Métodos Computacionais
Trabalho I

Daniel De Luca, Diego Jornada, Matthias Nunes

[1]Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

{daniel.luca, diego.jornada, matthias.nunes}@acad.pucrs.br

Resumo. *Este artigo descreve uma síntese sobre computação científica e alguns métodos iterativos, quando utilizados para achar números irracionais.*

1. Introdução

Computação científica é a área que estuda a construção de modelos matemáticos e técnicas para análises quantitativas para resolver problemas cinetíficos e de engenharia. Na prática, é a aplicação de simulações computacionais em análises numéricas, ciência da computação teórica e outros problemas científicos de disciplinas variadas.

No brasil, existem alguns institutos dedicados a computação científica, como por exemplo o LNCC (Laboratório Nacional de Computação Científica) ou a SBMAC (Sociedade Brasileira de Matemática Aplicada e Computacional). No mundo, os mais conhecidos são: SCG (Symbolic Computation Group), NAG (Numerical Algorithms Group), entre outros.

2. Métodos Iterativos

Um **método iterativo** é um procedimento matemático para resolução de equações que gera uma sequência de aproximações, cada vez melhores, que servem como solução para uma classe de problemas. Uma implementação específica de um método iterativo, incluindo sua condição de parada, é um algoritmo. Um método iterativo é considerado **convergente** se a sequência de valores geradas converge para uma dada aproximação inicial.

> Para uma definição mais teórica, o seguinte autor define:
>
> "Um método iterativo consiste em repetir uma determinada operação um certo número de vezes até que nos seja fornecida uma aproximação, que satisfaça as condições do problema e, para tal, a sequência de valores deve ser convergente."[Batista 2014]

Nos problemas do tipo encontre a raíz da equação, um método iterativo usa uma suposição inicial para gerar sucessivas aproximações à uma solução. Em Contraste, **métodos diretos** tentam resolver o problema em uma sequência *finita* de operações.

Um método iterativo é formado por quatro partes: [Cláudio 2000]

a) Estimativa inicial: uma ou mais aproximações para a raiz desejada.
b) Atualização: uma fórmula que atualize a solução aproximada.
c) Critério de parada: uma forma de estabelecer quando parar o processo iterativo em qualquer caso.
d) Estimador de exatidão: está associado ao critério de parada e provê uma estimativa do erro cometido.

3. Exercícios

Nessa seção vamos encontrar números irracionais conhecidos utilizando métodos iterativos.

3.1. Número de Ouro

O número de ouro, também conhecido como ϕ, é o resultado da seguinte equação:

$$x^2 - x - 1 \tag{1}$$

Existem algums algoritmos para realizar a aproximação neste artigo, sendo duas destas formas abordadas. A primeira utilizando frações continuas e o seguundo calculando as raizaes da equação utiizando método de newton.

3.1.1. Frações Continuadas

Frações conitinuadas são formas de representar números reais de tal forma que a expressão básica tem o seguinte formato:

$$a_0 + \frac{b_0}{a_1 + \frac{b_1}{a_n + \ldots}} \tag{2}$$

Para calcular o ϕ devemos substituir a e b por 1. Na Fração 2.

como é feita essa tabela?

Iteração	x	Iteração	x
1	2.00000000000000000000	26	1.61803398873830306393
2	1.50000000000000000000	27	1.61803398875432247195
3	1.66666666666666651864	28	1.61803398874820381081
4	1.60000000000000008882	29	1.61803398875054060824
5	1.62500000000000000000	30	1.61803398874964821097
6	1.61538461538461541878	31	1.61803398874998904944
7	1.61904761904761906877	32	1.61803398874985893130
8	1.61764705882352943789	33	1.61803398874990866929
9	1.61818181818181816567	34	1.61803398874988957346
10	1.61797752808988759554	35	1.61803398874989690093
11	1.61805555555555558023	36	1.61803398874989401435
12	1.61802575107296142676	37	1.61803398874989512457
13	1.61803713527851456000	38	1.61803398874989490253
14	1.61803278688524576623	39	1.61803398874989490253
15	1.61803444782168193150	40	1.61803398874989490253
16	1.61803381340012508716	41	1.61803398874989490253
17	1.61803405572755432118	42	1.61803398874989490253
18	1.61803396316670644595	43	1.61803398874989490253
19	1.61803399852180351814	44	1.61803398874989490253
20	1.61803398501735795634	45	1.61803398874989490253
21	1.61803399017559712547	46	1.61803398874989490253
22	1.61803398820532517988	47	1.61803398874989490253
23	1.61803398895790184753	48	1.61803398874989490253
24	1.61803398867044334608	49	1.61803398874989490253
25	1.61803398878024262686	50	1.61803398874989490253

Tabela 1. Convergência do número de ouro pelo método de frações continuadas

Como fica ilustrado na tabela, o número já se estabiliza antes mesmo de chegarmos à 50 iterações.

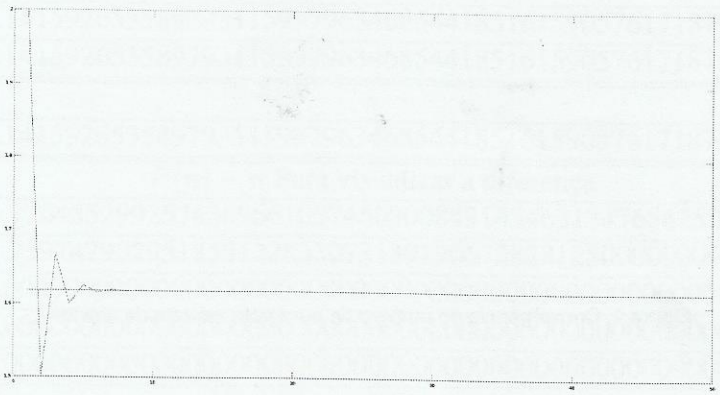

Figura 1. Convergência do número de ouro pelo método de frações continuadas

O método iterativo utilizado foi descrito pelo seguinte código:

qual o nome comp.?

```
function [x] = phi_frac(iteration=1)
  aux = 1;
  for i= 1:iteration
    aux = double(1 + 1/aux);
    x(i) = aux;
  end
end
```

não é um alg. mmm!

3.1.2. Método de Newton

A ideia do método de Newton é a partir de um valor inicial arbitrário, então a função é aproximada por sua reta tangente. O x que intercepta a reta e a função é computado, e ele será uma melhor aproximação que o chute inicial. O método, então, pode ser iterado.

Analizando novamente o número de ouro, mas com o método de newton, um número muito menor de iterações é observado.

Iteração	x	erro
1	1.66666666666666674068	0.33333333333333325932
2	1.61904761904761906877	0.04761904761904767192
3	1.61803444782168193150	0.00101317122593713727
4	1.61803398874998904944	0.00000045907169288206
5	1.61803398874989490253	0.00000000000009414691
6	1.61803398874989490253	0.00000000000000000000

Qual a fórmula?

Tabela 2. Convergência do número de ouro pelo método de newton

Isso pode ser observado no gráfico abaixo.

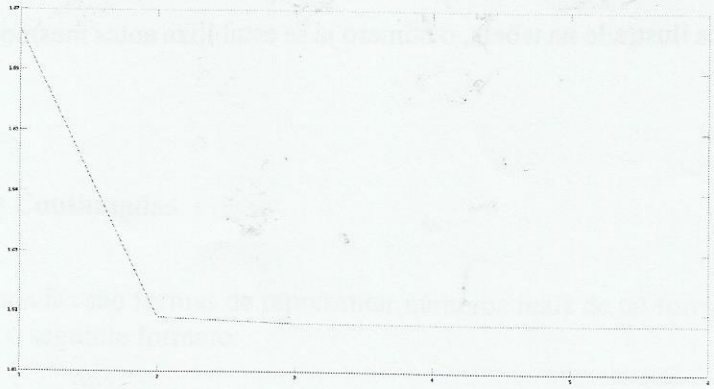

Figura 2. Convergência do número de ouro pelo método de newton

O método iterativo utilizado foi descrito pelo seguinte código:

```
function [x, ex] = newton( f, df, x0, tol, nmax)
  f = inline(f);
  df = inline(df);
  x(1) = double(x0 - (f(x0)/df(x0)));
  ex(1) = abs(x(1)-x0);
  k = 2;
  while k <= nmax && ex(k-1) > tol
     x(k) = double(double(x(k-1)) - double((f(x(k-1))/df(x(k-1)))));
     ex(k) = abs(x(k)-x(k-1));
     k = k+1;
  end
end
```

3.2. Pi(π)

3.2.1. Método Utilizando Funções Trigonométricas

Encontramos em um fórum de matemática um método iterativo que calcula π de uma forma aparentemente mais simples, apesar de sua complexidade estar escondida na função *sin*. O método está descrito a seguir:

$$P(n+1) = P(n) + sin(P(n)) \tag{3}$$

$P(n)$ seria a aproximação de π na iteração n. Esse método consegue convergir para π com um número muito baixo de iterações.

$P(n)$
3.14112000805986735230135309393517673015594482421875
3.14159265357219563696844488731585443019866943359375
3.14159265358979311599796346854418516159057617187500
3.14159265358979311599796346854418516159057617187500
3.14159265358979311599796346854418516159057617187500
π
3.14159265358979311599796346854418516159057617187500
$P(n) - \pi$ Para visualizar a diferença
-4.72645529925763696610374609008431434631347656250000
-1.75974790295185812283307313919067382812500000000000
0.00000000000000000000000000000000000000000000000000
0.00000000000000000000000000000000000000000000000000
0.00000000000000000000000000000000000000000000000000

Tabela 3. Convergência do π

O seguinte gráfico foi gerado com a análise dos resultos:

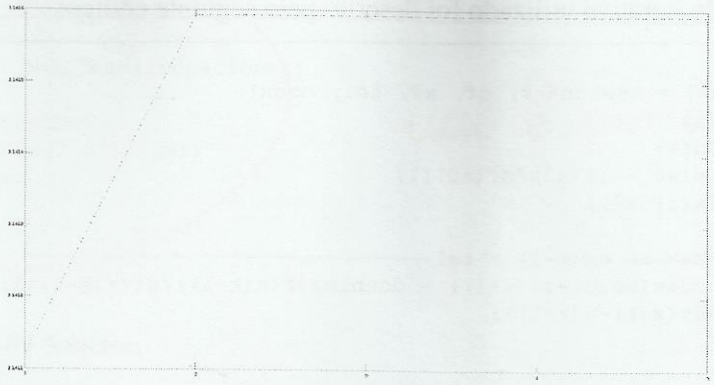

Figura 3. Convergência do π

O método iterativo utilizado foi descrito pelo seguinte código:

```
function [pif, pi_vec] = pi_it(iteration)
  pif(1) = 3 + sin(3);
  pi_vec(1) = pi
  for i = 2:iteration
    pif(i) = pif(i-1) + sin(pif(i-1));
    pi_vec(i) = pi;
    aux = pif(i);
  end
end
```

3.3. e^x

A função e^x é um função exponencial cuja base é o número de Euler, também conhecida como função exponencial natural.

Para o calculo da da função utilizamos a seguinte série de Taylor

$$\sum_{n=1}^{n} = \frac{x^n}{n!} \tag{4}$$

Para exemplificar a foram realizadas 40 iteração para a seguinte função e^5, e obtivemos os seguintes resultaodos:

Iteração	Valor Iteração	e^5	Diferença
1	1.000000	148.413159	147.413159
2	6.000000	148.413159	142.413159
3	18.500000	148.413159	129.913159
4	39.333333	148.413159	109.079826
5	65.375000	148.413159	83.038159
6	91.416667	148.413159	56.996492
7	113.118056	148.413159	35.295104
8	128.619048	148.413159	19.794111
9	138.307168	148.413159	10.105991
10	143.689457	148.413159	4.723703
11	146.380601	148.413159	2.032558
12	147.603849	148.413159	0.809311
13	148.113535	148.413159	0.299624
14	148.309568	148.413159	0.103591
15	148.379580	148.413159	0.033579
16	148.402917	148.413159	0.010242
17	148.410210	148.413159	0.002949
18	148.412355	148.413159	0.000804
19	148.412951	148.413159	0.000208
20	148.413108	148.413159	0.000051
21	148.413147	148.413159	0.000012
22	148.413156	148.413159	0.000003
23	148.413159	148.413159	0.000001
24	148.413159	148.413159	0.000000
25	148.413159	148.413159	0.000000
26	148.413159	148.413159	0.000000
27	148.413159	148.413159	0.000000
28	148.413159	148.413159	0.000000
29	148.413159	148.413159	0.000000
30	148.413159	148.413159	0.000000
31	148.413159	148.413159	0.000000
32	148.413159	148.413159	0.000000
33	148.413159	148.413159	0.000000
34	148.413159	148.413159	0.000000
35	148.413159	148.413159	0.000000
36	148.413159	148.413159	0.000000
37	148.413159	148.413159	0.000000
38	148.413159	148.413159	0.000000
39	148.413159	148.413159	0.000000
40	148.413159	148.413159	0.000000

Tabela 4. Convergência de e^5

Utilizando a série de Taylor foi percebido que o método converge após 24 iterações.A complexidade deste método encontra-se no cáculo de $n!$. Podemos observar na

Figura 2 a representação gráfica da convegência entre a série e e^5.

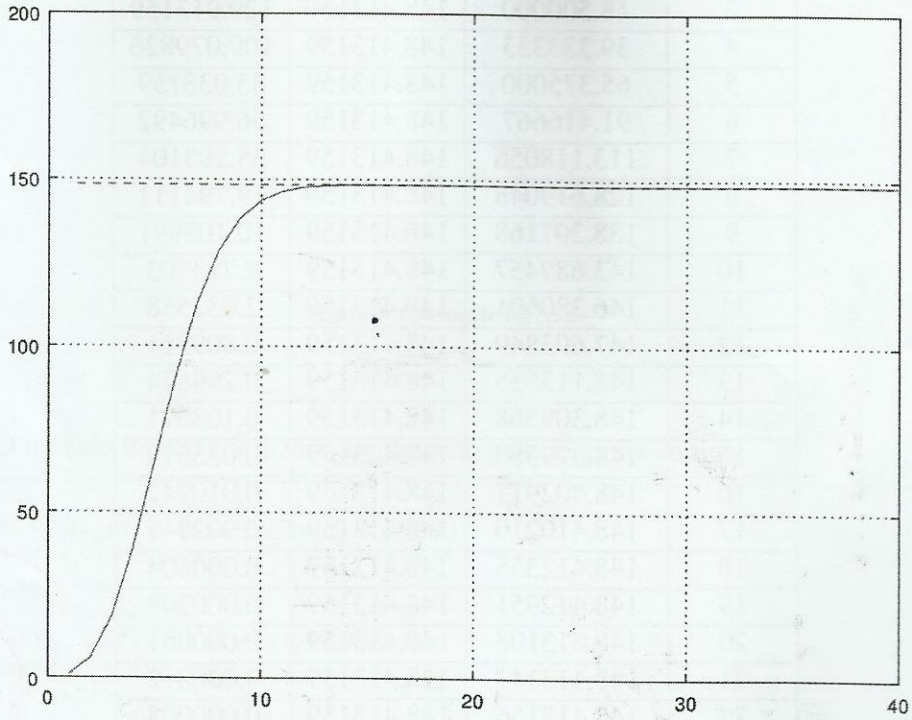

Figura 4. Convergência de e^5 em 40 iterações

Utilizamos o Algoritmo exponential implementado como mostrado a seguir, que retorna uma matriz de forma que cada coluna representa uma iteração e a primeira linha o valor calculado, a segunda o valor de e^5 e a tereceira linha é utilizada para guardar a diferença entre e^5 o valor calculado na iteração.

```
function [ex] = exponential(x, iteration)
  ex2 = e^x;
```

```
  ex(1,1) = 1;
  ex(2,1) = ex2;
  ex(3,1) = ex2 - ex(1,1);
  for k=2:iteration
    ex(1,k) = ex(1,k-1) + (power(x,k-1) / factorial(k-1));
    ex(2,k) = ex2;
    ex(3,k) = ex2 - ex(1,k);
  end
end
```

3.4. Erdős-Borwein

A constante de Erdős-Borwein é a soma dos inversos multiplicativos dos números de Mersenne, e por definição é:

$$E = \sum_{n=1}^{\infty} \frac{1}{2^n - 1} \approx 1.606695152415291763\ldots \tag{5}$$

Executando no *octave*, obtemos os seguintes resultados:

Iteração	E	Iteração	E
1	1.0000000000	19	1.6066932451
2	1.3333333333	20	1.6066941987
3	1.4761904762	21	1.6066946756
4	1.5428571429	22	1.6066949140
5	1.5751152074	23	1.6066950332
6	1.5909882232	24	1.6066950928
7	1.5988622390	25	1.6066951226
8	1.6027838076	26	1.6066951375
9	1.6047407548	27	1.6066951450
10	1.6057182719	28	1.6066951487
11	1.6062067917	29	1.6066951506
12	1.6064509919	30	1.6066951515
13	1.6065730771	31	1.6066951519
14	1.6066341160	32	1.6066951522
15	1.6066646345	33	1.6066951523
16	1.6066798935	34	1.6066951524
17	1.6066875230	35	1.6066951524
18	1.6066913377	36	1.6066951524

Tabela 5. Convergência do E

Podemos ver o número se estabilizando no seguinte gráfico, construído a partir dos resultados da tabela.

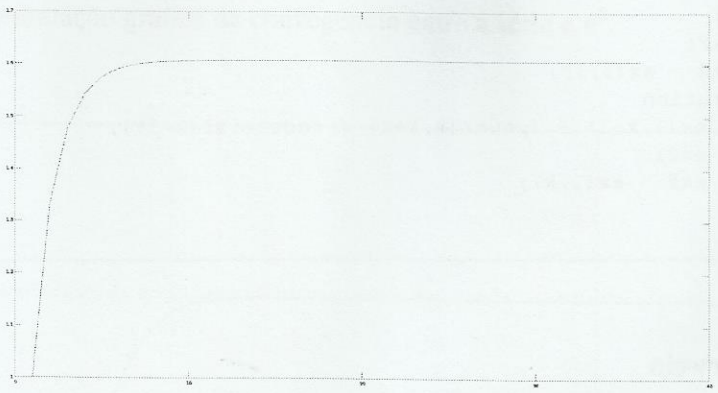

Figura 5. Convergência do erdős-borwein em 50 iterações

A definição de E foi codificada da seguinte maneira:

```
function [er] = erdos(iteration)
  er(1) = 1;
  for i=2:iteration
    er(i) = er(i-1) + (1 / ((2^i)-1));
  end
end
```

4. Conclusão

Métodos iterativos são extremamente importantes dentro da computação, pois eles nos permitem aproximar números irracionais, com precisão aceitável a ser usada no sistema de ponto flutuante dentro dos computadores. Observamos que diferentes métodos possuem diferentes características que podem complicar ou simplificar em alguns aspectos. Como na equação 3, onde aparenta ser simples, mas a complexidade está escondida dentro da função *sin*, que por sua vez, é implementada utilizando série de taylor por trás.

Referências

[Batista 2014] Batista, W. C. (2014). Métodos iterativos na resolução de equações. ? livro, artigo?

[Cláudio 2000] Cláudio, D. M. (2000). *Cálculo Numérico Computacional (teoria e prática): algoritmos em pseudo-linguagem.*