Universidade Federal de Santa Catarina
Curso de Pós-Graduação em Matemática e Computação Científica

O Método L-BFGS com Fatoração Incompleta para a Resolução de Problemas de Minimização

Melissa Weber Mendonça
Orientador: Prof. Dr. Mario César Zambaldi

Florianópolis
Fevereiro de 2005

Universidade Federal de Santa Catarina
Curso de Pós-Graduação em Matemática e Computação Científica

O Método L-BFGS com Fatoração Incompleta para a Resolução de Problemas de Minimização

Dissertação apresentada ao Curso de Pós-Graduação em Matemática e Computação Científica, do Centro de Ciências Físicas e Matemáticas da Universidade Federal de Santa Catarina, para a obtenção do grau de Mestre em Matemática, com Área de Concentração Honors Magister em Matemática Aplicada.

Melissa Weber Mendonça
Florianópolis
Fevereiro de 2005

O Método L-BFGS com Fatoração Incompleta para a Resolução de Problemas de Minimização

por

Melissa Weber Mendonça

Esta Dissertação foi julgada para a obtenção do Título de "Mestre", Área de Concentração em Matemática Aplicada, e aprovada em sua forma final pelo Curso de Pós-Graduação em Matemática e Computação Científica.

Prof. Dr. Igor Mozolevski
Coordenador

Comissão Examinadora

Prof. Dr. Mario César Zambaldi (MTM-UFSC-Orientador)

Prof. Dr. Antonio Carlos Moretti (MTM-UNICAMP)

Prof. Dr. Lício Hernanes Bezerra (MTM-UFSC)

Florianópolis, fevereiro de 2005.

Aos meus pais,
aos amigos que estiveram ao meu lado
e a Rodrigo.
Agradeço o apoio financeiro da CAPES.

Sumário

Lista de Figuras

Resumo

Neste trabalho, estudamos a resolução de problemas de minimização irrestrita por métodos quasenewtonianos, em particular o método BFGS, proposto na década de 60 por Broyden, Fletcher, Goldfarb e Shanno, bem como sua generalização para problemas de grande porte, o chamado método L-BFGS, proposto por Nocedal na década de 80. Apresentamos os resultados clássicos de convergência de ambos os métodos. No método L-BFGS, a matriz de recomeço utilizada é de grande importância na determinação da convergência do método. Neste sentido, propomos uma nova matriz de recomeço, utilizando a técnica de fatoração de Cholesky incompleta para matrizes simétricas positivas definidas, e situamos a fatoração incompleta dentro de seu contexto histórico como precondicionador para a resolução de sistemas lineares com o método do Gradiente Conjugado. Apresentamos testes numéricos, em que realizamos a decomposição de Cholesky incompleta da matriz Hessiana do problema em algumas iterações do algoritmo, e nos quais obtemos aceleração da convergência em relação a outras matrizes propostas anteriormente.

Abstract

Here, we study the unconstrained minimization problem and its resolution through Quasi-Newton methods, specially the BFGS method, proposed in the 60s by Broyden, Fletcher, Goldfarb and Shanno, as well as its generalization to large scale problems, the L-BFGS method, proposed by Nocedal in the 80s. We present the classical convergence results for both methods. For the L-BFGS method, the scaling matrix plays a big role in the convergence of the iterations. In this sense, we propose a new scaling matrix, by making use of the incomplete Cholesky factorization technique for symmetric positive definite matrices, and we present this factorization in its historical context as preconditioner to the Conjugate Gradient method for the resolution of systems of linear equations. We present some numerical results, obtained by applying the incomplete Cholesky factorization technique to the Hessian matrix of the minimization problem in some iterations of the L-BFGS method, and they show that we have obtained fastest convergence by comparison to the other scaling matrices proposed earlier.

Introdução

Muitos modelos matemáticos oriundos das ciências e engenharias de modo geral demandam métodos de otimização numérica. Com o avanço das modernas técnicas computacionais, modelos cada vez mais complexos são explorados, exigindo métodos mais elaborados e robustos.

Várias formulações destes modelos consistem em minimizar uma função não-linear sem restrições ou ainda restrita a um domínio no qual a solução encontra-se no interior do mesmo. Em geral, os métodos de otimização restrita estão fortemente fundamentados na teoria de otimização irrestrita [1], [2], [3], de modo que, estabelecer um método robusto para minimização de uma função não-linear sem restrições, por exemplo, é considerado uma tarefa muito relevante.

Entre os métodos mais famosos para minimização irrestrita, encontra-se o método BFGS [11], proposto na década de 60 por Broyden, Fletcher, Goldfarb e Shanno, simultaneamente. Trata-se de um método do tipo quasenewtoniano clássico, cuja fórmula de aproximação da matriz Hessiana se mantém simétrica e definida positiva, propriedades fundamentais para garantir boas taxas de convergência.

Embora a motivação dos métodos quasenewtonianos esteja, em princípio, fundamentada na economia gerada em não calcular derivadas de segunda ordem, sua principal vantagem atual reside no baixo custo computacional das iterações proveniente das aproximações das inversas da matriz Hessiana. Isso significa que, para a obtenção da direção de busca, somente um produto de uma matriz por um vetor é necessário, o que é muito mais vantajoso por exemplo que o método de Newton, no qual um sistema linear deve ser resolvido a cada iteração.

Todos esses aspectos tomam maior relevância ainda quando trabalhamos com problemas envolvendo milhares de variáveis, denominados problemas de grande porte, em que a Hessiana geralmente é uma matriz esparsa. Nesse contexto, Nocedal [16] desenvolveu um método muito eficiente denominado L-BFGS, o método BFGS com memória limitada. Este método é muito utilizado nos códigos computacionais mais utilizados em otimização numérica.

Neste trabalho estabelecemos uma nova estratégia para a utilização do método L-BFGS. Consiste basicamente em introduzir fatorações incompletas na formulação do método, produzindo aproximações muito mais precisas que aquelas utilizadas originalmente. Fatorações incompletas normalmente são empregadas como precondicionadores de métodos iterativos lineares [9], [10]. Procuram melhorar o condicionamento dos sistemas lineares de modo que o número de iterações seja reduzido substancialmente.

O trabalho está estruturado como se segue. No Capítulo 1 apresentamos os métodos de otimização irrestrita, no qual colocamos algumas propriedades e definições básicas para melhor compreensão do trabalho. Neste capítulo, é dada ênfase ao método BFGS, sua formulação e propriedades teóricas. No Capítulo 2 descrevemos a fatoração incompleta e seu uso para precondicionador no método do gradiente conjugado. No Capítulo 3 descrevemos o método L-BFGS, em que sua fórmula de memória limitada se ajusta bem à fatoração incompleta. No Capítulo 4 descrevemos os resultados dos testes numéricos que mostram a efetividade de nossa implementação, usando para tal o ambiente computacional CUTEr [15]. Terminamos o trabalho com seus aspectos conclusivos sinalizando futuras possibidades de continuidade.

Capítulo 1

Métodos para Minimização Irrestrita

1.1 Definições e Propriedades Básicas

Antes de começarmos a discussão de nosso trabalho, é necessário apresentar algumas definições importantes.

Condições necessárias e suficientes para o problema de minimização

Sabemos, do cálculo em uma variável, que o mínimo local de uma função pode ser encontrado através de condições que caracterizam este ponto. Não estaremos aqui interessados em extremos (máximos ou mínimos) globais, pois este problema é significativamente mais complexo do que o de se encontrar extremos locais. Salientamos ainda que o problema de se encontrar máximos é completamente análogo ao de se encontrar mínimos, portanto nossa discussão limitar-se-á ao segundo caso.

Lembramos do cálculo em uma variável que, se $x_\star$ é um minimizador local de f, então $f'(x_\star) = 0$ e $f''(x_\star) \geq 0$. Por outro lado, estas condições não garantem que $x_\star$ seja um minimizador de f, mas apenas um ponto crítico desta função; devemos exigir que $f''(x_\star) > 0$. Em várias variáveis, a situação é análoga: para que um ponto $x_\star \in \mathbb{R}^n$ seja mínimo de uma função $f : \mathbb{R}^n \to \mathbb{R}$, é necessário e suficiente que $\nabla f(x_\star) = 0$ e que $\nabla^2 f(x_\star)$ seja positiva definida, o que nos garante convexidade local.

Teorema 1.1. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes continuamente diferenciável no conjunto aberto e convexo $D \subset \mathbb{R}^n$. Se $x_\star \in D$ e $\nabla f(x_\star) = 0$, e se $\nabla^2 f$ for Lipschitz em $x_\star$ com $\nabla^2 f(x_\star)$ não singular, então $x_\star$ é um minimizador local de f se e somente se $\nabla^2 f(x_\star)$ for definida positiva.*

A demonstração pode ser encontrada em [1].

Podemos então relacionar o problema de minimização de uma função real de várias variáveis ao problema de encontrarmos o zero de uma função vetorial (ou a solução de um sistema de equações não-lineares), com a ressalva de que os problemas são relacionados, porém, não equivalentes. Como queremos nos concentrar nos problemas de minimização, trabalharemos aqui somente com o problema seguinte:

Seja $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes continuamente diferenciável. Encontre $x_\star$ tal que

$$f(x_\star) = \min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

Para mais detalhes sobre a resolução de sistemas de equações não-lineares, veja por exemplo [1], [5] ou [8].

Algoritmos de Otimização

O problema de minimização de uma função suave sem restrições tem recebido muita atenção nas últimas décadas. Existem inúmeras alternativas para a resolução deste tipo de problema, na forma de algoritmos computacionais. Vamos nos concentrar em um tipo destes métodos, que descreveremos ao longo deste trabalho.

Os algoritmos para minimização irrestrita exigem que o usuário forneça um ponto inicial, que chamaremos de x_0. Se o usuário possuir informações relevantes sobre o comportamento da função, poderá fornecer x_0 de forma que este esteja próximo da solução. Do contrário, devemos escolher x_0 de maneira arbitrária. Começando com este ponto x_0, os algoritmos de minimização geram uma seqüência de aproximações $x_1, x_2, x_3, \ldots$ que chegam ao fim quando não há mais nada a ser feito para o progresso do algoritmo ou quando encontra-se uma aproximação da solução com a precisão apropriada. Para decidir como mover-se de uma iterada x_k para a próxima, os algoritmos utilizam informações sobre a função f no ponto x_k, e, se possível, informações adicionais das iteradas anteriores, procurando encontrar assim uma nova iterada x_{k+1} em que o valor da função é menor. Existem métodos *não-monótonos* que não exigem um decréscimo da função em todo passo do algo-

ritmo, mas mesmo estes métodos acabam por, eventualmente, exigir um decréscimo na função após um certo número de iterações.

Para seguir do ponto atual x_k para o próximo, existem duas estratégias fundamentais. Basicamente, temos que escolher uma direção para seguir. A primeira estratégia, chamada *busca linear*, consiste em escolher uma direção p_k e, nesta direção, encontrar um ponto x_k que acarrete um decréscimo no valor da função. Desta forma, a iteração seguinte é calculada a partir da iteração anterior pela fórmula $x_{k+1} = x_k + \alpha_k p_k$, em que p_k é a direção escolhida e α_k é o tamanho do passo que tomaremos nesta direção.

Uma outra estratégia, cujos detalhes não serão discutidos aqui, chamada de estratégia de *região de confiança*, escolhe uma distância máxima do ponto atual x_k (o chamado *raio* da região de confiança) e procura nesta vizinhança do ponto x_k uma direção que gere uma nova iterada x_{k+1}, de forma que $f(x_{k+1}) < f(x_k)$. Nestes algoritmos, as informações que temos sobre f são usadas para construir um modelo m_k para a função f. Como este modelo pode não ser uma boa aproximação de f para x longe de x_k, restringimos a busca por um minimizador de m_k a uma região em torno de x_k. Em outras palavras, obtemos o candidato ao passo p resolvendo aproximadamente o subproblema seguinte:

$$\min_p m_k(x_k + p), \quad \text{com } x_k + p \text{ dentro da região de confiança.} \tag{1.2}$$

Se a solução não produz um decréscimo suficiente em f, concluimos que a região de confiança é muito grande, e diminuimos a região para resolver novamente (1.2). Normalmente, a região de confiança é uma bola definida por $||p||_2 \leq \Delta$, em que o escalar $\Delta > 0$ é o raio da região de confiança. Podem ser usadas também regiões de confiança elípticas e regiões em forma de caixa.

O modelo m_k é normalmente definido como uma função quadrática da forma

$$m_k(x_k + p) = f(x_k) + p^T \nabla f(x_k) + \frac{1}{2} p^T B_k p,$$

com B_k sendo a matriz Hessiana $\nabla^2 f(x_k)$ ou alguma aproximação desta matriz. Para mais detalhes sobre métodos de região de confiança, citamos [2].

A estratégia de busca linear será discutida adiante.

Convergência

Aqui, cabe mencionar os tipos de taxa de convergência observados nos nossos algoritmos de minimização.

Definição 1.1. *Seja* $\{x_k\}$ *uma seqüência em* $\mathbb{R}^n$ *que converge para* $x_\star$*. Dizemos que a convergência é* Q-linear *se existe uma constante* $r \in (0,1)$ *tal que*

$$\frac{||x_{k+1} - x_\star||}{||x_k - x_\star||} \leq r, \ \ \forall k \text{ suficientemente grande.}$$

Quando a convergência é Q-linear, o erro decresce a cada iteração por um fator constante (no mínimo). Usamos a letra Q para salientar que este tipo de convergência é definido por um quociente.

Definição 1.2. *A convergência de uma seqüência* $\{x_k\}$ *é dita* Q-superlinear *se*

$$\lim_{k\to\infty} \frac{||x_{k+1} - x_\star||}{||x_k - x_\star||} = 0.$$

Definição 1.3. *A convergência é* Q-quadrática *se*

$$\frac{||x_{k+1} - x_\star||}{||x_k - x_\star||^2} \leq M, \ \ \forall k \text{ suficientemente grande,}$$

com M *uma constante positiva, não necessariamente menor que* 1*.*

A velocidade da convergência depende do fator r (ou de M), cujos valores dependem tanto do algoritmo como das particularidades do problema que queremos resolver. No entanto, esses valores não descaracterizam a convergência de um método; um método que possui convergência Q-quadrática sempre irá convergir numa velocidade maior do que um método com convergência Q-linear.

Toda seqüência que converge Q-quadraticamente também converge Q-superlinearmente, e por sua vez, toda seqüência que converge Q-superlinearmente também converge Q-linearmente. Podemos falar de convergências mais rápidas (cúbica, quártica, etc.) mas estas não são, em geral, relevantes para os algoritmos práticos. Ao longo do nosso trabalho, vamos por vezes omitir a letra Q, falando em convergência linear, quadrática ou superlinear, salvo nos casos em que haja necessidade de salientar esta característica.

Direções de Descida

Como discutimos anteriormente, nossos métodos baseiam-se na escolha de uma direção e calculam a próxima iterada do algoritmo nesta direção. Podemos então tentar descobrir direções que causam diminuições no valor da função, já que este é o critério básico que seguiremos para calcular o próximo passo. Deste modo, vamos exigir que o passo vá numa direção p tal que $f(x_{k+1}) < f(x_k)$, para algum x_{k+1} na direção p, partindo de x_k. Uma direção que satisfaz isso chama-se *direção de descida*.

Matematicamente, p é uma direção de descida de x_k se a derivada direcional de f em x_k na direção de p for negativa, ou seja,

$$\frac{\partial f}{\partial p}(x) = \nabla f(x)^T p < 0. \tag{1.3}$$

Observação 1.1. *Quando falarmos em direção, estaremos considerando que uma normalização já foi realizada. Assim, uma direção p terá sempre norma unitária.*

Como queremos direções de descida, basta escolhermos direções p tais que

$$\nabla f(x_k)^T p < 0 \tag{1.4}$$

Se isso acontece, então para um $\delta > 0$ suficientemente pequeno, temos que $f(x_k + \delta p) < f(x_k)$.

Entre os métodos de direção de descida estão o método do gradiente, o método de Newton e os métodos quasenewtonianos. Porém, mesmo que um método seja baseado nas direções de descida, ainda nos resta decidir o tamanho do passo que iremos tomar a cada iteração, na direção escolhida.

Busca Linear

Queremos aqui discutir maneiras de calcular o tamanho do passo de um método de direção de descida para que tenhamos garantia de convergência. Nesta seção, consideraremos apenas o problema de minimização

$$\min_{x \in \mathbb{R}^n} f : \mathbb{R}^n \to \mathbb{R}$$

pois para este problema temos uma estratégia natural: queremos que, a cada passo, o valor de f diminua. Estratégias semelhantes podem ser desenvolvidas para encontrar

a solução de sistemas não-lineares, mas em geral exigem que, a cada passo, o valor de $F : \mathbb{R}^n \to \mathbb{R}^n$, a função que representa o sistema não-linear, diminua em alguma norma. Assim, estamos transformando o problema de achar a solução de um sistema não-linear em um problema de minimização de uma função f dada por:

$$f = \frac{1}{2}||F||_2^2 : \mathbb{R}^n \to \mathbb{R}.$$

Assim, discutiremos estas estratégias somente para o problema de minimização.

Tamanho do Passo

Basicamente, dada uma direção de descida p_k, queremos tomar um passo nesta direção que nos dê um x_{k+1} que forneça um decréscimo na função $(f(x_{k+1}) < f(x_k))$, e que este decréscimo seja suficiente. Ou seja:

Algoritmo 1.1. Busca Linear

Para cada iteração k

Calcule uma direção de descida p_k

Tome $x_{k+1} = x_k + \alpha_k p_k$ para algum $\alpha_k > 0$ que torne x_{k+1} uma iteração *aceitável* (nos termos acima).

Fim Para

Geometricamente, isto equivale a procurar x_{k+1} considerando uma secção unidimensional de f (na direção p_k), ou seja, queremos achar α_k minimizador da função unidimensional $\phi(\cdot)$ definida por

$$\phi(\alpha) = f(x_k + \alpha p_k), \;\; \alpha > 0$$

No entanto, não queremos gastar muito tempo com esta tarefa, já que nossa intenção é acelerar o método. Assim, ao invés de resolver esse problema exatamente, procuramos relaxar as condições impostas sobre α_k de modo que possamos obter uma iteração em direção à solução, sem gastar muito tempo com a escolha do tamanho do passo.

É importante que qualquer algoritmo que utilize esta estratégia permita que, perto da solução, possamos escolher $\alpha_k = 1$, para que se faça um passo inteiro na direção proposta, pois, se isto não for possível, a convergência pode ser prejudicada perto da solução.

Um algoritmo de busca linear genérico tem duas partes: primeiramente, escolhe-se um intervalo que contenha bons candidatos para α; depois, através de bissecção ou interpolação, acha-se um bom comprimento de passo dentro desse intervalo. Vamos ver que α satisfatórios não estão necessariamente próximos do minimizador de $\phi(\alpha)$.

Critérios para a escolha de α_k

Uma das exigências que precisamos impor sobre x_{k+1} para que este seja aceitável é que

$$f(x_{k+1}) = f(x_k + \alpha_k p_k) < f(x_k). \tag{1.5}$$

Esta condição sozinha não garante que $\{x_k\}$ convirja para um minimizador de f. Podemos ter reduções muito pequenas no valor de f relativamente ao tamanho dos passos. Portanto, gostaríamos de incluir nas nossas condições que não só $f(x_{k+1}) < f(x_k)$, mas que o valor da função diminua pelo menos a uma fração predeterminada do valor inicial. Além disso, também podemos tomar passos muito pequenos em relação à taxa de decréscimo inicial de f.

Para resolver estes problemas, algumas condições foram formuladas para obtermos α.

1. **Condição de Armijo:**

   Para evitar que a função tenha um decréscimo muito pequeno em relação ao seu valor anterior, podemos exigir que a redução em f seja proporcional ao tamanho do passo α_k e à sua derivada direcional na direção p_k, $\nabla f(x_k)^T p_k$, que representa a taxa de decréscimo da função nesta direção. Assim, queremos encontrar α_k entre os α que satisfazem

   $$f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla f(x_k)^T p_k \tag{1.6}$$

   com $c_1 \in (0, 1)$. Esta desigualdade é chamada de *Condição de Armijo*.

   Se α_k satisfaz (1.6), então a redução resultante em $f(x)$ satisfaz

   $$f(x_k) - f(x_{k+1}) \geq -c_1 \nabla f(x_k)^T (x_{k+1} - x_k) \tag{1.7}$$

   como queríamos.

Seja $\overline{\alpha_k}$ o menor valor positivo de α_k para o qual temos a igualdade

$$f(x_k + \alpha_k p_k) = f(x_k).$$

Então, reduções muito pequenas podem ocorrer se $\alpha_k \to \overline{\alpha_k}$ ou se $\alpha_k \to 0$. Assim, o objetivo da busca linear é achar α_k que nos dê um decréscimo significativo em f a cada iteração, e que não esteja próximo dos extremos do intervalo $[0, \overline{\alpha_k}]$. As condições para garantir isso devem ser tais que não excluam o minimizador de $\phi(\alpha)$ quando esta for uma quadrática com curvatura positiva.

No entanto, esta condição não é suficiente para garantir a eficiência do método, pois podemos ver que ela é satisfeita para todos os valores suficientemente pequenos de α, que gerariam passos extremamente curtos. Para solucionar este problema, introduzimos mais uma condição.

2. **Condição de Curvatura**

   Para evitarmos passos muito pequenos (ou seja, para evitar que α_k convirja a 0, o extremo esquerdo do intervalo de α aceitáveis), podemos incluir uma condição que teste a inclinação $\phi'(\alpha) = \nabla f(x_k + \alpha p_k)^T p_k$ decorrente da escolha de α:

   $$\nabla f(x_k + \alpha p_k)^T p_k \geq c_2 \nabla f(x_k)^T p_k \tag{1.8}$$

   com $c_2 \in (c_1, 1)$, c_1 escolhido na condição (1.6). Esta condição é devida a Wolfe.

   Podemos observar que esta condição equivale a exigir que $\phi'(\alpha_k)$ seja maior do que $c_2\phi'(0)$, ou seja, queremos que a inclinação de ϕ em α_k seja maior do que a inclinação de ϕ em 0. Lembrando que $\phi(0) = f(x_k)$, esta exigência garante que α_k não será escolhido pequeno demais, criando um limite inferior para α e assim excluindo o lado esquerdo de $[0, \overline{\alpha}_k]$. Isso faz sentido, pois, se a inclinação de $\phi(\alpha)$ é fortemente negativa, temos uma indicação de que podemos reduzir f significativamente na direção escolhida. Por outro lado, se a inclinação é pouco negativa ou mesmo positiva, é um sinal de que não podemos esperar muito decréscimo nesta direção, e assim deve fazer sentido terminar a busca linear.

Estas duas condições, juntas, são chamadas de *Condições de Wolfe*.

Quando exigimos que $0 < c_1 < c_2 < 1$, estamos garantindo que as duas condições podem ser satisfeitas simultaneamente, gerando um intervalo (α_k^1, α_k^2), de onde podemos escolher α_k para utilização no nosso algoritmo. Na prática, a condição (1.8) geralmente não precisa ser utilizada pois a estratégia de *backtracking* evita passos excessivamente pequenos.

Backtracking

Basicamente, o procedimento de busca linear com *backtracking* consiste no seguinte algoritmo:

Algoritmo 1.2. Backtracking

Escolha $\alpha > 0$; ρ, $c \in (0, 1)$

Repita até $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f(x_k)^T p_k$

$\alpha \leftarrow \rho\alpha$;

Fim Repita.

Termine com $\alpha_k = \alpha$.

Este algoritmo geralmente fornece resultados suficientemente bons para que sejam utilizados na prática, mas nem sempre garante que as condições de Wolfe sejam satisfeitas, sendo uma simplificação que só pode ser utilizada em casos específicos.

Resultados de Convergência

Aqui, enunciamos um resultado útil na análise de convergência da estratégia de busca linear e, conseqüentemente, de convergência de um método de direção de descida genérico utilizando a estratégia de busca linear.

Observação 1.2. *As condições de Wolfe são invariantes por mudança de escala: multiplicar a função objetivo (ou seja, a função que queremos minimizar) por uma constante ou fazer uma mudança de variáveis afim não as altera.*

Observação 1.3. *Quando dissermos que uma função $g : D \to \mathbb{R}$, $D \subset \mathbb{R}^n$ é tal que $g \in Lip_\gamma(D)$, estaremos dizendo que para todo $x, y \in D$, temos*

$$|g(x) - g(y)| \leq \gamma ||x - y||.$$

A função g é então chamada Lipschitz contínua no conjunto D com constante de Lipschitz γ.

Teorema 1.2. *Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^2(D)$, em que D é um aberto convexo de $\mathbb{R}^n$, e $\nabla^2 f \in Lip_\gamma(D)$. Considere uma seqüência $\{x_k\}$ gerada por $x_{k+1} = x_k + \alpha_k p_k$, em que $\nabla f(x_k)^T p_k < 0$, $\forall k$, e α_k é escolhido para satisfazer as condições de Wolfe com $c_1 < \frac{1}{2}$. Se $\{x_k\} \to x_\star \in D$, em que $\nabla^2 f(x_\star)$ é positiva definida, e se*

$$\lim_{k\to\infty} \frac{||\nabla f(x_k) + \nabla^2 f(x_k) p_k||_2}{||p_k||_2} = 0, \tag{1.9}$$

então existe um índice $k_0 \geq 0$ tal que, $\forall\, k \geq k_0$, $\alpha_k = 1$ é admissível.

Além disso, $\nabla f(x_\star) = 0$ e, se $\alpha_k = 1$ $\forall k \geq k_0$, então $\{x_k\}$ converge superlinearmente para $x_\star$.

A demonstração do Teorema 1.2 pode ser encontrada em [1].

Com estes resultados, fica claro que, se utilizarmos a estratégia de busca linear, os métodos de direção de descida convergem, desde que a escolha do passo satisfaça às condições de Wolfe, ou alguma condição equivalente. Portanto, daqui para frente, consideraremos sempre que o tamanho do passo tenha sido escolhido conforme estas condições, e assim restará somente analisar a taxa de convergência de cada método.

1.2 Métodos para Minimização Irrestrita

Agora, consideraremos o problema de achar qual direção nos leva a uma descida mais rápida no valor da função, em dada norma $||\cdot||$, na forma

$$\min_{p\in\mathbb{R}^n} \nabla f(x)^T p \qquad \text{sujeito a } ||p|| = 1$$

Na norma l_2, este problema tem solução $p = -\nabla f(x)/||\nabla f(x)||_2$. Assim, a direção na qual podemos diminuir a função mais rapidamente é o gradiente com sinal negativo. Esta direção é chamada de direção do gradiente (ou direção de máximo decréscimo).

O método que utiliza esta direção é chamado *método do gradiente*. Apesar da direção utilizada no método do gradiente ser garantidamente uma direção de descida, uma análise da taxa de convergência do método do gradiente mostra que ele é bastante ineficiente na prática.

Para analisar a taxa de convergência de qualquer algoritmo de minimização, começamos analisando seu desempenho quando aplicado a uma função quadrática, visto que a análise baseada em uma função quadrática geralmente é mais

simples em comparação a uma função não-linear qualquer, e que toda função suave (contínua e com derivadas contínuas) se comporta como uma função quadrática em uma região suficientemente pequena; podemos observar isto diretamente da aproximação da função pela sua expansão de Taylor de segunda ordem.

Seja $F(x)$ a função quadrática $c^T x + \frac{1}{2} x^T G x$, com c um vetor constante e G uma matriz simétrica positiva definida. Se o método do gradiente for aplicado a F, sua taxa de convergência é *linear*. Se $\lambda_{\max}$ e $\lambda_{\min}$ forem o maior e o menor autovalores de G, respectivamente, e utilizando a notação $||\cdot||_G$ para sinalizar a norma calculada de forma que, para uma matriz $G \in \mathbb{R}^{n \times n}$, $||x||_G^2 = x^T G x$, então é possível mostrar que

$$\begin{aligned} ||x_{k+1} - x_\star||_G^2 &\leq \frac{(\lambda_{\max} - \lambda_{\min})^2}{(\lambda_{\max} + \lambda_{\min})^2} ||x_k - x_\star||_G^2 \\ &= \frac{(\kappa - 1)^2}{(\kappa + 1)^2} (F(x_k) - F(x_\star)), \end{aligned}$$

em que κ denota $cond(G)$, o número de condição espectral de G. A demonstração pode ser encontrada em [7].

Este resultado implica que a constante de erro assintótico, que nos dá o fator de redução no erro em cada passo, pode ser muito próxima de 1. Por exemplo, se $\kappa = 100$ (o que não torna G *muito* mal condicionada), a constante de erro é $(99/101)^2 \approx .96$, e assim o ganho em precisão a cada iteração é mínimo. Na prática, o método do gradiente exige centenas de iterações para fazer pouco progresso na direção da solução. Um resultado similar de convergência nos diz que este comportamento também ocorre se aplicamos o método do gradiente em uma função geral.

Método de Newton

Gostaríamos de encontrar um método que fosse baseado nas direções de descida, com resultados de convergência melhores do que o obtido para o método do gradiente.

O método de Newton, como vamos observar, tem taxa de convergência Q-quadrática, o que significa que, a cada passo do algoritmo, o erro diminui quadraticamente.

Método de Newton para Problemas de Minimização

Aqui, temos uma função $f : \mathbb{R}^n \to \mathbb{R}$ e queremos achar um mínimo de f. Se $x_\star \in \mathbb{R}^n$ é um mínimo de f, então o problema de achar o mínimo de uma função $f : \mathbb{R}^n \to \mathbb{R}$ é equivalente ao problema

$$\textit{Encontrar } x_\star \textit{ tal que:}$$

$$\nabla f(x_\star) = 0 \tag{1.10}$$

$$e \quad \nabla^2 f(x_\star) \quad \textit{seja Positiva Definida.}$$

Assim, usaremos o modelo quadrático derivado da série de Taylor da função em torno do ponto x para aproximar nossa função. A série de Taylor de segunda ordem de f em torno de x é:

$$f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x) p + \mathcal{O}(||p||^3).$$

Nosso modelo quadrático será então:

$$q(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x) p. \tag{1.11}$$

Como queremos minimizar f, vamos começar tentando achar p que minimiza o modelo quadrático q. Assim, derivando o modelo em relação a p, obtemos:

$$\begin{aligned} q'(x+p) = \nabla f(x) + \nabla^2 f(x) p &= 0 \\ \nabla^2 f(x) p &= -\nabla f(x) \\ p &= -(\nabla^2 f(x))^{-1} \nabla f(x) \end{aligned} \tag{1.12}$$

Desta forma, o algoritmo do método de Newton para minimização de uma função segue abaixo.

Algoritmo 1.3. Método de Newton para Minimização

Dados $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes continuamente diferenciável, $x_0 \in \mathbb{R}^n$

Para $k = 0$ até *convergência*, faça:

Resolva $\nabla^2 f(x) p_k = -\nabla f(x)$ para p_k

Faça $x_{k+1} = x_k + p_k$

Fim Para.

Observação 1.4. *Note que a direção*

$$p_k = -B_k^{-1}\nabla f(x_k),$$

com $B_k = \nabla^2 f(x_k)$ *(ou alguma aproximação da matriz Hessiana), é uma direção de descida se* B_k *for positiva definida. De fato,*

$$\nabla f(x_k)^T p_k = -\nabla f(x_k)^T B_k^{-1}\nabla f(x_k) < 0,$$

se B_k^{-1} *é positiva definida. Isso garante também que o modelo quadrático terá um único minimizador.*

Abaixo, temos um resultado de convergência que garante que a taxa do método de Newton é quadrática. Além disso, este teorema mostra que somente podemos esperar que o método de Newton convirja se tivermos uma boa aproximação inicial para a solução.

Teorema 1.3. *Suponha que* $f : \mathbb{R}^n \to \mathbb{R}$ *é duas vezes continuamente diferenciável num conjunto aberto e convexo* $D \subset \mathbb{R}^n$ *e que* $\nabla^2 f(x) \in Lip_\lambda(N(x_\star, r))$*. Então existe* $\epsilon > 0$ *tal que para todo* $x_0 \in N(x_\star, \epsilon)$*, se* $\nabla^2 f(x)$ *é positiva definida para todo* $x \in N(x_\star, \epsilon)$*, o método de Newton está bem definido, a seqüência gerada pelo método converge para* $x_\star$*, e existe* $c > 0$ *tal que*

$$||x_{k+1} - x_\star|| \leq c||x_k - x_\star||^2. \tag{1.13}$$

A demonstração do Teorema 1.3 pode ser encontrada em [2].

Motivação para os Métodos Quasenewtonianos

O teorema que acabamos de enunciar nos garante que o método de Newton tem uma convergência excelente, quadrática. Infelizmente, também nos diz que esta convergência ocorre de forma no mínimo local, ou seja, somente vizinhança $N(x_\star, \epsilon)$ de $x_\star$. Portanto, se tivermos x_0 longe de $x_\star$ de forma que $x_0 \notin N(x_\star, \epsilon)$, o método pode não convergir.

Assim, o método de Newton pode ser muito bom e rápido quando temos boas aproximações iniciais; por outro lado, para muitos problemas não temos convergência global, e para implementar o método de Newton precisamos calcular a matriz $\nabla^2 f(x_k)$ em cada iteração. Isto pode ser bastante trabalhoso para problemas

de grande porte, ou até mesmo impossível, em problemas em que as derivadas não estão disponíveis (ou ainda, não temos a forma analítica da função), e precisamos em cada iteração resolver um sistema linear, que pode ser mal condicionado ou muito custoso computacionalmente.

Desta forma, os métodos Quasenewtonianos aparecem como uma tentativa de resolver ou melhorar estes problemas do método de Newton, porém procurando manter a boa convergência local do método.

Vamos então descrever um algoritmo genérico com essas propriedades.

Algoritmo 1.4. Algoritmo Quasenewtoniano Genérico para Minimização

Dados $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^2$, $x_0 \in \mathbb{R}^n$;

Para $k = 0$ até *convergência* faça

1. Calcule $\nabla f(x_k)$ e decida se deve continuar ou não.
   (Ex. $\nabla f(x_k) = 0$, então pare.)
2. Calcule $B_k = \nabla^2 f(x_k)$ ou alguma aproximação escolhida
   segundo as condições obtidas em (1.10).
3. Se B_k for mal condicionada, faça uma perturbação apropriada.
4. Resolva $B_k s_k^N = -\nabla f(x_k)$
5. Decida se deve tomar o passo de Newton $x_{k+1} = x_k + s_k^N$ ou
   se deve escolher x_{k+1} por alguma estratégia global

Fim Para.

O Passo 1 resume-se a analisar o critério de convergência desejado. Como queremos minimizar f, queremos achar $x_\star$ tal que $\nabla f(x_\star) = 0$, com $\nabla^2 f(x_\star)$ Positiva Definida. Portanto, no Passo 2 e no Passo 3, precisamos garantir que a aproximação para a Hessiana seja também positiva definida, para garantir que o método de Newton vá para um mínimo, e não para um outro ponto crítico de f. Se x_k não estiver suficientemente próximo da solução, $\nabla^2 f(x_k)$ pode não ser positiva definida. Neste caso, temos duas estratégias possíveis. Primeiramente, tentamos usar a forma do modelo quadrático para a função f, em particular as direções de curvatura negativa p, tais que

$$p^T \nabla^2 f(x_k) p < 0.$$

Assim, garantimos que f descresce rapidamente.

Outra maneira de fazer isto é modificar o modelo de forma que ele tenha um único minimizador, e utilizar este minimizador para definir o passo de

Newton.

Assim, fazemos uma perturbação da forma

$$\nabla^2 f(x_k) + \mu_k I, \qquad \mu_k > 0$$

com μ_k não muito maior que o menor μ que torne $B_k = \nabla^2 f(x_k) + \mu I$ positiva definida e razoavelmente bem condicionada. Fazer modificações desta forma garante que obteremos um resultado confiável. Não entraremos em detalhes sobre essas modificações; mais informações podem ser encontradas em [1].

Para o Passo 5, decidimos o tamanho do passo a ser tomado através de uma estratégia como a busca linear.

Nos capítulos seguintes, veremos as estratégias possíveis para cada passo deste algoritmo genérico para a resolução de problemas de minimização.

Observamos que, para utilizar o método de Newton como estratégia de resolução de um problema de minimização, em cada passo é necessário calcular o valor da função e da sua derivada (neste caso, o gradiente e a matriz Hessiana de f, respectivamente) no ponto atual. No entanto, isso pode ser caro computacionalmente ou mesmo impossível, pois por vezes, nos problemas reais, não temos a forma analítica da função e de suas derivadas.

Neste capítulo, apresentaremos métodos desenvolvidos como alternativas para o método de Newton para a resolução de problemas de minimização. Queremos minimizar uma função $f : \mathbb{R}^n \to \mathbb{R}$. Para isto, estabelecemos em (1.10) as condições necessárias para que $x_\star$ seja um minimizador de f: primeiramente, $x_\star$ deve ser solução de

$$\nabla f(x) = 0. \tag{1.14}$$

Observe que quando apresentamos o método de Newton, pela equação (1.12) obtivemos a direção de Newton definida por

$$p_k^N = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k).$$

Se não quisermos utilizar a matriz Hessiana de f em cada iteração, podemos substituir esta direção por uma aproximação

$$p_k = -B_k^{-1} \nabla f(x_k).$$

Então (1.9) é equivalente a

$$\lim_{k\to\infty} \frac{||(B_k - \nabla^2 f(x_\star))p_k||}{||p_k||} = 0, \tag{1.15}$$

o que significa que as matrizes de aproximação geradas pelos métodos quasenewtonianos não precisam necessariamente convergir para $\nabla^2 f(x_\star)$; basta que, a cada k, B_k seja uma boa aproximação de $\nabla^2 f(x_\star)$ ao longo da direção p_k. Com isto, pode-se demonstrar o resultado seguinte.

Teorema 1.4. *Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^3(\mathbb{R}^n)$. Considere a iteração $x_{k+1} = x_k + \alpha_k p_k$ e $p_k = -B_k^{-1}\nabla f(x_k)$. Suponha que $\{x_k\} \to x_\star$ em que $\nabla f(x_\star) = 0$ e $\nabla^2 f(x_\star)$ é positiva definida. Suponha ainda que $\alpha_k = 1$ em todas as iterações.*

Então $\{x_k\}$ converge superlinearmente se e somente se (1.15) é válida.

Este teorema pode ser encontrado em [2], e será utilizado mais à frente para mostrar que os métodos quasenewtonianos apresentados convergem superlinearmente.

Agora, observe que se aproximarmos f pelo modelo quadrático derivado da sua expansão de Taylor, teremos

$$m_k(p) = f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T B_k p,$$

e o minimizador deste modelo quadrático é exatamente a direção p_k obtida acima.

Gostaríamos então que, na próxima iteração, a matriz B_{k+1} de aproximação fosse obtida através da aproximação anterior B_k e das informações obtidas no passo anterior: $f(x_k)$ e $\nabla f(x_k)$. Para isto, é natural exigir que o gradiente do modelo m_{k+1} seja igual ao gradiente de f ao menos nas duas últimas iteradas, x_k e x_{k+1}. Assim, como $\nabla m_{k+1}(0) = \nabla f(x_{k+1})$, basta exigirmos que

$$\nabla m_{k+1}(-\alpha_k p_k) = \nabla f(x_{k+1}) - \alpha_k B_{k+1} p_k = \nabla f(x_k),$$

ou seja,

$$B_{k+1}\alpha_k p_k = \nabla f(x_{k+1}) - \nabla f(x_k).$$

Simplificando a notação, chamando $s_k = x_{k+1} - x_k$ e $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$, temos a *equação secante*:

$$B_{k+1} s_k = y_k, \tag{1.16}$$

Assim, depois de calcular x_{k+1}, calcula-se uma nova aproximação B_{k+1}, levando em consideração B_k e as novas informações de curvatura obtidas com esta nova iteração. Uma fórmula de correção é uma definição de B_k da forma

$$B_{k+1} = B_k + U_k,$$

em que U_k é a matriz de correção. Então, exigimos que B_{k+1} assim definida satisfaça a equação secante (1.16). Em cada iteração obtêm-se informações novas sobre a curvatura somente em uma direção; portanto, é natural exigir que a correção seja uma matriz de posto 1 ou 2. Se a correção for uma matriz de posto 1, tomará a forma

$$B_{k+1} = B_k + uv^T, \tag{1.17}$$

para vetores $u, v \in \mathbb{R}^n$. De (1.16), obtemos então que, se $v^T s_k \neq 0$, $u = \frac{1}{v^T s_k}(y_k - B_k s_k)$, e assim

$$B_{k+1} = B_k + \frac{1}{v^T s_k}(y_k - B_k s_k)v^T. \tag{1.18}$$

Como a equação secante não determina U_k unicamente, devemos impor mais condições para que B_{k+1} tenha certas propriedades. Note que, mesmo que B_k seja simétrica, não podemos garantir que B_{k+1} também o será, e simetria é uma propriedade importante da matriz Hessiana que não queremos desprezar. Portanto, procuramos correções que mantenham a propriedade seguinte: Se B_k for simétrica, B_{k+1} também deverá ser. Para uma correção de posto 1, esta exigência determina B_{k+1} unicamente. Note que, para que (1.17) mantenha a simetria de B_k, v deve ser um múltiplo de u. Assim, a correção de posto 1 toma a forma

$$B_{k+1} = B_k + \frac{1}{(y_k - B_k s_k)^T s_k}(y_k - B_k s_k)(y_k - B_k s_k)^T,$$

com $y_k - B_k s_k$ e $(y_k - B_k s_k)^T s_k$ não nulos. Esta fórmula define a *correção simétrica de posto 1*, e o método que utiliza esta fórmula é denominado SR1. Mais detalhes sobre este método podem ser encontrados em [2], por exemplo.

Observação 1.5. Como vimos na introdução deste trabalho, se $x_\star$ satisfaz (1.14), $x_\star$ pode ser um minimizador, maximizador ou ponto de sela da função f. Portanto, para garantir a convexidade da função f, devemos exigir que $\nabla^2 f(x)$ seja positiva definida, ao menos em uma vizinhança de $x_\star$. Assim, seria bom que, a partir de um

certo k, para todo $x \in \mathbb{R}^n$, $x^T B_k x > 0$. Em particular, para $x = s_k$,

$$0 < {s_k}^T B_{k+1} s_k = {s_k}^T y_k \Leftrightarrow {s_k}^T y_k > 0$$

Observe que esta condição é a condição de curvatura (1.8).

Infelizmente, é fácil ver que, mesmo se B_k for positiva definida, B_{k+1} gerada pelo SR1 pode não possuir esta propriedade. Existem métodos, como os baseados em região de confiança, que não dependem desta característica da matriz de aproximação; no entanto, se utilizarmos a estratégia de busca linear, como é o caso dos métodos que estamos estudando, esta característica é fundamental para o bom funcionamento do algoritmo.

Além disso, existe outro problema: o denominador $(y_k - B_k s_k)^T s_k$ pode ficar muito próximo de zero. Apesar de existirem estratégias para corrigir este problema, vamos procurar estender nossa análise anterior de modo a obter métodos que preservem mais informação sobre o problema original.

Correções de Posto 2

Similarmente ao desenvolvimento para as correções de posto 1, dada uma matriz simétrica B_k, fazemos $B^0 = B_k$, e procuramos obter B^1 usando a fórmula (1.17):

$$B^1 = B^0 + uv^T,\ v^T s_k \neq 0.$$

Como vimos, esta matriz satisfaz a equação secante, porém não é simétrica. Por isso, vamos definir

$$B^2 = \frac{1}{2}(B^1 + B^{1T}).$$

No entanto, como B^2 nem sempre satisfará a equação secante, o processo é repetido. Geramos assim uma seqüência de matrizes:

$$\begin{cases} B_{k+1}^{2i+1} &= B_{k+1}^{2i} + \dfrac{(y_k - B_k^{2i} s_k)v^T}{v^T s_k} \\ B_{k+1}^{2i} &= \dfrac{1}{2}(B_{k+1}^{2i+1} + (B_{k+1}^{2i+1})^T) \end{cases} \tag{1.19}$$

Powell [18] mostrou que esta seqüência converge para

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)v^T + v(y_k - B_k s_k)^T}{v^T s_k} - \frac{(y_k - B_k s_k)^T s_k}{(v^T s_k)^2} v v^T. \tag{1.20}$$

Observe que $B_{k+1} - B_k$ é uma matriz simétrica de posto 2, e B_{k+1} satisfaz a equação secante. Esta fórmula está bem definida para todo v que não seja ortogonal a s_k. Se escolhermos $v = s_k$, obteremos a fórmula que define o método *Powell-Symmetric-Broyden* (PSB).

Teorema 1.5. *Seja $B_k \in \mathbb{R}^{n\times n}$ simétrica, $s_k, y_k \in \mathbb{R}^n$, $s_k \neq 0$ (ou seja, $x_k \neq x_{k+1}$). Então, a solução única de*

$$\min_{B\in\mathbb{R}^{n\times n}} ||B - B_k||_F$$

$$s/a\ Bs_k = y_k,\ (B - B_k)\ \textit{simétrica}$$

é dada por (1.20), com $v = s_k$.

Este resultado pode ser encontrado em [12].

Se tomarmos $v = y_k$ em (1.20), obteremos a fórmula chamada *Davidon-Fletcher-Powell* (DFP):

$$B_{k+1} = B_k - \frac{1}{s_k^T B_k s_k} B_k s_k s_k^T B_k + \frac{1}{y_k^T s_k} y_k y_k^T + (s_k^T B_k s_k) w_k w_k^T, \tag{1.21}$$

com

$$w_k = \frac{y_k}{y_k^T s_k} - \frac{B_k s_k}{s_k^T B_k s_k}.$$

Podemos observar que $w_k^T s_k = 0$. Portanto, qualquer múltiplo da matriz de posto 1 $w_k w_k^T$ pode ser adicionado a B_{k+1} sem afetar a condição da equação secante. Assim, obtemos uma família a um parâmetro de fórmulas de correção, definidas pela fórmula seguinte:

$$B_{k+1}^{\phi} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + \phi_k (s_k^T B_k s_k) w_k w_k^T, \tag{1.22}$$

em que o escalar ϕ_k depende de y_k e de $B_k^{\phi} s_k$, e $w_k = \dfrac{y_k}{y_k^T s_k} - \dfrac{B_k s_k}{s_k^T B_k s_k}$.

1.3 O Método BFGS

A escolha de ϕ_k é objetivo de muitos estudos teóricos, e podemos observar que a fórmula DFP (1.21) é obtida se tomarmos $\phi_k \equiv 1$ em (1.22). Se escolhermos $\phi_k \equiv 0$, obteremos o método que costuma ser o mais eficiente na prática, chamado *Broyden-Fletcher-Goldfarb-Shanno* (BFGS):

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \tag{1.23}$$

Cabe perguntar agora se esta aproximação mantém B_{k+1} positiva definida, já que, como vimos anteriormente, esta característica pode representar a garantia de que nosso método irá convergir para um mínimo da função, e não somente para um ponto crítico geral. Além disso, observamos que, se B_{k+1} for positiva definida, o modelo quadrático local terá um mínimo único, e a direção de busca definida por $p_k = -B_k^{-1}\nabla f(x_k)$ é de descida. Vamos mostrar então que a fórmula BFGS gera matrizes positivas definidas.

Supondo que B_k é positiva definida, pode-se encontrar R tal que $B_k = R^T R$. Então, podemos escrever (1.23) como

$$B_{k+1} = R^T W R, \tag{1.24}$$

com a matriz W dada por

$$W = I - \frac{\overline{s}\,\overline{s}^T}{\overline{s}^T\overline{s}} + \frac{\overline{y}\,\overline{y}^T}{\overline{y}^T\overline{s}}, \tag{1.25}$$

em que $\overline{s} = Rs_k$ e $\overline{y} = (R^T)^{-1}y_k$. Pela equação (1.24), B_{k+1} será positiva definida sempre que W for positiva definida. Chame então $\Pi(A)$ o determinante de A, que é o produto dos seus autovalores. Sabemos que, para quaisquer duas matrizes A e M, $\Pi(AM) = \Pi(A)\Pi(M)$. Se calcularmos o produto dos autovalores das matrizes de ambos os lados de (1.24), teremos que

$$\begin{aligned} \Pi(B_{k+1}) &= \Pi(R^T W R) \\ &= (\Pi(R))^2 \Pi(W). \end{aligned}$$

Como W é uma modificação da matriz identidade I por uma correção de posto 2, W possui $n-2$ autovalores unitários, se $n \geq 3$. Sejam λ_1 e λ_2 os dois autovalores restantes. De (1.25) podemos ver que os autovetores associados a λ_1 e λ_2

são combinações lineares de $\overline{s}$ e $\overline{y}$. Usando as propriedades de traço e determinante de uma matriz, odemos verificar que

$$\lambda_1 + \lambda_2 = \frac{(\overline{y}^T\overline{s} + \overline{y}^T\overline{y})}{\overline{y}^T\overline{y}},$$

e

$$\lambda_1\lambda_2 = \frac{\overline{y}^T\overline{s}}{\overline{s}^T\overline{s}}.$$

Como temos que $\overline{s}^T\overline{s} > 0$, ambos λ_1 e λ_2 serão positivos se $\overline{y}^T\overline{s}$ for positivo. Como $\overline{y}^T\overline{s} = y_k^T s_k$, a fórmula BFGS tem a propriedade de gerar matrizes positivas definidas se e somente se $y_k^T s_k > 0$.

Observação 1.6. *A condição $y_k^T s_k > 0$ equivale a*

$$(\nabla f(x_{k+1}) - \nabla f(x_k))^T s_k > 0,$$

ou seja,

$$\nabla f(x_{k+1})^T s_k > \nabla f(x_k)^T s_k,$$

o que nos diz que a derivada direcional de $f(x)$ na direção s_k é maior em x_{k+1} do que em x_k. Esta condição tem que ser satisfeita se o algoritmo implementa a segunda condição da busca linear que diz que

$$s_k^T \nabla f(x_{k+1}) > \beta s_k^T \nabla f(x_k), \beta \in (0, 1).$$

Portanto, a fórmula BFGS mantém as propriedades das matrizes positivas definidas.

1.4 Convergência do Método BFGS

Para analisar a convergência dos métodos secantes para minimização, vamos considerar o tamanho do passo $\lambda_k = 1$, $\forall k$. Faremos isso para manter a coerência com os resultados clássicos obtidos nesta área, já que, na prática, próximo à solução, os algoritmos tomam $\lambda_k = 1$, o que de fato gera um decréscimo em f. Seguiremos aqui o trabalho feito em [13].

Primeiramente, vamos provar um teorema que garante a convergência do método BFGS, se ele satisfizer uma propriedade semelhante à deterioração limitada obtida em [1].

Lema 1.1. *Suponha que* $F : \mathbb{R}^n \to \mathbb{R}^n$ *seja diferenciável no conjunto aberto e convexo* D*, e suponha que para algum* $x_\star \in D$ *e* $p > 0$*,*

$$||F'(x) - F'(x_\star)|| \leq K||x - x_\star||^p. \tag{1.26}$$

Então, para todo $u, v \in D$*,*

$$||F(v) - F(u) - F'(x_\star)(v - u)|| \leq K \max\{||v - x_\star||^p, ||u - x_\star||^p\}||v - u||. \tag{1.27}$$

Além disso, se $F'(x_\star)$ *for inversível, então existem* $\epsilon > 0$ *e* $\rho > 0$ *tais que, se* $\max\{||v - x_\star||, ||u - x_\star||\} \leq \epsilon$*, então* $u, v \in D$ *e*

$$\frac{1}{\rho}||v - u|| \leq ||F(v) - F(u)|| \leq \rho||v - u||. \tag{1.28}$$

Na análise a seguir, utilizaremos uma norma de matrizes $||\cdot||_M$, definida como $||A||_M = ||MAM||$, e uma norma matricial induzida pela norma de vetores $||\cdot||$ que podem ser relacionadas pois todas as normas num espaço vetorial de dimensão finita são equivalentes, ou seja, existe $\eta > 0$ tal que

$$||A|| \leq \eta||A||_M. \tag{1.29}$$

Teorema 1.6. *Seja* $F : \mathbb{R}^n \to \mathbb{R}^n$ *diferenciável em um conjunto aberto e convexo* D*. Suponha que, para algum* $x_\star \in D$ *e* $p > 0$*, a desigualdade (1.26) seja satisfeita com* $F(x_\star) = 0$ *e* $F'(x_\star)$ *não singular. Seja* $U : \mathbb{R}^n \times L(\mathbb{R}^n) \to P\{L(\mathbb{R}^n)\}$ *definida em uma vizinhança* $N = N_1 \times N_2$ *de* $(x_\star, F'(x_\star))$*, com* $N_1 \subset D$ *e* N_2 *contendo somente matrizes não singulares. Suponha que existem constantes não negativas* α_1 *e* α_2 *tais que, para cada* (x, B) *em* N *e para* $\overline{x} = x - B^{-1}F(x)$*, a função* U *satisfaça*

$$\begin{aligned} ||\overline{B} - F'(x_\star)||_M &\leq [1 + \alpha_1 \max\{||\overline{x} - x_\star||^p, ||x - x_\star||^p\}] \cdot ||B - F'(x_\star)||_M \\ &+ \alpha_2 \max\{||\overline{x} - x_\star||^p, ||x - x_\star||^p\}, \end{aligned} \tag{1.30}$$

para todo $\overline{B}$ *em* $U(x, B)$*. Então, para cada* $r \in (0, 1)$*, existem constantes positivas* $\epsilon(r)$ *e* $\delta(r)$ *tais que, para* $||x_0 - x_\star|| \leq \epsilon(r)$ *e* $||B_0 - F'(x_\star)||_M < \delta(r)$*, e qualquer* $B_{k+1} \in U(x_k, B_k)$*, a seqüência*

$$x_{k+1} = x_k - B_k^{-1}F(x_k)$$

está bem definida e converge para $x_\star$. Além disso,

$$||x_{k+1} - x_\star|| \leq r||x_k - x_\star||$$

para cada $k \geq 0$, e $\{||B_k||\}$, $\{||B_k^{-1}||\}$ são uniformemente limitadas.

Demonstração: Seja $r \in (0,1)$ dado, e tome $\gamma \geq ||F'(x_\star)||$. Agora, escolha $\delta(r) = \delta$ e $\epsilon(r) = \epsilon$ tais que

$$[2\alpha_1\delta + \alpha_2]\frac{\epsilon^p}{1 - r^p} \leq \delta, \tag{1.31}$$

e para η dado por (1.29),

$$\gamma(1 + r)[K\epsilon^p + 2\eta\delta] \leq r. \tag{1.32}$$

Se necessário, sempre podemos restringir ϵ e δ de forma que $(x, B) \in N$ sempre que $||B - F'(x^\star)||_M < 2\delta$ e $||x - x_\star|| < \epsilon$. Agora, suponha que $||B_0 - F'(x_\star)||_M < \delta$ e $||x_0 - x_\star|| < \epsilon$. Então, $||B_0 - F'(x_\star)|| < \eta\delta < 2\eta\delta$ e, de (1.32), temos

$$2\gamma(1 + r)\eta\delta \leq r. \tag{1.33}$$

Vamos agora usar o seguinte lema:

Lema 1.2. *Seja $||\cdot||$ uma norma consistente, $E \in \mathbb{R}^{n\times n}$, $||I|| = 1$. Se $||E|| < 1$, então $(I - E)^{-1}$ existe e*

$$||(I - E)^{-1}|| \leq \frac{1}{1 - ||E||}$$

Se A é não-singular e $||A^{-1}(B - A)|| < 1$, então B é não singular e

$$||B^{-1}|| \leq \frac{||A^{-1}||}{1 - ||A^{-1}(B - A)||}.$$

Este resultado pode ser encontrado em [10]. Do Lema 1.2, temos que

$$||B_0^{-1}|| \leq (1 + r)\gamma.$$

Agora, do Lema 1.1, segue que:

$$\begin{aligned} ||x_1 - x_\star|| &\leq ||B_0^{-1}||[||F(x_0) - F(x_\star) - F'(x_\star)(x_0 - x_\star)|| + ||B_0 - F'(x_\star)|| \cdot ||x_0 - x_\star||] \\ &\leq \gamma(1+r)[K\epsilon^p + 2\eta\delta]||x_0 - x_\star|| \end{aligned}$$

e, de (1.32), conclui-se que $||x_1 - x_\star|| \leq r||x_0 - x_\star||$. Assim, $||x_1 - x_\star|| < \epsilon$ e, portanto, $x_1 \in D$. Para completar a prova, seguiremos com um argumento de indução.

Suponha que $||B_k - F'(x_\star)||_M \leq 2\delta$ e $||x_{k+1} - x_\star|| \leq r||x_k - x_\star||$ para $k = 0, 1, \ldots, m-1$. Segue de (1.30) que:

$$||B_{k+1} - F'(x_\star)||_M - ||B_k - F'(x_\star)||_M \leq 2\alpha_1\delta\epsilon^p r^{kp} + \alpha_2\epsilon^p r^{kp},$$

e, somando ambos os lados de $k = 0$ até $m-1$, temos que

$$||B_m - F'(x_\star)||_M \leq ||B_0 - F'(x_\star)||_M + [2\alpha_1\delta + \alpha_2]\frac{\epsilon^p}{1 - r^p},$$

o que implica, por (1.31), que $||B_m - F'(x_\star)||_M \leq 2\delta$. Agora, basta mostrarmos que $||x_{m+1} - x_\star|| \leq r||x_m - x_\star||$. Isto segue de um argumento semelhante ao que usamos para $m = 1$. De fato, como $||B_m - F'(x_\star)|| \leq 2\eta\delta$, o Lema de Banach e (1.33) implicam que

$$||B_m^{-1}|| \leq (1+r)\gamma.$$

Assim, segue do Lema 1.1 que

$$\begin{aligned} ||x_{m+1} - x_\star|| &\leq ||B_m^{-1}||[||F(x_m) - F(x_\star) - F'(x_\star)(x_m - x_\star)|| \\ &\quad + ||B_m - F'(x_\star)|| \cdot ||x_m - x_\star||] \\ &\leq \gamma(1+r)[K\epsilon^p + 2\eta\delta]||x_m - x_\star||, \end{aligned}$$

e, portanto, $||x_{m+1} - x_\star|| \leq r||x_m - x_\star||$ segue de (1.32). ■

O Teorema 1.6 garante a convergência local de qualquer método que satisfaça uma propriedade do tipo (1.30), mas não estabelece sua taxa de convergência. Vamos estabelecer no corolário abaixo que a taxa é Q-superlinear.

Corolário 1.1. *Suponha que as hipóteses do Teorema 1.6 são satisfeitas. Se alguma subseqüência de $\{||B_0 - F'(x_\star)||_M\}$ converge para zero, então $\{x_k\}$ converge Q-superlinearmente para $x_\star$.*

Demonstração: Queremos mostrar que

$$\lim_{k\to\infty} \frac{||x_{k+1} - x_\star||}{||x_k - x_\star||} = 0.$$

Seja $r \in (0,1)$ dado. Pelo Teorema 1.6 existem números $\epsilon(r)$ e $\delta(r)$ tais que, se $||B_0 - F'(x_\star)||_M \leq \delta(r)$ e $||x_0 - x_\star|| \leq \epsilon(r)$, então $||x_{k+1} - x_\star|| \leq r||x_k - x_\star||$ para todo $k \geq 0$. No entanto, como existe uma subseqüência de $\{||B_0 - F'(x_\star)||_M\}$ que converge para zero, podemos escolher $m > 0$ tal que $||B_m - F'(x_\star)||_M < \delta(r)$ e $||x_m - x_\star|| \leq \epsilon(r)$ e, assim, $||x_{k+1} - x_\star|| \leq r||x_k - x_\star||$ para cada $k \geq m$. Como $r \in (0,1)$ é arbitrário, basta tomarmos r suficientemente pequeno para que a seqüência convirja superlinearmente. ■

Acabamos de mostrar então que, se nossa fórmula de correção satisfizer a uma propriedade do tipo (1.30), análoga à propriedade de deterioração limitada utilizada na análise de convergência do método de Broyden para a resolução de sistemas não-lineares [1], nosso método possuirá convergência local. Se, além disso, ele satisfizer uma propriedade a mais (como possuir uma seqüência de $\{||B_0 - F'(x_\star)||_M\}$ convergindo para zero), então a convergência do método será superlinear. Nosso objetivo então será mostrar que o método BFGS satisfaz a propriedade dada por (1.30) e portanto possui taxa de convergência superlinear.

Para isto, vamos escrever a fórmula de correção do método BFGS de uma maneira apropriada para que possamos obter estimativas sobre seu comportamento a cada iteração.

Observação 1.7. *O desenvolvimento dos resultados de convergência seguirá os resultados clássicos obtidos em [13]. Portanto, os teoremas serão desenvolvidos no contexto de um método geral com fórmula de correção*

$$\overline{B} = B + \frac{(y - Bs)c^T + c(y - Bs)^T}{c^T s} - \frac{s^T(y - Bs)cc^T}{(c^T s)^2},$$

com $y, c, s \in \mathbb{R}^n$, *e* $c^T s \neq 0$. *Note que esta fórmula é análoga à fórmula (1.20), e já mostramos que tomando* $c = s$, *chegamos à família de Broyden, que define o método BFGS. Portanto, os resultados aqui apresentados de forma geral aplicam-se ao método BFGS.*

Lema 1.3. *Seja $B \in \mathcal{L}(\mathbb{R}^n)$ simétrica, e*

$$\overline{B} = B + \frac{(y - Bs)c^T + c(y - Bs)^T}{c^T s} - \frac{s^T(y - Bs)cc^T}{(c^T s)^2}, \tag{1.34}$$

com $y, c, s \in \mathbb{R}^n$ e $c^T s \neq 0$. Se $A \in \mathcal{L}(\mathbb{R}^n)$ for simétrica, e $M \in \mathcal{L}(\mathbb{R}^n)$ não singular e simétrica, então

$$\overline{E} = P^T E P + \frac{M(y - As)(Mc)^T}{c^T s} + \frac{Mc(y - As)^T MP}{c^T s} \tag{1.35}$$

com

$$P = I - \frac{(M^{-1}s)(Mc^T)}{c^T s},$$

e $\overline{E} = M(\overline{B} - A)M$, $E = M(B - A)M$.

Vamos agora estimar cada termo desta identidade.

Lema 1.4. *Seja $M \in \mathcal{L}(\mathbb{R}^n)$ uma matriz simétrica não singular tal que*

$$||Mc - M^{-1}s|| \leq \beta ||M^{-1}s|| \tag{1.36}$$

para algum $\beta \in [0, \frac{1}{3}]$ e para $c, s \in \mathbb{R}^n$ com $s \neq 0$. Então,

(a) $$(1 - \beta)||M^{-1}s||^2 \leq c^T s \leq (1 + \beta)||M^{-1}s||^2,$$

e, para qualquer $E \in \mathcal{L}(\mathbb{R}^n)$,

(b) $$\left|\left| E\left[I - \frac{(M^{-1}s)(M^{-1}s)^T}{c^T s}\right]\right|\right|_F \leq \sqrt{1 - \alpha\theta^2}||E||_F,$$

(c) $$\left|\left| E\left[I - \frac{M^{-1}s(Mc)^T}{c^T s}\right]\right|\right|_F \leq \left[\sqrt{1 - \alpha\theta^2} + (1 - \beta)^{-1}\frac{||Mc - M^{-1}s||}{||M^{-1}s||}\right]||E||_F,$$

com

$$\alpha = \frac{1 - 2\beta}{1 - \beta^2} \in [\frac{3}{8}, 1] \tag{1.37}$$

e

$$\theta = \frac{||EM^{-1}s||}{||E||_F||M^{-1}s||} \in [0, 1]. \tag{1.38}$$

Além disso, para todo $y \in \mathbb{R}^n$,

(d) $$\left|\left| \frac{(y - As)(Mc)^T}{c^T s}\right|\right|_F \leq 2\frac{||y - As||}{||M^{-1}s||}.$$

Demonstração: Para provar (a), note que

$$c^T s = (Mc)^T (M^{-1}s) = (Mc - M^{-1}s)^T M^{-1}s + ||M^{-1}s||^2,$$

e, de (1.36) e da desigualdade de Cauchy-Schwartz, temos

$$|(Mc - M^{-1}s)^T M^{-1}s| \leq \beta ||M^{-1}s||^2.$$

Para provarmos a parte (b), observamos que, da parte (a), temos $c^T s > 0$. Mas $||Q||_F^2$ é o traço de $(Q^T Q)$ e, portanto, cálculos diretos mostram que

$$||E[I - uv^T]||_F^2 = ||E||_F^2 - 2v^T E^T Eu + ||Eu||^2 ||v||^2,$$

e, em particular,

$$\left|\left|E\left[I - \frac{(M^{-1}s)(M^{-1}s)^T}{c^T s}\right]\right|\right|_F^2 = ||E||_F^2 + (-2c^T s + ||M^{-1}s||^2)\frac{||EM^{-1}s||^2}{(c^T s)^2}.$$

Agora, pelas estimativas feitas em (a),

$$\begin{aligned}\left|\left|E\left[I - \frac{(M^{-1}s)(M^{-1}s)^T}{c^T s}\right]\right|\right|_F^2 &\leq ||E||_F^2 - \left(\frac{1-2\beta}{1-\beta}\right)\frac{||EM^{-1}s||^2}{c^T s} \\ &\leq ||E||_F^2 - \alpha\left(\frac{||EM^{-1}s||}{||M^{-1}s||}\right)^2,\end{aligned}$$

que, usando (1.38) e fatorando $||E||_F^2$, se transforma em (b). Devido a (b), para provarmos a parte (c) só precisamos mostrar que

$$\left|\left|E\left[\frac{M^{-1}s(M^{-1}s - Mc)^T}{c^T s}\right]\right|\right| \leq (1-\beta)^{-1}\left(\frac{||Mc - M^{-1}s||}{||M^{-1}s||}\right)||E||_F.$$

Mas, claramente,

$$\left|\left|\frac{M^{-1}s(M^{-1}s - Mc)^T}{c^T s}\right|\right|_F = \frac{||M^{-1}s||\,||M^{-1}s - Mc||}{c^T s},$$

e o resultado segue de (a). A parte (d) segue de um raciocínio similar. De fato,

$$\left|\left|\frac{(y - As)(Mc)^T}{c^T s}\right|\right|_F = \frac{||y - As||\,||Mc||}{c^T s}(||Mc - M^{-1}s|| + ||M^{-1}s||)$$

e (1.36) implica que

$$\left|\left|\frac{(y-As)(Mc)^T}{c^Ts}\right|\right|_F \leq (1+\beta)||y-As||\left(\frac{||M^{-1}s||}{c^Ts}\right).$$

Agora o resultado segue de (a) já que $1+\beta \leq 2(1-\beta)$. ■

Usando os resultados deste lema, é fácil mostrar que a fórmula de correção (1.34) satisfaz as hipóteses do Teorema 1.6 se c e s forem escolhidos apropriadamente. Para estimar P^TEP, primeiramente note que, da parte (c) do Lema 1.4,

$$||P^TEP||_F \leq \left(1+(1-\beta)^{-1}\frac{||Mc-M^{-1}s||}{||M^{-1}s||}\right)||P^TE||_F.$$

Mas $||P^TE||_F = ||E^TP||_F$, e assim, novamente pelo Lema 1.4, como E é simétrica,

$$||E^TP||_F \leq \left(\sqrt{1-\alpha\theta^2}+(1-\beta)^{-1}\frac{||Mc-M^{-1}s||}{||M^{-1}s||}\right)||E||_F$$

em que

$$\theta = \frac{||EM^{-1}s||}{||E||_F||M^{-1}s||}.$$

Juntando as observações acima com a desigualdade (1.36) temos que:

$$||P^TEP||_F \leq \left(\sqrt{1-\alpha\theta^2}+\left(\frac{5}{2}\right)(1-\beta)^{-1}\frac{||Mc-M^{-1}s||}{||M^{-1}s||}\right)||E||_F.$$

Os outros termos de (1.35) podem ser estimados mais facilmente. De fato,

$$\left|\left|\frac{M(y-As)(Mc)^T}{c^Ts}\right|\right|_F \leq 2||M||_F\frac{||y-As||}{||M^{-1}s||},$$

e então, fazendo $E=I$ na parte (c) do Lema 1.4 temos

$$||P||_F \leq [1+(1-\beta)^{-1}\beta]\sqrt{n} \leq 2\sqrt{n}.$$

Assim, essas estimativas resultam no seguinte resultado:

Lema 1.5. *Seja $M \in \mathcal{L}(\mathbb{R}^n)$ uma matriz simétrica não singular satisfazendo a desigualdade (1.36) para algum $\beta \in [0, \frac{1}{3}]$ e para $c, s \in \mathbb{R}^n$ com $s \neq 0$. Seja $B \in \mathcal{L}(\mathbb{R}^n)$ simétrica e defina $\overline{B}$ por (1.34) com $y \in \mathbb{R}^n$. Se $||\cdot||_M$ for a norma matricial definida por $||Q||_M = ||MQM||_F$, então para qualquer matriz $A \in \mathcal{L}(\mathbb{R}^n)$ simética com*

$B \neq A$,

$$||\overline{B} - A||_M \leq \left[\sqrt{1-\alpha\theta^2} + \left(\frac{5}{2}\right)(1-\beta)^{-1}\frac{||Mc - M^{-1}s||}{||M^{-1}s||}\right]||B - A||_M + 2(1+2\sqrt{n})||M||_F\frac{||y - As||}{||M^{-1}s||} \quad (1.39)$$

com

$$\alpha = \frac{1-2\beta}{1-\beta^2} \in \left[\frac{3}{8}, 1\right]$$

e

$$\theta = \frac{||M[B-A]s||}{||B-A||_M||M^{-1}s||}.$$

Demonstração: O resultado segue das estimativas feitas anteriormente e da definição da norma $||\cdot||_M$. ■

Para que possamos aplicar estes resultados, vamos supor que $F : \mathbb{R}^n \to \mathbb{R}^n$ satisfaz as hipóteses do Lema 1.1 e que, além disso, $F(x_\star) = 0$. Vamos escolher o vetor c de modo que, para todo (x, B) em uma vizinhança N' de $(x_\star, F'(x_\star))$,

$$\frac{||Mc - M^{-1}s||}{||M^{-1}s||} \leq \mu_1||s||^p, \quad s \neq 0. \quad (1.40)$$

para alguma constante $\mu_1 \geq 0$, e para alguma matriz simétrica e não singular $M \in \mathcal{L}(\mathbb{R}^n)$, ambos independentes de c e s.

Além disso, $F'(x_\star)$ e todas as matrizes de correção devem ser simétricas; portanto, $F'(x_\star)$ deve pertencer ao subespaço de dimensão $n(n+1)/2$ das matrizes simétricas de ordem n. Então, quando nos referirmos a uma vizinhança de $F'(x_\star)$, estaremos considerando uma vizinhança nesse subespaço de $\mathcal{L}(\mathbb{R}^n)$.

Teorema 1.7. *Seja F satisfazendo as hipóteses do Teorema 1.6 e suponha além disso que $F'(x_\star)$ seja simétrica. Se a desigualdade (1.40) for satisfeita para alguma constante $\mu_1 \geq 0$, para alguma matriz simétrica não singular $M \in \mathcal{L}(\mathbb{R}^n)$ e para todo (x, B) em uma vizinhança N' de $(x_\star, F'(x_\star))$, então a função de correção $U(x, B) = \{\overline{B}\}$ com*

$$\overline{B} = B + \frac{(y - Bs)c^T + c(y - Bs)^T}{c^T s} - \frac{s^T(y - Bs)cc^T}{(c^T s)^2}$$

e B simétrica, estará bem definida em uma vizinhança N de $(x_\star, F'(x_\star))$, e a iteração correspondente

$$x_{k+1} = x_k - B_k^{-1}F(x_k)$$

com $B_{k+1} \in U(x_k, B_k)$, convergirá local e Q-superlinearmente para $x_\star$.

Demonstração: Suponha que $0 < \mu_1 ||s||^p \leq \frac{1}{3}$ e façamos $A = F'(x_\star)$ no Lema 1.5. Então, de (1.40) temos que

$$\begin{aligned} ||\overline{B} - F'(x_\star)||_M &\leq [[\sqrt{1-\alpha\theta^2} + \left(\frac{5}{2}\right)(1-\beta)^{-1}\mu_1||s||^p]||B - F'(x_\star)||_M \\ &+ 2(1+2\sqrt{n})||M||_F \frac{||y - F'(x_\star)||}{||M^{-1}s||}. \end{aligned}$$

Mas, para qualquer (x, B) em N',

$$||s|| \leq 2\max\{||\overline{x} - x_\star||, ||x - x_\star||\},$$

e se $\overline{x} \in D$, pelo Lema 1.1 temos que

$$||y - F'(x_\star)s|| \leq K\max\{||\overline{x} - x_\star||^p, ||x - x_\star||^p\}||s||.$$

Assim, se $\overline{x} \in D$,

$$\begin{aligned} ||\overline{B} - F'(x_\star)||_M &\leq [\sqrt{1-\alpha\theta^2} + \alpha_1\max\{||\overline{x} - x_\star||^p, ||x - x_\star||^p\}||B - F'(x_\star)||_M \\ &+ \alpha_2\max\{||\overline{x} - x_\star||^p, ||x - x_\star||^p\} \end{aligned} \tag{1.41}$$

com $\alpha_1 = \left(\frac{5}{2}\right)(1-\beta)^{-1}\mu_1 2^p$, $\alpha_2 = 2K(1+2\sqrt{n})||M||_F||M||$ e, se $B \neq F'(x_\star)$,

$$\theta = \frac{||M[B - F'(x_\star)]s||}{||B - F'(x_\star)||_M||M^{-1}s||}$$

Agora, observe que, se existir uma subseqüência de $\{B_k\}$ convergindo para $F'(x_\star)$, o resultado segue do Corolário 1.1; senão, $\{||B_k - F'(x_\star)||_M\}$ é arbitrariamente maior do que zero. Agora, como $\sqrt{1-\alpha} \leq 1 - (\frac{\alpha}{2})$, então de (1.41) temos que

$$\begin{aligned} (3/16)\theta_k^2||B_k - F'(x_\star)||_M &\leq ||B_k - F'(x_\star)||_M - ||B_{k+1} - F'(x_\star)||_M \\ &+ \max\{||x_{k+1} - x_\star||^p, ||x_k - x_\star||^p\}[\alpha_1||B_k - F'(x_\star)||_M + \alpha_2], \end{aligned}$$

e portanto

$$\sum_{k=1}^{\infty} \theta_k^2||B_k - F'(x_\star)||_M < +\infty.$$

Mas $\{||B_k - F'(x_\star)||\}$ é limitada superiormente, e assim,

$$\lim_{k\to\infty} \frac{||[B_k - F'(x_\star)]s_k||}{||s_k||} = 0. \tag{1.42}$$

Agora, observe que $B_k s_k = -F(x_k)$. Logo,

$$||[B_k - F'(x_\star)]s_k|| \geq ||F(x_{k+1})|| - ||F(x_{k+1}) - F(x_k) - F'(x_\star)s_k||.$$

Portanto, como $||x_{k+1} - x_\star|| \leq ||x_k - x_\star||$, o Lema 1.1 implica que

$$||[B_k - F'(x_\star)]s_k|| \geq \frac{1}{\rho}||x_{k+1} - x_\star|| - K||x_k - x_\star||^p||s_k||.$$

Mas $||s_k|| \leq 2||x_k - x_\star||$ e assim

$$\frac{||[B_k - F'(x_\star)]s_k||}{||s_k||} \geq (2\rho)^{-1}\frac{||x_{k+1} - x_\star||}{||x_k - x_\star||} - K||x_k - x_\star||^p$$

que, junto com (1.42), diz que

$$\lim_{k\to\infty} \frac{||x_{k+1} - x_\star||}{||x_k - x_\star||} = 0,$$

que é exatamente o que queríamos provar. ■

O desempenho do método BFGS, assim como seus resultados teóricos são bastante influenciados pela qualidade das aproximações. Estas, por sua vez, dependem muito da aproximação para a Hessiana no ponto inicial. Veremos no Capítulo 4 que esse fato também é muito relevante na formulação do método L-BFGS. Nossa estrátégia de aproximação está baseada na fatoração incompleta de Cholesky, que descrevemos no capítulo que se segue.

Capítulo 2

Fatoração de Cholesky Incompleta como Precondicionador de Sistemas Lineares

A fatoração de Cholesky incompleta pode ser útil como precondicionador para a resolução de sistemas de equações lineares de grande porte, cujas matrizes de coeficientes são, em geral, esparsas.

2.1 Fatoração Incompleta no Gradiente Conjugado

A fatoração incompleta é uma ferramenta muito utilizada na resolução de sistemas lineares como precondicionador de algum método iterativo, como por exemplo o Gradiente Conjugado (CG), e suas generalizações. Descreveremos aqui brevemente sua aplicação no método do Gradiente Conjugado. Os detalhes podem ser encontrados em vários textos; ver [5], [8], [9] e [10].

Considere um sistema linear

$$Ax = b,$$

com $A \in \mathbb{R}^{n\times n}$ e $x, b \in \mathbb{R}^n$. Queremos encontrar $x_\star$ tal que $x_\star = A^{-1}b$. Observamos que, se A for simétrica positiva definida, minimizar a função

$$\phi(x) = \frac{1}{2}x^T Ax - x^T b$$

implica escolher $x_\star$ tal que $\nabla\phi(x_\star) = 0$. Ou seja, $x_\star$ deve ser solução de

$$Ax = b.$$

Vamos minimizar a função ϕ através de uma das estratégias mais simples, o método do gradiente, que já discutimos brevemente. Assim, nossa direção de busca na iteração atual é a direção em que ϕ decresce mais rapidamente: $-\nabla\phi = b - Ax_c$, em que x_c é o ponto atual. Chamaremos de $r_c = b - Ax_c$ de resíduo de x_c. Note que, se o resíduo for diferente de zero, então existirá α positivo tal que $\phi(x_c + \alpha r_c) < \phi(x_c)$. Assim, no método do gradiente com busca linear exata, tomamos $\alpha = r_c^T r_c / r_c^T A r_c$, minimizando assim

$$\phi(x_c + \alpha r_c) = \phi(x_c) - \alpha r_c^T r_c + \frac{1}{2}\alpha^2 r_c^T A r_c.$$

No entanto, sabemos dos problemas do método do gradiente; ele pode convergir muito lentamente para a solução, ou por vezes nem convergir. Portanto, queremos acelerar a convergência deste método. Para isto, vamos minimizar ϕ em uma seqüência de direções $\{p_1, p_2, \ldots\}$ que não correspondem necessariamente aos resíduos $\{r_0, r_1, \ldots\}$. Pode-se mostrar que $\alpha = \alpha_k = p_k^T r_{k-1} / p_k^T A p_k$ minimiza $\phi(x_{k-1} + \alpha p_k)$. Desta forma,

$$\phi(x_{k-1} + \alpha_k p_k) = \phi(x_{k-1}) - \frac{1}{2}\frac{(p_k^T r_{k-1})^2}{p_k^T A p_k}.$$

Assim, para garantirmos que o valor de ϕ terá um decréscimo, precisamos exigir que p_k *não* seja ortogonal a r_{k-1}. Além disso, gostaríamos que x_k fosse facilmente calculado a cada passo, a partir de x_{k-1}. Note que, segundo esta formulação,

$$x_k \in x_0 + \text{span}\{p_1, \ldots, p_k\} \equiv \{x_0 + \gamma_1 p_1 + \ldots + \gamma_k p_k : \gamma_i \in \mathbb{R}\}.$$

Logo, x_k é da forma

$$x_k = x_0 + P_{k-1} y + \alpha p_k,$$

com $P_{k-1} = [p_1, \ldots, p_{k-1}]$, $y \in \mathbb{R}^{k-1}$ e $\alpha \in \mathbb{R}$. Assim,

$$\phi(x_k) = \phi(x_0 + P_{k-1} y) + \alpha y^T P_{k-1}^T A p_k + \frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0.$$

Se $p_k \in \text{span}\{Ap_1, \ldots, Ap_{k-1}\}^\perp$, então o termo $\alpha y^T P_{k-1}^T A p_k$ é zero, e podemos agora dividir o problema de minimizar $\phi(x_k)$ em dois problemas distintos, um para y e outro para α:

$$\begin{aligned}
\min_{x_k \in x_0 + \text{span}\{p_1,\ldots,p_k\}} \phi(x_k) &= \min_{y,\alpha} \phi(x_0 + P_{k-1}y + \alpha p_k) \\
&= \min_{y,\alpha} \left(\phi(x_0 + P_{k-1}y) + \frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0 \right) \\
&= \min_{y} \phi(x_0 + P_{k-1}y) + \min_{\alpha} \left(\frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0 \right).
\end{aligned}$$

Agora, note que se y_{k-1} resolve o primeiro problema de minimização, então $x_{k-1} = x_0 + P_{k-1}y_{k-1}$ minimiza ϕ em $x_0 + \text{span}\{p_1, \ldots, p_{k-1}\}$. Além disso, a solução para o problema de minimização em α é $\alpha_k = p_k^T r_0 / p_k^T A p_k$. Mas, pelas propriedades que mostramos, é fácil ver que $p_k^T r_0 = p_k^T r_{k-1}$. Assim, desde que tenhamos a direção p_k, $x_k = x_{k-1} + \alpha_k p_k$ e assim podemos resolver o sistema linear. No entanto, ainda precisamos escolher a direção $p_k \in \text{span}\{Ap_1, \ldots, Ap_{k-1}\}^\perp$ tal que $p_k^T r_{k-1} \neq 0$, pois assim $\alpha_k \neq 0$.

Para escolher a melhor direção, vamos procurar p_k tal que

$$p_k = \min_p ||p - r_{k-1}||_2,$$

com $p \in \text{span}\{Ap_1, \ldots Ap_{k-1}\}^\perp$. Assim, queremos resolver este problema particular de minimização de forma eficiente em um algoritmo. É possível mostrar que, para resolver este problema, p_k deve ser uma combinação linear simples de p_{k-1} e de r_{k-1}, ou seja,

$$p_k = r_{k-1} + \beta_k p_{k-1},$$

com

$$\beta_k = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}}.$$

Assim, usando as propriedades dos resíduos, o algoritmo do gradiente conjugado (CG) toma a forma seguinte.

Algoritmo 2.1. Gradiente Conjugado

$k = 0$

$r_0 = b - Ax_0$

Enquanto $r_k \neq 0$ faça

$k = k + 1$

Se $k = 1$ então

$p_1 = r_0$

Senão

$\beta_k = r_{k-1}^T r_{k-1} / r_{k-1}^T r_{k-2}$

$p_k = r_{k-1} + \beta_k p_{k-1}$

Fim Se

$\alpha_k = r_{k-1}^T r_{k-1} / p_k^T A p_k$

$x_k = x_{k-1} + \alpha_k p_k$

$r_k = r_{k-1} - \alpha_k A p_k$

Fim Enquanto

$x = x_k$.

O que fizemos foi minimizar o erro em uma seqüência de subespaços V_k de dimensão crescente construídos recursivamente, adicionando um novo vetor $A^k r_0$ à base do supespaço anterior V_{k-1}, ou seja,

$$V_k = V_{k-1} \oplus \{A^k r_0\}, k = 1, 2, 3, \ldots$$

com $V_0 = \{r_0\}$. Assim, V_k é gerado pelos vetores $r_0, Ar_0, \ldots, A^k r_0$ que compõem o chamado subespaço de Krylov associado a A e r_0.

Como os vetores de Krylov eventualmente geram o espaço inteiro, este algoritmo tem a propriedade de terminação finita. No entanto, devido a arredondamentos e erros acumulados nas iterações, pode ser necessário acelerar de alguma forma a convergência do método, principalmente nos casos em que n é grande. Assim, surge a idéia do *precondicionamento*.

Em [7], mostra-se que o algoritmo do gradiente conjugado produz uma seqüência $\{x_k\}$ que satisfaz

$$||x - x_k||_A \leq 2||x - x_0||_A \left(\frac{\sqrt{cond(A)} - 1}{\sqrt{cond(A)} + 1} \right)^k.$$

Portanto, é interessante que $cond(A) \approx 1$. Se tivermos $cond(A)$ grande,

a convergência pode ser bastante lenta.

Queremos então aplicar o algoritmo do gradiente conjugado (CG) a um sistema transformado

$$\tilde{A}\tilde{x} = \tilde{b}$$

com $\tilde{A} = C^{-1}AC^{-1}$, $\tilde{x} = Cx$ e $\tilde{b} = C^{-1}b$, em que C é simétrica positiva definida. Queremos escolher C de forma que $\tilde{A}$ seja uma matriz bem condicionada. No entanto, na implementação, queremos evitar referências à matriz C^{-1}; definimos então $\tilde{p}_k = Cp_k$, $\tilde{x}_k = Cx_k$, e $\tilde{r}_k = C^{-1}r_k$. Assim, é fácil ver que, se definirmos $M = C^2$ (que também será positiva definida) e resolvermos $Mz_k = r_k$ para z_k, teremos o seguinte algoritmo:

Algoritmo 2.2. Gradiente Conjugado Precondicionado

$k = 0$
$r_0 = b - Ax_0$
Enquanto $(r_k \neq 0)$ faça
  Resolva $Mz_k = r_k$
  $k = k + 1$
  Se $k = 1$ então
    $p_1 = z_0$
  Senão
    $\beta_k = r_{k-1}^T z_{k-1} / r_{k-2}^T z_{k-2}$
    $p_k = z_{k-1} + \beta_k p_{k-1}$
  Fim Se
  $\alpha_k = r_{k-1}^T z_{k-1} / p_k^T A p_k$
  $x_k = x_{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k A p_k$
Fim Enquanto
$x = x_k$.

Observação 2.1. *Salientamos que as propriedades do resíduo e das direções p_k continuam a ser satisfeitas com esta transformação do problema. Além disso, os denominadores $r_{k-2}^T z_{k-2} = z_{k-2}^T M z_{k-2}$ são sempre não nulos, pois M é positiva definida. Para garantirmos a eficiência do método, é importante que o sistema $Mz = r$ seja resolvido de forma rápida e simples.*

Uma das maneiras de encontrar a matriz C é através da fatoração de Cholesky incompleta de A. Sabemos que existe uma matriz ortogonal Q tal que

$C = QH^T$, por exemplo, H^T sendo o fator triangular superior de uma fatoração QR de C. Assim, se C é simétrica positiva definida, $C = C^T$ e, logo,

$$\begin{aligned}\tilde{A} &= C^{-1}AC^{-1} = C^{-T}AC^{-1} \\ &= (HQ^T)^{-1}A(QH^T)^{-1} = Q(H^{-1}GG^TH^{-T})Q^T \approx I, \text{ se } H \approx G,\end{aligned}$$

em que $A = GG^T$ é a fatoração de Cholesky completa de A. Portanto, quanto mais H aproximar o fator G, melhor o número de condição de $\tilde{A}$, e assim melhor o desempenho do método do Gradiente Conjugado. Assim, o precondicionador é tomado como $M = HH^T$, calculado através do Algoritmo 2.3 (ver adiante) ou alguma de suas variações.

2.2 Fatoração de Cholesky Incompleta

Sabemos que toda matriz não singular $A \in \mathbb{R}^{n\times n}$ pode ser decomposta de forma iterativa em $A = LU$, com L triangular inferior da forma

$$L = \begin{bmatrix} 1 & & & & & & 0 \\ a_{21}^{(1)}/a_{11}^{(1)} & 1 & & & & & \\ \vdots & \vdots & & \ddots & & & \\ a_{m1}^{(1)}/a_{11}^{(1)} & a_{m2}^{(2)}/a_{22}^{(2)} & \cdots & a_{m,t-1}^{(t-1)}/a_{t-1,t-1}^{(t-1)} & 0 & \cdots & 1 \end{bmatrix}_{n\times n}$$

em que $a_{ij}^{(k)}$ representa o elemento na posição i, j da matriz, e k representa a iteração na qual o termo correspondente foi calculado. U triangular superior tem entradas definidas por

$$a_{ij}^{(s+1)} = a_{ij}^{(s)} - a_{is}^{(s)}{a_{ss}^{(s)}}^{-1}a_{sj}^{(s)}.$$

Quando fatoramos uma matriz esparsa $A \in \mathbb{R}^{n\times n}$ usando a eliminação Gaussiana tradicional, durante o processo de fatoração de A, algumas entradas que eram nulas tornam-se não nulas. Estas entradas são chamadas *fill-in*. Por exemplo, se tivermos $a_{ij} = 0$, mas a_{ik} e a_{kj} diferentes de zero, no passo $k + 1$ do algoritmo de fatoração substituiremos uma entrada nula por

$$a_{ij}^{(k+1)} = -a_{ik}^{(k)}a_{kk}^{(k)-1}a_{kj}^{(k)}$$

Podemos observar o que acontece na figura:

$$\begin{pmatrix} \diamondsuit & & \times & & \times \\ & \times & & & \times \\ \otimes & & \times & \times & \square \\ & & \times & \times & \times \\ \otimes & \times & \square & \times & \times \end{pmatrix}$$

Para anular os elementos denotados por $\otimes$, criam-se entradas não nulas $\square$ em posições anteriormente nulas. Infelizmente, a tendência é que o *fill-in* aumente a cada iteração, destruindo o padrão de esparsidade da matriz original, já que as entradas de *fill-in* criadas a cada passo tendem a criar novos *fill-in* em iterações posteriores.

A idéia de um método de fatoração incompleta é descartar estas entradas geradas como *fill-in* que, ou geram elementos não nulos de forma a diminuir o grau de esparsidade da matriz, ou são pequenos demais em relação aos elementos diagonais da matriz. Desta forma, definimos uma fatoração de A na forma

$$A = \tilde{L}\tilde{U} - R, \tag{2.1}$$

em que R é a matriz de erro da fatoração. Se A for positiva definida, podemos fazer com que $C = \tilde{L}\tilde{U}$ também o seja, pois a escolha da proximidade entre A e sua fatoração incompleta pode ser feita pelo usuário. Assim, mantemos as características principais da matriz A. Além disso, no nosso caso, será útil trabalharmos com matrizes simétricas positivas definidas; portanto, queremos obter uma decomposição incompleta baseada na decomposição de Cholesky, da forma $C = \tilde{L}\tilde{L}^T$.

Como podemos impor um padrão de esparsidade arbitrário nas matrizes, a construção de $\tilde{L}$ para obtermos $C = \tilde{L}\tilde{L}^T$ e a solução dos sistemas envolvendo C que ocorrem a cada iteração podem ser realizados com menos esforço computacional do que na construção e solução dos fatores correspondentes da fatoração completa. Podemos, assim, controlar o *fill-in* gerado a cada iteração, especificando *a priori* posições onde o *fill-in* será aceito, ou aceitando entradas geradas por *fill-in* somente quando seu valor absoluto for satisfatório comparado às entradas diagonais da matriz, por exemplo.

Algoritmo da Fatoração Incompleta Considere uma matriz $A = [a_{ij}] \in \mathbb{R}^{n\times n}$, que na prática será esparsa. Similarmente ao método da fatoração por eliminação Gaussiana tradicional, um método de fatoração incompleta trabalha recursivamente. O próximo pivô é escolhido a cada iteração, e na implementação tradicional, as entradas da coluna do pivô são eliminadas através de operações com as linhas. No entanto, em contraste com o método de fatoração completa, não aceitaremos entradas em L ou em U que sejam não nulas em posições que não pertencem a uma estrutura de esparsidade $\mathcal{S}$, que é um subconjunto de $\{(i,j)/1 \leq i \leq n, 1 \leq j \leq n\}$. Esta estrutura de esparsidade pode ser escolhida antes da fatoração ser realizada, ou durante a fatoração, de forma que, no estágio r da eliminação, são descartadas as entradas que forem suficientemente pequenas para satisfazer

$$|a_{ij}^{(r+1)}| \leq c|a_{ii}^{(r+1)}a_{jj}^{(r+1)}|^{\frac{1}{2}}, \;\; 0 < c < 1.$$

Note que $c = 0$ nos dá uma fatoração completa, e $c = 1$ resulta numa matriz diagonal se A for simétrica positiva definida.

Seja $A^{(r)} = [a_{ij}^{(r)}]$ a matriz no r-ésimo passo da eliminação, com $A^{(1)} = A$ e $a_{r,r}^{(r)}$ o pivô no passo atual. Naturalmente, devemos supor que $a_{rr}^{(r)} \neq 0$. Para tal, é comum permutar linhas ou colunas, ou ambos, até que um pivô não nulo seja encontrado. No entanto, isto pode alterar a estrutura de esparsidade da matriz. Portanto, não utilizaremos permutações entre linhas ou colunas.

Como vimos anteriormente, o passo básico na eliminação Gaussiana é calcular

$$a_{ij}^{(r+1)} = a_{ij}^{(r)} - a_{ir}^{(r)}{a_{rr}^{(r)}}^{-1}a_{rj}^{(r)}, \;\; r+1 \leq i \leq n, r+1 \leq j \leq n. \tag{2.2}$$

As entradas $a_{ij}^{(r+1)}$ da matriz $A^{(r+1)}$ são da forma:

(a) Definidas por (2.2), para $r+1 \leq i, j, \leq n$;

(b) $a_{ij}^{(r+1)} = a_{ij}^{(r)}$, para $1 \leq i \leq r, 1 \leq j \leq n$, e $1 \leq i \leq n, 1 \leq j \leq r-1$;

(c) $a_{ir}^{(r+1)} = 0, r+1 \leq i \leq n$.

No entanto, na fatoração incompleta, as entradas $a_{ij}^{(r+1)}$ são aceitas somente se $(i,j) \in \mathcal{S}^{(r)}$, em que $\mathcal{S}^{(r)} \subseteq \mathcal{S}$ é um conjunto de índices que no estágio r define a posição onde entradas não nulas são aceitas, com $i, j \geq r+1$. Por simplicidade, vamos aqui considerar somente padrões de esparsidade simétricos, ou seja, se $(i,j) \in \mathcal{S}, 1 \leq i \leq n$, então $(j,i) \in \mathcal{S}$. O algoritmo da fatoração incompleta

para o caso geral, ou seja, $A = \tilde{L}\tilde{U} - R$ (a matriz A não é necessariamente simétrica positiva definida) toma então a seguinte forma:

Algoritmo 2.3. Fatoração Incompleta $A = \tilde{L}\tilde{U} - R$**:**

Para $r = 1$ até $n - 1$ faça
  $d = 1/a(r, r)$
  Para $i = r + 1$ até n faça
    Se $(i, r) \in \mathcal{S}$ então
      $e = a(i, r) \cdot d$
      $a(i, r) = e$
      Para $j = r + 1$ até n faça
        Se $(i, j) \in \mathcal{S}$ e $(r, j) \in \mathcal{S}$ então
          $a(i, j) = a(i, j) - e \cdot a(r, j)$
        Fim Se
      Fim Para
    Fim Se
  Fim Para
Fim Para

Se $a_{rr}^{(r)} \neq 0$, $r = 1, 2, \ldots, n - 1$, podemos continuar o algoritmo até o estágio final, com $r = n - 1$, para formar uma matriz de fatoração incompleta $\tilde{L}\tilde{U}$, com L triangular inferior com entradas $a_{ir}^{(r)}/a_{rr}^{(r)}$, $i = r, r + 1, \ldots, n$, $r = 1, 2, \ldots, n$, e U triangular superior com entradas $a_{rj}^{(r)}$, $j = r, r + 1, \ldots, n$, $r = 1, 2, \ldots, n$. Estas matrizes têm a mesma forma das matrizes da fatoração completa, porém com menor número de entradas não nulas. No nosso caso particular, $U = L^T$, portanto o Algoritmo 2.3 pode ser aplicado nesta forma, ou simplificado para o caso de matrizes simétricas.

Exemplo 2.1. *Vamos aplicar o algoritmo da fatoração incompleta de Cholesky como precondicionador para a solução do sistema gerado pela aproximação por diferenças finitas, usando 5 pontos, do Laplaciano negativo na malha de 16 pontos gerada pela discretização do retângulo* $[0, 1] \times [0, 1]$*. Para a construção da fatoração incompleta, não vamos aceitar* fill-in*, ou seja,* $\mathcal{S}$ *é constante em todas as iterações e idêntico ao padrão de esparsidade da matriz original* A*.*

Para resolver o sistema $Ax = b$*, em que* A *é a matriz descrita acima e* b *é o lado direito definido de forma que* $A \cdot 1 = b$*, em que* $1 = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$*,*

primeiro utilizamos o método do gradiente conjugado tradicional (CG), e depois utilizamos a fatoração de Cholesky incompleta de A como precondicionador para o método do gradiente conjugado (PCG). No gráfico abaixo, a linha pontilhada representa os resultados obtidos com a primeira estratégia, enquanto que a linha contínua representa os resultados obtidos para o sistema modificado pelo precondicionamento.

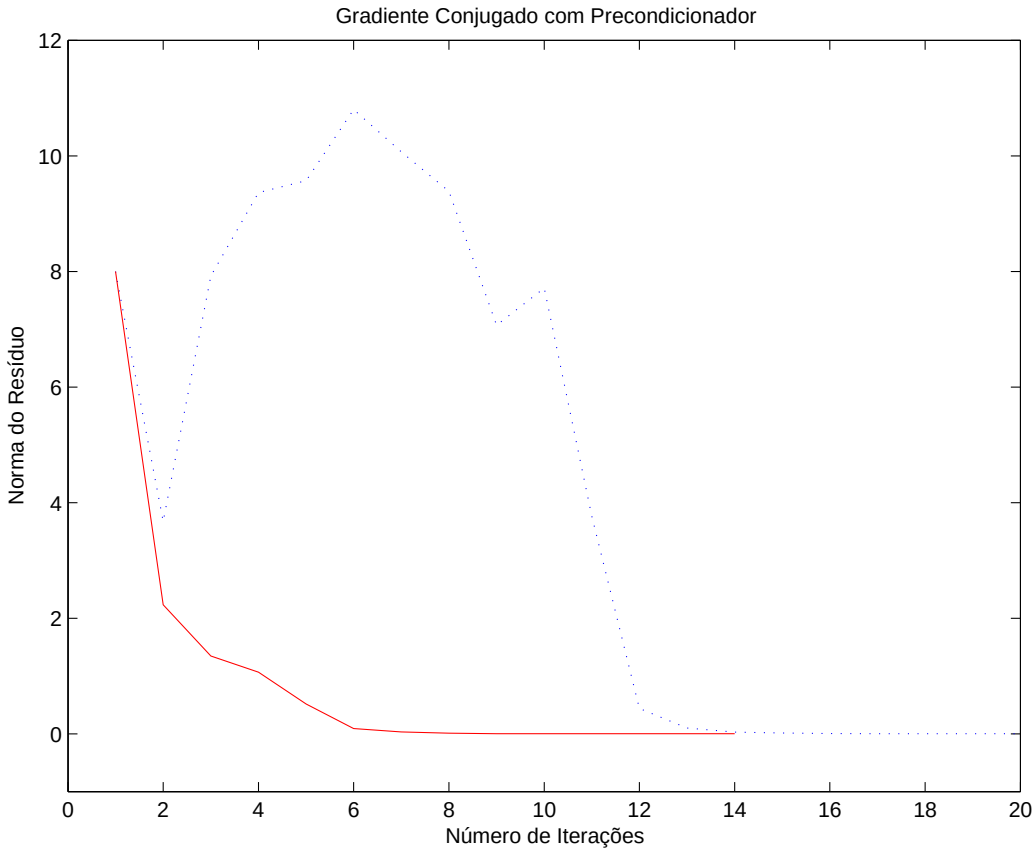

Figura 2.1: Resultados para o CG e CG com precondicionamento

Observamos que o método do gradiente conjugado tradicional não convergiu com a tolerância desejada dentro do número máximo de iterações (20), enquanto que o PCG convergiu em 14 iterações.

Nas próximas figuras, podemos observar o padrão de esparsidade da matriz A na Figura 2.2, o padrão de esparsidade da sua fatoração incompleta na Figura 2.3 e o padrão resultante depois de realizarmos a fatoração completa na Figura 2.4. Em todos os casos, nz representa o número total de elementos não nulos de cada matriz.

O método da fatoração incompleta pode ser modificado de várias ma-

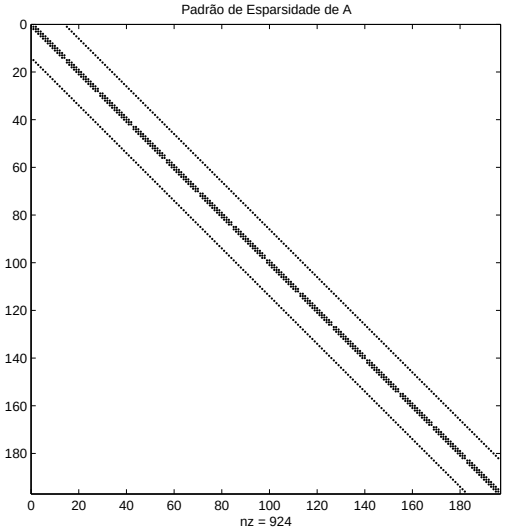

Figura 2.2: Padrão de Esparsidade de A

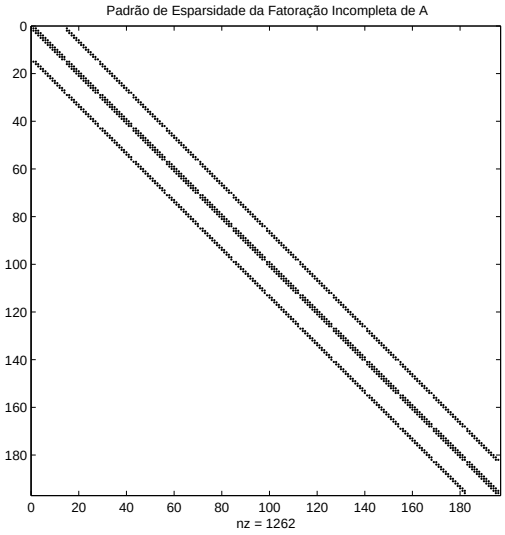

Figura 2.3: Padrão de Esparsidade da Fatoração Incompleta de A

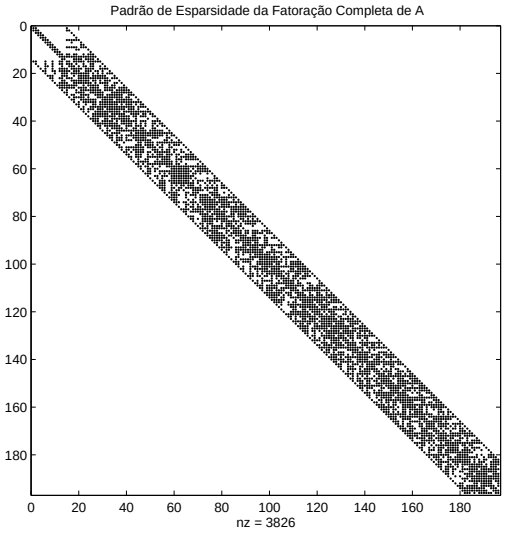

Figura 2.4: Padrão de Esparsidade da Fatoração Completa de A

neiras. Por exemplo, podemos somar as entradas descartadas ao elemento diagonal da mesma linha, preservando a soma da linha da matriz original, ou seja, $\tilde{L}\tilde{L}^T e = Ae$, $e = (1, 1, \ldots, 1)^T$. Assim, o laço j do Algoritmo 2.3 toma a forma seguinte:

Algoritmo 2.4. Laço j modificado

Se $(r, j) \in \mathcal{S}$ então
  Se $(i, j) \in \mathcal{S}$ então
    $a(i, j) = a(i, j) - e \cdot a(r, j)$
  Senão
    $a(i, i) = a(i, i) - e \cdot a(r, j)$
  Fim Se
Fim Se

A versão mais geral desta modificação tem o último passo da forma

$$a(i, i) = a(i, i) - \omega \cdot e \cdot a(r, j), \quad \text{se } (i, j) \notin \mathcal{S}.$$

Aqui, ω é um parâmetro, $\omega \leq 1$. Para $\omega = 0$ temos a fatoração dada pelo algoritmo 2.3, e para $\omega = 1$ temos a versão modificada dada pelo algoritmo 2.4.

Finalmente, podemos modificar a entrada do pivô, adicionando um número positivo, geralmente pequeno, de modo a garantir que as entradas dos pivôs terão valor absoluto suficientemente grande. Assim, o segundo passo do algoritmo

2.3 toma a forma

$$d = (a_{rr}^{(r)} + \alpha_r)^{-1},$$

com α_r suficientemente grande.

Resta definir como escolher o conjunto $\mathcal{S}$ que define a esparsidade da matriz A. A escolha mais comum é

$$\mathcal{S} = \mathcal{S}_0 = \{i, j / a_{ij} \neq 0\}.$$

Vamos definir a esparsidade de ordem q da seguinte maneira. Seja $\mathcal{R}(q)$ o conjunto de esparsidade definido pelas posições das entradas geradas por *fill-in* durante a fatoração incompleta baseada no conjunto de esparsidade $\mathcal{S}(q)$, e seja

$$\mathcal{S}(q+1) = \mathcal{S}(q) \cup \mathcal{R}(q), \;\; q = 0, 1, \ldots,$$

com $\mathcal{S}(0) = \mathcal{S}_0$. Esta fórmula define uma seqüência de conjuntos de esparsidade razoáveis, ao menos para valores não muito grandes de q.

Observação 2.2. *A fatoração incompleta nem sempre existe para matrizes positivas definidas. No entanto, podemos garantir a existência do método reduzindo as entradas positivas fora da diagonal, fazendo modificações para que não existam problemas durante a fatoração.*

Quando A é mal condicionada, no entanto, eliminar *fill-in* de valor absoluto pequeno pode piorar o condicionamento de A. Portanto, a técnica para construir o conjunto $\mathcal{S}$ que define o *fill-in* a ser descartado deve ser escolhida com cuidado.

Note que, como queremos uma aproximação de $A = \nabla^2 f(x)$, gostaríamos que a decomposição incompleta $\tilde{L}\tilde{L}^T$ fosse de tal forma que $(\tilde{L}\tilde{L}^T)^{-1}A$ seja próximo da matriz identidade. Porém, note que, sendo R a matriz de erro como definida em (2.1), temos que

$$(\tilde{L}\tilde{L}^T)^{-1}A = (\tilde{L}\tilde{L}^T)^{-1}[(\tilde{L}\tilde{L}^T) - R] = I - (\tilde{L}\tilde{L}^T)^{-1}R.$$

O Teorema 2.1 nos diz que $(\tilde{L}\tilde{L}^T)^{-1}$ será uma boa aproximação de A^{-1} se e somente se $||(\tilde{L}\tilde{L}^T)^{-1}R||$ for pequena para alguma norma matricial $||\cdot||$.

Teorema 2.1. *Suponha que $\tilde{L}\tilde{L}^T - R$ seja uma decomposição de uma matriz A não*

singular, e que o produto $\tilde{L}\tilde{L}^T$ *seja não singular. Então*

$$\frac{||(\tilde{L}\tilde{L}^T)^{-1}R||}{cond(A)} \leq \frac{||(\tilde{L}\tilde{L}^T)^{-1} - A^{-1}||}{||A^{-1}||} \leq ||(\tilde{L}\tilde{L}^T)^{-1}R||, \tag{2.3}$$

em que $cond(A)$*, o número de condição de* A*, é definido como* $cond(A) = ||A|| \cdot ||A^{-1}||$*. Se, além disso,* x *for a solução de* $Ax = b$ *e* $\tilde{x}$ *satisfizer* $\tilde{L}\tilde{L}^T\tilde{x} = b$*, então*

$$\frac{||x - \tilde{x}||}{||x||} \leq ||(\tilde{L}\tilde{L}^T)^{-1}R||. \tag{2.4}$$

Demonstração:

$$(\tilde{L}\tilde{L}^T)^{-1}R = (\tilde{L}\tilde{L}^T)^{-1}(\tilde{L}\tilde{L}^T - A) = I - (\tilde{L}\tilde{L}^T)^{-1}A = [A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}]A$$
$$||(\tilde{L}\tilde{L}^T)^{-1}R|| \leq ||A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}||\,||A|| = \frac{||(\tilde{L}\tilde{L}^T)^{-1} - A^{-1}||}{||A^{-1}||}||A||\,||A^{-1}|| \tag{2.5}$$

Dividindo ambos os lados por $||A||\,||A^{-1}||$, obtemos a primeira desigualdade de (2.3). A segunda desigualdade segue de (2.5):

$$(\tilde{L}\tilde{L}^T)^{-1}R = [A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}]A$$

$$(\tilde{L}\tilde{L}^T)^{-1}RA^{-1} = [A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}]$$

$$||A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}|| \leq ||(\tilde{L}\tilde{L}^T)^{-1}R||\,||A^{-1}||.$$

Dividindo por $||A^{-1}||$, obtemos a desigualdade desejada. Finalmente:

$$x - \tilde{x} = A^{-1}b - (\tilde{L}\tilde{L}^T)^{-1}b = [A^{-1} - (\tilde{L}\tilde{L}^T)^{-1}]Ax$$

Juntando com (2.5), temos $x - \tilde{x} = (\tilde{L}\tilde{L}^T)^{-1}Rx$. Para obter (2.4), basta tomar a norma da expressão. ■

Teorema 2.2. *Suponha que* $\tilde{L}\tilde{L}^T - R$ *seja uma decomposição de uma matriz não singular* A *e que* $||A^{-1}R|| < 1$*. Então, a matriz* $\tilde{L}\tilde{L}^T$ *será não singular e*

$$cond[(\tilde{L}\tilde{L}^T)^{-1}A] \leq \frac{1 + ||A^{-1}R||}{1 - ||A^{-1}R||}. \tag{2.6}$$

Demonstração: Suponha que $\tilde{L}\tilde{L}^T x = 0$. Então,

$$\tilde{L}\tilde{L}^T x = 0 \Leftrightarrow (A + R)x = 0 \Leftrightarrow (I + A^{-1}R)x = 0 \Rightarrow ||A^{-1}Rx|| = ||x|| \leq ||A^{-1}R||\,||x||$$

Como $||A^{-1}R|| < 1$, isto implica que $||x|| = 0$ e assim $x = 0$. Isto prova que $\tilde{L}\tilde{L}^T$ é não singular.

Agora, observe que:

$$\begin{aligned} cond[(\tilde{L}\tilde{L}^T)^{-1}A] &= cond[(A+R)^{-1}A] = cond[(I+A^{-1}R)^{-1}] \\ &= ||(I+A^{-1}R)^{-1}||\,||I+A^{-1}R|| \leq ||(I+A^{-1}R)^{-1}||(1+||A^{-1}R||) \end{aligned}$$

Pelo Lema 1.2,

$$||(I+A^{-1}R)^{-1}|| \leq \frac{1}{1-||A^{-1}R||}.$$

Assim, o resultado está provado. ■

Este teorema pode ser encontrado em [20].

A matriz de erro R estabelece a diferença entre a fatoração de Cholesky (completa) e a fatoração incompleta. Quando a Hessiana é esparsa e sua estrutura não possui um padrão adequado para a fatoração (estrutura banda, bloco diagonal, etc..) o cálculo da fatoração completa geralmente é proibitivo. A estratégia de trabalhar somente com a estrutura de dados original para a fatoração incompleta gera bons precondicionadores em muitas aplicações, ou seja, os erros são relativamente controlados em muitos casos. Essa é a idéia na qual está baseada o seu uso na formulação do método L-BFGS que apresentaremos no próximo capítulo.

Capítulo 3

Método BFGS para Sistemas de Grande Porte (L-BFGS) com Fatoração Incompleta

3.1 O Método L-BFGS

Quando os problemas de minimização são de grande porte (grande número de variáveis), as técnicas de atualização dos métodos quasenewtonianos tradicionais tornam-se caras computacionalmente, e a aproximação para o Hessiano ocupa um grande espaço de memória. Porém, existem modificações que podemos fazer nestes métodos para que eles se tornem competitivos em problemas de grande porte.

Neste trabalho, analisaremos um método que faz parte de uma classe chamada de "memória limitada"; nestes métodos, obtemos aproximações para o Hessiano que podem ser armazenadas em somente alguns vetores de dimensão n, em que n é o número de variáveis do problema. Estes métodos são baratos e fáceis de implementar, mas nem sempre convergem rapidamente. No entanto, apresentam claras vantagens sobre outros métodos no caso de problemas de grande porte, já que a quantidade de memória utilizada pode ser diretamente controlada pelo usuário. Em geral, eles apresentam um desempenho satisfatório. Estes métodos vêm sendo desenvolvidos desde o final da década de 70, e o método que vamos analisar neste trabalho é o método BFGS com memória limitada, descrito por Nocedal [16], e a análise de convergência segue o trabalho feito em [17].

Aqui, ao invés de armazenar aproximações densas de dimensão n da

matriz Hessiana associada ao problema, armazenaremos apenas alguns vetores de dimensão n. O método L-BFGS é baseado na idéia de armazenar informação de curvatura apenas das últimas iterações. Estas provavelmente serão mais relevantes na construção da aproximação do Hessiano do que as informações de iterações mais antigas.

Cada passo do BFGS tradicional tem a forma

$$\begin{aligned} x_{k+1} &= x_k - \alpha_k H_k \nabla f_k \\ H_{k+1} &= V_k^T H_k V_k + \rho_k s_k s_k^T \end{aligned} \tag{3.1}$$

em que $H_k = B_k^{-1}$, ou seja, representa a aproximação para $\nabla^2 f(x_k)^{-1}$, e α_k é o tamanho do passo escolhido por alguma estratégia, por exemplo usando as condições de Wolfe;

$$\rho_k = \frac{1}{y_k^T s_k}, \qquad V_k = I - \rho_k y_k s_k^T, \tag{3.2}$$

$$s_k = x_{k+1} - x_k, \qquad y_k = \nabla f_{k+1} - \nabla f_k. \tag{3.3}$$

Dizemos que a matriz H_{k+1} é obtida atualizando H_k, usando o par $\{s_k, y_k\}$.

Observação 3.1. *Vamos trabalhar aqui com a aproximação inversa H_k de forma que a resolução do sistema $B_k \alpha_k s_k = -\nabla f(x_k)$ não esteja incluida no nosso algoritmo. Salientamos que todo o trabalho pode ser feito de maneira análoga para a aproximação direta B_k; pela fórmula de Sherman-Morrisson-Woodbury, de (1.25), $H_{k+1} = B_{k+1}^{-1} = R^{-1}W^{-1}R^{-T} = V_k^T B_k^{-1} V_k + \rho_k s_k s_k^T$.*

A aproximação inversa H_k é geralmente densa, e assim o custo de armazenamento e manipulação é proibitivo. Para contornar este problema, armazenamos uma versão modificada de H_k implicitamente, guardando apenas m pares de vetores $\{s_i, y_i\}$ usados nas fórmulas (3.1), (3.2) e (3.3). O produto $H_k \nabla f_k$ pode ser obtido se realizarmos uma seqüência de produtos internos e somas de vetores envolvendo ∇f_k e os pares $\{s_i, y_i\}$. Após o cálculo da nova aproximação, o mais antigo dos pares $\{s_i, y_i\}$ é descartado, e substituído pelo novo par $\{s_k, y_k\}$ obtido no passo atual. Assim, o conjunto de pares de vetores contém sempre informação de curvatura dos últimos m passos.

Na prática, se m for pequeno (em relação a n), teremos melhores resultados. Podemos implementar o L-BFGS de maneira similar ao BFGS, inclusive

utilizando a mesma estratégia de busca linear para escolha do tamanho do passo.

Fórmula de Atualização

Na iteração k, a iterada atual é x_k e o conjunto de pares de vetores contém $\{s_i, y_i\}$ para $i = k-m, \ldots, k-1$. Primeiramente, escolhemos uma aproximação inicial H_k^0 (que pode, inclusive, variar a cada iteração). De 3.1 temos:

$$\begin{aligned} H_k &= (V_{k-1}^T \cdots V_{k-m}^T) H_k^0 (V_{k-m} \cdots V_{k-1}) + \\ &\quad + \rho_{k-m} (V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1} \cdots V_{k-1}) + \\ &\quad + \rho_{k-m+1} (V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2} \cdots V_{k-1}) + \\ &\quad + \cdots + \\ &\quad + \rho_{k-1} s_{k-1} s_{k-1}^T. \end{aligned} \tag{3.4}$$

Assim, podemos construir um algoritmo que calcula $H_k \nabla f_k$ recursivamente:

Algoritmo 3.1. Cálculo do produto $H_k \nabla f_k$:

$q \leftarrow \nabla f_k$
Para $i = k-1, \ldots, k-m$
  $v_i \leftarrow \rho_i s_i^T q$
  $q \leftarrow q - v_i y_i$
Fim Para
$r \leftarrow H_k^0 q$
Para $i = k-m, \ldots, k-1$
  $\beta \leftarrow \rho_i y_i^T r$
  $r \leftarrow r + s_i(v_i - \beta)$
Fim Para.

Sem considerar a multiplicação $H_k^0 q$, o algoritmo realiza $4mn$ multiplicações. Se H_k^0 for diagonal, n multiplicações a mais são realizadas. Uma das vantagens deste algoritmo, além do seu baixo custo computacional, é que a multiplicação pela matriz inicial fica isolada do resto dos cálculos, o que permite que a escolha da matriz varie de uma iteração para a outra. Podemos, inclusive, definir uma escolha implícita de H_k^0, usando uma aproximação inicial B_k^0 para o Hessiano e obter r através da resolução do sistema $B_k^0 r = q$.

Algoritmo 3.2. L-BFGS

Escolha $x_0, m > 0$.

$k \leftarrow 0$

Para $k = 0$ até *convergência* faça

Escolha H_k^0

Calcule $p_k \leftarrow -H_k \nabla f_k$ usando o Algoritmo 3.1

$x_{k+1} \leftarrow x_k + \alpha_k p_k$, com α_k satisfazendo as condições de Wolfe

Se $k > m$,

Descarte o par de vetores $\{s_{k-m}, y_{k-m}\}$

Fim Se

Calcule e armazene $s_k \leftarrow x_{k+1} - x_k$, $y_k \leftarrow \nabla f_{k+1} - \nabla f_k$

$k \leftarrow k + 1$

Fim Para.

Durante as primeiras $m - 1$ iterações, o Algoritmo 3.2 é equivalente ao BFGS, com $H_k^0 = H_0^{BFGS}$. No entanto, quando $m > \frac{n}{2}$, o L-BFGS se torna mais caro que o BFGS.

Esta estratégia de guardar os últimos m pares de vetores funciona bem na prática, apesar de não ser recomendável para problemas muito mal condicionados, em que os autovalores são distantes uns dos outros. Outra opção seria guardar apenas os pares de vetores que gerem matrizes bem condicionadas.

3.2 Análise de Convergência

Nesta seção, apresentaremos resultados de convergência para o método L-BFGS. Mostraremos que ele é globalmente convergente em problemas uniformemente convexos, e que sua taxa de convergência é r-linear. Os resultados serão enunciados em termos da aproximação direta B_k para a matriz Hessiana ao invés da aproximação inversa H_k que vínhamos apresentando, simplesmente para facilitar o trabalho de análise, segundo os resultados obtidos em [17]; salientamos que isto não modifica nenhum dos resultados obtidos aqui. Vamos assim considerar o algoritmo abaixo, análogo ao Algoritmo 3.2:

Algoritmo 3.3. *BFGS com Memória Limitada*

Escolha x_0, m, e uma matriz inicial B_0 simétrica positiva definida.

Para $k = 0$ até *convergência* faça

Calcule $p_k = -B_k^{-1}\nabla f(x_k)$

$x_{k+1} = x_k + \alpha_k p_k$ com α_k satisfazendo as condições de Wolfe

Seja $\tilde{m} = \min\{k+1, m\}$ e defina uma matriz simétrica positiva definida $B_k^{(0)}$. Escolha um conjunto crescente de inteiros $\mathcal{L}_k = \{j_0, \ldots, j_{\tilde{m}-1}\} \subseteq \{0, \ldots, k\}$.

Atualize $B_k^{(0)}$ $\tilde{m}$ vezes usando os pares $\{y_{jl}, s_{jl}\}_{l=0}^{\tilde{m}-1}$, por exemplo, para $l = 0, \ldots, \tilde{m}-1$ calcule

$$B_k^{(l+1)} = B_k^{(l)} - \frac{B_k^{(l)} s_{jl} s_{jl}^T B_k^{(l)}}{s_{jl}^T B_k^{(l)} s_{jl}} + \frac{y_{jl} y_{jl}^T}{y_{jl}^T s_{jl}} \tag{3.5}$$

Faça $B_{k+1} = B_k^{\tilde{m}}$

Fim Para.

Como veremos mais à frente, existem várias escolhas possíveis para $B_k^{(0)}$. Aqui, vamos supor somente que a seqüência de matrizes $B_k^{(0)}$ e a seqüência das suas inversas são ambas limitadas. Este algoritmo é idêntico ao método BFGS quando $k < m$.

Vamos então estabelecer o resultado de convergência para este algoritmo.

Hipóteses:

(1) A função objetivo f é duas vezes continuamente diferenciável;

(2) O conjunto de nível $D = \{x \in \mathbb{R}^n / f(x) \leq f(x_0)\}$ é convexo;

(3) Existem constantes positivas M_1 e M_2 tais que

$$M_1 ||z||^2 \leq z^T \nabla^2 f(x) z \leq M_2 ||z||^2 \tag{3.6}$$

para todo $z \in \mathbb{R}^n$ e todo $x \in D$. Note que isto implica que f tem um único minimizador $x_\star$ em D.

O teorema seguinte pode ser encontrado em [17].

Teorema 3.1. *Seja x_0 um ponto inicial para o qual f satisfaz as hipóteses acima, e suponha que as matrizes $B_k^{(0)}$ são escolhidas de forma que $\{||B_k^{(0)}||\}$ e $\{||B_k^{(0)^{-1}}||\}$ sejam limitadas. Então, para qualquer matriz positiva definida B_0, o Algoritmo 3.3*

gera uma seqüência $\{x_k\}$ *que converge para* $x_\star$*. Além disso, existe uma constante* $0 \leq r < 1$ *tal que*

$$f(x_k) - f(x_\star) \leq r^k[f(x_0) - f(x_\star)], \tag{3.7}$$

o que implica que $\{x_k\}$ *converge r-linearmente.*

Demonstração: Se definirmos

$$G_k = \int_0^1 \nabla^2 f(x_k - \tau s_k)\, d\tau,$$

então

$$y_k = G_k s_k. \tag{3.8}$$

De (3.7) e (3.8) temos que

$$M_1||s_k||^2 \leq y_k^T s_k \leq M_2||s_k||^2, \tag{3.9}$$

e

$$\frac{||y_k||^2}{y_k^T s_k} = \frac{s_k^T G_k^2 s_k}{s_k^T G_k s_k} \leq M_2. \tag{3.10}$$

Vamos denotar por $tr(B)$ o traço de B. Então, de (3.5), (3.10) e como $\{||B_k^{(0)}||\}$ é limitada,

$$tr(B_{k+1}) \leq tr(B_k^{(0)}) + \sum_{t=0}^{\tilde{m}-1} \frac{||y_{jl}||^2}{y_{jl}^T s_{jl}} \leq tr(B_k^{(0)}) + \tilde{m}M_2 \leq M_3, \tag{3.11}$$

para alguma constante positiva M_3. Existe também uma expressão simples para o determinante:

$$det(B_{k+1}) = det(B_k^{(0)}) \prod_{l=0}^{\tilde{m}-1} \frac{y_{jl}^T s_{jl}}{s_{jl}^T B_k^{(l)} s_{jl}} = det(B_k^{(0)}) \prod_{l=0}^{\tilde{m}-1} \frac{y_{jl}^T s_{jl}}{s_{jl}^T s_{jl}} \frac{s_{jl}^T s_{jl}}{s_{jl}^T B_k^{(l)} s_{jl}}. \tag{3.12}$$

Como, por (3.11), o maior autovalor de $B_k^{(l)}$ é menor que M_3, temos, usando (3.9) e observando que $\{||B_k^{(0)^{-1}}||\}$ é limitada,

$$det(B_{k+1}) \geq det(B_k^{(0)})(M_1/M_3)^{\tilde{m}} \geq M_4, \tag{3.13}$$

para alguma constante positiva M_4. Assim, de (3.11) e (3.13), concluimos que existe

uma constante $\delta > 0$ tal que

$$\cos\theta_k \equiv \frac{s_k^T B_k s_k}{||s_k||\,||B_k s_k||} \geq \delta.$$

Pode-se mostrar que as condições de Wolfe e as Hipóteses enunciadas acima implicam que existe uma constante $c > 0$ tal que

$$f(x_{k+1}) - f(x_\star) \leq (1 - c\cos^2\theta_k)(f(x_k) - f(x_\star)).$$

Usando a definição de $\cos\theta_k$, obtemos (3.7). De (3.6), temos que

$$\frac{1}{2}M_1||x_k - x_\star||^2 \leq f(x_k) - f(x_\star),$$

o que, junto com (3.7), implica que $||x_k - x_\star|| \leq r^{k/2}[2(f(x_0) - f(x_\star)/M_1]^{1/2}$, e assim a seqüência $\{x_k\}$ também é r-linearmente convergente. ■

Matrizes de Recomeços

Sabe-se que a escolha da matriz $H_k^{(0)}$ na fórmula do método L-BFGS tem um papel importantíssimo na convergência do algoritmo 3.2. Gostaríamos de obter uma matriz $H_k^{(0)}$ para o método L-BFGS que acelerasse de alguma forma sua convergência. A escolha desta matriz pode ser feita a cada iteração, pois a multiplicação pela matriz $H_k^{(0)}$ fica isolada do resto dos cálculos. Poderíamos então nos perguntar qual estratégia seria melhor para escolher estas matrizes.

Em geral, definimos $H_k^{(0)}$ como um múltiplo da matriz identidade. Se usarmos somente a matriz identidade, a convergência torna-se bastante lenta, e em alguns casos nem obtemos convergência. Um método bastante utilizado para o cálculo de $H_k^{(0)}$ é definirmos $H_k^{(0)} = \gamma_k I$, em que

$$\gamma_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}.$$

Note que γ_k é um fator de escala que tenta estimar o valor da Hessiana verdadeira ao longo da direção de busca mais recente, pois este quociente aproxima o maior autovalor da matriz Hessiana inversa. Esta escolha faz com que a direção de busca tenha uma escala mais apropriada, e assim o tamanho do passo α_k toma o valor 1 na maioria das iterações. Este método foi proposto por Nocedal [17].

Outra possibilidade é tentar encontrar uma matriz diagonal que satisfaça aproximadamente a equação secante com respeito aos últimos m passos. Seja x_k a iteração atual, e suponha que $k > m$. Queremos encontrar a matriz diagonal D_k que minimiza

$$||D_k Y_{k-1} - S_{k-1}||_F,$$

com $||\cdot||_F$ sendo a norma de Frobenius, e $Y_{k-1} = [y_{k-1}, \ldots, y_{k-m}]$, $S_{k-1} = [s_{k-1}, \ldots, s_{k-m}]$. A solução é $D_k = \text{diag}(d_k^i)$, em que

$$d_k^i = \frac{s_{k-1}^i y_{k-1}^i + \cdots + s_{k-m}^i y_{k-m}^i}{(y_{k-1}^i)^2 + \cdots + (y_{k-m}^i)^2}, \quad i = 1, \ldots, n. \tag{3.14}$$

Como um elemento d_k^i pode ser negativo ou muito próximo de zero, utiliza-se um teste de segurança: (3.14) só é utilizada se o denominador for maior que uma certa tolerância, e se todos os elementos diagonais estiverem dentro de um certo intervalo pré definido. Senão, utiliza-se $d_k^i = \gamma_k$.

Testes mostram que as estratégias que utilizam $\gamma_k I$ e (3.14) são as melhores na prática; como $\gamma_k I$ tem implementação mais barata, ela é geralmente a preferida nos algoritmos. Para mais detalhes e testes numéricos, veja [17] e [19].

As aproximações já apresentadas funcionam na prática, porém gostaríamos de utilizar uma aproximação mais precisa, de forma a diminuir o número de iterações necessárias para obter convergência em problemas de grande porte, sem aumentar significativamente o custo computacional de cada iteração.

É natural imaginarmos que uma aproximação da matriz Hessiana original do problema nos traria melhores resultados; no entanto, o processo de calcular a matriz Hessiana a cada iteração torna-se proibitivo na prática, devido ao grande número de variáveis envolvidas no problema. Além disso, em problemas de grande porte, freqüentemente a matriz associada ao problema é esparsa, ou seja, possui um grande número de entradas nulas. Se optarmos por uma aproximação não diagonal, gostaríamos que esta aproximação mantivesse, até certo ponto, o padrão de esparsidade da matriz Hessiana, já que isto pode garantir um certo controle sobre o custo a cada iteração, e também que a solução do sistema $B_k s_k = -F(x_k)$ pudesse ser obtida facilmente. Uma alternativa para calcular esta aproximação é a *fatoração incompleta*, que vamos utilizar como matriz de recomeço para o método L-BFGS. Este trabalho torna-se vantajoso, pois realizamos esta decomposição incompleta somente a cada m iterações, o que acelera a convergência do método sem acrescentar cálculos excessivos, como seriam necessários para, por exemplo, utilizar a própria matriz

Hessiana do problema na aplicação do método L-BFGS. Além disso, podemos exigir que a decomposição incompleta seja, de certa forma, mais bem condicionada que a aproximação $\gamma_k I$ proposta por Nocedal, como poderemos ver nos testes numéricos apresentados mais à frente.

Esta matriz (às vezes chamada *fator de escala* para o método), será obtida através dos pares de vetores que guardam informações recentes sobre a matriz Hessiana do problema. Como observamos, o algoritmo do L-BFGS guarda m vetores de forma a aproximar as informações de curvatura mais recentes do problema. Nosso algoritmo, a cada m passos, toma a matriz Hessiana do problema calculada no ponto x_{im}, em que $i = k(\text{mod } m)$, e realizamos sua fatoração de Cholesky incompleta:

$$\nabla^2 f(x_{im}) = L_{im} L_{im}^T - R_{im},$$

e assim

$$B_{im} = L_{im} L_{im}^T.$$

A partir deste ponto, nas próximas m iterações, a matriz H_{im} é atualizada através da fórmula (3.4).

No próximo capítulo, será possível analisar os resultados numéricos obtidos com esta modificação, mostrando assim que esta alternativa pode ser bastante útil em problemas que possuem taxa de convergência lenta, e que permitem a realização da fatoração incompleta em algumas iterações. Salientamos ainda que, de acordo com o problema, pode ser cogitada uma alteração em nossa proposta de atualização, permitindo que a fatoração incompleta seja realizada mais ou menos freqüentemente, mantendo ainda uma taxa de convergência razoável, além de podermos controlar, até certo ponto, a complexidade numérica da fatoração incompleta, de modo que esta alternativa seja competitiva com os métodos que apresentam matrizes iniciais diagonais.

Capítulo 4

Implementação: Testes numéricos e Ambiente CUTE

Para analisar na prática um algoritmo de otimização, precisamos compará-lo com outros métodos já existentes em problemas variados, para que sua eficiência seja comprovada. Para isto, utilizamos a coleção de problemas do Ambiente CUTEr [15], que fornece problemas específicos e um ambiente de programação onde podemos testar nossos algoritmos.

4.1 CUTEr

O ambiente de testes CUTEr [15] consiste de uma coleção de problemas e ferramentas (subrotinas escritas em Fortran) idealizadas para facilitar o trabalho de teste de um algoritmo de otimização e sua comparação com outros métodos. Ele nos fornece uma grande variedade de problemas, classificados conforme suas características (restrições, número de variáveis, informações sobre derivadas, por exemplo), escritos no formato SIF (*Standard Input Format*). Para transformá-los em arquivos de dados e para leitura pelo Fortran 77, utilizamos o decodificador SifDec, fornecido com a distribuição do CUTEr. Uma vez decodificados, estes arquivos podem ser acessados através das ferramentas disponíveis no ambiente, e assim tornam-se facilmente utilizáveis em algoritmos de otimização.

O CUTEr está disponível para várias plataformas UNIX, inclusive LINUX, onde nosso trabalho foi desenvolvido.

4.2 Implementação e Resultados Numéricos

No CUTEr, os problemas são introduzidos dinamicamente, ou seja, o usuário pode contribuir em determinada ocasião, introduzindo um novo problema para fazer parte da numerosa coleção. Cada problema é nomeado, assim como codificado e classificado. Desse modo, podemos selecionar tipos de problemas específicos, como por exemplo, problemas com mais de 100 váriáveis com função objetivo quadrática e menos de 50 restrições lineares de desiguadade. No nosso caso, como estamos tratando de problemas irrestritos com número de varáveis arbitrárias, selecionamos alguns problemas com funções objetivo não lineares gerais e com número arbitrário de variáveis. Para selecionar os problemas que satisfazem estes critérios, utilizamos a classificação fornecida para os problemas da biblioteca CUTEr.

Um problema da biblioteca é classificado pela seqüência de caracteres

XXXr-XX-n-m

O primeiro caractere define o tipo de função objetivo do problema: N indica que nenhuma função objetivo é fornecida; C indica que a função objetivo é constante; L diz que a função objetivo é linear; Q diz que a função objetivo é quadrática; S indica que a função objetivo é uma soma de quadrados, e O indica que a função objetivo é de outro tipo, diferente dos listados anteriormente. O segundo caractere define o tipo de restrições do problema. No nosso caso, a única escolha possível era U, pois trabalhamos somente com problemas irrestritos. O terceiro caractere indica a suavidade do problema. Como queríamos problemas regulares, escolhemos sempre R. Na quarta posição da seqüência, devemos inserir um inteiro (r) que corresponde ao maior grau de diferenciação disponível analiticamente dentro da descrição do problema. Os valores possíveis são 0, 1 e 2; no nosso caso, como queríamos trabalhar com a Hessiana dos problemas, escolhemos somente os problemas com o valor 2.

O caractere que segue o primeiro hífen representa a origem do problema; este pode ser acadêmico (A), parte de um exercício de modelagem, cujo valor da solução não é utilizado em nenhuma aplicação prática (M), ou um problema com aplicações reais da solução (R). Aqui, não tínhamos nenhuma restrição quanto à escolha dos problemas, pois nossos testes se destinavam a quaisquer destes tipos de problema.

Logo em seguida, temos um caracter que indica se a descrição do problema contém (Y) ou não (N) variáveis internas explícitas. Aqui também nossa

escolha era indiferente.

Entre o segundo e o terceiro hífens, estão indicados o número de variáveis do problema. Aqui, utilizamos tanto problemas com número fixo de variáveis (em geral, um número inteiro maior que 1000, pois queríamos testar o desempenho do nosso algoritmo em problemas razoavelmente grandes), quanto problemas com número de variáveis a ser escolhido, cujo caractere correspondente na classificação é V.

O último símbolo indica o número de restrições do problema; como trabalhamos com problemas irrestritos, usamos sempre o valor 0.

Em geral, nossos problemas têm a classificação

OUR2-XX-X-0.

Para ilustrar a classificação, citamos o problema DIXMAANA, cuja classificação é OUR2-AN-V-0.

A Tabela 4.1 abaixo ilustra os testes computacionais realizados com os problemas selecionados, apresentando um comparação entre o número de iterações realizadas por cada estratégia. Indicamos na primeira coluna o nome e o número de variáveis do problema correspondente; DIXMAANA(1500) significa que este problema possui 1500 variáveis. Nas outras colunas, indicamos m, que é o intervalo de recomeços utilizados no L-BFGS. Denotamos por $H_k^{(0)} = \gamma_k I$, proposta por Nocedal, e por $H_k^{(0)_{mod}} = L_k L_k^T - R_k$, a fatoração incompleta de Cholesky.

Problema	$m = 5$		$m = 7$		$m = 10$	
	$H_k^{(0)}$	$H_k^{(0)_{mod}}$	$H_k^{(0)}$	$H_k^{(0)_{mod}}$	$H_k^{(0)}$	$H_k^{(0)_{mod}}$
DIXMAANA (1500)	13	16	16	15($\diamond$)	19	15($\diamond$)
DIXMAANB (1500)	44	22($\diamond$)	46	21($\diamond$)	58	23($\diamond$)
DIXMAANC (1500)	40	30($\diamond$)	48	32($\diamond$)	63	33($\diamond$)
DIXMAAND (1500)	42	35($\diamond$)	38	37($\diamond$)	53	38($\diamond$)
EDENSCH (2000)	41	41	(?)	35	113	39
ENGVAL1 (1000)	47	23	73	23	92	23
SCHMVETT (1000)	197(*)	29	150(*)	29	135(*)	28
TOINTPSP (50)	181	91	243	83	286	66

Tabela 4.1: Testes Numéricos

Observação 4.1. *O símbolo (*) indica situações em que a função não atingiu o valor esperado, e o simbolo ($\diamond$) indica situações em que a fatoração de Cholesky incompleta teve que realizar modificações em pivôs negativos ou nulos encontrados durante a fatoração. O símbolo (?) indica um problema que não convergiu com a matriz indicada.*

Os resultados da tabela mostram a efetividade da metodologia introduzida neste trabalho. De fato, a maior proximidade da Hessiana nas iterações de recomeços, dada pela fatoração incompleta, fornece uma considerável diminuição no número de interações.

As Tabelas 4.2, 4.3 e 4.4, respectivamente associadas a $m = 5$, $m = 7$ e $m = 10$, mostram uma comparação do número de avaliações da função, do seu gradiente e do Hessiano, representados nas tabelas por Feval, Geval e Heval, respectivamente, e do tempo computacional associado a cada estratégia.

Problema	$H_k^{(0)}$				$H_k^{(0)_{mod}}$			
	Feval	Geval	Heval	Tempo	Feval	Geval	Heval	Tempo
DIXMAANA (1500)	31	30	0	0.18	20	19	4	0.31
DIXMAANB (1500)	249	248	0	1.34	26	25	5	0.47
DIXMAANC (1500)	200	199	0	1.11	20	19	4	0.32
DIXMAAND (1500)	127	126	0	0.75	40	39	7	0.64
EDENSCH (2000)	164	163	0	1.78	67	66	9	1.08
ENGVAL1 (1000)	170	169	0	0.69	30	29	5	0.20
SCHMVETT (1000)	558	557	0	4.20	59	58	6	0.59
TOINTPSP (50)	427	426	0	0.10	205	204	19	0.06

Tabela 4.2: Resultados para $m = 5$

Problema	$H_k^{(0)}$				$H_k^{(0)_{mod}}$			
	Feval	Geval	Heval	Tempo	Feval	Geval	Heval	Tempo
DIXMAANA (1500)	32	31	0	0.21	19	18	3	0.30
DIXMAANB (1500)	255	254	0	1.31	25	24	3	0.47
DIXMAANC (1500)	257	256	0	1.41	38	37	5	0.59
DIXMAAND (1500)	159	158	0	0.90	42	41	6	0.66
EDENSCH (2000)	244	243	0	2.63	57	56	5	0.92
ENGVAL1 (1000)	295	294	0	1.17	30	29	4	0.20
SCHMVETT (1000)	420	419	0	3.35	59	58	5	0.58
TOINTPSP (50)	541	540	0	0.10	183	182	12	0.05

Tabela 4.3: Resultados para $m = 7$

Problema	$H_k^{(0)}$				$H_k^{(0)_{mod}}$			
	Feval	Geval	Heval	Tempo	Feval	Geval	Heval	Tempo
DIXMAANA (1500)	34	33	0	0.23	19	18	2	0.29
DIXMAANB (1500)	299	298	0	1.67	27	26	3	0.49
DIXMAANC (1500)	388	387	0	2.07	39	38	4	0.62
DIXMAAND (1500)	272	271	0	1.53	43	42	4	0.69
EDENSCH (2000)	575	574	0	6.24	59	58	4	0.95
ENGVAL1 (1000)	355	354	0	1.46	30	29	3	0.20
SCHMVETT (1000)	403	402	0	3.08	57	56	3	0.54
TOINTPSP (50)	597	596	0	0.15	143	142	7	0.04

Tabela 4.4: Resultados para $m = 10$

A seguir, as figuras ilustram a evolução do algoritmo em alguns dos problemas acima. Em todos os casos, as estrelas $\star$ representam as iterações realizadas com o algoritmo que utiliza nossa matriz inicial, e os círculos $\circ$ representam as iterações do algoritmo que utiliza a matriz inicial $\gamma_k I$, proposta por Nocedal [2].

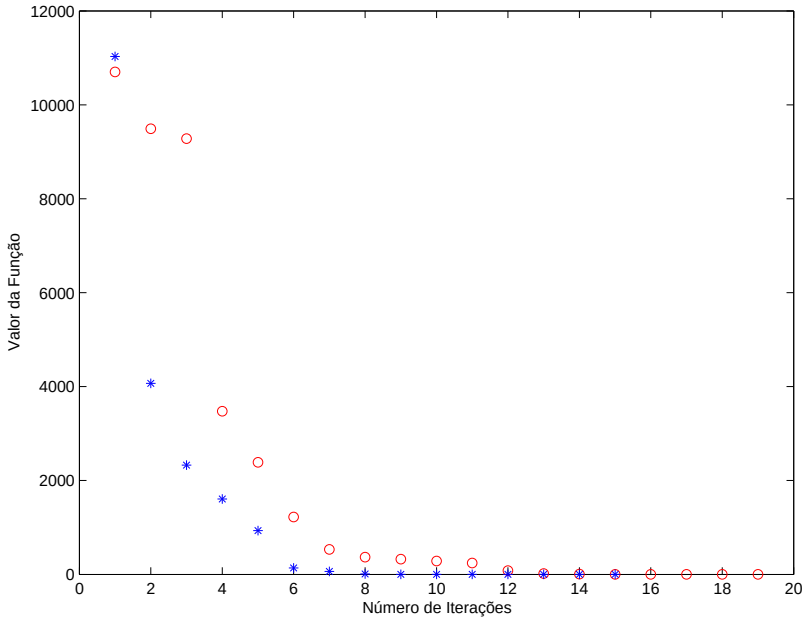

Figura 4.1: Problema DIXMAANA, de 1500 variáveis, com m=10

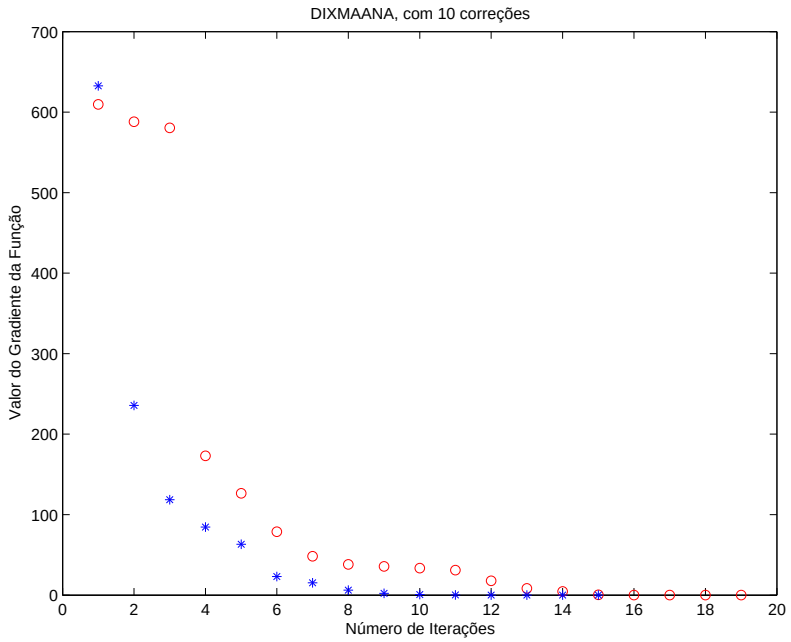

Figura 4.2: Problema DIXMAANA, de 1500 variáveis, com m=10, gráfico Iterações $\times$ Valor do Gradiente

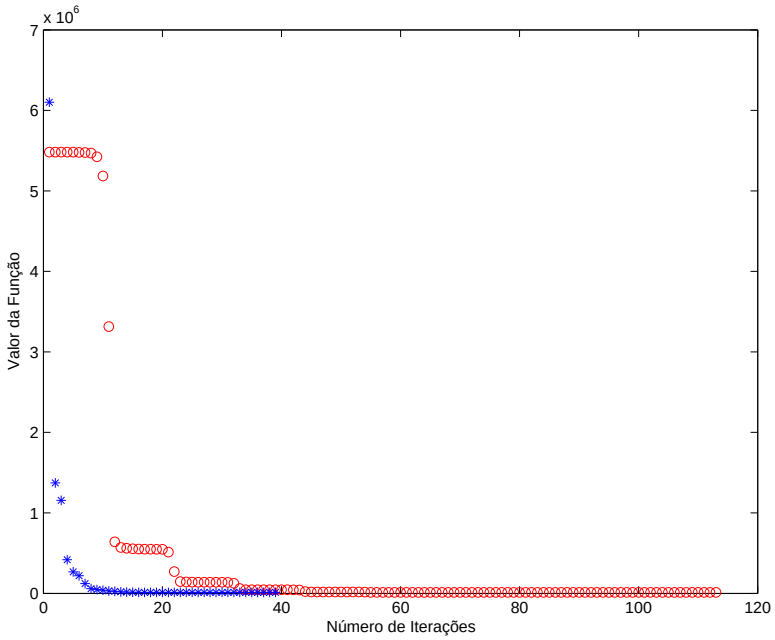

Figura 4.3: Problema EDENSCH, de 2000 variáveis, com m=10

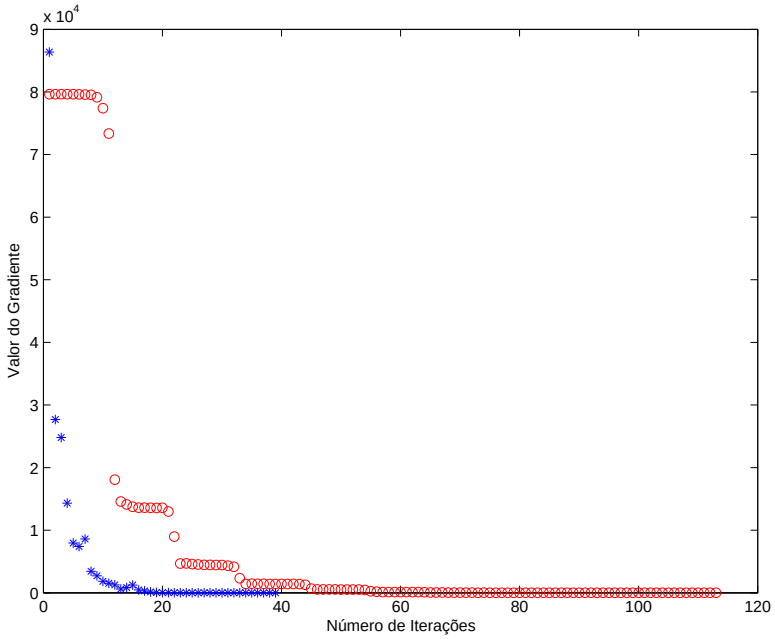

Figura 4.4: Problema EDENSCH, de 2000 variáveis, com m=10, gráfico Iterações × Valor do Gradiente

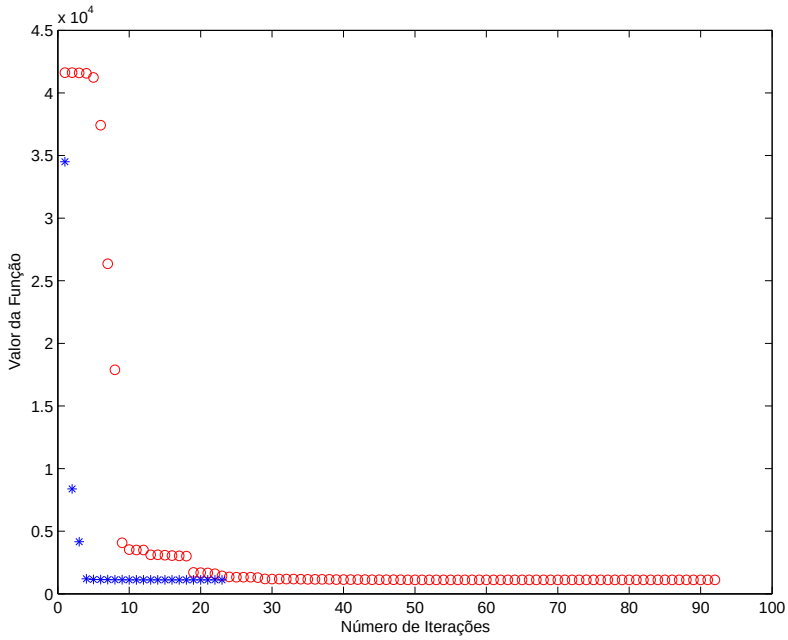

Figura 4.5: Problema ENGVAL1, de 1000 variáveis, com m=10

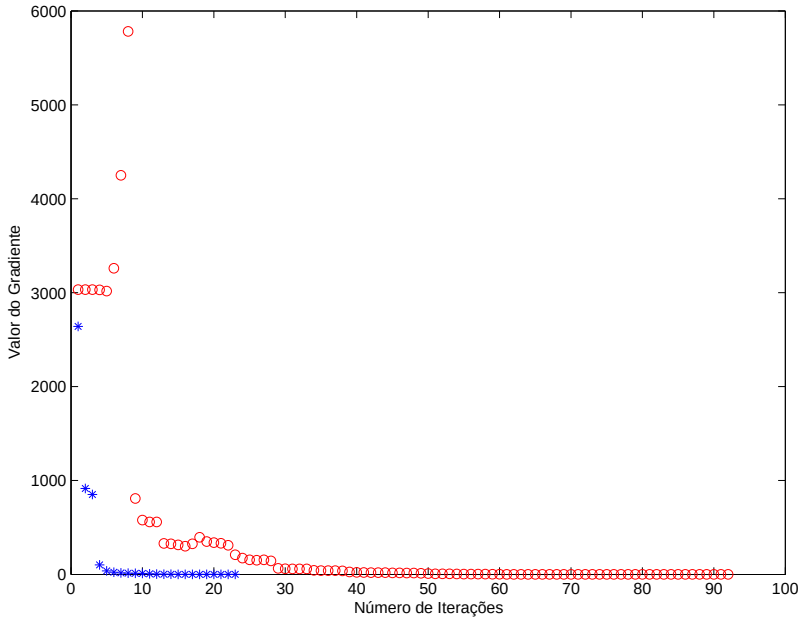

Figura 4.6: Problema ENGVAL1, de 1000 variáveis, com m=10, gráfico Iterações × Valor do Gradiente

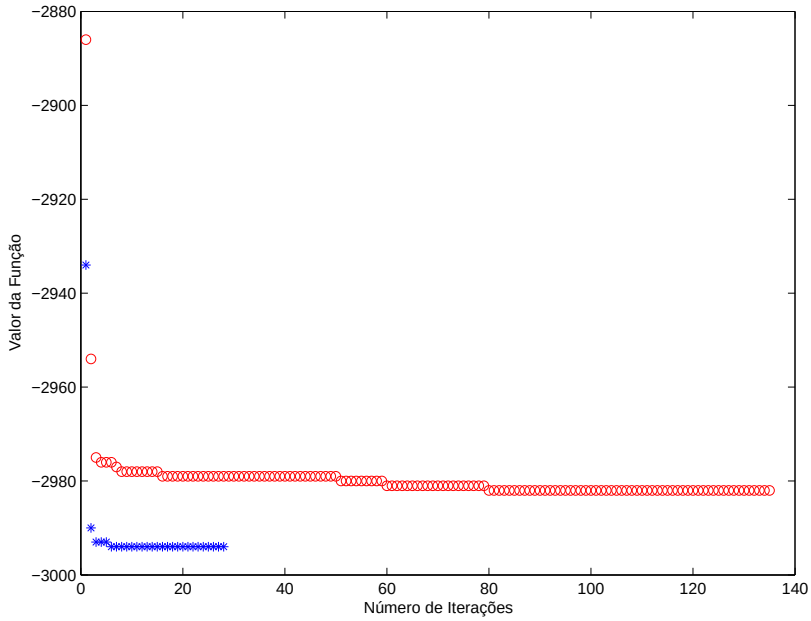

Figura 4.7: Problema SCHMVETT, de 1000 variáveis, com m=10

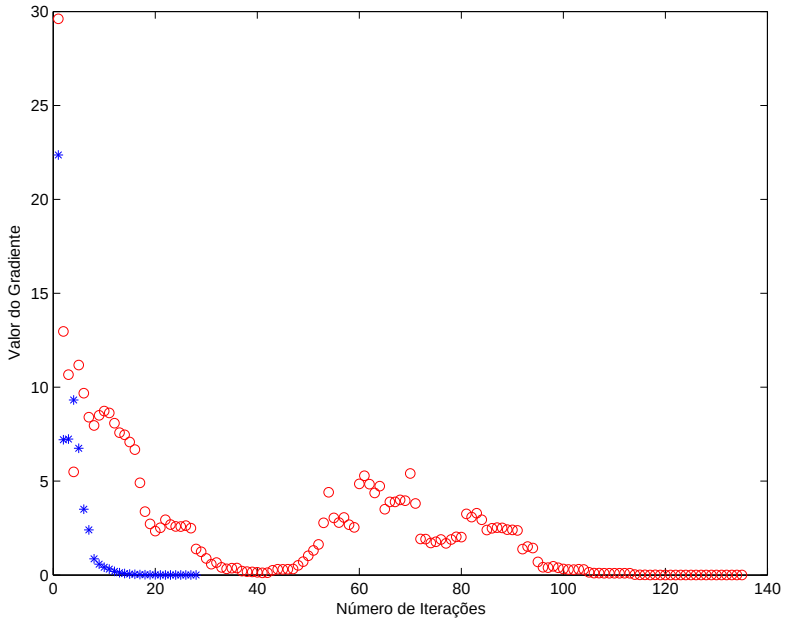

Figura 4.8: Problema SCHMVETT, de 1000 variáveis, com m=10, gráfico Iterações × Valor do Gradiente

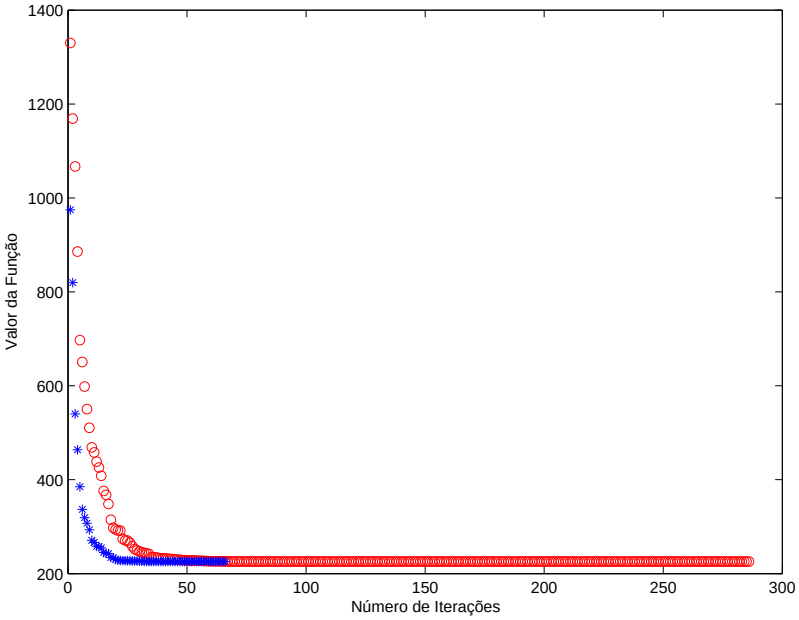

Figura 4.9: Problema TOINTPSP, de 50 variáveis, com m=10

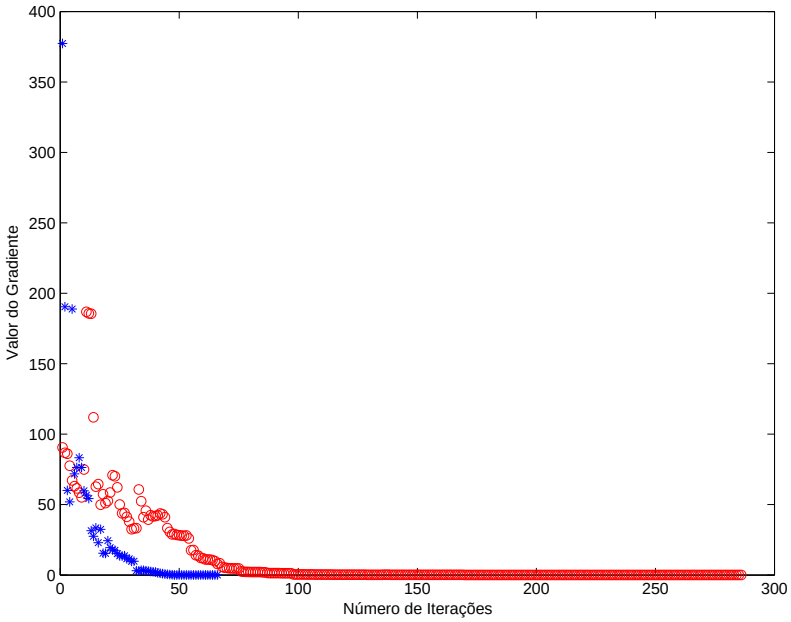

Figura 4.10: Problema TOINTPSP, de 50 variáveis, com m=10, gráfico Iterações × Valor do Gradiente

Os gráficos também ilustram a efetividade da fatoração incompleta no método L-BFGS. Podemos observar que em praticamente todas as iterações o valor do gradiente da função objetivo é menor do que o obtido com a estratégia usual. Em geral, os métodos secantes fornecem um histórico de iterações oscilantes. Podemos verificar a minimização deste fenômeno pela introdução da fatoração incompleta.

Capítulo 5

Conclusões

Um dos métodos mais efetivos para resolver problemas de otimização é o método BFGS, considerando suas boas propridedades teóricas e principalmente pelas suas possibilidades de implementação computacional. Quando o problema é de grande porte, as vantagens computacionais sobre o método de Newton, por exemplo, tornam-se ainda mais evidentes. A versão L-BFGS é a mais conhecida na literatura de otimização, tendo sido incorporada a vários códigos computacionais. Um dos pontos cruciais para o bom desempenho deste método são os recomeços da sua fórmula de memória limitada.

Neste trabalho, introduzimos a técnica da fatoração incompleta para a aproximação da Hessiana nas iterações de recomeços. A idéia natural e ordinária da fatoração incompleta é o seu uso como precondicionador de método iterativos lineares. Embora tenhamos sido tentados a criar o termo "precondicionador para métodos quasenewtonianos", acreditamos não ser o caso para tal terminologia, uma vez que estaríamos nos referindo a precondicionar um método para programação não-linear. Na verdade, o que obtivemos foi uma forma de proximidade do Hessiano que apresenta melhor desempenho do que a forma original do L-BFGS e muito adequada às especificidades da implementação de grande porte.

Devido à simetria, empregamos a fatoração de Cholesky incompleta. Atenção foi dada à possibilidade de mau condicionamento ou singularidade na construção da fatoração, como ilustrado nos resultados computacionais. Podemos notar nos resultados numéricos que a convergência normalmente se dá em um número relativamente pequeno de iterações, mesmo para dimensões grandes.

Como futuras possibilidades, poderíamos pensar em novas formas de correção da fatoração incompleta existentes na literatura, de modo a introduzir uma

proximidade ainda maior do Hessiano. Outra abordagem seria trabalhar com outros métodos quasenewtonianos e testá-los com esta mesma estratégia.

Referências Bibliográficas

[1] DENNIS, J. E., SCHNABEL, R. B., "Numerical Methods for Unconstrained Optimization and Nonlinear Equations", SIAM Classics in Applied Mathematics, 16, SIAM, 1996.

[2] NOCEDAL, J., WRIGHT, S. J., "Numerical Optimization", Springer Series In Operations Research, Springer, 1999.

[3] FLETCHER, R., "Practical Methods of Optimization", 2a. Edição, John Wiley & Sons, 1987.

[4] KELLEY, C. T., "Iterative Methods for Optimization", SIAM Frontiers in Applied Mathematics, SIAM, 1999.

[5] KELLEY, C. T., "Iterative Methods for Linear and Nonlinear Equations", SIAM Frontiers in Applied Mathematics, SIAM, 1995.

[6] GILL, P. E., MURRAY, W., WRIGHT, M. H., "Practical Optimization", Academic Press, 1981.

[7] LUENBERGER, D., "Introduction do Linear and Nonlinear Programming", Addison Wesley, 2a. ed., 1984.

[8] ORTEGA, J. M., RHEINBOLDT, W. C., "Iterative Solution of Nonlinear Equations in Several Variables", Academic Press, 1970.

[9] AXELSSON, O. "Iterative Solution Methods", Cambridge University Press, 1996.

[10] GOLUB, G. H., VAN LOAN, C. F., "Matrix Computations", The Johns Hopkins University Press, 3rd. Ed., 1996.

[11] BROYDEN, C. G., "A Class of Methods for Solving Nonlinear Simultaneous Equations", Mathematics of Computation, Vol. 19, No. 92. 1965.

[12] DENNIS, J. E., MORÉ, J. J., "Quasi-Newton Methods, Motivation and Theory", SIAM Review, V. 9 No. 1, 1977.

[13] BROYDEN, C. G., DENNIS, J. E., MORÉ, J. J., "On the Local and Superlinear Convergence of Quasi-Newton Methods", J. Inst. Math. Appl., 12, 1973.

[14] DENNIS, J. E., MORÉ, J. J., "A Characterization of Superlinear Convergence and Its Applications to Quasi-Newton Methods", Math. Comp. 28, 126, 1974.

[15] GOULD, N. I. M., ORBAN, D., TOINT, Ph. L., "CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, Revisited", CERFACS Technical Report n. TR/PA/01/04.

[16] NOCEDAL, J., "Updating quasi-Newton matrices with limited storage", Mathematics of Computation 35, 773-782, 1980.

[17] LIU, D.C., NOCEDAL, J., "On the Limited Memory BFGS Method for Large Scale Optimization", Math. Programming 45, 503, 1989.

[18] POWELL, M. J. D., "A new algorithm for unconstrained optimization", Nonlinear Programming, Academic Press, 1970.

[19] GILBERT, J. C., LEMARÉCHAL, C., "Some Numerical Experiments With Variable-Storage Quasi-Newton Algorithms", Math. Programming 45, 407, 1989.

[20] PLOEG, A. V. D., "Preconditioning Techniques for Non-Symmetric Matrices with Application to Temperature Calculations of Cooled Concrete", Int. Journal for Numerical Methods in Engineering, V. 35, 1992.

[21] AXELSSON, O., "Preconditioned Conjugate Gradient Methods", BIT 29:4, 1989.