Ministério da Educação
Universidade Federal do Agreste
de Pernambuco

Relatório Final
Métodos de Otimização

$\propto$

Ministério da Educação
Universidade Federal do Agreste de Pernambuco

Relatório Final
Métodos de Otimização

Mateus Baltazar de Almeida
Matheus Machado Vieira
Orientador: Gersonilo Oliveira da Silva

$\propto$

Sumário

Capítulo 1

Métodos Matemáticos de Otimização

1.1 O Conceito de Otimização

Diz-se otimização, o processo que tem como objetivo encontrar condições que minimizam ou maximizam algo (seja energia, tempo, dinheiro, etc). Sendo este, muitas vezes um trabalho árduo, custoso.

Dessa maneira, na matemática, este processo é amplamente utilizado quando busca-se valores pertencentes ao conjunto A (que pode ter restrições), com o objetivo de encontrar uma solução ótima, aplicando os valores de A em numa função objetivo predefinida.

Podendo assim, ser representada da seguinte forma:

Dada a função

$$f : A \to \mathbb{R} \tag{1.1}$$

- Maximização pode ser definida como:

  busca pelo elemento $x_0 \in A$, que satisfaz:

  $$f(x_0) \geq f(x); \tag{1.2}$$

  para todo $x \in A$.

- Minimização pode ser definida como:

  busca pelo elemento $x_0 \in A$, que satisfaz:

  $$f(x_0) \leq f(x); \tag{1.3}$$

  para todo $x \in A$.

Com isso, podemos agora entender como esse processo pode ser custoso. Iniciando com fato de que existem os pontos máximos e mínimos (pontos críticos), locais e globais, no espaço das funções. Sendo os pontos críticos locais, aqueles que não são os menores ou maiores valores para a minimização e maximização, respectivamente.

E os pontos globais, aqueles que representam o menor ou maior valor no espaço das funções, para a minimização e maximização, respectivamente.

Criando assim, uma certa incerteza ao encontrar um valor crítico numa função, já que é estritamente difícil saber se o ponto crítico encontrado é local ou global. Como pode-se perceber na Figura 1.1.

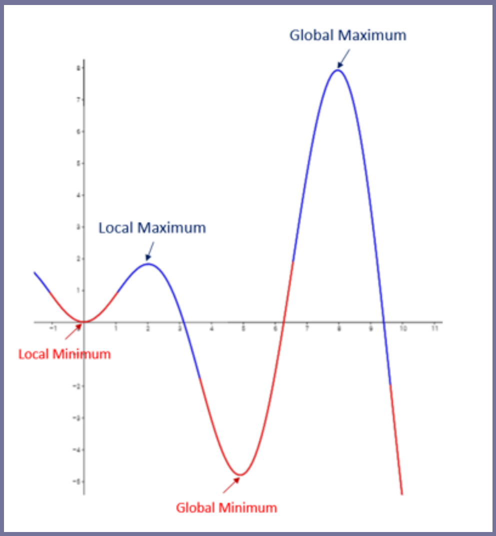

Figura 1.1: Exemplo de pontos críticos locais e globais indicados no gráfico de uma função

1.2 Otimização de Funções à Uma Variável Real

Evidente que funções possuem as variáveis dependentes (a qual representa o objeto da otimização) e variáveis independentes (cujo suas grandezas podem ser selecionadas), podemos denotar que, para a equação

$$y = f(x), \tag{1.4}$$

quando buscamos otimizá-la, temos como objetivo encontrar valores que quando aplicados à x, temos o mínimo ou máximo valor y (seja ele local, ou preferencialmente global).

Partindo dessa perspectiva, acaba surgindo a necessidade de utilizar algum recurso para encontrar o os pontos críticos. E nesse sentindo, pode-se utilizar a técnica de **derivação**, donde, oferece como recurso a possibilidade de identificar tais pontos.

A derivada é a representação da taxa de variação de uma função, em relação a um ponto. A partir daí, podemos observar um fator interessante; por exemplo, quando a função está num ponto máximo, existem duas possibilidades, a primeira, sendo a função parando de crescer e em seguinda se tornando indefinida; e a segunda possibilidade quando a função para de crescer e começa a decrescer. Com isso, é importante ressaltar que por definição, quando a derivada (taxa de variação) em um ponto é positiva, a função cresce, e quando negativa a função decresce. Conclui-se que, quando a taxa de variação é zero, a função ou para de crescer ou de decrescer, sendo assim um ponto de máximo ou de mínimo.

Com o uso da derivada, podemos pensar num método de otimização bastante simples, considerando $f(x)$ a função que queremos otimizar e $f'(x)$ sua função derivada, podemos dizer que o conjunto O possui todos os ótimos locais e globais de $f(x)$:

$$O := \{f(x)|f'(x) = 0\} \tag{1.5}$$

Portanto, aplicando um filtro em O para obeter o máximo e mínimo do conjunto, acabamos por obeter o máximo e mínimo de $f(x)$:

$$max(f(x)) = max(O) \tag{1.6}$$

$$min(f(x)) = min(O) \tag{1.7}$$

Então podemos perceber dois problemas; determinar como encontrar os pontos onde a derivada se anule, e determinar se temos de fato todos os pontos.

Considerando por agora, apenas o problema de encontrar os pontos críticos;

Entendendo melhor, a taxa de variação de uma função $y = f(x)$ em relação a x, é dada pela relação $\Delta y/\Delta x$. Sendo esse resultado correspondente a tangente do ângulo formado pela intersecção entre a reta e a curva da função y; o coeficiênte ângular da reta à curva.

Sendo o cálculo da taxa de variação demonstrado pela Figura 1.2.

$$\frac{dy}{dx} = f'(x) = y' = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Figura 1.2: Derivada: taxa de variação dy/dx.

Partindo desse princípio, é intríseco saber que uma função pode ser derivada mais de uma vez, sendo essas derivadas, denominadas de 'primeira derivada', 'segunda derivada' e por ai em diante. De modo que, a segunda derivada é a taxa de variação da primeira derivada. Concluindo-se que dada a função $f(x)$, sua primeira derivada é $df/dx = p(x)$, e sua segunda derivada $dp/dx = s(x)$.

Considerando que a primeira e segunda derivada são as comunmente utilizadas, pode-se agora entender a relação entre elas e a função original.

Já sabido que a primeira derivada representa a taxa de variação de um ponto na curva, a segunda derivada proporciona informações complementares, como por exemplo, se é um ponto de máximo, mínimo ou inflexão, de modo que ela determina a concavidade da curva naquele ponto. Como pode-se ver na Figura 1.3. E de modo construtivo, é importante ressaltar que é importante o estudo do gráfico das derivadas de uma função.

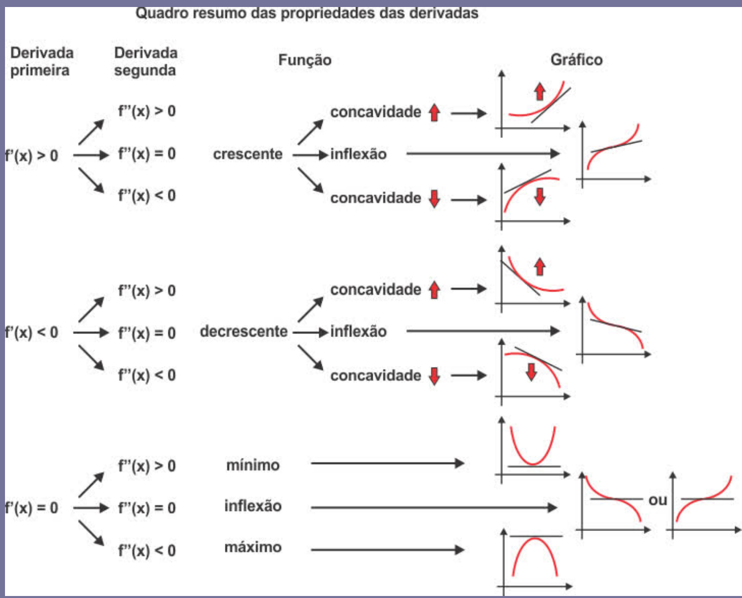

Figura 1.3: Propriedades e relação da primeira e segunda derivada.

1.3 Programando o Método

A própria definição da derivada já nos oferece uma uma visão simples de como ela pode ser implementada num programa de computador. Mas com alguns nuances que devem ser levados em consideração. Segue a tal implementação:

Listing 1.1: Calculo de Derivada em Rust.

```
pub fn derive1x1_v1<F>(f: &F, x: f64) -> f64
where
    F: Fn(f64) -> f64,
{
    (f(x + DBL_EPS) - f(x)) / DBL_EPS
}
```

Essa implementação é ingenua no que se diz respeito a precisão da operação. O uso depende da aplicação, caso se queira prezar por velocidade de calculo e muito pouco sobre precisão, talvez essa solução seja boa o suficiente. O problema com com precisão desta implementação é que não se é levado em consideração o aspecto infinitesimal da variação (ΔX na figura 1.2, ou a variável h na implementação em Rust), já que é por este aspecto que definimos a derivada. Não temos como ter uma variável com tal propriedade num programa de computador. Como consequência, $f(x+h) - f(x)$ já é calculada de forma falha, entregando uma variação diferente da que seria quando se tem uma variável infinitesimal. O que de fato é entregue é a variação um pouco mais a frente do ponto esperado.

A partir dai podemos reformular considerando esse deslocamento, redefinindo da seguinte forma:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} \tag{1.8}$$

Que é equivalente à:

$$f'(x) = \frac{f(x+h) - f(x-h)}{(x+h) - (x-h)} \tag{1.9}$$

Usaremos essa segunda formulação por motivos de custo de multiplicação de de números reais se comparado a somas e subtrações, além da possível perca de precisão. Assim conseguimos compensar o tal deslocamento numa nova função insignificantemente mais custosa e mais precisa:

Listing 1.2: Calculo de Derivada em Rust.

```
pub fn derive1x1_v2<F>(f: &F, x: f64) -> f64
where
    F: Fn(f64) -> f64,
{
    let x1 = x - DBL_EPS;
    let x2 = x + DBL_EPS;

    let y1 = f(x1);
    let y2 = f(x2);
    return (y2 - y1) / (x2 - x1);
}
```

1.4 Otimização de Funções à Várias Variáveis

Capítulo 2

Métodos Clássicos de Otimização

2.1 O Método de Newton

O Método de Newton nos diz que uma sequência $\{x_k\}$ converge para o mínimo de $f(x)$:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \tag{2.1}$$

O que este método faz de fato é encontrar zeros de uma função $g(x)$ a partir da mesma sequência $\{x_k\}$, afim de que x_k converga para uma entrada x^* de $g(x)$, tal que a mesma se anule:

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)} \tag{2.2}$$

$$g(x^*) = 0 \tag{2.3}$$

Com isso temos um método que minimiza uma função que pode ser derivada duas vezes (caso não possa, aproximações de suas primeiras e segundas derivadas podem ser boas o suficiente) a partir de um valor de entrada. O método é simples, entrega muitas vezes ótimos locais próximos ao ponto inicial, mas tem seu destaque pode ser curto e facilmente computável.

Problemas de maximização podem ser vistos sob o seguinte olhar:

$$max(f(x)) = min(-1 * f(x)) \tag{2.4}$$

Que com isto podem ser otimizados pelo Método de Newton também.

O movimento de x_k dentro da sequência, é determinado pela relação das quantidades e propriedades que tanto a primeira quanto a segunda derivada oferecem. As quantidades determinam a velocidade do motivento, e os sinais indicam a direção do movimento. De certa forma podemos ver esse movimento da sequência $\{x_k\}$ como instantes do movimento de uma bola numa ladeira, que no começo de sua descida é acelerada, e, conforme chega ao plano no fim da ladeira, começa a reduzir sua velocidade, até supostamente chegar no ponto mais baixo.

2.2 Outros Métodos

2.3 Programando os Métodos

Método de Newton

A forma mais simples e mais util de implementar o metodo de Newton é na forma de busca das raizes, que uma vez implementada, só precisamos por como entrada a priemira e a segunda derivada da função que desejamos minimizar, já que o método não precisa saber qual a função de fato. A seguir temos a implementação na liguagem de programação Rust:

Listing 2.1: Implementação do Método de Newton para uma variável em Rust.

```
pub fn newton1x1<F>(funcao_derivada: &F, x: f64) -> (usize, f64)
where
    F: Fn(f64) -> f64,
{
    let mut entrada_atual = x;
    let maximo_iteracoes = 100;

    for iteracao_atual in 1..=maximo_iteracoes {
        let diferenca: f64 =
            funcao_derivada(entrada_atual.clone())
            /
            derive1x1(&funcao_derivada, &entrada_atual);

        println!("diferenca: {}", diferenca);
        entrada_atual -= diferenca;

        if diferenca.abs() < 0.0000001 {
            return (iteracao_atual, entrada_atual);
        }
    }

    return (maximo_iteracoes, entrada_atual);
}
```

Os parâmetros da função são:

- Uma função $f : \mathbb{R} \to \mathbb{R}$
- Uma entrada x sendo o chute inicial do otimo.

A função derive1x1 recebe como parametro uma função e um ponto, tendo como retorno a derivada da função entregue, no ponto especificado. Restingindo-se à funções do tipo $f : \mathbb{R} \to \mathbb{R}$, donde deve ser escrita pelo úsuario como bem entender.

2.4 O Método de Newton para Várias Variáveis

Capítulo 3

Os Métodos Modernos de Otimização

3.1 Breve Relato Histórico

3.2 Métodos de Um

3.2.1 O Método - Uma breve descrição

3.2.2 Exemplos Aplicações

3.2.3 Possíveis Aplicações

3.3 Métodos de Dois

3.3.1 O Método - Uma breve descrição

3.3.2 Exemplos Aplicações

3.3.3 Possíveis Aplicações

3.4 Um com o outro

Capítulo 4

Aplicações à Mecânica Celeste

4.1 Entendendo o Problema de N Corpos

4.2 A Otimização na Mecânica

4.3 Resultados Numéricos

Capítulo 5

Demais Resultados

5.1 Outros Resultados

5.1.1

Referências Bibliográficas

[1] *Garcia, Arnaldo. Elementos de Álgebra. Rio de Janeiro. Associação instituto Nacional de Matemática Pura e Aplicada, 2003.*

[2] *Lima, Elon Lajes Álgebra Linear. 7º Edição. Rio de Janeiro; IMPA, 2008.*

[3] *Halmos, Paul R. Espaços Vetoriais de Dimensão Finita. Tradução [de] Guilherme de la Penha. Rio de Janeiro: Campus, 1978.*