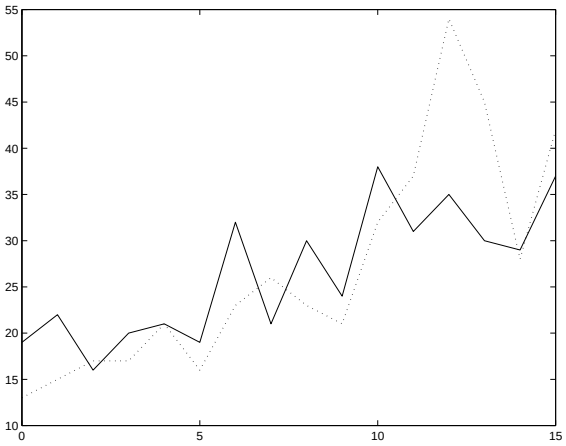

Figura 2: Desempenho de C e $\hat{C}$

5 Conclusões

Salientamos o bom desempenho dos precondicionadores estudados. Ambos mostraram–se excelentes, especialmente quando aplicados a problemas muito mal condicionados. Quando fazemos uma comparação emtre eles, C mostrou-se melhor em 7 problemas, enquanto $\hat{C}$ realizou um número menor de iterações em 4 dos problemas testados. Mas devemos ressaltar a superioridade de $\hat{C}$ no primeiro teste da tabela 2 e, especialmente, no terceiro teste da tabela 3.

Precondicionadores mais sofisticados, como o de banda, poderiam ser ainda mais eficientes em termos de redução no número de iterações de QUACAN, mas exigiriam um maior esforço computacional por iteração.

Este trabalho proporcionou um estudo aprofundado de vários tópicos relativos aos algoritmos de minimização, com enfoque no caso especial de minimização de quadráticas gerais em caixa. Para unir assuntos tão diversos foi necessário estudá-los profundamente em separado, o que foi certamente proveitoso para a percepção do espírito da pesquisa em otimização.

Do ponto de vista computacional, o trabalho trouxe resultados importantes para o uso do software QUACAN. O emprego de precondicionadores diagonais a este algoritmo e os bons resultados a partir disso estimulam a pesquisa de precondicionadores mais sofisticados que possam melhorar ainda mais o desempenho do QUACAN.

Referências

[1] Axelsson, O. & Barker, V.A., *Finite Element Solution of Boundary Value Problems*, Academic Press, 1984

[2] Conn, A.R. ; Gould, N.I.M. & Toint, Ph.L., *LANCELOT – A Fortran Package for Large-scale Nonlinear Optimization (Release A)*, Springer Verlag, Berlin, Heidelberg, New York, 1992.

[3] Diniz-Ehrhardt, M.A. ; Gomes-Ruggiero, M.A. & Santos, S.A., Numerical Analysis of Leaving-face Parameters in Bound-Constraines Quadratic Minimization, Optimization Methods and Software, 15, 45 – 66, 2001.

[4] Friedlander, A., *Elementos de Programação Não-Linear*, Editora da UNICAMP, 1994.

[5] Friedlander, A. ; Martínez, J.M. & Santos, S.A., A new trust region algorithm for bound constrained minimization, *Applied Mathematics and Optimization*, 30, 235 – 266, 1994.

[6] Gomes–Ruggiero, M.A. & Lopes, V.L.R., *Cálculo Numérico – Aspectos Teóricos e Computacionais*, 2ª edição, Makron Books, 1997.

[7] Kelley, C.T., *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, 1995.

[8] Luenberger, D.G., *Linear and Nonlinear Programming*, 2ª edição, Nova York, Addison – Wesley Publishing Company, 1986.

[9] Martínez, J.M. & Santos, S.A., *Métodos Computacionais de Otimização*, DMA – IMECC – UNICAMP, 1995.

[10] Moré, J.J. & Toraldo, G, Algorithms for bound constrained quadratic programming problems, *Numerische Mathematik*, 55, 377 – 400, 1989.