Universidade Federal de São Carlos

Campus Sorocaba

Projeto 1 de Cálculo Numérico

Lucas Granja Toniello   RA: 726560

Sumário

1) Os métodos propostos foram todos feitos usando a linguagem *python*, na versão 3.4.3 e usando como base os algoritmos passados no livro texto [1]. Para maior facilidade, também foi usada a biblioteca *math* [2] da linguagem, dela foram utilizadas as seguintes funções matemáticas:

- math.pow(a, b): retorna o valor de “a” elevado a “b”.
- math.sqrt(x): retorna a raiz quadrada de x.
- math.sin(x): retorna o valor do seno de x.
- math.cos(x): retorna o valor do cosseno de x.
- math.e: a constante matemática denominada como Número de Euler.

Antes da execução do programa, é necessário colocar a função a ser usada no começo do código, como no exemplo abaixo

```
def f(x):
    return math.pow(x, 3) - 9*x + 3
```

Exemplo da função $x^3 - 9x + 3$ no código

Antes de cada exercício está escrito, entre aspas, o que deve ser colocado dentro da função f(x), da função de iteração φ (Método do Ponto Fixo) e a derivada de f(x)(Método de Newton-Raphson) para a correta execução. Nos códigos entregues ao final do arquivo os dados das funções são referentes ao exercício 2.

Além da função o usuário também deve colocar como valores de entrada alguns dados iniciais (intervalo, precisão e aproximação inicial) após iniciar o programa. Seria melhor que as funções usadas também fossem valores de entrada do usuário, porém, tal solução é extremamente complexa e complicada de ser feita sem restringir a quantidade de funções que poderiam ser usadas.

A seguir seguem imagens com os resultados da execução dos métodos dos exercícios 2, 3 e 4. As imagens contém os resultados obtidos com todas as iterações, seguidas dos valores da raiz x, dos resultados das condições de parada e, por fim, dos resultados finais da execução. Os valores utilizados para a precisão e da aproximação inicial foram escolhidos com base nos exemplos do livro texto.

2) $x^3 - 9x + 3$

No código:

“return math.pow(x, 3) - 9*x + 3” para a f(x) de todos os métodos.
“return (math.pow(x, 3) / 9) + (1/3)” para a g(x) do Método do Ponto Fixo.
“return 3*math.pow(x, 2) – 9” para a df(x) do Método de Newton-Raphson.

Método da Bissecção

```
Intervalo inicial:
a: 0
b: 1
Precisão e: 0.001

Iteração 01: x = 0.5000000000, b - a = 0.5000000000
Iteração 02: x = 0.2500000000, b - a = 0.2500000000
Iteração 03: x = 0.3750000000, b - a = 0.1250000000
Iteração 04: x = 0.3125000000, b - a = 0.0625000000
Iteração 05: x = 0.3437500000, b - a = 0.0312500000
Iteração 06: x = 0.3281250000, b - a = 0.0156250000
Iteração 07: x = 0.3359375000, b - a = 0.0078125000
Iteração 08: x = 0.3398437500, b - a = 0.0039062500
Iteração 09: x = 0.3378906250, b - a = 0.0019531250
Iteração 10: x = 0.3369140625, b - a = 0.0009765625

A média aproximada é 0.33740234375, e foram necessárias 10 iterações
```

Método do Ponto Fixo, com $\varphi = \frac{x^3}{9} + \frac{1}{3}$

```
Precisões iniciais:
e1: 0.0005
e2: 0.0005
Aproximação inicial: 0.5

Iteração 01: |x1| = 0.3472222222, |x1 - x0| = 0.1527777778
Iteração 02: |x1| = 0.3379846941, |x1 - x0| = 0.0092375281
Iteração 03: |x1| = 0.3376232474, |x1 - x0| = 0.0003614467

A média aproximada é 0.33762324738031935, e foram necessárias 3 iterações
```

Método de Newton-Raphson, com $f'(x)=3x^2-9$

```
Precisões iniciais:
e1: 0.0001
e2: 0.0001
Aproximação inicial: 0.5

Iteração 01: |x1| = 0.3333333333, |x1 - x0| = 0.1666666667
Iteração 02: |x1| = 0.3376068376, |x1 - x0| = 0.0042735043

A média aproximada é 0.33760683760683763, e foram necessárias 2 iterações
```

Neste exemplo podemos ver que, mesmo possuindo uma precisão menor o método da Bissecção teve muitas mais iterações que os outros dois métodos, que foram resolvidos rapidamente.

3) $x^3 - x - 1$

No código:

“return math.pow(x, 3) - x - 1” para a f(x) de todos os métodos.
“return math.pow(x + 1, 1/3)” para a g(x) do Método do Ponto Fixo.
“return 3*math.pow(x, 2) – 9” para a df(x) do Método de Newton-Raphson.

Método da Bissecção

```
Intervalo inicial:
a: 1
b: 2
Precisão e: 0.000001

Iteração 01: x = 1.5000000000, b - a = 0.5000000000
Iteração 02: x = 1.2500000000, b - a = 0.2500000000
Iteração 03: x = 1.3750000000, b - a = 0.1250000000
Iteração 04: x = 1.3125000000, b - a = 0.0625000000
Iteração 05: x = 1.3437500000, b - a = 0.0312500000
Iteração 06: x = 1.3281250000, b - a = 0.0156250000
Iteração 07: x = 1.3203125000, b - a = 0.0078125000
Iteração 08: x = 1.3242187500, b - a = 0.0039062500
Iteração 09: x = 1.3261718750, b - a = 0.0019531250
Iteração 10: x = 1.3251953125, b - a = 0.0009765625
Iteração 11: x = 1.3247070312, b - a = 0.0004882812
Iteração 12: x = 1.3249511719, b - a = 0.0002441406
Iteração 13: x = 1.3248291016, b - a = 0.0001220703
Iteração 14: x = 1.3247680664, b - a = 0.0000610352
Iteração 15: x = 1.3247375488, b - a = 0.0000305176
Iteração 16: x = 1.3247222900, b - a = 0.0000152588
Iteração 17: x = 1.3247146606, b - a = 0.0000076294
Iteração 18: x = 1.3247184753, b - a = 0.0000038147
Iteração 19: x = 1.3247165680, b - a = 0.0000019073
Iteração 20: x = 1.3247175217, b - a = 0.0000009537

A média aproximada é 1.3247179985046387, e foram necessárias 20 iterações
```

Método do Ponto Fixo, com $\varphi=(x+1)^{(1/3)}$

```
Precisões iniciais:
e1: 0.000001
e2: 0.000001
Aproximação inicial: 1

Iteração 01: |x1| = 1.2599210499, |x1 - x0| = 0.2599210499
Iteração 02: |x1| = 1.3122938367, |x1 - x0| = 0.0523727868
Iteração 03: |x1| = 1.3223538191, |x1 - x0| = 0.0100599825
Iteração 04: |x1| = 1.3242687446, |x1 - x0| = 0.0019149254
Iteração 05: |x1| = 1.3246326253, |x1 - x0| = 0.0003638807
Iteração 06: |x1| = 1.3247017485, |x1 - x0| = 0.0000691233
Iteração 07: |x1| = 1.3247148784, |x1 - x0| = 0.0000131299
Iteração 08: |x1| = 1.3247173724, |x1 - x0| = 0.0000024940
Iteração 09: |x1| = 1.3247178462, |x1 - x0| = 0.0000004737

A média aproximada é 1.3247178461621456, e foram necessárias 9 iterações
```

Método de Newton-Raphson, com $f'(x)=x^2-1$

```
Precisões iniciais:
e1: 0.000001
e2: 0.000001
Aproximação inicial: 0

Iteração 01: |x1| = 1.0000000000, |x1 - x0| = 1.0000000000
Iteração 02: |x1| = 0.5000000000, |x1 - x0| = 0.5000000000
Iteração 03: |x1| = 3.0000000000, |x1 - x0| = 2.5000000000
Iteração 04: |x1| = 2.0384615385, |x1 - x0| = 0.9615384615
Iteração 05: |x1| = 1.3902821472, |x1 - x0| = 0.6481793912
Iteração 06: |x1| = 0.9116118977, |x1 - x0| = 0.4786702495
Iteração 07: |x1| = 0.3450284967, |x1 - x0| = 0.5665834010
Iteração 08: |x1| = 1.4277507040, |x1 - x0| = 1.0827222073
Iteração 09: |x1| = 0.9424179125, |x1 - x0| = 0.4853327915
Iteração 10: |x1| = 0.4049493572, |x1 - x0| = 0.5374685553
Iteração 11: |x1| = 1.7069046452, |x1 - x0| = 1.3019552880
Iteração 12: |x1| = 1.1557563611, |x1 - x0| = 0.5511482841
Iteração 13: |x1| = 0.6941918133, |x1 - x0| = 0.4615645477
Iteração 14: |x1| = 0.7424942987, |x1 - x0| = 1.4366861121
Iteração 15: |x1| = 2.7812959407, |x1 - x0| = 2.0388016420
Iteração 16: |x1| = 1.9827252470, |x1 - x0| = 0.7985706936
Iteração 17: |x1| = 1.5369273798, |x1 - x0| = 0.4457978673
Iteração 18: |x1| = 1.3572624832, |x1 - x0| = 0.1796648966
Iteração 19: |x1| = 1.3256630944, |x1 - x0| = 0.0315993888
Iteração 20: |x1| = 1.3247187886, |x1 - x0| = 0.0009443058
Iteração 21: |x1| = 1.3247179572, |x1 - x0| = 0.0000008314

A média aproximada é 1.3247179572453902, e foram necessárias 21 iterações
```

Podemos ver que o número de iterações para achar a raiz aproximada dessa função aumentou para todos os métodos. Para a Bissecção e para o Ponto Fixo podemos atribuir isso a precisão maior, porém o mesmo não pode ser dito para o método de Newton-Raphson que apresenta muitos valores erráticos durante as iterações e demora a se aproximar da raiz.

4) Para este exemplo não polinomial, foi escolhida a função $4\,sen(x)-e^{x}$

No código:
“return 4*math.sin(x) - math.pow(math.e, x)” para a f(x) de todos os métodos.
“return x - 2*math.sin(x) + 0.5*math.pow(math.e, x)” para a g(x) do Método do Ponto Fixo.
“return 4*math.cos(x) - math.pow(math.e, x)” para a df(x) do Método de Newton-Raphson.

Método da Bissecção

```
Intervalo inicial:
a: 0
b: 1
Precisão e: 0.00001

Iteração 01: x = 0.5000000000, b - a = 0.5000000000
Iteração 02: x = 0.2500000000, b - a = 0.2500000000
Iteração 03: x = 0.3750000000, b - a = 0.1250000000
Iteração 04: x = 0.3125000000, b - a = 0.0625000000
Iteração 05: x = 0.3437500000, b - a = 0.0312500000
Iteração 06: x = 0.3593750000, b - a = 0.0156250000
Iteração 07: x = 0.3671875000, b - a = 0.0078125000
Iteração 08: x = 0.3710937500, b - a = 0.0039062500
Iteração 09: x = 0.3691406250, b - a = 0.0019531250
Iteração 10: x = 0.3701171875, b - a = 0.0009765625
Iteração 11: x = 0.3706054688, b - a = 0.0004882812
Iteração 12: x = 0.3703613281, b - a = 0.0002441406
Iteração 13: x = 0.3704833984, b - a = 0.0001220703
Iteração 14: x = 0.3705444336, b - a = 0.0000610352
Iteração 15: x = 0.3705749512, b - a = 0.0000305176
Iteração 16: x = 0.3705596924, b - a = 0.0000152588
Iteração 17: x = 0.3705520630, b - a = 0.0000076294

A média aproximada é 0.3705558776855469, e foram necessárias 17 iterações
```

Método do Ponto Fixo, com $\varphi=2\,sen(x)+0.5(\varepsilon^{x})$

```
Precisões iniciais:
e1: 0.00001
e2: 0.00001
Aproximação inicial: 0.5

Iteração 01: |x1| = 0.3655095581, |x1 - x0| = 0.1344904419
Iteração 02: |x1| = 0.3712831916, |x1 - x0| = 0.0057736335
Iteração 03: |x1| = 0.3704569785, |x1 - x0| = 0.0008262131
Iteração 04: |x1| = 0.3705722577, |x1 - x0| = 0.0001152792
Iteração 05: |x1| = 0.3705561138, |x1 - x0| = 0.0000161440

A média aproximada é 0.37055611376057707, e foram necessárias 5 iterações
```

Método de Newton-Raphson, com $f'(x)=4\cos(x)-e^x$

```
Precisões iniciais:
e1: 0.00001
e2: 0.00001
Aproximação inicial: 0.5

Iteração 01: |x1| = 0.3555116101, |x1 - x0| = 0.1444883899
Iteração 02: |x1| = 0.3704194511, |x1 - x0| = 0.0149078410
Iteração 03: |x1| = 0.3705580838, |x1 - x0| = 0.0001386326

A média aproximada é 0.37055808376128907, e foram necessárias 3 iterações
```

Por fim, para a última função temos que, assim como no exercício 2, o Método da Bissecção tem um número de iterações muito maior que os outros 2 métodos

Conclusões Finais

Inicialmente, o método da Bissecção aparente ser o que requer maior esforço computacional visto que ela apresenta maior número de iterações, entretanto um estudo mais preciso mostra que existem outros aspectos a serem analisados.

O método da Bissecção necessita apenas de um intervalo contínuo contendo uma raiz para sua execução, ao mesmo modo que os dois outros métodos necessitam além de valores iniciais, funções auxiliares (a derivada da f(x) no caso do Método de Newton-Raphson, e uma função equivalente no Método do Ponto Fixo) que podem ser difíceis de serem calculadas, seja pelo computador ou por humanos. Uma escolha errada na aproximação também pode levar a um número de iterações muito maior que o esperado, como visto no exercício 3 do Método de Newton-Raphson.

Portanto, podemos concluir que, conforme implementadas corretamente e com boas escolhas iniciais, tanto o Método do Ponto Fixo como o Método de Newton-Raphson solucionam os problemas de forma mais rápida que o método da Bissecção, que geralmente é uma escolha melhor para soluções mais simples.

Referências

[1] Cálculo Numérico: Aspectos Teóricos e Computacionais, 2ª edição, Márcia A. Gomes Ruggiero e Vera Lúcia da Rocha Lopes
[2] https://docs.python.org/3/library/math.html

Anexos

Método da Bissecção, código fonte em python 3.4.3:

```
import math

def f(x):
	return math.pow(x, 3) - 9*x + 3

def bisseccao(a, b):
	return (a + b) / 2

def metodoBisseccao(a, b, e):
	M = f(a)
	x = 0.0
	k = 0

	while (b - a) > e:
		x = bisseccao(a, b)

		if (M * f(x)) > 0:
			a = x
		else:
			b = x

		k = k + 1
		print("Iteração {:02d}: x = {:.10f}, b - a = {:.10f}" .format(k, x, b-a))

	return bisseccao(a, b), k

print("\nIntervalo inicial:")
a = (float)(input("a: "))
b = (float)(input("b: "))
e = (float)(input("Precisão e: "))
print()

raizAproximada, iteracoes = metodoBisseccao(a, b, e)
print('\nA média aproximada é {}, e foram necessárias {} iterações' .format(raizAproximada, iteracoes))
```

Método do Ponto Fixo, código fonte em python 3.4.3:

```
import math

def f(x):
	return math.pow(x, 3) - 9*x + 3

def g(x):
	return (math.pow(x, 3) / 9) + (1/3)

def modulo(x):
	if x < 0:
		return (-1) * x
	else:
		return x

def metodoPontoFixo(e1, e2, x0):
	x1 = x0
	k = 1

	if modulo(f(x0)) < e1:
		return x0, 0
	else:
		x1 = g(x0)
		print("Iteração {:02d}: |x1| = {:.10f}, |x1 - x0| = {:.10f}" .format(k, modulo(x1), modulo(x1-x0)))

	while (modulo(f(x1)) > e1) and (modulo(x1 - x0) > e2):
		x0 = x1
		x1 = g(x0)
		k = k + 1
		print("Iteração {:02d}: |x1| = {:.10f}, |x1 - x0| = {:.10f}" .format(k, modulo(x1), modulo(x1-x0)))

	return x1, k

print("\nPrecisões iniciais:")
e1 = (float)(input("e1: "))
e2 = (float)(input("e2: "))
x0 = (float)(input("Aproximação inicial: "))

raizAproximada, iteracoes = metodoPontoFixo(e1, e2, x0)
print('\nA média aproximada é {}, e foram necessárias {} iterações' .format(raizAproximada, iteracoes))
```

Método de Newton-Raphson, código fonte em python 3.4.3:

```python
import math

def f(x):
	return math.pow(x, 3) - 9*x + 3

def df(x):
	return 3*math.pow(x, 2) - 9

def modulo(x):
	if x < 0:
		return (-1) * x
	else:
		return x

def metodoNewtonRaphson(e1, e2, x0):
	x1 = x0
	k = 1

	if modulo(f(x0)) < e1:
		return x0, 0
	else:
		x1 = x0 - (f(x0) / df(x0))
		print("Iteração {:02d}: |x1| = {:.10f}, |x1 - x0| = {:.10f}" .format(k, modulo(x1), modulo(x1-x0)))

	while (modulo(f(x1)) > e1) and (modulo(x1 - x0) > e2):
		x0 = x1
		x1 = x0 - (f(x0) / df(x0))
		k = k + 1
		print("Iteração {:02d}: |x1| = {:.10f}, |x1 - x0| = {:.10f}" .format(k, modulo(x1), modulo(x1-x0)))

	return x1, k

print("\nPrecisões iniciais:")
e1 = (float)(input("e1: "))
e2 = (float)(input("e2: "))
x0 = (float)(input("Aproximação inicial: "))

raizAproximada, iteracoes = metodoNewtonRaphson(e1, e2, x0)
print('\nA média aproximada é {}, e foram necessárias {} iterações' .format(raizAproximada, iteracoes))
```