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, tal processo é amplamente utilizado quando busca-se valores em conjunto um A (que pode ter restrições), com o objetivo de encontrar uma solução ótima (a melhor resposta para o problema), aplicando os valores de A em uma função objetivo predefinida.

Podendo assim, ser representada da forma como a seguir.

Dada uma função

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

A maximização pode ser definida como; a busca pelo elemento $x^* \in A$, que satisfaz:

$$f(x^*) \geq f(x), \tag{1.2}$$

para todo $x \in A$.

A minimização pode ser definida como; a busca pelo elemento $x^* \in A$, que satisfaz:

$$f(x^*) \leq f(x), \tag{1.3}$$

para todo $x \in A$.

Com isso, podemos agora entender como esse processo pode ser custoso. Iniciando com o fato de que existem os pontos máximos e mínimos (pontos críticos[1]), locais e globais, na imagem 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 na imagem das funções, para a minimização e maximização, respectivamente.

[1] Iremos denotar como pontos críticos apenas aqueles onde a função tem derivada e esta se anula.

Criando, desse modo, 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.

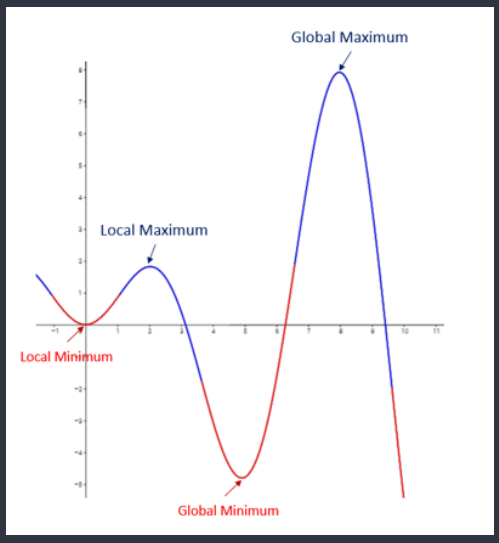

Figura 1.1: Exemplo de pontos críticos locais e globais indicados no gráfico de uma função.
Essa imagem foi coletada de https://www.onlinemathlearning.com.

Como na grande maioria dos casos não é possível conhecer antecipadamente todo o espaço gerado pela função, a busca pelo ótimo pode resultar em pontos críticos locais. Donde, dependendo do problema, tais resultados chegam a ser satisfatórios. No entanto, quando o objetivo é estritamente encontrar os máximos ou mínimos globais, o trabalho necessário acaba sendo mais custoso, pelo fato de que há a incerteza da existência de mais pontos na grande maioria dos problemas.

Ademais, o conjunto A, o qual contém os valores aplicáveis no problema, podem respeitar restrições, fazendo com que o campo de respostas do problema, não seja contínuo.

1.2 Otimização de Funções à Uma Variável Real

Evidentemente, as funções possuem as variáveis dependentes (que representam o objeto da otimização) e as variáveis independentes (que 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, x, que quando aplicados à $f(x)$, 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 os pontos críticos. E nesse sentindo, pode-se utilizar a técnica de **derivação**, que oferece o recurso de identificar tais pontos.

Podemos observar um fator interessante; por exemplo, quando a função está num ponto máximo, existem duas possibilidades, a primeira, a função para de crescer e, em seguinda, torna-se 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 é negativa a função decresce. Conclui-se que, quando a taxa de variação é zero, a função ou para de crescer ou de decrescer, o que configura um ponto de máximo ou de mínimo. Mas também, é quando podo ocorrer o que denominamos ponto de sela (é um comportamento errático da função, que precisa de uma análise mais refinada). Ou em última instância, quando a função é constante.

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, abaixo definido, 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 obter o máximo e mínimo do conjunto, acabamos por obter 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 três problemas:

- Determinar como encontrar os pontos onde a derivada se anule, isto é, os pontos críticos;
- Determinar se temos de fato todos os pontos;
- Classificação dos pontos críticos, que nos revela quais são os máximos e os mínimos, e, ademais, da sua relevância quanto a serem locais ou globais.

Consideraremos, por agora, apenas o problema de encontrar os pontos críticos.

A taxa de variação de uma função, $y = f(x)$, em relação a x, é dada pela relação $\Delta y/\Delta x$. Sendo o cálculo da derivada demonstrado pela Figura 1.2, o que nos leva ao fato de que os pontos críticos provém do conjunto solução da equação $f'(x) = 0$, como definido na equação 1.5. Além disso, caso a função esteja definida num intervalo fechado, temos pelo Teorema do Valor Absoluto, a garantia que ela atingirá o valor dos seus pontos extremos, ou seja, seus valores de máximo e mínimo serão atingidos naquele intervalo. O que culmina nossa busca pelos pontos críticos.

Partindo desse pressuposto, é nítido que a derivada de uma função é também uma função. Portanto, fica evidente que uma função pode ser derivada mais de uma vez, sendo essas derivadas, denominadas de “primeira derivada”, “segunda

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

Figura 1.2: Derivada: taxa de variação dy/dx.
Essa imagem foi coletada de https://brasilescola.uol.com.br.

derivada" e por ai em diante. De modo que, a segunda derivada é a derivada da primeira derivada. Concluindo-se que dada a função $f(x)$, sua primeira derivada é $df/dx = p(x)$, e sua segunda derivada $dp/dx = s(x)$.

Agora vamos tratar da classificação dos pontos críticos.

É sabido que a primeira derivada representa a taxa de variação instantânea de um ponto na curva, e, que, 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 ser visto na Figura 1.3. E, de modo construtivo, devemos ressaltar que é importante o estudo do gráfico das derivadas de uma função, pois é necessariamente através do uso da intrepretação do comportamento da primeira derivada e/ou da segunda derivada, que obtemos a classificação dos pontos críticos seguindo os critérios que são apontados na Figura 1.3.

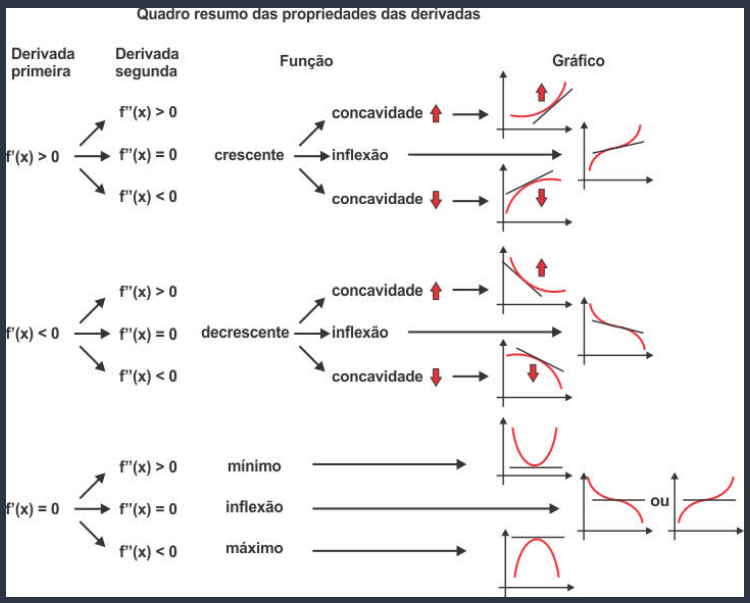

Figura 1.3: Propriedades e relação da primeira e segunda derivada.
Essa imagem foi coletada de https://www.alfaconnection.pro.br.

1.3 Programando o Método

A própria definição da derivada já nos oferece 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:

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

Essa implementação é ingênua no que diz respeito a precisão da operação. O uso, depende da aplicação, caso se queira prezar por velocidade de cálculo e muito pouco sobre precisão, então provavelmente, essa solução seja boa o suficiente. O problema com a 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 à frente do ponto esperado.

A partir daí 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 perda de precisão. Assim, conseguimos compensar o tal deslocamento numa nova função insignificantemente mais custosa, e mais precisa. Como vemos no código a seguir:

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

    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

Escrito por Isaas Newton na obra *De analysi per aequationes número terminorum infinitas* publicada em 1711, o Método de Newton passou por alterações, sendo aprimorado por Joseph Raphson para qualquer tipo de função real, em 1690, sendo novamente publicado em *De metodis fluxionum et serierum infinitarum* no ano de 1736, fazendo com que o dado método também seja amplamente conhecido por o Método de Newton-Raphson.

É válido também salientar que o método não foi apresentado como hoje é trabalhado. Houve a intervenção de vários grandes matemáticos como *Cauchy*, *Fourier* e *Kantorovich*, contribuindo em sua prova.

O Método de Newton, foi desenvolvido com o objetivo de encontrar estimativas para as raízes de uma função. De modo que, a execução do método é feita de forma iterativa, repetindo sempre o mesmo processo, atualizando o mesmo valor.

2.1.1 Entendendo o Método

Este método faz uso do recurso de derivação. Existindo uma relação muito forte com o ângulo da reta tangente à curva, que é o gráfico da função que foi derivada. Ademais, é importante ressaltar a necessidade de um "palpite" inicial, que represente o valor de x, para o qual será buscado uma estimativa da raiz. E com isso vamos entender como o método funciona.

O método em princípio consiste na utilização da reta tangente à um ponto sobre à curva, para gerar valores cada vez mais próximos da raiz daquela função. Analisamos a interseção da reta tangente com o eixo das abscissas. Obtendo o ponto P1(x_p, y_p). O novo valor de entrada na função será construído com base nas coordenadas desse ponto.

A equação da reta que passa por um ponto de coordenada (x_0, y_0) é dada por:

$$(y - y_0) = m(x - x_0). \tag{2.1}$$

E levando em conta que temos como objetivo encontrar x, que é um ponto

sobreposto no eixo x, podemos considerar $y = 0$. Logo:

$$-y_0 = m(x - x_0). \tag{2.2}$$

Com isso sabemos que y_0 é a imagem da função $f(x_0)$, e m representa o ângulo da reta tangente ao ponto (x_0, y_0). Como vimos no Capítulo 1, $m = f'(x_0)$. Desenvolvendo essa equação, temos:

$$-f(x_0) = f'(x_0)(x - x_0), \tag{2.3}$$

$$-f(x_0) = f'(x_0)x - f'(x_0)x_0, \tag{2.4}$$

$$0 = f'(x_0)x - f'(x_0)x_0 + f(x_0), \tag{2.5}$$

$$0 = x - x_0 + \frac{f(x_0)}{f'(x_0)}, \tag{2.6}$$

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{2.7}$$

Exemplificaremos no gráfico a seguir. Considerando a função $f(x) = -x^2+2x$, e que o palpite inicial é $x_0 = 1.5$. $P1$ representa o ponto quando aplicamos x_0 a $f(x)$. Podemos observar que como fruto da intersecção da reta tangente em azul encontramos o valor x_1 que será usado na próxima iteração.

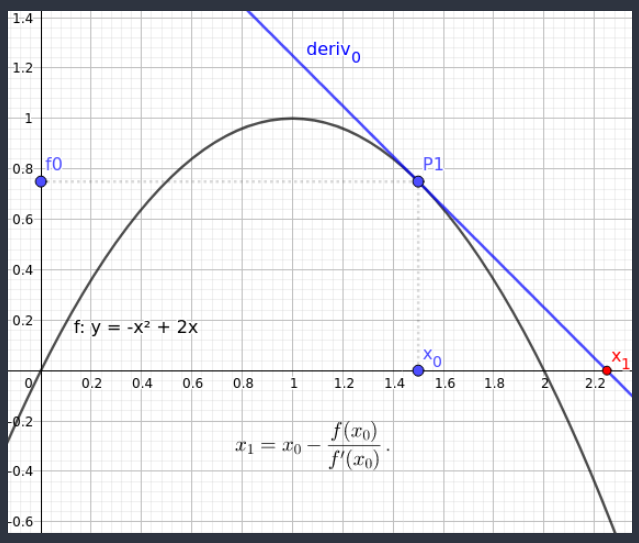

Figura 2.1: Primeira iteração do Método de Newton.

Agora podemos entender melhor como a iteração funcionará. O valor gerado, x_1, será aplicado no mesmo procedimento. E, a partir disso, poderemos construir as seguintes equações:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}, \tag{2.8}$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{2.9}$$

Desse modo, gera-se uma sequência $\{x_k\}$, que deve convergir para a raiz da função (este é o cerne do Teorema de Convergência do Método de Newton). Reconsiderando a função $f(x)$ do exemplo mostrado na Figura 2.1, encontramos a próxima iteração que apresentamos na Figura 2.2:

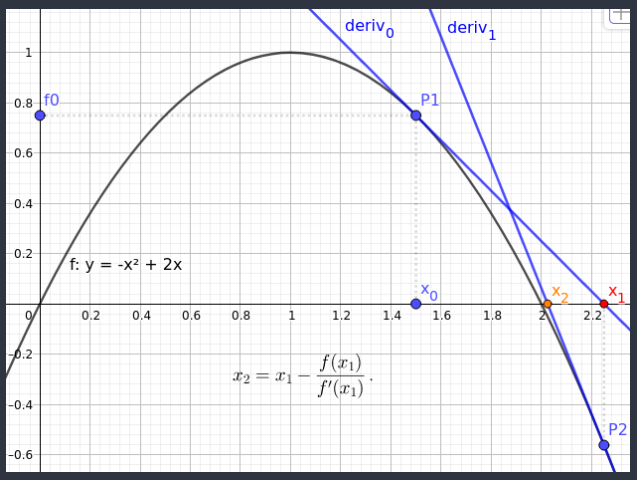

Figura 2.2: Segunda iteração do Método de Newton.

Podemos notar que x_2 é um ponto mais próximo da raiz $f(x)$ do que x_1 ou x_0. O que evidencia a provável convergência da sequência x_k para a raiz.

É válido ressaltar que o denominador da equação 2.9 tem que ser diferente de 0 (ou seja, a reta tangante num ponto P não pode ser paralela ao eixo x). Caso contrário, teríamos que a dada função não possui raiz na proximidade daquele ponto. Ademais, o palpite, denotado por x_0, deve ser suficientemente próximo das raizes, para garantir cerelidade da convergência.

Para aprimorar os resultados expostos, descrevemo-los como no seguinte teorema:

aloo

2.1.2 Encontrando Mínimos

Agora podemos utilizar este método para encontrar mínimos de uma função. O Método de Newton calcula raízes. É possível vincular isto, com o que foi estudado no Capítulo 1. Obtemos os mínimos de uma função que como vimos podem ser representados como raízes de sua derivada. Considerando $g(x)$ uma função duas vezes diferenciável, e tendo como objetivo encontrar seus pontos de mínimo, pode-se utilizar o Método de Newton para resolver tal problema, gerando a seguinte sequência:

$$x_{k+1} = x_k - \frac{g'(x_k)}{g''(x_k)}. \tag{2.10}$$

A convergência da sequência 2.10, é um valor x^* que satisfaz a equação:

$$g'(x^*) = 0. \tag{2.11}$$

O método é simples, entrega muitas vezes ótimos locais próximos ao ponto inicial, mas tem seu destaque, que é ser, facilmente computável.

Problemas de maximização podem ser vistos sob o seguinte olhar:

$$max(f(x)) = min(-1 * f(x)). \tag{2.12}$$

O que faz com que o Método de Newton possa ser utilizado em ambos sentidos no processo de otimizar.

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 movimento, e os sinais, indicam a direção do movimento. De certa forma, podemos interpretar o 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

Com o advento do Método de Newton, acabou surgindo uma família de métodos similares. E como foi visto, é possível a partir dele, encontrar tanto raízes de funções, quanto máximos ou mínimos, tal característica é herdada pelos vários métodos que deviram dele. Dentre eles temos; Métodos Quasi-Newton e Método do Gradiente Descendente. Tais métodos podem ser vistos como uma complementação ao Método de Newton, no entanto, esses fornecem uma grande flexibilidade em como deseja-se utilizar os recursos de precisão e poder computacional.

Métodos Quasi-Newton se fazem necessários, quando nem sempre se tem acesso ou recursos suficientes para se calcular a derivada de segunda ordem, ou até mesmo a de primeira ordem. Já os métodos do Gradiente Descendente, como sugerido pelo nome, utiliza-se majoritariamente, da derivada de primeira ordem em seu processo.

Os quais, descreveremos em alguma escala nas sessões seguintes.

2.2.1 Métodos Quasi-Newton e Gradiente Descendente

Aqui voltaremos a considerar uma função $f(x)$. E considerando que o Método de Newton encontra o $min(f(x))$ através de uma sequência $\{x_k\}$:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}. \tag{2.13}$$

A qual, pode ser reescrita da seguinte forma:

$$x_{k+1} = x_k - \frac{1}{f''(x)} * f'(x_k). \tag{2.14}$$

Sabemos que quem determina a direção da convergência é $f'(x)$, portanto não é completamente necessário o uso de $\frac{1}{f''(x)}$, que tem como principal papel controlar o tamanho do "passo dado" na iteração. De modo que, a maioria dos Métodos Quasi-Newton fazem a substituição da dada fração por aproximações boas o suficiente. E com isso, levando em conta α como a representação dessa aproximação, temos:

$$x_{k+1} = x_k - \alpha * f'(x_k). \tag{2.15}$$

Onde α satisfaz:

$$f(x_{k+1}) = f(x_k - \alpha * f'(x_k)) < f(x_x). \tag{2.16}$$

A partir da equação 2.15, podemos escolher um α, de modo que, seja menos custoso encontrá-lo do que calcular a segunda derivada da função objetivo, ou até, podemos nem calculá-lo, basta considerar um α fixo, e tão pequeno que, quando a restrição 2.16 não for cumprida, já temos uma aproximação boa o suficiente para o mínimo da função.

Podemos dizer que, esse modelo de resolução é útil e flexível o suficiente. Tendo isso em mente, e seguindo para o problema de encontrar o melhor α, já sabemos

que sendo um valor pequeno o suficiente, minimiza a função (mas, não da melhor forma). Já é sabido que o α tem como propósito, regular o tamanho do passo dado nas iterações. Ele pode ser usado em conjunto com outros elementos, afim de otimizar o processo de otimização.

Comumente usamos algum dos seguintes métodos para encontrar o tamanho do passo:

- Um valor fixado para α;
- Um método que atualize α de acordo com alguma situação;
- Um método que escolha um valor ótimo ou quase ótimo para α;
- Um elemento que forneça mais informações sobre a função, trabalhando em conjunto com algum α definido nos métodos anteriores.

Simplesmente considerando α fixo e pequeno pode funcionar, mas não seguramente para qualquer situação. Como por exemplo, uma função que possui "movimentos bruscos", em um escala minúscula. Ou ainda, caso se tenha considerado um ponto inicial muito distante de qualquer ótimo, resultando em um funcionamento extremamente custoso do algoritmo.

Atualizar o valor de α, de acordo com a situação do processo iterativo de otimização, pode ser uma boa opção, quando não se tem tanto poder computacional. Comumente, pode ser encontrado implementações, que iniciam com um α não tão pequeno (isso ajuda a resolver o problema de iniciar com um ponto longe da solução), mas ao passar das iterações, reduz-se o seu valor por algum fator. Fator este, que pode ter alguma relação com a magnitude da derivada, já que, ela pode nos dar uma dica do quão longe o ótimo está do ponto atual, ou ainda, considerar o fator de redução havendo uma relação com a iteração atual. Como por exemplo se considerarmos um $\alpha_0 \in \mathbb{R}^+$ e $k \in \mathbb{N}$ poderemos considerar a k-ésima iteração como:

$$\alpha_{k+1} = \frac{\alpha_k}{k}, \tag{2.17}$$

ou ainda:

$$\alpha_{k+1} = \frac{\alpha_0}{2^k}. \tag{2.18}$$

Vale a pena ressaltar os métodos mais famosos para tal tipo de atualização, que também podem ser chamadas de clássicos por serem um ponto inicial, como por exemplo, o método de Barzilai–Borwein, que pode ser considerado uma atualização ingenua, porém efetiva e barata de realizar o cálculo. O α é dado pela seguinte expressão:

$$\alpha_{k+1} = \frac{\langle \Delta x, \Delta g \rangle}{\langle \Delta g, \Delta g \rangle}, \tag{2.19}$$

onde $\langle \cdot, \cdot \rangle$ significa o produto interno entre vetores (no caso de funções que possuem uma variável apenas, podemos considerar um vetor de apenas uma entrada), e:

$$\begin{aligned} \Delta x &= x_k - x_{k-1}, \\ \Delta g &= g_k - g_{k-1}. \end{aligned} \tag{2.20}$$

Podemos considerar que $g_k = f'(x_k)$ nos entrega a noção de direção de descida, noção essa, que ainda pode ser melhorada, como veremos mais a frente. Então, por agora, podemos ficar com a ideia de que g_k é a direção de descida apenas, independente da forma com que foi obtida.

Tem-se mais alguns métodos semelhantes, como o método Polak-Ribière 1997 e outro gerado pelos estudos de Fletcher e Reeves 1964. **??**.

Quando começamos a observar outras formas de melhorar o valor de α, acabamos por entrar em uma recursão, pois começa um estudo sobre como otimizar um parâmetro específico de um otimizador. Os métodos mais famosos que buscam valores ótimos, ou quase ótimos para α, são os métodos classificados como *line search*. Os quais, normalmente, trabalham com condições específicas sobre a otimalidade de α, como as condições de Wolfe. Tais métodos utilizam-se dos artifícios citados anteriormente, já que, são uma peça básica na construção do otimizador, ou ainda, podem utilizar métodos fora da família dos newtonianos.

Agora, antes de seguirmos para a analise de estatégias para obter um melhor resultado, se faz necessária uma nova observação sobre a estrutura básica dos métodos. Até o momento, viemos considerando apenas a derivada de primeira ordem como sendo a direção certa a ser tomada. Vamos observar a seguinte equação:

$$x_{k+1} = x_k - \alpha * \frac{1}{f''(x_k)} * f'(x_k). \tag{2.21}$$

Se tomarmos $\alpha = 1$, temos o método em sua forma natural. Mas podemos procurar um valor para α que melhore ainda mais a iteração. E, complementando, sabemos sobre as acusações que a derivada de segunda ordem faz sobre a otimalidade, com isso podemos dizer que a direção de otimização em verdade é dada por:

$$g_k = \beta(x_k) * f'(x_k). \tag{2.22}$$

Então podemos reescrever a sequência de iteração na seguinte forma:

$$x_{k+1} = x_k - \alpha * g_k. \tag{2.23}$$

Daí, temos normalmente três opções no que diz respeito ao $\beta(x_k)$, que são:

1. Considerar $\beta(x_k) = \frac{1}{f''(x_k)}$. O que nos dá o Método de Newton novamente, e o α não passaria de um mero ajuste;

2. Considerar $\beta(x_k) \approx \frac{1}{f''(x_k)}$. O que abre um leque de possibilidades no que diz respeito ao cálculo dessa aproximação e tornando-o assim num método Quasi-Newton. Sendo esse formato um dos mais populares. São aplicados

nos métodos de otimização padrão em bibliotecas cientificas, como *scipy* em Python, e *GSL* em C. Tendo como um dos mais famosos o método BFGS (Broyden-Fletcher-Goldfarb-Shanno).

3. Considerar $\beta(x_k)$ neutro. O que possibilita considerar $g_k = f'(x_k)$. Dando-nos o Método do Gradiente Descendente que, como dito anteriormente, normalmente é resolvido usando a estratégia de *line search*.

2.2.2 Simplex

O Método Simplex pode ser classificado como clássico, por ser um dos métodos mais famosos. Formulado por George B. Dantzig, fruto de uma sugestão de T. S. Motzkin, que contribuiu para diversas áreas da matemática.

Os problemas que o Método Simplex resolve fazem parte de um conjunto de problemas de Otimização Linear (ou Programação Linear). Os quais se restringem apenas funções lineares. Além disso, tais problemas genericamente acompanham restrições sobre as entradas da função a ser otimizada.

Um problema de otimização linear pode ser transcrito da seguinte forma. Considerando a função objetivo:

$$Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n, \tag{2.24}$$

e tendo restrições também lineares para a função objetivo, como por exemplo:

$$\begin{gathered} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2, \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m. \end{gathered} \tag{2.25}$$

$$x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0.$$

O tipo de problema que o Simplex resolve, se desenvolve como a seguir:

$$max\{c^t x | Ax \leq b, x \geq 0\} \tag{2.26}$$

A forma de operação desse método é semelhante ao que uma pessoa comumente faria ao se deparar com o problema de forma gráfica. O problema, que é completamente linear, gera retas que acabam por formar regiões de possíveis soluções, e dessas regiões, o que queremos é saber qual a melhor solução. Para isso, bastaria procurar dentro de tal região, onde o valor da função objetivo Z é o maior possível. Uma vez que a região tenha sido construída pelas retas, teremos a característica de ser convexa, como um polígono convexo no plano ou um poliedro convexo no espaço. O que acaba facilitando a busca pelo máximo, dentro dessa região, uma vez que basta olhar os vértices de tal região. Esses fatos podem ser provados e tais demonstrações podem ser encontradas em [??].

Repara-se que o Método Simplex não se utiliza de artifícios e ferramentas do Cálculo, como nos métodos anteriormente apresentados, mas sim, de formas diferentes de analisar o problema, que por certos aspectos é simples. O método completo é constituído por operações em uma matriz específica, denominada Tableau, que representa o problema, o qual não é necessário ser apresentado aqui, por se distanciar da "família" de métodos de otimização que é nos interessa apresentar.

2.3 Programando os Métodos

2.3.1 Método de Newton

A forma mais simples e mais útil de implementar o Método de Newton é na busca das raízes da derivada da função objetivo. Que uma vez implementada precisamos por como entrada a primeira 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 linguagem de programação *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);
}
```

Esta implementação define um procedimento que encontra alguma raiz de uma função f. Que tem os seguintes parâmetros:

- A função f tem como domínio um intervalo $\mathbb{I} \subseteq \mathbb{R}$ e imagem um valor Real;
- E uma entrada x sendo como o "palpite" inicial do ótimo.

Além disso, temos o procedimento *derive1x1*, que recebe como parâmetro uma função e um valor x, tendo como retorno a derivada da função entregue no ponto especificado. Restringindo-se à funções do tipo $f : \mathbb{I} \subseteq \mathbb{R} \to \mathbb{R}$.

2.3.2 Método do Gradiente Descendente

A implementação desse método exige apenas que seja definido o cálculo da derivada da função objetivo, o valor de α, um palpite inicial para o valor de x e a quantidade de iterações desejada. Também, pode ser indicado um valor que se refere à diferença dos dois últimos valores da sequência $\{x_k\}$, podendo assim efetuar a parada da execução do método. A seguir, temos a implementação na linguagem de programação *C++*:

```
// Sendo f(x) = x^4 - 3*x^3 + 2
#include <bits/stdc++.h>
#define decimal long double
using namespace std;
// Calcula a derivada da funcao
decimal df(decimal x) {
        return 4 * pow(x, 3) - 9 * pow(x, 2);
}
int main() {
        // Palpite inicial
        decimal x_proximo = 5.5;
        // Variavel para iterar
        decimal x_atual;
        // Quantidade maxima de iteracoes
        int iteracoes = 100000;
        decimal alpha = 0.00001;
        // Precisao para condicao de parada
        decimal precisao = 0.000000001;
        // Comeca as iteracoes
        while(iteracoes > 0) {
                x_atual = x_proximo;
                x_proximo = x_atual - (alpha * df(x_atual));
                decimal precisao_atual = x_atual - x_proximo;
                // Se atingir a precisao desejada
```

```
            if(abs(precisao_atual) < precisao)
                    break; // Para as iteracoes
            // Conclui uma iteracao
            iteracoes -= 1;
        }
        cout << ``x que minimiza f(x) = '' << x_proximo << endl;
        // Saida do algoritmo:
        // --> x que minimiza f(x) = 2.25
        return 0;
}
```

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] *Liqun Qi, Kok Lay Teo, Xiao Qi Yang. Applied Optimization. Springer, Optimization and control with applications. 2005.*