Análise Numérica
Trabalho 1
Grupo 8

Diogo Bogas (201202482), Diogo Fernandes (201403232),
Iúri Pena (201304604), João Ferreira (201304824)

19 de Fevereiro de 2017

Conteúdo

1 Exercícios

Todos os exercícios foram resolvidos usando a linguagem Java, numa máquina Linux 64 bits.
Nos exercícios de cálculo de erro, o valor exato foi extraído da biblioteca *Math* da linguagem.
Os exercícios abordam erros e apróximações de séries a um valor concreto. Neste trabalho os valores a calcular foram Epsilon máquina, constante de Euler, contante Pi e log(2).

Os resultados destes problemas são aproximações com determinados erros a um valor exacto através de programas que incidem em somatórios de expressões.

1.1 Cálculo do valor de *epsilon* máquina (*eps*) a usar

De maneira a calcular o valor de *eps* foi elaborado um simples algoritmo que permite identificar quando a máquina em questão deixa de conseguir distinguir o valor de 1+eps de 1.
O Epsilon máquina, majorante do erro relativo associado aos arrendamentos de aritmética em vírgula flutuante, é visto como o menor valor que somado à unidade produza um resultado diferente de um (eps+1¿1).
Dependendo da conveção a usar, o valor considerado pode ser 1.1102230246251565E-16 ou 2.220446049250313E-16.
O algoritmo pode ser consultado no Anexo A.

1.2 Cálculo da constante de *Euler*

Foram fornecidas duas expressões para o cálculo da constante de *Euler*, sendo elas:

$$a_n = (1 + \frac{1}{n})^n \tag{1}$$

$$b_m = \sum_{k=0}^{m} \frac{1}{k!} \tag{2}$$

Foram obtidos os seguintes resultados:

Figura 1:

```
a(1) = 2.0
a(2) = 2.25
a(3) = 2.37037037037037
a(4) = 2.44140625
a(5) = 2.4883199999999994
a(6) = 2.5216263717421135
a(7) = 2.546499697040712
a(8) = 2.565784513950348
a(9) = 2.5811747917131984
a(10) = 2.5937424601000023
a(11) = 2.6041990118975287
a(12) = 2.613035290224676
a(13) = 2.6206008878857308
a(14) = 2.6271515563008685
a(15) = 2.6328787177279187
a(16) = 2.6379284973666
a(17) = 2.64241437518311
a(18) = 2.6464258210976865
a(19) = 2.650034326640442
a(20) = 2.653297705144422
a(21) = 2.656263213926108
a(22) = 2.658969858537786
a(23) = 2.6614501186387796
a(24) = 2.663731258068599
a(25) = 2.665836331487422
Valor real = 2.718281828459045
Erro = 0.05244549697162304
```

```
b(1) = 2.0
b(2) = 2.5
b(3) = 2.6666666666666665
b(4) = 2.708333333333333
b(5) = 2.7166666666666663
b(6) = 2.7180555555555554
b(7) = 2.7182539682539684
b(8) = 2.71827876984127
b(9) = 2.7182815255731922
b(10) = 2.7182818011463845
b(11) = 2.718281826198493
b(12) = 2.7182818282861687
b(13) = 2.7182818284467594
b(14) = 2.71828182845823
b(15) = 2.718281828458995
b(16) = 2.718281828459043
b(17) = 2.7182818284590455
b(18) = 2.7182818284590455
b(19) = 2.7182818284590455
b(20) = 2.7182818284590455
b(21) = 2.7182818284590455
b(22) = 2.7182818284590455
b(23) = 2.7182818284590455
b(24) = 2.7182818284590455
b(25) = 2.7182818284590455
Valor real = 2.718281828459045
Erro = -4.440892098500626E-16
```

Ao olharmos para os resultados reparamos que para o mesmo número de iterações, a segunda expressão é claramente melhor a calcular a constante de Euler pois apresenta um erro na casa dos 10^{-16} enquanto que a primeira, apresenta um erro na casa dos 10^{-2}. Note-se também que o erro da segunda equação é negativo porque a constante da linguagem usada tem uma casa decimal a menos, o que faz que o erro seja maior e a subtração negativa.
A implementação usada para o cálcudo da expressão (1) pode ser consultada no Anexo A assim como para a expressão (2).

1.3 Cálculo de um valor aproximado de π

Para o cálculo de π, foi usada a expressão dada:

$$\pi = \sum_{k=0}^{\infty} \frac{2^{k+1} k!^2}{(2k+1)!} \tag{3}$$

Era também pedido que o resultado tivesse um erro absoluto inferior ao intervalo de erros entre 10^{-5} e 10^{-12}. Nestas condições, os valores foram encontrados, por ordem, nas iterações 16 (erro: 6.257552732868987^{-6}), 19 (erro: 7.259146475036005^{-7}), 22 (erro: 8.503615944732701^{-8}), 26 (erro: 4.929842312151322^{-9}), 29 (erro: 5.863114438398043^{-10}),
32 (erro: 7.004707924807008^{-11}), 35 (erro: 8.400835582733635^{-12}) e 39 (erro: 5.000444502911705^{-13}).
Para poder computar a série pedida neste exercício teve que se estabelecer uma recorrência entre os termos da sucessão da série. Sem isto, ao fim de algumas iterações seria impossivel guardar alguns valores devido ao seu tamanho.
O código utilizado pode ser consultado no Anexo A.

1.4 Cálculo da soma da série

A série a calcular era:

$$\sum_{i=1}^{\infty} (-1)^{i+1} \frac{1}{i} \tag{4}$$

Era também pedido que o erro fosse inferior a $5*10^{-8}$, o que se obteve para i = 9999968, sendo o valor da série 0.693147130559946 com um erro de 4.999999914101494E-8 e, posteriormente, para $5 * 10^{-10}$ onde se obteve um i = 998596072, sendo o valor da série 0.693147130559946 com um erro de 4.99999930347883E-10. Neste exercício é mais visível a próximidade do erro obtido ao esperado. Para o primeiro teste, o erro obtido menos o esperado é, aproximadamente, $8*10^{-16}$ e no segundo, cerca de $6*10^{-16}$. Relacionando estes resultados com o valor de Epsilon obtido, conseguimos observar que para o segundo erro, este está perto do valor mínimo detectável pela máquina. De notar que a série representada é equivalente a log(2), valor representado pela biblioteca da linguagem usada.

2 Conclusão

No final da resolução dos exercícios, pensamos ter obtido resultados corretos, visto que estes se aproximam dos valores pretendidos e o erro entre esses valores e o valor real diminuir ao longo das iterações.
Ao longo do trabalho encontramos certos problemas como questionar se os resultados que estavam a ser determinados estavam corretos. Usando erros absolutos foi mais fácil entender que quanto menor o erro, mais próximos do resultado exacto as nossas soluções se encontravam.

A Código

Código 1: Cálculo do *epsilon* máquina a ser usado

```
double eps = 1;
  while(eps+1>1)
     eps/=2;

  System.out.println("O Epsilon maquina (1) e: " + eps);
  System.out.println("O Epsilon maquina (2) e: " + eps*2);
```

Código 2: Cálculo da constante de *Euler* através da expressão (1)

```
  double result=0;
  for(int i=1; i<=15; i++){
     double k = (double)i;
     double base = (1+((double)(1/k)));
     result = Math.pow((double)base, (double)k);
     System.out.println("a("+i+") = "+result);
  }

  double error = Math.exp(1)-result;
  System.out.println("Valor Real = " + Math.exp(1));
  System.out.println("Erro = " + error);
```

Código 3: Cálculo da constante de *Euler* através da expressão (2)

```
  double result=0;
  double fact;
  for(int m=0; m<=25; m++){
     result=0;
     fact=1;
     for(int k=0; k<=m; k++){
        if(k!=0)
           fact*=k;
        result+=(1/fact);
     }
     System.out.println("b("+m+") = "+result);
  }

  double error = Math.exp(1)-result;
  System.out.println("Valor Real = " + Math.exp(1));
  System.out.println("Erro = " + error);
```

Código 4: Cálculo de um valor aproximado de π através da expressão (3)

```
    int inf = Integer.MAX_VALUE;
  double pi=0, num, denum, num2, fact = 1, num1, checkError=0.00001, error;
  int conta=5;
  boolean check=false;

  for(int i=0; i<inf; i++){
     if(i!=0)
        fact*=i;
     num1 = Math.pow((double)2, (double)(i+1));
     num2 = Math.pow((double)fact, (double)2);
```

```
      num = num1*num2;
      denum = factorial(2*i+1);
      pi+=num/denum;
      error = Math.PI-pi;
      if(error<checkError && !check){
         System.out.println("Para um erro absoluto de 10^-"+conta+":");
         System.out.println("O numero n de termos usados na srie: " + i);
         System.out.println("Sn, o valor aproximado de Pi: "+ pi);
         System.out.println("O erro absoluto efetivamente cometido no clculo de Pi, En = |Pi-Sn|:
             " + error);
         System.out.println();
         conta++;
         checkError/=10;
         if(conta==13){
            check=true;
            break;
         }
      }
   }
}

public static double factorial(int n){
    double fact = 1; // this will be the result
     for(int i=1; i<=n; i++)
       fact *= (double)i;

     return fact;
}
```

Código 5: Cálculo da soma da série (4)

```
   int max=Integer.MAX_VALUE;
   double res=0, first, second,
   first_error=0.00000005, second_error=0.0000000005,
   exact_value = Math.log(2), error;
   int i;
   boolean check = false;

   for(i=1; i<max; i++){
      if(i%2!=0)first=1;
      else first=-1;
      second=(double)1/i;
      res+=(double)first*second;
      error = Math.abs(exact_value-res);
      if(error<first_error && !check){
         System.out.println("Para um erro de 5.0x10^-8:");
         System.out.println("A pesquisa correu "+i+" vezes (N="+i+")");
         System.out.println("A pesquisa obteve um resuldado de " + res);
         System.out.println();
         check=true;
      }
      if(error<second_error){
         System.out.println("Para um erro de 5.0x10^-10:");
         System.out.println("A pesquisa correu "+i+" vezes (N="+i+")");
         System.out.println("A pesquisa obteve um resuldado de " + res);
         break;
      }
   }
```