

# Primeiro Exercício-Programa

Norton Trevisan Roman

17 de abril de 2018

## 1 Método de Newton-Raphson

O cálculo do “zero” de uma função consiste em, dada uma função  $f(x) = 0$ , achar o valor (ou possivelmente conjunto de valores) de  $x$ , para o qual essa expressão torna-se verdade. Por exemplo, considere a função  $f(x) = x^2 + 4$ . Um zero dessa função seria:

$$\begin{aligned} f(x) &= x^2 - 4 = 0 \\ x^2 &= 4 \\ x &= 2 \end{aligned}$$

sendo que o outro zero estaria em  $x = -2$ .

Esse mesmo raciocínio pode ser usado também para se resolver equações. Por exemplo, considere a equação  $x^2 = 4$ . Ela pode ser reduzida ao problema de se encontrar o zero de  $F(x) = x^2 - 4$ .

Naturalmente, nesse exemplo a equação é bastante simples e direta. Contudo, o poder do método se mostra em equações mais complexas. Por exemplo, imagine que você faz uma determinada aplicação inicial  $D_0$  em um fundo de investimento, em uma determinada data  $t_0$ . A partir dessa data, e em dias esporádicos, você faz novos depósitos  $D_i$  nesse mesmo fundo. Passado um certo tempo, você verifica o saldo do fundo, e então se pergunta qual teriam sido os juros (compostos) médios mensais praticados no período?

Essa resposta seria razoavelmente simples, caso os depósitos tivessem sido constantes. Contudo, eles foram esporádicos: você somente depositou quando o acaso fez sobrar algum dinheiro.

Como fazer então? Imagine que foram feitos  $n$  depósitos (além do inicial, ou seja,  $n + 1$  no total). Para simplificar considere que todo depósito, quando feito, é feito no dia primeiro do mês.

<i>depósito</i>	<i>data</i>
$D_0$	$t_0$
$D_1$	$t_1$
$\dots$	$\dots$
$D_n$	$t_n$

Em uma determinada data  $t_f$ , você verifica o saldo  $S$ . Supondo uma taxa mensal de juros  $0 \leq j \leq 1$  constante, podemos aplicar esses juros a cada um dos depósitos, resultando em

$$D_0(1+j)^{t_f-t_0} + D_1(1+j)^{t_f-t_1} + \dots + D_n(1+j)^{t_f-t_n} = S$$

Nesse caso, como estamos interessados nos juros mensais, podemos tomar  $t_f - t_n$ , por exemplo, como o número de meses entre  $t_f$  e  $t_n$ . Qual seria então a taxa de juros? Note que o problema é, na verdade, encontrar o zero da função

$$f(j) = D_0(1+j)^{t_f-t_0} + D_1(1+j)^{t_f-t_1} + \dots + D_n(1+j)^{t_f-t_n} - S$$

E como funciona o método de Newton-Raphson para zeros de funções? O método de Newton-Raphson parte de uma aproximação inicial para a variável buscada (um palpite para  $j$ ; 0.5 por exemplo), e incrementalmente vai chegando cada vez mais perto da solução. Formalmente, o método diz que<sup>1</sup>:

$$j_0 \leftarrow \frac{1}{2}$$

$$j_{k+1} \leftarrow j_k - \frac{f(j_k)}{f'(j_k)}$$

para  $k = 0, 1, 2, \dots$ . Ou seja, obtemos  $j_1$  fazendo  $j_1 = j_0 - \frac{f(j_0)}{f'(j_0)}$ ,  $j_2$  fazendo  $j_2 = j_1 - \frac{f(j_1)}{f'(j_1)}$  e assim por diante. Note que quanto mais alto o valor de  $k$ , melhor a aproximação de  $f(j)$  dada por  $j_{k+1}$ . Nessa equação,  $f'(j_k)$  é a derivada de  $f(j)$  no ponto  $j_k$ . No nosso exemplo, a derivada de

$$f(j) = D_0(1+j)^{t_f-t_0} + D_1(1+j)^{t_f-t_1} + \dots + D_n(1+j)^{t_f-t_n} - S$$

é

$$f'(j) = (t_f - t_0)D_0(1+j)^{t_f-t_0-1} + (t_f - t_1)D_1(1+j)^{t_f-t_1-1} + \dots + (t_f - t_n)D_n(1+j)^{t_f-t_n-1}$$

Esse processo deve ser repetido enquanto  $|j_{k+1} - j_k| \geq \epsilon$ , onde  $\epsilon$  é um número positivo que representa a precisão do cálculo. Assim, a aproximação de  $f(j)$  será o primeiro valor  $j_{k+1}$  para o qual  $|j_{k+1} - j_k| < \epsilon$ .

Por exemplo, suponha que sejam feitos 5 depósitos, conforme a tabela abaixo:

---

<sup>1</sup>Para um método mais geométrico de se obter essa equação, veja <http://www.youtube.com/watch?v=6ueocPki25I>. Para uma dedução com base na série de Taylor, consulte <http://omnis.if.ufrj.br/sandra/MetComp/2011-1/Newton.Raphson.pdf>.

<i>Valor</i>	<i>Data</i>
1.000,00	03/2015
1.200,00	04/2015
100,00	05/2015
1.100,00	07/2015
900,00	09/2015

Suponha agora que, em 12/2015, o saldo seja de 5.000,00. Assim,  $f(j)$ , com cada  $D_i$  dado pela tabela acima,  $t_f = 12/2015$  e  $S = 5.000,00$ , será

$$f(j) = 1.000,00(1+j)^9 + 1.200,00(1+j)^8 + 100,00(1+j)^7 + 1.100,00(1+j)^5 + 900,00(1+j)^3 - 5.000,00$$

e

$$f'(j) = 9 \cdot 1.000,00(1+j)^8 + 8 \cdot 1.200,00(1+j)^7 + 7 \cdot 100,00(1+j)^6 + 5 \cdot 1.100,00(1+j)^4 + 3 \cdot 900,00(1+j)^2$$

Segundo o método de Newton-Raphson,  $f(j)$ , com  $\epsilon = 0.001$  fica:

iteração	$j_k$	$j_{k+1}$	$ j_{k+1} - j_k $
1	0.5	0.3229471438873768	0.17705285611262322
2	0.3229471438873768	0.17719505698725935	0.14575208690011743
3	0.17719505698725935	0.07603810202653498	0.10115695496072437
4	0.07603810202653498	0.031151323858092876	0.0448867781684421
5	0.031151323858092876	0.023807799064652573	0.007343524793440304
6	0.023807799064652573	0.023637054483966326	1.7074458068624607E-4

e a resposta será 0.023637054483966326. Ou seja, para que essa série de depósitos tenha gerado o montante de 5.000,00, é necessário que os juros mensais tenham sido de  $\approx 0.023$  (ou  $\approx 2,3\%$ ) ao mês. Note que usamos apenas as três primeiras casas do valor calculado. Isso se dá pelo fato de nossa precisão ser de 0.001, ou seja, admitimos erros somente a partir da quarta casa decimal. Então as três primeiras casas são as únicas que podemos garantir.

## 2 Tarefa

Escreva um método que receba como parâmetro o valor de  $\epsilon$ , retornando  $f(j)$  acima. Seu método deve necessariamente ter a seguinte assinatura (constante do arquivo `NewtonRaphson.java`):

```
static double newton(double epsilon)
```

O método deve implementar  $f(j)$  para  $d$  depósitos (além de um saldo final). Para isso ele deve usar o arranjo `depositos`, também já incluído em `NewtonRaphson.java`, contendo  $d + 1$

posições (as  $d$  primeiras para os depósitos, inclusive o inicial, e a última,  $(d+1)^a$ , para o saldo final). As datas do depósito estão armazenadas no arranjo `datas`, que também conterá  $d+1$  posições.

Para simplificar, as datas referem-se ao número do mês dentro de um ano (calculamos o rendimento de depósitos dentro de um determinado ano apenas). Atente para o fato de que pode ser feito mais de um depósito no mesmo mês, então o arranjo de datas (e depósitos) pode conter números duplicados. Além disso, os arranjos de depósitos e datas terão, no mínimo, tamanho 2 (depósito inicial mais saldo final), sendo o máximo em aberto, pois não há limite para o número de depósitos (apenas saiba que serão no mesmo ano).

## 2.1 Entrada

A entrada é composta pelos dois arranjos descritos acima (ou seja, pela inclusão de valores nesses arranjos), além do parâmetro  $0 < \epsilon < 1$ , indicando a precisão do cálculo.

Lembre que o número de elementos de um arranjo pode ser obtido a partir de seu atributo `length`. É desse atributo que você pode inferir o número de depósitos  $d$ .

## 2.2 Saída

Como saída, o método retorna o valor de  $j$  com precisão  $\epsilon$ , ou  $-1$ , caso  $\epsilon \leq 0$  ou  $\epsilon \geq 1$ .

## 2.3 Material a Ser Entregue

Deverá ser entregue o arquivo `NewtonRaphson.java`. No início do arquivo preencha o cabeçalho informativo como segue:

```
/*  
**      <nome do(a) aluno(a)>                <número USP>                **/  
**  
**  
**      <data de entrega>                    **/  
*/
```

A entrega será feita única e exclusivamente via `tidia`, até a data marcada para entrega. Deverá ser postado no `tidia` um zip com o arquivo `.java` entregue a você, com o método `newton` implementado. O arquivo deve ser compactado (zip ou rar), tendo seu número USP como nome, ou seja:

`número_usp.zip`

Somente este arquivo zip deve ser postado no `tidia`. A responsabilidade de postagem nele é exclusiva do aluno. Por isso, problemas referentes ao uso do sistema devem ser resolvidos com antecedência.

### 3 Avaliação

Para avaliação, serão observados os seguintes quesitos:

1. Documentação: se há comentários explicando o que se faz nos passos mais importantes e para que serve o programa (Tanto o método quanto o programa em que está inserido)
2. Apresentação visual: se o código está legível, indentado etc
3. Corretude: se o programa funciona

Além disso, algumas observações pertinentes ao trabalho, que influem em sua nota, são:

- Este exercício-programa deve ser elaborado individualmente.
- Não será tolerado plágio, em hipótese alguma.
- Exercícios com erro de sintaxe (ou seja, erros de compilação), receberão nota ZERO

**Atenção!** Para avaliação, apenas o método `newton(double epsilon)` será invocado diretamente. Em especial, qualquer código dentro do `main` será ignorado. Então tenha certeza de que o problema é resolvido chamando-se diretamente somente esse método. Além disso, será necessário, para os testes, o uso dos arranjos definidos acima (globais). Assim, não mude seus nomes ou tipos.

Caso ache necessário, você pode criar outros métodos, desde que o problema seja resolvido chamando-se apenas `newton(double epsilon)`. Por fim, é imperativo que sua resposta esteja correta até o número de casas definidos pela precisão indicada. É possível, pelo próprio método, obter-se em uma iteração uma precisão maior. Isso está aceitável. Inaceitável é uma precisão menor que o  $\epsilon$  desejado.