Capítulo 2

Raízes de equações

2.1 O método da bissecção

A primeira classe de problemas abordados neste curso consiste nos métodos utilizados para obtenção de raízes de equações algébricas e transcendentais. As primeiras buscas por raízes de equações na história da matemática datam de civilizações Babilônias e Egípcias, cujos registros históricos indicam que ambas as civilizações já utilizavam métodos matemáticos para solução de polinômios de primeiro e segundo grau. Esses registros datam de cerca de 1800 aC a 1650 aC. Ao longo da história da matemática a busca por expressões gerais fechadas para estimativa de raízes de polinômios de ordem mais alta ocupou mentes brilhantes ao longo de vários séculos.

Dentre as funções $f(x)$ de interesse nesse tópico, classificamo-as entre funções algébricas e funções transcendentais. As primeiras são aquelas que podem ser formadas por operações algébricas de potências de x que envolvem adição, subtração, multiplicação, divisão e radicalização. Estas podem ser expressas em termos gerais como:

$$f_0 + f_1 y + f_2 y^2 + ... + f_n y^n = 0,$$

em que f_0, f_1, f_2, f_n são polinômios de ordem 0, 1, 2 e n respectivamente, dados por:

$$f_n(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + ... + a_n x^n.$$

Exemplos de equações algébricas são

$$f(x) = \frac{2}{\sqrt{1+3x^4}}, \;\; f(x) = 5x^5 - (3x+5)^{1/3}, \;\; f(x) = \frac{3x-4}{1+\sqrt{2-x^5}}$$

Já as equações transcendentais envolvem funções matemáticas especiais como exponenciais, logaritmos, funções trigonométricas, entre outras. Exemplos de equações transcendentais são:

$$f(x) = \sin(x) - e^{-2x}, \;\; f(x) = \cosh^{-1}(x) + \frac{1}{x}, \;\; f(x) = \ln 2x^5.$$

Problemas físicos relacionados à determinação de um ou mais pontos da variável de entrada x que satisfaçam $f(x) = 0$ aparecem em diferentes campos de aplicação, da engenharia civil e ambiental à engenharia mecânica, elétrica e aeroespacial.

Existem diferentes métodos numéricos destinados à solução deste tipo de problema, cada qual com suas vantagens e desvantagens. O objetivo aqui é começarmos de um ponto

de partida simples que nos permita compreender como estes métodos são concebidos e após a construção de métodos mais rudimentares tentar melhorá-los em função de uma análise crítica criteriosa com relação às deficiências de cada método.

A maneira mais grosseira de localizarmos uma raiz seria por uma busca visual num gráfico que nos mostre o comportamento de $f(x) \times x$. Esse método consistiria em ir ampliando a região do gráfico onde a curva cruza (ou toca) o eixo x para tentar localizar em quais valores de x isso ocorre. Esse método é evidentemente muito grosseiro e tem sua precisão dependente da qualidade da visão do observador. Portanto, precisamos conceber métodos melhores.

Nesse sentido, uma primeira abordagem consiste em definir um intervalo de busca em x. Podemos definir esse intervalo como $x \in [x_i, x_s]$, em que x_i e x_s representam os valores inferior e superior do intervalo. Em alguns contextos poderemos também nos referir a estas variáveis como x_l e x_u, em que os sub-índices l e u denotam *lower* e *upper*. Uma vez definido o intervalo de busca podemos verificar se a função muda de sinal nesse intervalo. Uma forma simples de verificar isso é calculando o produto $f(x_i)f(x_s)$. Caso $f(x_i)f(x_s) < 0$, sabemos que existe pelo menos uma raiz nesse intervalo de busca. Em seguida, podemos dividir esse intervalo pela metade e calcular o ponto médio em x como $x_m = (x_i + x_s)/2$ e checar se a raiz encontra-se no intervalo $[x_i, x_m]$ ou no intervalo $[x_m, x_s]$. Essa verificação pode ser feita da mesma maneira, observando os sinais de $f(x_i)f(x_m)$ e $f(x_m)f(x_s)$. Uma vez descoberto o novo intervalo de localização da raiz vamos refinando essa divisão pela metade sucessivamente até termos um intervalo tão pequeno quanto uma tolerância. Essa é a ideia do método da bissecção.

Uma das vantagens claras desse método é sua simplicidade em termos de implementação. Outra vantagem do método da bissecção é que o número de iterações necessárias para chegarmos à solução dentro de uma certa tolerância é conhecida à priori. Para compreender isto, note que na iterção 0, ou seja, no momento em que definimos o intervalo de busca inicial, o erro máximo da nossa solução $E^0_{a,max}$ é Δx, o que significa que na pior das hipóteses, caso nossa raiz esteja dentro do intervalo, a distância máxima entre ela e o ponto mais extremo do intervalo é o próprio tamanho do intervalo. Dessa forma, podemos dizer que:

$$E^0_{a,max} = \Delta x,$$

porém, já na primeira divisão (iteração 1) esse erro máximo cai para

$$E^1_{a,max} = \frac{\Delta x}{2}.$$

Na próxima divisão erro erro cai pela metade, para

$$E^2_{a,max} = \frac{\Delta x}{2^2}.$$

Na n-ésima iteração, o erro vai para

$$E^n_{a,max} = \frac{\Delta x}{2^n}. \tag{2.1}$$

Para sabermos o número de iterações necessárias para reduzir esse erro a um certo valor desejado $E_{a,d}$, basta igualarmos $E^n_{a,max} = E_{a,d}$ e isolar n da equação (2.1):

$$n = \log_2 \left(\frac{\Delta x}{E_{a,d}} \right).$$

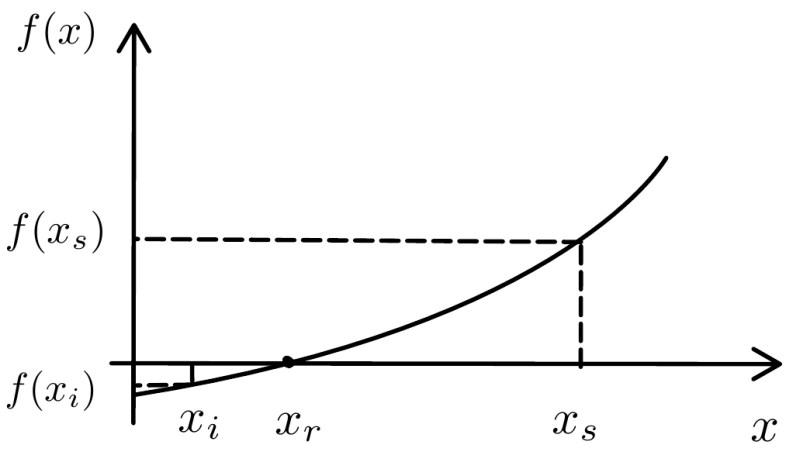


Figura 2.1: Ilustração do efeito da proximidade dos valores de $f(x_i)$ e $f(x_s)$ em relação ao eixo y na distância da raiz x_r em relação aos extermos do intervalo de busca.

2.2 O método da falsa posição

Uma desvantagem clara do método da bissecção é que ele não leva em consideração por exemplo a proximidade da função $f(x)$ com relação ao eixo x nos extermos do intervalo. Caso $f(x_i)$ esteja mais próxima da origem do eixo y do que $f(x_s)$, a raiz deverá estar mais perto de x_i do que de x_s (ver figura 2.10) e isso não é levado em consideração pelo método da bissecção.

Uma primeira forma de incorporar essas informações na construção de um novo método consiste em ligar por uma reta $f(x_i)$ e $f(x_s)$ e usando o princípio da semelhança de triângulos aplicado aos triângulos formados pelo cruzamento dessa reta com o eixo x (ver figura 2.11), obter a seguinte expressão para o valor de x desta reta que cruza exatamente o eixo em questão:

$$x_r = x_s - \frac{f(x_s)(x_i - x_s)}{f(x_i) - f(x_s)}. \tag{2.2}$$

A substituição da equação (2.2) no local do cálculo de x_m pelo método da bissecção é a essência do método da falsa posição, que agora leva em conta a proximidade da função $f(x)$ com relação à origem do eixo y nos extremos do intervalo de busca. Esse método tende a precisar de um número menor de iterações para convergir.

2.2.1 Exercícios: bissecção e falsa posição

Baseado nessa contextualização, sua tarefa consiste em escrever um programa de computador (FORTRAN, C++ ou Python) que encontre a raíz real das seguintes equações utilizando o método da bissecção e da falsa posição:

$$f(x) = -0.5x^2 + 2.5x + 4.5 \text{ com aproximações iniciais: } x_l^0 = 5, x_u^0 = 10 \tag{2.3}$$

$$f(x) = 5x^3 - 5x^2 + 6x - 2 \text{ com aproximações iniciais: } x_l^0 = 0, x_u^0 = 1 \tag{2.4}$$

$$f(x) = -25 + 82x - 90x^2 + 44x^3 - 8x^4 + 0.7x^5 \text{ com: } x_l^0 = 0.5, x_u^0 = 1 \tag{2.5}$$

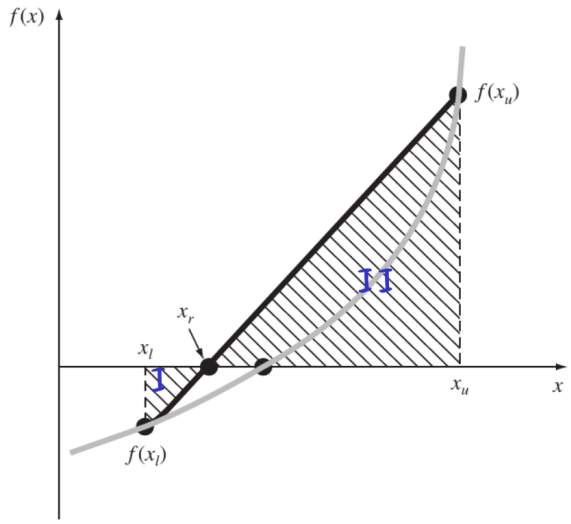


Figura 2.2: Representação visual dos triângulos formados pelo cruzamento da reta que conecta os valores da função $f(x)$ nos extermos do intervalo com o eixo x utilizada para dedução do método da falsa posição.

Obtenha também a primeira raiz não-trivial para a seguinte equação transcendental:

$$\sin(x) = x^3 \text{ com aproximações iniciais: } x_l^0 = 0.5, x_u^0 = 1 \tag{2.6}$$

e a raiz real positiva de

$$\ln(x^4) = 0.7 \text{ com aproximações iniciais: } x_l^0 = 0.5, x_u^0 = 2 \tag{2.7}$$

Seu programa deverá:

1. Resolver a mesma equação pelos dois métodos até obter um erro relativo para ambos os métodos dentro da mesma faixa de tolerância;
2. Gerar uma saída de texto tabelando o erro relativo em função do número de iterações para cada um dos métodos;
3. Plotar um gráfico comparando esses erros para cada um dos métodos para cada equação resolvida;

2.3 O método da falsa posição modificado

Vimos até esse momento o uso de alguns métodos intervalares para lidar com o problema de busca de raízes, dentre eles o método da bissecção e o método da falsa posição. Chamamos esses métodos de intervalares pelo fato de que partem da escolha inicial de um intervalo fechado onde presume-se a localização das raízes que estamos buscando. Veremos mais a seguir que nem todos os métodos numéricos de buscas de raízes operam dessa maneira.

Pelo que vimos até aqui entendemos o método da falsa posição como uma evolução do método da bissecção, uma vez que o mesmo incorpora características da função $f(x)$ que o método da bissecção não percebe a fim de localizar mais rapidamente a raiz. No exemplo

utilizado em sala de aula para ilustrarmos a vantagem do método da falsa posição sobre o método da bissecção o método da falsa posição se mostrou vantajoso. No entanto, para algumas funções isso não é bem verdade.

Para a função $f(x) = x^{10} - 1$, cujas raízes múltiplas sabemos ser $x_r = 1$, a aplicação do método da falsa posição leva a uma convergência mais lenta, como visto em sala de aula. O que nos leva a máxima de que as generalizações são muito perigosas no campo dos métodos numéricos. O motivo para o pior desempenho do método da falsa posição nesse exemplo é que o mesmo leva a prisão de uma das extremidades de busca ao longo do processo, como ilustrado na figura (2.6). Esse fenômeno da extremidade presa é uma característica da forma da função $f(x)$ escolhida nesse caso.

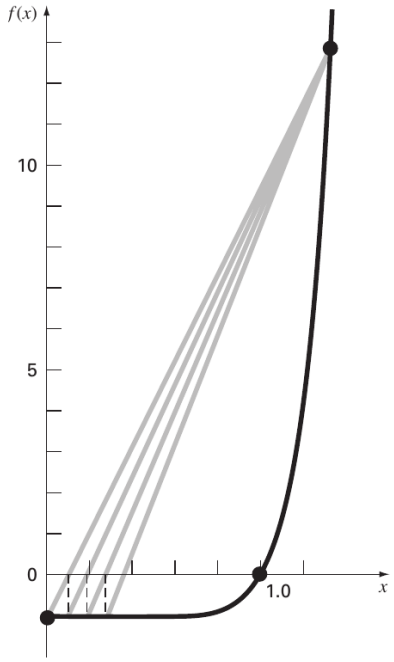


Figura 2.3: Ilustração de uma extremidade presa no processo de busca pelo método da falsa posição

Uma variante natural do método da falsa posição para lidar com esse problema é simplesmente aplicar um contador ao nosso código e caso identifiquemos a prisão de uma raiz ao longo do processo de busca, damos um chega pra lá nessa extremidade presa e avançamos. Com isso acabamos de criar o *método da falsa posição modificado*. O problema dessa abordagem força bruta é que dependendo do *chega-pra-lá* dado na extremidade presa a gente pode fazer com que o intervalo de busca encolha subitamente de uma iteração para outra e pule pela raiz que está buscando sem vê-la no caminho.

2.4 O método de Newton-Raphson

É interessante recapitularmos sempre o caminho percorrido para que entendamos não só as diferenças de concepção, performance e aplicação dos métodos vistos, mas percebamos também o processo criativo por trás da criação de um novo método numérico. A atividade do numericista é também uma atividade de criação além do óbvio aspecto técnico envolvido.

Além dos métodos intervalares vistos até o momento, temos também os métodos abertos, nos quais partimos de um valor inicial da variável de busca $x = x_0$ e vamos evoluindo-a incrementalmente ao longo de um caminho visando chegar nas raízes a frente de x_0. Dos métodos abertos, o mais famoso é o *método de Newton-Raphson*. Nesse método usamos a definição de derivada como reta tangente ao ponto e vamos caminhando ao longo da própria função projetando sua derivada no eixo x até encontrarmos a raiz. Fica mais fácil compreender o método por meio de uma figura. Para isso, considere a figura (2.4). Nessa figura, podemos ver que a derivada $f'(x)$ é dada por:

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}} \longrightarrow x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{2.8}$$

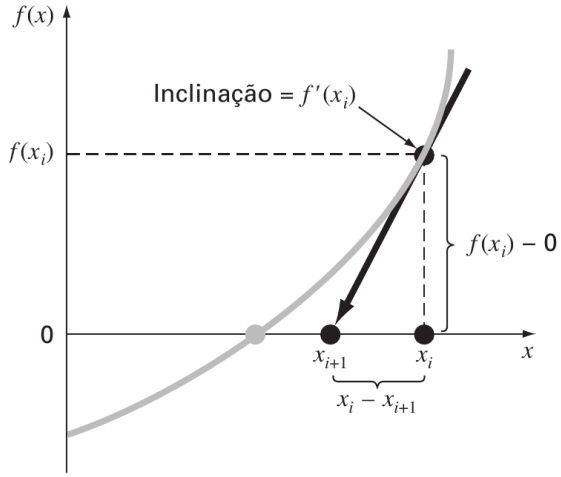


Figura 2.4: Projeção da derivada no alvo ao longo do eixo x pelo método de Newton-Raphson para evoluir rumo à raiz iterativamente

Essa fórmula para evolução de x é conhecida como *fórmula de Newton-Raphson* e é um dos métodos mais utilizados para encontrar raízes de funções cuja cara conhecemos bem.

2.5 O método da secante e da secante modificada

Em alguns problemas temos apenas os valores numéricos de $f(x)$ na forma de uma grande tabela de números. Nesses casos, uma variante natural do método de Newton-Raphson é o método da secante, que calcula a derivada $f'(x)$ numericamente adicionando um ponto extra x_{i-1} para dar partida no método e gerando a seguinte fórmula:

$$x_{i+1} = x_i - \frac{f(x_i)(x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}, \tag{2.9}$$

em que as variáveis x_{i-1} e x_i utilizadas para estimativa numérica de $f'(x_i)$ estão representadas na figura (2.5).

Em seguida, vimos também que é possível modificar o método da secante para padronizar a distância entre x_i e x_{i-1} por meio de uma pequena perturbação δx_i de tal sorte que $x_{i-1} = x_i - \delta x_i$, o que nos leva ao método da secante modificado, em que a solução evolui

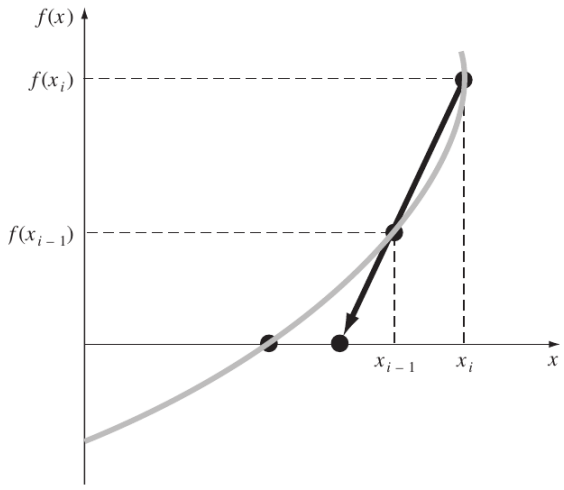


Figura 2.5: Adição de um ponto extra para estimativa numérica da derivada função pelo método da secante.

de forma aberta numa busca incremental iterativa de acordo com a seguinte fórmula de recorrência

$$x_{i+1} = x_i - \frac{\delta x_i f(x_i)}{f(x_i) - f(x_i - \delta x_i)}. \tag{2.10}$$

2.6 Raízes múltiplas

Todos esses métodos abertos podem levar a problemas em contextos de raízes múltiplas. O motivo para isso é simples: a derivada $f'(x)$ é nula nesses pontos para funções que possuem raízes múltiplas. Além disso, sabemos que raízes de multiplicidade par levam a um comportamento da função $f(x)$ no qual ela não chega a cruzar o eixo na raiz, o que também nos leva a problemas na aplicação dos métodos intervalares.

Uma solução para este problema é a aplicação do *método de Newton-Raphson modificado para raízes múltiplas* que se vale do fato de que para uma função $f(x)$ de múltiplas raízes, podemos construir uma função $u(x)$ composta por $f(x)$ e $f'(x)$ que possui as mesmas raízes de $f(x)$ e é dada por:

$$u(x) = \frac{f(x)}{f'(x)},$$

de tal sorte que pelo método de Newton-Raphson teríamos

$$x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)}.$$

Derivando $u(x)$ com relação à x e manipulando algebricamente o resultado para escrevermos x_{i+1} em função de x_i e $f(x_i)$ chegamos na fórmula de Newton-Raphson modificada para raízes múltiplas, dada por:

$$x_{i+1} = x_i - \frac{f(x_i)f'(x_i)}{f'(x_i)^2 - f(x_i)f''(x_i)} \tag{2.11}$$

2.6.1 Exercício: Método de Newton-Raphson para raízes múltiplas

Escolha um polinômio de quarta ordem de sua preferência e na construção desse polinômio conceba-o de tal sorte que o mesmo possua raízes múltiplas. Aplique na mão o método de Newton Raphson modificado para raízes múltiplas para encontrar todas as suas raízes de tal sorte que o erro aboluto fique abaixo de 1% para todas as raízes.

Nessa tarefa, como você montará o polinômio de antemão, você já possui a vantagem de conhecer todas as raízes que está buscando. Dessa forma, você poderá escolher um valor inicial para partir para a sua busca incremental já próximo de onde deverá chegar a fim de não precisar fazer muitas iterações na mão. Tente limitar suas iterações a um máximo de 4 para cada raiz.

A ideia dessa atividade é fazer com que você pare e aplique com calma, manualmente o método numérico de Newton-Raphson modificado para raízes múltiplas a fim de compreender a estratégia de construção e aplicação do método antes de implementar a solução deste problema no computador.

Você deverá digitar essa tarefa em LaTeX, mostrando o passo a passo de seus cálculos. Ao final da solução você deverá organizar seus resultados parciais na forma de uma tabela que mostre como sua solução evolui ao longo das iterações até os valores corretos das raízes do polinômio escolhido juntamente com a evolução do erro verdadeiro.

2.7 Métodos especiais para polinômios e aplicáveis à raízes complexas

Uma outra classe importante de métodos numéricos voltados à solução de zeros de funções, são os métodos desenvolvido especificamente para obtenção de raízes de polinômios. Esses métodos são particularmente úteis no sentido em que atacam por uma perspectiva numérica problemas que vem sendo explorados no campo analítico há muitos anos pelos matemáticos. A procura por fórmulas fechadas para obtenção de raízes de polinômios de ordem geral n é uma busca antiga da comunidade matemática, pautada não só pelas diversas aplicações de polinômios na enunciação de problemas práticos, mas também pela elegância e generalidade dos polinômios como estruturas matemáticas.

A fim de explorarmos a relação entre os polinômios e a prática de engenharia, considere um determinado sistema físico, regido por uma equação diferencial ordinária do tipo:

$$a_2\frac{d^2y}{dt^2} + a_1\frac{dy}{dt} + a_0y = F(t), \tag{2.12}$$

em que a_2, a_1, a_0 representam coeficientes constantes, y representa uma variável cujo comportamento transiente desejamos saber e $F(t)$ é um termo forçante. A equação (2.12) é um modelo geral para diferentes classes de problemas de engenharia que vão desde a modelagem de um problema de vibrações utilizando um sistema massa-mola-amortecedor até o problema de determinação de correntes transientes é um circuito RLC, também conhecido como um oscilador harmônico. Um caso particular de interesse consiste na solução da equação (2.12) no contexto em que $F(t) = 0$. Nesse caso, o que estamos afirmando é que o sistema não encontra-se sujeito a uma ação do mundo externo, de tal sorte que a solução dessa equação no limite $F(t) \to 0$ revelará aspectos fundamentais intrínsecos à estrutura do sistema físico de interesse. Por esse motivo, chamamos essa solução de solução geral e dizemos que a equação diferencial em questão é uma EDO homogênea.

No contexto de uma equação diferencial ordinária de segunda ordem, linear, com coeficientes constantes e homogênea, a solução geral é dada por $y = e^{rt}$, em que r é um coeficiente a ser determinado. Essa forma geral é resultado da cara da nossa equação diferencial que relaciona a função $y(t)$ que desejamos conhecer com suas derivadas diretamente por meio de coeficientes. A função exponencial é a melhor candidata a solução, uma vez que as derivadas dessa função envolvem ela mesmo a menos de coeficientes decorrentes da aplicação da regra da cadeia. Dessa forma, substituindo a proposta de solução geral $y(t) = e^{rt}$ na equação diferencial de segunda ordem homogênea que desejamos resolver, obtemos:

$$a_2 r^2 e^{rt} + a_1 r e^{rt} + a_0 e^{rt} = 0, \tag{2.13}$$

o que nos leva à

$$a_2 r^2 + a_1 r + a_0 = 0. \tag{2.14}$$

A equação (2.14) é chamada de equação característica ou polinômio característico do sistema. A solução desse polinômio vai gerar valores de r, dependentes dos coeficientes a_0, a_1, a_2 que atendam a física regida por essa equação. Mais ainda, como este é um polinômio de ordem n, em que n é a ordem da equação diferencial a ser resolvida, teremos n raízes para o polinômio característico. Dessa forma, a solução geral da equação homogênea vai ser dada por uma combinação linear de soluções com a cara $y(t) = e^{rt}$ com as diferentes raízes obtidas para r pela solução do polinômio característico. Por exemplo, para o caso em que $n = 2$ e as raízes para r são denotadas por r_1, r_2, a solução da equação homogênea é dada por:

$$y(t) = c_1 e^{r_1 t} + c_2 e^{r_2 t}, \tag{2.15}$$

em que as constantes c_1 e c_2 são determinadas a partir das condições de contorno do problema.

2.7.1 O método de Müller

Agora, que a relação entre polinômios e problemas práticos de engenharia foi introduzida, vamos investigar alguns métodos numéricos destinados especialmente à obtenção de raízes de polinômios. O primeiro desses métodos a ser apresentado aqui é o método de Müller, concebido para ser uma extensão do método da secante, em que ao invés de seccionarmos a função $f(x)$, cuja raiz desejamos obter, por uma reta utilizando dois pontos, fazemos isso por meio de uma parábola utilizando três pontos, como ilustrado na figura (2.6).

A essência do método de Müller consiste em descobrir os coeficientes dessa parábola e então projetá-la no eixo x para descobrir onde ela tocará o eixo. Em seguida evoluímos iterativamente esses passos de construção e projeção da parábola até chegarmos o mais perto possível da raiz. Para compreendermos como o método é construído considere a função $f_2(x) = a(x-x_2)^2+b(x-x_2)+c$, em que a, b, c são constantes a serem determinadas e x_2 é um ponto ao longo do eixo x que passa sobre a função $f(x)$ no ponto $f(x_2)$. É importante notar aqui que $f(x) \neq f_2(x)$. Quando falamos da função $f(x)$ estamos falando da função que queremos obter a raiz, já $f_2(x)$ é a parábola criada por nós para representar o comportamento de $f(x)$ nas proximidades da raiz que estamos buscando. Dessa forma, o primeiro passo para obtenção dos valores de a, b, c que atendam essa demanda consiste em montar o seguinte sistema de equações:

$$\begin{aligned} f(x_0) &= a(x_0 - x_2)^2 + b(x_0 - x_2) + c \\ f(x_1) &= a(x_1 - x_2)^2 + b(x_1 - x_2) + c \\ f(x_2) &= a(x_2 - x_2)^2 + b(x_2 - x_2) + c, \end{aligned} \tag{2.16}$$

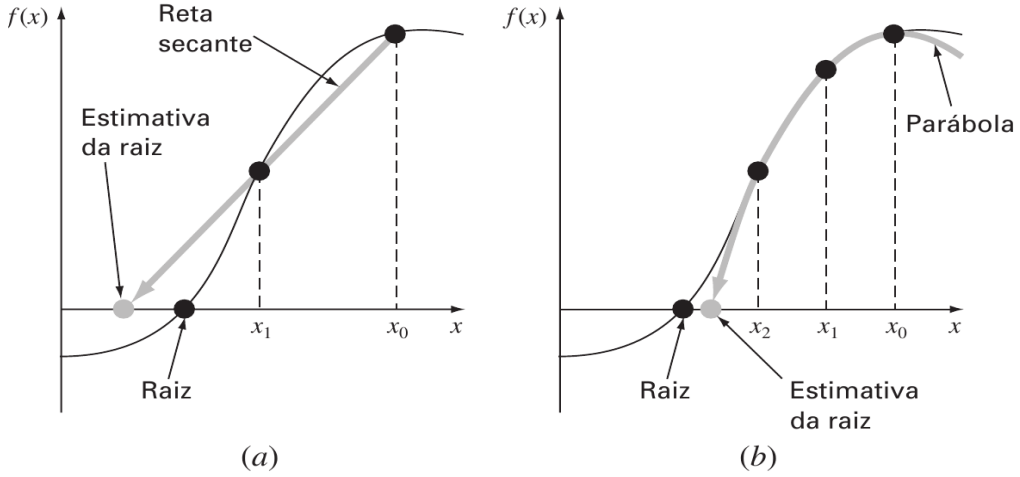


Figura 2.6: Diferença entre o método da secante (a) e o método de Müller (b).

em que x_0, x_1, x_2 são pontos inicialmente escolhidos pelos quais a parábola que estamos buscando deve passar. Da última equação no sistema (2.16) é fácil ver que $c = f(x_2)$. Utilizando essa informação e substituindo-a nas outras equações do sistema, após uma série de manipulações algébricas é possível mostrar que a, b, c são dadas por:

$$a = \frac{\delta_1 - \delta_0}{h_1 + h_0}, \quad b = ah_1 + \delta_1, \quad c = f(x_2), \tag{2.17}$$

em que $h_0, h_1, \delta_0, \delta_1$ são dados por:

$$h_0 = x_1 - x_0, \quad h_1 = x_2 - x_1, \quad \delta_0 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \quad \delta_1 = \frac{f(x_2) - f(x_1)}{x_2 - x_1}. \tag{2.18}$$

Combinando as equações (2.17) e (2.18) é possível escrever a, b, c em termos de x_0, x_1, x_2, $f(x_0)$, $f(x_1)$ e $f(x_2)$. Isso significa que uma vez que saibamos a cara da função $f(x)$ cujas raízes queremos encontrar, basta utilizarmos as equações (2.17) e (2.18) para descobrir os coeficientes a, b, c necessários para montar a parábola que intercepta essa função $f(x)$ nas proximidades da raiz. A finalização do método consiste em retomar a forma de escrevermos a função $f_2(x)$ aplicando a substituição de variáveis $x^* = x - x_2$, de tal sorte que

$$f_2(x) = ax^{*2} + bx^* + c, \tag{2.19}$$

cujas raízes são

$$x^* = x - x_2 = x_{i+1} - x_i = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{2.20}$$

Note porém que a aplicação da fórmula quadrática clássica como a conhecemos pode levar a um potencial problema numérico no limite em que $a \to 0$. Nesse caso, o discriminante $\Delta = b^2 - 4ac \to b^2$, o que levaria a uma subtração de valores muito próximos para a operação do numerador $-b + \sqrt{\Delta}$, o que poderia gerar uma propagação indesejada de erros de arredondamento. Dessa forma, utilizamos para determinação das raízes da parábola $f_2(x^*)$ uma versão alternativa da fórmula quadrática, obtida pela aplicação das fórmulas de Viète, dada por

$$x^* = x_{i+1} - x_i = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}. \tag{2.21}$$

Essa versão, pouco familiar, da fórmula quadrática é famosa por ser aplicada justamente no método de Müller. Note ainda que a solução que buscamos x_{i+1} evolui de acordo com a seguinte fórmula:

$$x_{i+1} = x_i - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}. \tag{2.22}$$

Como desejamos caminhar o mais rápido possível rumo à solução, precisamos minimizar as distâncias entre x_{i+1} e x_i. Pelo fato da parábola ter duas raízes, precisamos escolher uma delas para solução que buscamos. Fazemos isso escolhendo a opção de soma no denominador $b + \sqrt{\Delta}$, o que evita a subtração de termos muito próximos no denominador e minimiza a diferença entre x_{i+1} e x_i acelerando a convergência do método na busca da raiz pelo método de projeção da parábola sobre o eixo x. Uma outra vantagem do método é que ele não está restrito apenas a raízes reais, mas vale também para raízes complexas uma vez que não existe nenhuma restrição com relação à positividade do discriminante da parábola construída para representar o comportamento de $f(x)$ nas vizinhanças da raiz.

2.7.2 Programa para casa: método de Müller × método da secante

Considere um polinômio de ordem 5, dado por

$$f(x) = (x-1)(x-3)(x-5)(x-7)(x-9), \tag{2.23}$$

cujas raízes sabemos ser $x_r = [1, 3, 5, 7, 9]$. Para esse cenário, escreva um programa de computador que permita ao usuário escolher entre os métodos da secante e de Müller para encontrar aproximações numéricas dessas raízes reais. O programa deverá iniciar a busca a partir de valores definidos pelo usuário. Para fins de análise, considere os seguintes valores fixos pré-definidos de x_0, próximos das raízes, conforme a lista:

$$x_0 = [0.5,\ 2.0,\ 4.1,\ 6.5,\ 8.4]$$

Para cada valor de x_0, os métodos deverão construir os pontos adicionais conforme descrito:

- Método da Secante: utilizar os pontos iniciais x_0 e $x_1 = x_0 + \delta x$;
- Método de Müller: utilizar os pontos iniciais x_0, $x_1 = x_0 + \delta x$ e $x_2 = x_1 + \delta x$;

Utilize o valor de $\delta x = 0.05$ para ambos os métodos. O programa deverá calcular e registrar, a cada iteração: a estimativa atual da raiz e o erro relativo entre duas iterações sucessivas.

Os dados de cada simulação devem ser salvos em arquivos .dat, e ao final, o programa (ou um script auxiliar) deverá produzir um gráfico comparando a evolução do erro relativo em função do número de iterações para os dois métodos, com curvas separadas para cada valor de x_0. O gráfico final deverá utilizar escala logarítmica no eixo vertical para melhor visualização das ordens de grandeza dos erros.

O objetivo do exercício é comparar a eficiência e o comportamento numérico dos dois métodos quando aplicados a um polinômio com múltiplas raízes reais bem espaçadas, a partir de diferentes aproximações iniciais.

2.7.3 O método de Bairstow

Um dos esquemas numéricos mais robustos para a determinação de raízes de polinômios é o método de Bairstow. Esse método foi proposto pelo Engenheiro Aeronáutico Leonard Bairstow em um livro de Aerodinâmica Aplicada publicado em 1920. Um fato interessante é que o método é apresentado apenas no apêndice do livro. Esse esquema numérico utiliza apenas álgebra real e ainda assim consegue fornecer tanto raízes reais quanto raízes complexas de polinômios de ordem n geral.

A ideia central do método consiste em dividir o polinômio $f(x)$ cujas raízes desejamos obter por um fator quadrático $D(x) = x^2 - rx - s$, em que r e s são coeficientes reais a serem determinados, para em seguida avaliar o resto dessa divisão. Caso o resto não seja nulo, aplica-se o método de Newton-Raphson para ajustar-se os valores de r e s até que o divisor se anule. Quando isso ocorre, aplica-se a fórmula quadrática sobre o divisor $D(x)$ para obtenção de um par de raízes do problema. Esse processo é então repetido para o polinômio resultante até que todas as raízes sejam determinadas.

Para que possamos compreender como esses passos são executados a fim de que entender a construção do método, precisamos falar um pouco sobre *deflação polinomial*, termo dado ao processo de divisão de um polinômio por outro. Para isso, vamos começar com um exemplo simples. Considere o seguinte polinômio $f(x)$:

$$f(x) = (x+1)(x-4)(x-5)(x+3)(x-2), \tag{2.24}$$

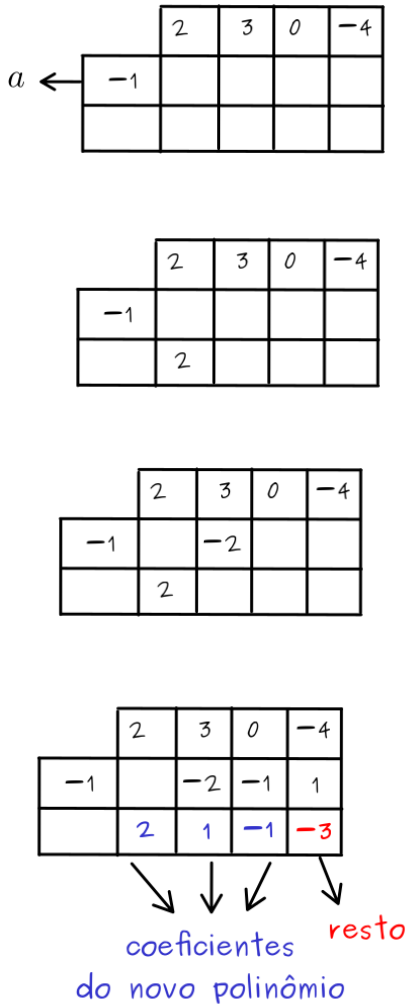
sabemos que este é um polinômio de quinta ordem, com 5 raízes reais, dadas por $x_r = [-3, -1, 2, 4, 5]$. Caso façamos a divisão desse polinômio por um monômio do tipo $(x-t)$, em que $t = x_r$, o resto dessa divisão deve ser nulo. Entretanto, caso façamos a divisão por um monômio no qual $t \neq x_r$, obteremos um resto dessa divisão. De um modo geral, para o caso da divisão de um polinômio $f(x)$ de grau n por um monômio $D(x)$ temos como resultado um polinômio $Q(x)$ de grau $n-1$ com um resto $R(x)$ que é na verdade um polinômio de grau zero. Em outras palavras, temos que:

$$\frac{f(x)}{D(x)} = Q(x) + R(x). \tag{2.25}$$

O processo de divisão de um polinômio por outro é conhecido na literatura como *deflação polinomial* e o algoritmo utilizado para realizar essa deflação é nomeado de *divisão sintética*. Um algoritmo famoso de divisão sintética aplicado à divisão de polinômios por monômios é o algoritmo de Briot-Ruffini. Para ilustrarmos o princípio de funcionamento do algoritmo de Briot-Ruffini considere a divisão do polinômio $f(x) = 2x^3 + 3x^2 - 4$ pelo monômio $D(x) = x+1$. Esse algoritmo pode ser enunciado por meio dos seguintes passos:

Algoritmo de Briot-Ruffini

1. Reescreve-se $D(x)$ na forma $D(x) = x - (-1)$ e chamamos o fator -1 de a;
2. Transcrevemos os coeficientes de $f(x)$ e a na seguinte forma:
3. Em seguida, passa-se o primeiro coeficiente para a última linha:
4. Multiplica-se então esse primeiro coeficiente por a e passa-se o resultado para cima e para a próxima casa à direita:



5. Soma-se os valores da nova coluna e repete-se os passos 3 e 4 para as demais casas até completar-se toda a tabela:

Dessa forma, para esse exemplo específico, temos que a divisão do polinômio $f(x) = 2x^3 + 3x^2 - 4$ pelo monômio divisor $D(x) = (x + 1)$ resulta num novo polinômio $Q(x) = 2x^2 + x - 1$ com um resto $R(x) = -3$, ou seja:

$$2x^3 + 3x^2 - 4 = (x + 1)(2x^2 + x - 1) - 3 \rightarrow \frac{f(x)}{D(x)} = Q(x) + R(x). \tag{2.26}$$

Note que no caso da divisão por um monômio o resto é um polinômio de grau zero. Podemos generalizar essa relação para processos de deflação polinomial envolvendo divisores de ordem mais alta. Para a divisão de um polinômio por um fator quadrático, o resto é um polinômio de grau 1. De um modo geral, a divisão de um polinômio de grau n por um polinômio de grau $m < n$ gera um polinômio resultante de grau $n - m$ e um resto de grau $m - 1$.

Uma forma mais conveniente de expressarmos esse processo de divisão sintética consiste em escrever o polinômio $f(x)$ de grau n como um polinômio $f_n(x)$ dado por

$$f_n(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n, \tag{2.27}$$

em que $a_0, a_1, a_2, ..., a_n$ são coeficientes reais. Nesse caso, a divisão deste polinômio por um monômio geral $(x - t)$ leva como resultado um polinômio $f_{n-1}(x)$ com novos coeficientes mais um resto. Escrevendo $f_{n-1}(x)$ como:

$$f_n(x) = b_1 + b_2x + b_3x^2 + ... + b_nx^{n-1} \tag{2.28}$$

e utilizando a relação final expressa em (2.26), temos:

$$a_0 + a_1x + a_2x^2 + ... + a_nx^n = \left(b_1 + b_2x + b_3x^2 + ... + b_nx^{n-1}\right)(x - t) + b_0. \tag{2.29}$$

Note que aqui chamamos o resto $R(x) = b_0$. Comparando os coeficientes das potências de x na equação (2.29) é possível obter as seguintes relações de recorrência para calcular os coeficientes $b_i's$:

$$\begin{aligned} b_n &= a_n \\ b_i = a_i + b_{i+1}, &\quad \text{para: } i = (n-1) \to 0. \end{aligned} \tag{2.30}$$

As relações de recorrência expressas em (2.30) são equivalentes à aplicação do algoritmo de Briot-Ruffini para a divisão de um polinômio geral por um monômio. Agora que todos os fundamentos foram colocados, podemos então apresentar o processo de construção do método de Bairstow. A ideia geral é realizar o processo de divisão sintética do polinômio $f_n(x)$ pelo fator quadrático $x^2 - rx - s$, de tal sorte que:

$$a_0+a_1x+a_2x^2+...+a_nx^n = \left(b_2 + b_3x + b_4x^2 + ... + b_nx^{n-2}\right)(x^2-rx-s)+[b_0+b_1(x-r)]. \tag{2.31}$$

Note que agora, pelo fato do divisor ser quadrático, o resto é um polinômio de ordem 1, que aqui foi representado como $R(x) = b_0 + b_1(x - r)$. Utilizando uma comparação entre os coeficientes das potências de x para a equação (2.31), Bairstow obtém a seguinte relação de recorrência para estimativa dos coeficientes $b_i's$:

$$\begin{aligned} b_n &= a_n \\ b_{n-1} &= a_{n-1} + rb_n \\ b_i = a_i + rb_{i+1} + sb_{i+2}, &\quad \text{para: } i = (n-2) \to 0. \end{aligned} \tag{2.32}$$

A ideia do método consiste em supor um valor inicial do par (r_0, s_0), realizar o processo de divisão sintética pelo divisor quadrático por meio das relações de recorrência apresentadas em (2.32) e em seguida ajustar os valores do par (r, s) para fazer com que b_0 e b_1 sejam nulos. Para isso, utiliza-se a ideia de série de Taylor na qual escrevemos:

$$\begin{aligned} b_1(r + \Delta r, s + \Delta s) &= b_1 + \frac{\partial b_1}{\partial r}\Delta r + \frac{\partial b_1}{\partial s}\Delta s \\ b_0(r + \Delta r, s + \Delta s) &= b_0 + \frac{\partial b_0}{\partial r}\Delta r + \frac{\partial b_0}{\partial s}\Delta s. \end{aligned} \tag{2.33}$$

O que procuramos aqui são valores de Δr e Δs que anulem $b_1(r + \Delta r, s + \Delta s)$ e $b_0(r + \Delta r, s + \Delta s)$, ou seja:

$$\begin{aligned} \frac{\partial b_1}{\partial r}\Delta r + \frac{\partial b_1}{\partial s}\Delta s &= -b_1 \\ \frac{\partial b_0}{\partial r}\Delta r + \frac{\partial b_0}{\partial s}\Delta s &= -b_0. \end{aligned} \tag{2.34}$$

Se conhecermos os valores das derivadas parciais de b_0 e b_1 com relação à r e s, o sistema de equações (2.34) se torna um sistema linear de 2 equações e 2 icógnitas, que pode ser resolvido pela regra de Cramer por exemplo para determinar os próximos passos Δr e Δs que deverão ser dados para irmos ajustando os valores de r e s a fim de zerar o resto dessa divisão. Bairstow mostrou que essas derivadas parciais podem ser obtidas por meio de um processo de divisão sintética análogo ao utilizado para a determinação dos próprios $b_i's$, de tal sorte que:

$$\begin{aligned} c_n &= b_n \\ c_{n-1} &= b_{n-1} + rc_n \\ c_i = b_i + rc_{i+1} + sc_{i+2}, &\quad \text{para: } i = (n-2) \to 1, \end{aligned} \tag{2.35}$$

em que

$$c_1 = \frac{\partial b_0}{\partial r}, \;\; c_2 = \frac{\partial b_0}{\partial s} = \frac{\partial b_1}{\partial r}, \;\; c_3 = \frac{\partial b_1}{\partial s}. \tag{2.36}$$

Uma vez que o sistema linear expresso em (2.34) tenha sido resolvido, basta atualizarmos r e s de acordo com

$$r_i = r_{i-1} + \Delta r \quad \text{e} \quad s_i = s_{i-1} + \Delta s \tag{2.37}$$

até que o resto da divisão seja nulo. Quando isso acontece, temos os valores de r e s que deverão ser utilizados na fórmula quadrática $x^2 - rx - s$ para obtenção das raízes, dadas por:

$$x_r = \frac{r \pm \sqrt{r^2 + 4s}}{2}. \tag{2.38}$$

Caso o resultado $f_{n-2}(x)$ da divisão de $f_n(x)$ pelo fator quadrático seja um polinômio de grau maior ou igual a 3, o método é aplicado novamente ao polinômio resultante $f_{n-2}(x)$ até irmos reduzindo a ordem do polinômio ao seu último grau. Caso o resultado seja um polinômio de ordem 2, basta aplicar a fórmula quadrática e obtêm-se as últimas raízes. Finalmente, caso seja um polinômio de ordem 1, então a última raiz é dada por $x_r = -s/r$. Um resumo do método é apresentado na figura (2.7).

Uma das vantagens desse método é que pelo fato de utilizar-se um divisor quadrático ele sempre encontra raízes aos pares. Sua convergência também costuma ser rápida pelo fato de utilizar o método de Newton-Raphson para a busca dos pares r, s que anulem o resto da divisão. No entanto, dependendo da escolha inicial de valores de r e s, o método pode apresentar problemas de divergência. Um recurso bem interessante no campo da análise numérica desse método consiste na elaboração de um mapa denominado fractais de Bairstow. Esse mapa consiste em atribuir uma cor associada por exemplo ao número de iterações necessárias para garantir a convergência do método para diversos pares (r_0, s_0) testados. Valores que levem à divergência do método podem ser pintados por exemplo de preto. A figura (2.8) ilustra alguns exemplos de estruturas de fractais de Bairstow. Esse é um ponto interessante de intersecção entre análise numérica e análise dinâmica não-linear que pode também ser explorado para outros métodos. O mapeamento de zonas de estabilidade/instabilidade no comportamento tanto de métodos numéricos quanto sistemas dinâmicos fornece pistas com relação ao comportamento número/físico de sistemas.

2.7.4 Proposta de programa: método de Bairstow

Sua tarefa consiste em escrever um programa de computador que encontre as raízes de um polinômio de ordem n pelo método de Bairstow e em seguida plotar fractais de Bairstow para diferentes polinômios testados.

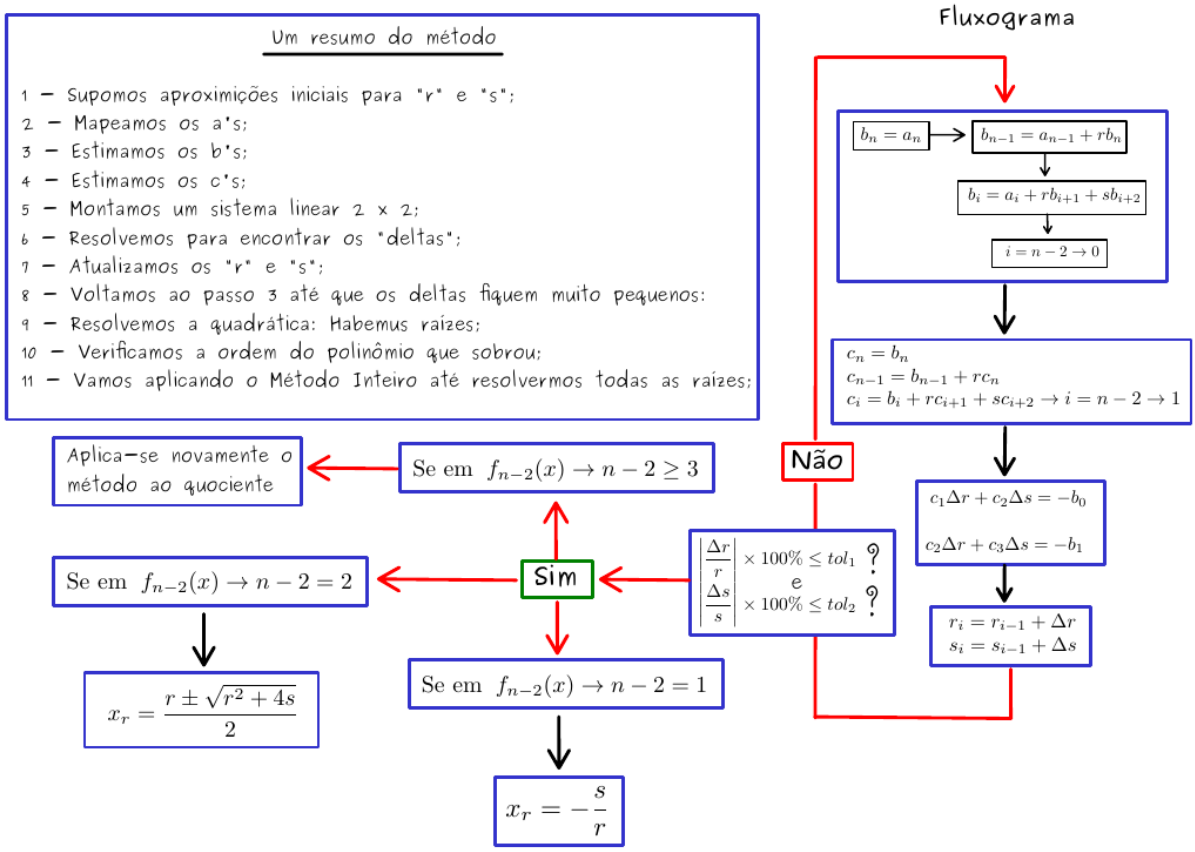


Figura 2.7: Fluxograma do método de Bairstow

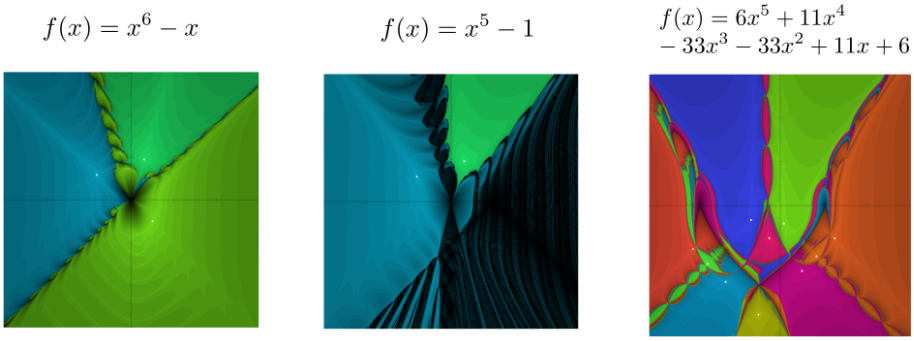


Figura 2.8: Exemplos de fractais de Bairstow: o eixo x contém valores de $r_0[= -3, 3]$ e o eixo y contém valores de $s_0 = [-3, 3]$.

A fim de ir ganhando confiança na tarefa, primeiro implemente o código considerando um único valor inicial de (r_0, s_0) e teste o método para um polinômio cujas raízes você conhece de antemão. Você pode montar um polinômio pelo processo de multiplicação de monômios com raízes conhecidas para testar a convergência do método.

Uma vez que o seu método funcione, você pode então automatizar o processo de varredura para diferentes valores de (r_0, s_0). A sugestão é que seu programa escreva como saída um arquivo de texto contendo 3 colunas, nas quais você irá alocar as informações referentes aos valores de r_0, s_0, it, em que it representa o número de iterações até a convergência do método. Caso o método não convirja após um elevado número de iterações

(digamos por exemplo $it = 999$) você pode considar que o método divergiu e mandar o programa escrever o valor 999 na coluna responsável pela iteração.

Um script simples em gnuplot voltado para ler esse arquivo de texto e construir o fractal de Bairstow correspondente é dado a seguir:

Gnuplot Copiar código

```
# Configurações gerais
set terminal pngcairo size 1000,800
enhanced font 'Arial,10'
set output 'mapa_bairstow.png'

set title "Fractal de Bairstow - Número
de Iterações até Convergência"
set xlabel "r₀ (chute inicial)"
set ylabel "s₀ (chute inicial)"
set cblabel "Número de Iterações"

set palette defined ( \
    0 "#440154", \
    1 "#482777", \
    2 "#3E4989", \
    3 "#31688E", \
    4 "#26828E", \
    5 "#1F9E89", \
    6 "#35B779", \
    7 "#6CCE59", \
    8 "#B4DE2C", \
    9 "#FDE725")

set view map

# Trata pontos divergentes (ex: 999) como
"NaN"
set datafile missing "999"

# Plotagem
splot 'dados_bairstow_simulado.txt' using
1:2:3 with image
```

Figura 2.9: Script em Gnuplot para plotar um fractal de Bairstow com base num arquivo de texto 'dados_bairstow_simulado.txt'.

Para executar esse script, basta ter o programa Gnuplot instalado no seu sistema Linux, abrí-lo num terminal e carregar o script com o comando 'load fractal_bairstow.gnu', em que fractal_bairstow.gnu é o nome do arquivo no qual você deverá salvar seu script.

2.8 O método de Newton-Raphson aplicado a sistemas não lineares

Até esse momento vimos algumas opções de métodos intervalares e abertos, destinados à obtenção de raízes de funções algébricas e transcendentais. Dentre os métodos vistos, um dos mais utilizados em diversas classes de problemas é o método de Newton-Raphson, que possui algumas variantes, como sua versão modificada adaptada para a obtenção de raízes múltiplas. Na verdade, muitos métodos numéricos possuem a característica de serem adaptados para se adequarem à solução de diferentes classes de problemas. Um outro exemplo interessante onde podemos aplicar o método de Newton-Raphson é na solução de sistemas de equações não-lineares. Nesse cenário, o método de Newton-Raphson não chega a resolver o problema, mas nos permite reformulá-lo a fim de transformarmos um sistema de equações não-lineares em múltiplos sistemas lineares que são resolvidos de maneira iterativa por um método próprio para essa finalidade.

Existem diversos métodos utilizados para a solução de equações não lineares, que vão desde uso de algoritmos genéticos, método da descida íngreme, métodos de ponto fixo, entre outros. Um método de fácil compreensão e ao mesmo tempo muito eficaz consiste no método de Newton. Geralmente alunos de cursos de cálculo numérico estão familiarizados com o método de Newton para a determinação de zeros de funções e em certo sentido é essa a ideia por trás de seu uso para resolver sistemas de equações não lineares. A ideia do método de Newton é que uma função $f(x)$ qualquer (linear ou não) pode ser aproximada por uma série de Taylor no ponto $x = x_0 + \Delta x$ por seu valor e de sua derivada no ponto x_0 como

$$f(x) \approx f(x_0) + \frac{df}{dx}(x_0)\Delta x \tag{2.39}$$

aqui $\Delta x = x - x_0$. Caso desejemos encontrar alguma raíz da função $f(x)$, ou seja, determinar o valor (ou valores) de x que fazem com que $f(x) = 0$, igualamos a equação (2.39) e isolamos x, este procedimento fornece

$$x = x_0 - \frac{f(x_0)}{f^{'}(x_0)} \tag{2.40}$$

aqui $f^{'}(x_0)$ denota a primeira derivada de f com respeito a x avaliada em x_0. Como o método se baseia na aproximação dada por (2.39) o uso da equação (2.40) não fornecerá a resposta exata com apenas uma única computação. Na realidade o método de Newton é um método iterativo, em que

$$x^{i+1} = x^i - \frac{f(x^i)}{f^{'}(x^i)}, \quad i = 0, 1, 2, \cdots \tag{2.41}$$

A partir do momento em que a diferença entre dois valores consecutivos de x for suficientemente pequena (dentro de uma tolerância pré-estabelecida) dizemos que o método convergiu. Considere por exemplo que desejemos determinar uma raíz da função dada por

$$f(x) = x^2 - 4, \tag{2.42}$$

aplicando o método descrito em (2.41) para a equação (2.42) obtemos

$$x^{i+1} = x^i - \frac{x^{i^2} - 4}{2x^i}, \quad i = 0, 1, 2, \cdots \tag{2.43}$$

i	x^i	x^{i+1}	$\lvert x^{i+1} - x^i \rvert$
0	0.5	4.25	3.75
1	4.25	2.5956	1.6544
2	2.5956	2.0683	0.5273
3	2.0683	2.0011	0.0672
4	2.0011	2.0000003183	0.0011284441
5	2.0000003183	2.0	3.18×10^{-7}

Tabela 2.1: Determinação de uma das raízes de uma equação não linear algébrica de segundo grau simples via Método de Newton.

Sabemos que a equação (2.42) possui duas raízes reais, dadas por 2 e -2. Considerando $x_0 = 0.5$ podemos montar a seguinte tabela:

Note que a partir da sexta iteração a diferença entre dois valores consecutivos da raíz obtida é da ordem de 10^{-7}. Essa tabela mostra a eficácia do método e sua rápida convergência. O uso do método de Newton para a solução de sistemas não lineares se baseia na generalização deste para encontrar N raízes de N equações não lineares organizadas na forma de um sistema não linear. Um sistema não linear de N equação acopladas pode ser organizado em termos vetoriais como uma função $\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}$. Essa função $\boldsymbol{f}$ é um vetor com N componentes em que N é o número de equações a ser resolvida. Como exemplo considere o seguinte sistema de duas equações não lineares

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \begin{pmatrix} x_1 + 2x_2 - 2 \\ x_1^2 + 4x_2^2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \tag{2.44}$$

considere ainda a confecção da seguinte matriz $\boldsymbol{J}$

$$\boldsymbol{J} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ \\ 2x_1 & 8x_2 \end{pmatrix}. \tag{2.45}$$

Essa matriz é denominada *matriz Jacobiana do sistema* e é utilizada para escrever a equação de recorrência (2.43) em forma vetorial como

$$\boldsymbol{x^{i+1}} = \boldsymbol{x^i} - \boldsymbol{J}(\boldsymbol{x^i})^{-1} \cdot \boldsymbol{f}(\boldsymbol{x^i}). \tag{2.46}$$

Note que na equação (2.46) surge a matriz inversa de $\boldsymbol{J}$. Isso não significa necessariamente que precisamos inverter a matriz $\boldsymbol{J}$ para resolver o sistema. Chamando o termo $-\boldsymbol{J}(\boldsymbol{x^i})^{-1} \cdot \boldsymbol{f}(\boldsymbol{x^i})$ de $\boldsymbol{s^i}$ temos

$$\boldsymbol{s^i} = -\boldsymbol{J}(\boldsymbol{x^i})^{-1} \cdot \boldsymbol{f}(\boldsymbol{x^i}) \tag{2.47}$$

multiplicando os dois lados da equação (2.47) por $\boldsymbol{J}(\boldsymbol{x^i})$ obtemos

$$\boldsymbol{J}(\boldsymbol{x^i}) \cdot \boldsymbol{s^i} = -\boldsymbol{J}(\boldsymbol{x^i}) \cdot \boldsymbol{J}(\boldsymbol{x^i})^{-1} \cdot \boldsymbol{f}(\boldsymbol{x^i}) = -\boldsymbol{I} \cdot \boldsymbol{f}(\boldsymbol{x^i}) = -\boldsymbol{f}(\boldsymbol{x^i}), \tag{2.48}$$

dessa forma temos

$$\boldsymbol{x^{i+1}} = \boldsymbol{x^i} + \boldsymbol{s^i}, \;\; i = 0, 1, 2, \cdots \tag{2.49}$$

com $\boldsymbol{s^i}$ dado pela solução do sistema linear

$$\boldsymbol{J(x^i)} \cdot \boldsymbol{s^i} = -\boldsymbol{f(x^i)}. \tag{2.50}$$

Para o exemplo em questão considere um vetor inicial $\boldsymbol{x^0} = [1 \quad 2]^T$, dessa forma $\boldsymbol{f(x^0)} = [3 \quad 13]^T$ e

$$\boldsymbol{J(x^0)} = \begin{pmatrix} 1 & 2 \\ \\ 2 & 16 \end{pmatrix} \tag{2.51}$$

Resolvendo o sistema

$$\begin{pmatrix} 1 & 2 \\ \\ 2 & 16 \end{pmatrix} \cdot \boldsymbol{s^0} = \begin{pmatrix} -3 \\ 13 \end{pmatrix}, \tag{2.52}$$

obtemos $\boldsymbol{s^0} = [-1.83 \quad -0.58]^T$ e $\boldsymbol{x^1} = [-0.83 \quad 1.42]^T$. Para um próximo ponto temos $\boldsymbol{f(x^1)} = [0 \quad 4.72]^T$ e

$$\boldsymbol{J(x^1)} = \begin{pmatrix} 1 & 2 \\ \\ -1.67 & 11.3 \end{pmatrix} \tag{2.53}$$

Resolvendo o sistema

$$\begin{pmatrix} 1 & 2 \\ \\ -1.67 & 11.3 \end{pmatrix} \cdot \boldsymbol{s^1} = \begin{pmatrix} 0 \\ 4.72 \end{pmatrix}, \tag{2.54}$$

obtemos $\boldsymbol{s^1} = [0.64 \quad -0.32]^T$ e $\boldsymbol{x^1} = [-0.19 \quad 1.1]^T$, repetindo o procedimento diversas vezes a solução convirgirá para $\boldsymbol{x} = [0 \quad 1]^T$.

Note que o uso do Método de Newton para a solução de sistemas não lineares consiste na transformação de um único sistema não linear em diversos sitemas lineares resolvidos de modo progressivo em um procedimento iterativo até a convergência. A ideia por trás deste método é a de que uma função não linear pode ser aproximada como uma soma de diferentes funções lineares em torno de pequenos intervalos. A próxima seção mostra um exemplo do uso deste método na solução da equação de Navier-Stokes para um problema de camada limite laminar sobre placa plana de um fluido incompressível.

2.8.1 Exemplo de aplicação: solução de uma EDP não linear

Considere a seguinte equação diferencial parcial

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} \tag{2.55}$$

a equação (2.55) é conhecida como equação de Navier-Stokes (sua componente apenas na direção x para um problema de camada limite). Considere a seguinte imagem ilustrativa que representa o domínio de cálculo discretizado no qual desejamos resolver a equação (2.55). A figura (2.11) ilustra um modelo esquemático do problema de camda limite a ser abordado.

As funções escalares u e v são funções de x e y (coordenadas espaciais), de modo que

$$u = u(x, y) \quad \text{e} \quad v = v(x, y) \tag{2.56}$$

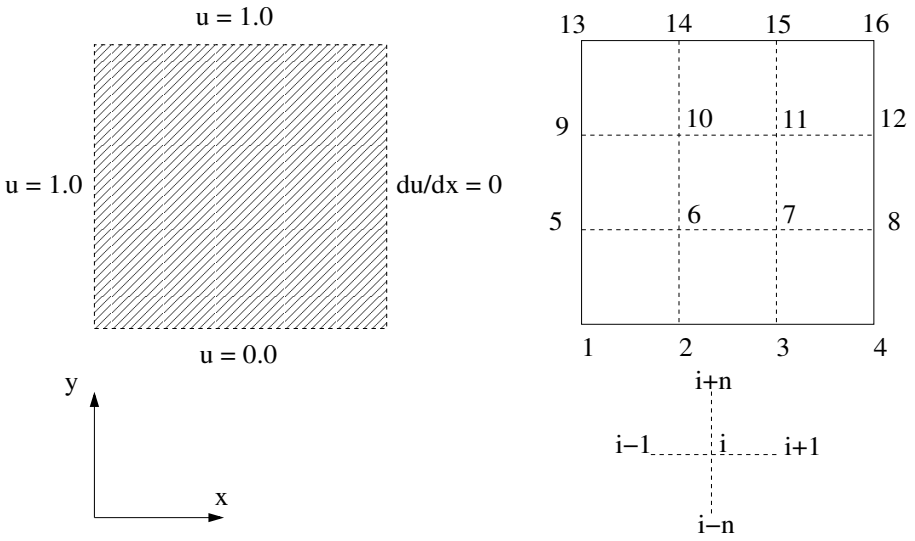


Figura 2.10: O domínio de cálculo discretizado consiste em uma malha com 16 nós, sendo 4 internos e 12 externos sujeitos às condições de contorno descritas na figura.

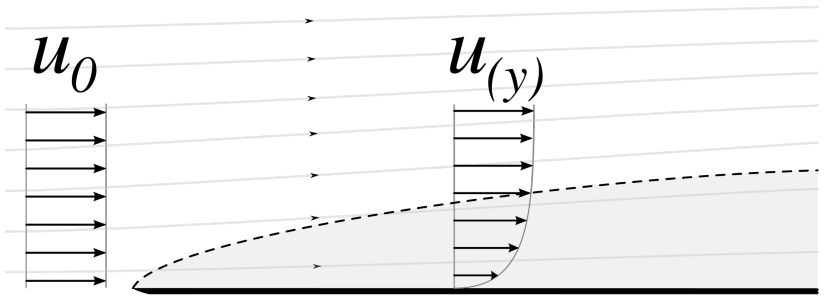


Figura 2.11: Modelo esquemático do problema de camada limite hidrodinâmica.

A versão discretizada da equação (2.55) é dada por

$$u^i \frac{(u^i - u^{i-1})}{\Delta x} + v^i \frac{(u^i - u^{i-n})}{\Delta x} = \frac{1}{Re} \frac{(u^{i+n} + u^{i-n} - 2u^i)}{\Delta y^2} \tag{2.57}$$

isolando u^i temos

$$u^i \left(\frac{u^i}{\Delta x} + \frac{v^i}{\Delta y} + \frac{2}{Re\Delta y^2} - \frac{u^{i-1}}{\Delta x} \right) - \frac{v^i u^{i-n}}{\Delta y} - \frac{u^{i+n}}{Re\Delta y^2} - \frac{u^{i-n}}{Re\Delta y^2} = 0 \tag{2.58}$$

Note que a equação acima se refere à um nó i qualquer e a maneira com a qual este se relaciona com seus vizinhos. Para fins de simplificação considere o caso em que $v = 0$, de modo que

$$u^i \left(\frac{u^i}{\Delta x} + \frac{2}{Re\Delta y^2} - \frac{u^{i-1}}{\Delta x} \right) - \frac{u^{i+n}}{Re\Delta y^2} - \frac{u^{i-n}}{Re\Delta y^2} = 0 \tag{2.59}$$

Podemos escrever esse sistema de equações não lineares para a malha em questão (com 16 nós) da forma

$$\boldsymbol{f}(\boldsymbol{u}) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 - 1 \\ \frac{u_6^2}{\Delta x} - \frac{u_6 u_5}{\Delta x} + \frac{2u_6}{Re\Delta y^2} - \frac{u_{10}}{Re\Delta y^2} - \frac{u_2}{Re\Delta y^2} \\ \frac{u_7^2}{\Delta x} - \frac{u_7 u_6}{\Delta x} + \frac{2u_7}{Re\Delta y^2} - \frac{u_{11}}{Re\Delta y^2} - \frac{u_3}{Re\Delta y^2} \\ u_8 - u_7 \\ u_9 - 1 \\ \frac{u_{10}^2}{\Delta x} - \frac{u_{10} u_9}{\Delta x} + \frac{2u_{10}}{Re\Delta y^2} - \frac{u_{14}}{Re\Delta y^2} - \frac{u_6}{Re\Delta y^2} \\ \frac{u_{11}^2}{\Delta x} - \frac{u_{11} u_{10}}{\Delta x} + \frac{2u_{11}}{Re\Delta y^2} - \frac{u_{15}}{Re\Delta y^2} - \frac{u_7}{Re\Delta y^2} \\ u_{12} - u_{11} \\ u_{13} - 1 \\ u_{14} - 1 \\ u_{15} - 1 \\ u_{16} - 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.60}$$

Considere a matriz Jacobiana $\boldsymbol{J}$ do sistema definida como

$$\boldsymbol{J} = \begin{pmatrix}
\frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \frac{\partial f_1}{\partial u_3} & \frac{\partial f_1}{\partial u_4} & \frac{\partial f_1}{\partial u_5} & \frac{\partial f_1}{\partial u_6} & \frac{\partial f_1}{\partial u_7} & \frac{\partial f_1}{\partial u_8} & \frac{\partial f_1}{\partial u_9} & \frac{\partial f_1}{\partial u_{10}} & \frac{\partial f_1}{\partial u_{11}} & \frac{\partial f_1}{\partial u_{12}} & \frac{\partial f_1}{\partial u_{13}} & \frac{\partial f_1}{\partial u_{14}} & \frac{\partial f_1}{\partial u_{15}} & \frac{\partial f_1}{\partial u_{16}} \\
\frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \frac{\partial f_2}{\partial u_3} & \frac{\partial f_2}{\partial u_4} & \frac{\partial f_2}{\partial u_5} & \frac{\partial f_2}{\partial u_6} & \frac{\partial f_2}{\partial u_7} & \frac{\partial f_2}{\partial u_8} & \frac{\partial f_2}{\partial u_9} & \frac{\partial f_2}{\partial u_{10}} & \frac{\partial f_2}{\partial u_{11}} & \frac{\partial f_2}{\partial u_{12}} & \frac{\partial f_2}{\partial u_{13}} & \frac{\partial f_2}{\partial u_{14}} & \frac{\partial f_2}{\partial u_{15}} & \frac{\partial f_2}{\partial u_{16}} \\
\frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} & \frac{\partial f_3}{\partial u_3} & \frac{\partial f_3}{\partial u_4} & \frac{\partial f_3}{\partial u_5} & \frac{\partial f_3}{\partial u_6} & \frac{\partial f_3}{\partial u_7} & \frac{\partial f_3}{\partial u_8} & \frac{\partial f_3}{\partial u_9} & \frac{\partial f_3}{\partial u_{10}} & \frac{\partial f_3}{\partial u_{11}} & \frac{\partial f_3}{\partial u_{12}} & \frac{\partial f_3}{\partial u_{13}} & \frac{\partial f_3}{\partial u_{14}} & \frac{\partial f_3}{\partial u_{15}} & \frac{\partial f_3}{\partial u_{16}} \\
\frac{\partial f_4}{\partial u_1} & \frac{\partial f_4}{\partial u_2} & \frac{\partial f_4}{\partial u_3} & \frac{\partial f_4}{\partial u_4} & \frac{\partial f_4}{\partial u_5} & \frac{\partial f_4}{\partial u_6} & \frac{\partial f_4}{\partial u_7} & \frac{\partial f_4}{\partial u_8} & \frac{\partial f_4}{\partial u_9} & \frac{\partial f_4}{\partial u_{10}} & \frac{\partial f_4}{\partial u_{11}} & \frac{\partial f_4}{\partial u_{12}} & \frac{\partial f_4}{\partial u_{13}} & \frac{\partial f_4}{\partial u_{14}} & \frac{\partial f_4}{\partial u_{15}} & \frac{\partial f_4}{\partial u_{16}} \\
\frac{\partial f_5}{\partial u_1} & \frac{\partial f_5}{\partial u_2} & \frac{\partial f_5}{\partial u_3} & \frac{\partial f_5}{\partial u_4} & \frac{\partial f_5}{\partial u_5} & \frac{\partial f_5}{\partial u_6} & \frac{\partial f_5}{\partial u_7} & \frac{\partial f_5}{\partial u_8} & \frac{\partial f_5}{\partial u_9} & \frac{\partial f_5}{\partial u_{10}} & \frac{\partial f_5}{\partial u_{11}} & \frac{\partial f_5}{\partial u_{12}} & \frac{\partial f_5}{\partial u_{13}} & \frac{\partial f_5}{\partial u_{14}} & \frac{\partial f_5}{\partial u_{15}} & \frac{\partial f_5}{\partial u_{16}} \\
\frac{\partial f_6}{\partial u_1} & \frac{\partial f_6}{\partial u_2} & \frac{\partial f_6}{\partial u_3} & \frac{\partial f_6}{\partial u_4} & \frac{\partial f_6}{\partial u_5} & \frac{\partial f_6}{\partial u_6} & \frac{\partial f_6}{\partial u_7} & \frac{\partial f_6}{\partial u_8} & \frac{\partial f_6}{\partial u_9} & \frac{\partial f_6}{\partial u_{10}} & \frac{\partial f_6}{\partial u_{11}} & \frac{\partial f_6}{\partial u_{12}} & \frac{\partial f_6}{\partial u_{13}} & \frac{\partial f_6}{\partial u_{14}} & \frac{\partial f_6}{\partial u_{15}} & \frac{\partial f_6}{\partial u_{16}} \\
\frac{\partial f_7}{\partial u_1} & \frac{\partial f_7}{\partial u_2} & \frac{\partial f_7}{\partial u_3} & \frac{\partial f_7}{\partial u_4} & \frac{\partial f_7}{\partial u_5} & \frac{\partial f_7}{\partial u_6} & \frac{\partial f_7}{\partial u_7} & \frac{\partial f_7}{\partial u_8} & \frac{\partial f_7}{\partial u_9} & \frac{\partial f_7}{\partial u_{10}} & \frac{\partial f_7}{\partial u_{11}} & \frac{\partial f_7}{\partial u_{12}} & \frac{\partial f_7}{\partial u_{13}} & \frac{\partial f_7}{\partial u_{14}} & \frac{\partial f_7}{\partial u_{15}} & \frac{\partial f_7}{\partial u_{16}} \\
\frac{\partial f_8}{\partial u_1} & \frac{\partial f_8}{\partial u_2} & \frac{\partial f_8}{\partial u_3} & \frac{\partial f_8}{\partial u_4} & \frac{\partial f_8}{\partial u_5} & \frac{\partial f_8}{\partial u_6} & \frac{\partial f_8}{\partial u_7} & \frac{\partial f_8}{\partial u_8} & \frac{\partial f_8}{\partial u_9} & \frac{\partial f_8}{\partial u_{10}} & \frac{\partial f_8}{\partial u_{11}} & \frac{\partial f_8}{\partial u_{12}} & \frac{\partial f_8}{\partial u_{13}} & \frac{\partial f_8}{\partial u_{14}} & \frac{\partial f_8}{\partial u_{15}} & \frac{\partial f_8}{\partial u_{16}} \\
\frac{\partial f_9}{\partial u_1} & \frac{\partial f_9}{\partial u_2} & \frac{\partial f_9}{\partial u_3} & \frac{\partial f_9}{\partial u_4} & \frac{\partial f_9}{\partial u_5} & \frac{\partial f_9}{\partial u_6} & \frac{\partial f_9}{\partial u_7} & \frac{\partial f_9}{\partial u_8} & \frac{\partial f_9}{\partial u_9} & \frac{\partial f_9}{\partial u_{10}} & \frac{\partial f_9}{\partial u_{11}} & \frac{\partial f_9}{\partial u_{12}} & \frac{\partial f_9}{\partial u_{13}} & \frac{\partial f_9}{\partial u_{14}} & \frac{\partial f_9}{\partial u_{15}} & \frac{\partial f_9}{\partial u_{16}} \\
\frac{\partial f_{10}}{\partial u_1} & \frac{\partial f_{10}}{\partial u_2} & \frac{\partial f_{10}}{\partial u_3} & \frac{\partial f_{10}}{\partial u_4} & \frac{\partial f_{10}}{\partial u_5} & \frac{\partial f_{10}}{\partial u_6} & \frac{\partial f_{10}}{\partial u_7} & \frac{\partial f_{10}}{\partial u_8} & \frac{\partial f_{10}}{\partial u_9} & \frac{\partial f_{10}}{\partial u_{10}} & \frac{\partial f_{10}}{\partial u_{11}} & \frac{\partial f_{10}}{\partial u_{12}} & \frac{\partial f_{10}}{\partial u_{13}} & \frac{\partial f_{10}}{\partial u_{14}} & \frac{\partial f_{10}}{\partial u_{15}} & \frac{\partial f_{10}}{\partial u_{16}} \\
\frac{\partial f_{11}}{\partial u_1} & \frac{\partial f_{11}}{\partial u_2} & \frac{\partial f_{11}}{\partial u_3} & \frac{\partial f_{11}}{\partial u_4} & \frac{\partial f_{11}}{\partial u_5} & \frac{\partial f_{11}}{\partial u_6} & \frac{\partial f_{11}}{\partial u_7} & \frac{\partial f_{11}}{\partial u_8} & \frac{\partial f_{11}}{\partial u_9} & \frac{\partial f_{11}}{\partial u_{10}} & \frac{\partial f_{11}}{\partial u_{11}} & \frac{\partial f_{11}}{\partial u_{12}} & \frac{\partial f_{11}}{\partial u_{13}} & \frac{\partial f_{11}}{\partial u_{14}} & \frac{\partial f_{11}}{\partial u_{15}} & \frac{\partial f_{11}}{\partial u_{16}} \\
\frac{\partial f_{12}}{\partial u_1} & \frac{\partial f_{12}}{\partial u_2} & \frac{\partial f_{12}}{\partial u_3} & \frac{\partial f_{12}}{\partial u_4} & \frac{\partial f_{12}}{\partial u_5} & \frac{\partial f_{12}}{\partial u_6} & \frac{\partial f_{12}}{\partial u_7} & \frac{\partial f_{12}}{\partial u_8} & \frac{\partial f_{12}}{\partial u_9} & \frac{\partial f_{12}}{\partial u_{10}} & \frac{\partial f_{12}}{\partial u_{11}} & \frac{\partial f_{12}}{\partial u_{12}} & \frac{\partial f_{12}}{\partial u_{13}} & \frac{\partial f_{12}}{\partial u_{14}} & \frac{\partial f_{12}}{\partial u_{15}} & \frac{\partial f_{12}}{\partial u_{16}} \\
\frac{\partial f_{13}}{\partial u_1} & \frac{\partial f_{13}}{\partial u_2} & \frac{\partial f_{13}}{\partial u_3} & \frac{\partial f_{13}}{\partial u_4} & \frac{\partial f_{13}}{\partial u_5} & \frac{\partial f_{13}}{\partial u_6} & \frac{\partial f_{13}}{\partial u_7} & \frac{\partial f_{13}}{\partial u_8} & \frac{\partial f_{13}}{\partial u_9} & \frac{\partial f_{13}}{\partial u_{10}} & \frac{\partial f_{13}}{\partial u_{11}} & \frac{\partial f_{13}}{\partial u_{12}} & \frac{\partial f_{13}}{\partial u_{13}} & \frac{\partial f_{13}}{\partial u_{14}} & \frac{\partial f_{13}}{\partial u_{15}} & \frac{\partial f_{13}}{\partial u_{16}} \\
\frac{\partial f_{14}}{\partial u_1} & \frac{\partial f_{14}}{\partial u_2} & \frac{\partial f_{14}}{\partial u_3} & \frac{\partial f_{14}}{\partial u_4} & \frac{\partial f_{14}}{\partial u_5} & \frac{\partial f_{14}}{\partial u_6} & \frac{\partial f_{14}}{\partial u_7} & \frac{\partial f_{14}}{\partial u_8} & \frac{\partial f_{14}}{\partial u_9} & \frac{\partial f_{14}}{\partial u_{10}} & \frac{\partial f_{14}}{\partial u_{11}} & \frac{\partial f_{14}}{\partial u_{12}} & \frac{\partial f_{14}}{\partial u_{13}} & \frac{\partial f_{14}}{\partial u_{14}} & \frac{\partial f_{14}}{\partial u_{15}} & \frac{\partial f_{14}}{\partial u_{16}} \\
\frac{\partial f_{15}}{\partial u_1} & \frac{\partial f_{15}}{\partial u_2} & \frac{\partial f_{15}}{\partial u_3} & \frac{\partial f_{15}}{\partial u_4} & \frac{\partial f_{15}}{\partial u_5} & \frac{\partial f_{15}}{\partial u_6} & \frac{\partial f_{15}}{\partial u_7} & \frac{\partial f_{15}}{\partial u_8} & \frac{\partial f_{15}}{\partial u_9} & \frac{\partial f_{15}}{\partial u_{10}} & \frac{\partial f_{15}}{\partial u_{11}} & \frac{\partial f_{15}}{\partial u_{12}} & \frac{\partial f_{15}}{\partial u_{13}} & \frac{\partial f_{15}}{\partial u_{14}} & \frac{\partial f_{15}}{\partial u_{15}} & \frac{\partial f_{15}}{\partial u_{16}} \\
\frac{\partial f_{16}}{\partial u_1} & \frac{\partial f_{16}}{\partial u_2} & \frac{\partial f_{16}}{\partial u_3} & \frac{\partial f_{16}}{\partial u_4} & \frac{\partial f_{16}}{\partial u_5} & \frac{\partial f_{16}}{\partial u_6} & \frac{\partial f_{16}}{\partial u_7} & \frac{\partial f_{16}}{\partial u_8} & \frac{\partial f_{16}}{\partial u_9} & \frac{\partial f_{16}}{\partial u_{10}} & \frac{\partial f_{16}}{\partial u_{11}} & \frac{\partial f_{16}}{\partial u_{12}} & \frac{\partial f_{16}}{\partial u_{13}} & \frac{\partial f_{16}}{\partial u_{14}} & \frac{\partial f_{16}}{\partial u_{15}} & \frac{\partial f_{16}}{\partial u_{16}}
\end{pmatrix} \tag{2.61}$$

Determinando as derivadas especificadas em (2.61) para o sistema definido em (2.60) obtemos

$$
\boldsymbol{J} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \left[-\frac{1}{Re\Delta y^2}\right] & 0 & 0 & 0 & \left[-\frac{u_6}{\Delta x}\right] & \left[\frac{1}{\Delta x}(2u_6 - u_5) + \frac{2}{Re\Delta y^2}\right] & \left[-\frac{u_7}{\Delta x}\right] & 0 & \left[\frac{1}{\Delta x}(2u_7 - u_6) + \frac{2}{Re\Delta y^2}\right] & 0 & 0 & 0 & \left[-\frac{1}{Re\Delta y^2}\right] \\
0 & 0 & \left[-\frac{1}{Re\Delta y^2}\right] & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix} \tag{2.62}
$$

Considere o caso em que $\Delta x = \Delta y = 0.1$ e $Re = 100$, essa matriz é dada por

$$\boldsymbol{J} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & -10u_6 & 10\,(2u_6 - u_5) + 2 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & -10u_7 & 10\,(2u_7 - u_6) + 2 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix} \tag{2.63}$$

O método de solução consiste em supor uma condição inicial $\boldsymbol{u_0}$ para o vetor $\boldsymbol{u}$, substituir essa condição na matriz $\boldsymbol{J}(\boldsymbol{u_0})$ e resolver o seguinte sistema linear

$$\boldsymbol{J}(\boldsymbol{u_0}) \cdot \boldsymbol{u_0^*} = -\boldsymbol{f}(\boldsymbol{u_0}) \tag{2.64}$$

em seguida nós atualizamos o novo valor do vetor $\boldsymbol{u_1}$ fazendo

$$\boldsymbol{u_1} = \boldsymbol{u_0} + \boldsymbol{u_0^*}, \tag{2.65}$$

com esse novo valor de $\boldsymbol{u_1}$ nós resolvemos o seguinte sistema

$$\boldsymbol{J}(\boldsymbol{u_1}) \cdot \boldsymbol{u_1^*} = -\boldsymbol{f}(\boldsymbol{u_1}), \tag{2.66}$$

em seguida fazemos

$$\boldsymbol{u_2} = \boldsymbol{u_1} + \boldsymbol{u_1^*}, \tag{2.67}$$

e assim sucessivamente até que o método convirja para um valor dentro de uma determinada faixa de tolerância. Podemos resumir o método nos seguintes passos

1. Supor um valor inicial para o vetor $\boldsymbol{u_0}$
2. Determinar a matriz Jacobiana $\boldsymbol{J}(\boldsymbol{u_0})$
3. Determinar o vetor $\boldsymbol{f}(\boldsymbol{u_0})$
4. Resolver o sistema linear $\boldsymbol{J}(\boldsymbol{u_0}) \cdot \boldsymbol{u_0^*} = -\boldsymbol{f}(\boldsymbol{u_0})$

5. Corrigir o próximo valor de $\boldsymbol{u_1}$ com $\boldsymbol{u_1} = \boldsymbol{u_0} + \boldsymbol{u_0^*}$
6. Fazer para j de 1 à N:
   - Resolver o sistema linear $\boldsymbol{J}(\boldsymbol{u_j}) \cdot \boldsymbol{u_j^*} = -\boldsymbol{f}(\boldsymbol{u_j})$
   - Corrigir o próximo valor de $\boldsymbol{u_j}$ com $\boldsymbol{u_j} = \boldsymbol{u_j} + \boldsymbol{u_j^*}$
7. Repetir o procedimento até convergir;

Para fins de exemplificação considere o seguinte valor inicial para o vetor $\boldsymbol{u_0}$

$$\boldsymbol{u_0} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{2.68}$$

Para esses valores iniciais, com $\Delta x = \Delta y = 0.1$ e $Re = 100$, temos que $\boldsymbol{J}(\boldsymbol{u_0})$ é dada por

$$\boldsymbol{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.69}$$

e o vetor $\boldsymbol{f}(\boldsymbol{u_0})$ é dado por

$$\boldsymbol{f}(\boldsymbol{u_0}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.70}$$

Dessa forma precisamos resolver inicialmente o seguinte sistema

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -10 & 12 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} u_0^{*1} \\ u_0^{*2} \\ u_0^{*3} \\ u_0^{*4} \\ u_0^{*5} \\ u_0^{*6} \\ u_0^{*7} \\ u_0^{*8} \\ u_0^{*9} \\ u_0^{*10} \\ u_0^{*11} \\ u_0^{*12} \\ u_0^{*13} \\ u_0^{*14} \\ u_0^{*15} \\ u_0^{*16} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.71}$$

A solução desse sistema linear pelo método de Jacobi (método iterativo) com 500 iterações

fornece

$$\boldsymbol{u_0^*} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -0.083916068 \\ -0.1548242 \\ -0.1548242 \\ 0 \\ -0.006993004 \\ -0.018729515 \\ -0.018729515 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.72}$$

Façamos agora a seguinte correção

$$\boldsymbol{u_1} = \boldsymbol{u_0} + \boldsymbol{u_0^*}, \tag{2.73}$$

de modo que

$$\boldsymbol{u_1} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0.9160839 \\ 0.8451759 \\ 0.8451759 \\ 1 \\ 0.9930070 \\ 0.9812705 \\ 0.9812705 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{2.74}$$

Agora precisamos resolver o seguinte sistema

$$\boldsymbol{J}(\boldsymbol{u_1}) \cdot \boldsymbol{u_1^*} = -\boldsymbol{f}(\boldsymbol{u_1}), \tag{2.75}$$

note que a matriz $\boldsymbol{J}$ muda a cada nova iteração. Montando o sistema e resolvendo para

$\boldsymbol{u_1^*}$ pelo mesmo método (Jacobi com 500 iterações) obtemos o novo valor de $\boldsymbol{u_2}$ dado por

$$\boldsymbol{u_2} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0.9092013 \\ 0.8277108 \\ 0.8277108 \\ 1 \\ 0.9923854 \\ 0.9790677 \\ 0.9790677 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{2.76}$$

Repetindo esse procedimento 10 vezes obtemos $\boldsymbol{u_{10}}$, comparando o valor de $\boldsymbol{u_{10}}$ com o valor de $\boldsymbol{u_9}$ temos

$$\boldsymbol{u_9} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0.9091543 \\ 0.8274714 \\ 0.8274714 \\ 1 \\ 0.9923811 \\ 0.9790405 \\ 0.9790405 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad \boldsymbol{u_{10}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0.9091544 \\ 0.8274716 \\ 0.8274716 \\ 1 \\ 0.9923812 \\ 0.9790406 \\ 0.9790406 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{2.77}$$

Para análise da convergência do processo repare no vetor dado por $|\boldsymbol{u_{10}} - \boldsymbol{u_9}|$

$$|\boldsymbol{u_{10}} - \boldsymbol{u_9}| = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1.1920929E-07 \\ 1.1920929E-07 \\ 1.1920929E-07 \\ 0 \\ 5.9604645E-08 \\ 5.9604645E-08 \\ 5.9604645E-08 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.78}$$

O valor máximo dessa diferença, nesse caso $1.1920929E-07$ pode ser utilizado como o parâmetro que deverá ser menor que determinada tolerância pré-definida pelo usuário para garantir a convergência do método.