METODA NUMERIK

PRAKTIKUM 1 : PENDAHULUAN KOMPUTASI SAINTIFIK

Tujuan	Memahami berbagai aturan dan realisasi komputasi pada komputer
Kompetensi Dasar	1. Memahami dan mengimplementasikan error mutlak dan error relatif pada aproksimasi faktorial dengan formula Stirling. 2. Memahami dampak rounding error komputer terhadap aproksimasi bilangan natural e . 3. Mengaproksimasi e dengan polynomial Taylor.

1. Aproksimasi nilai faktorial dengan formula Stirling

Sebagaimana dipelajari bahwa zaman dahulu orang menghitung nilai $n!$ untuk n cukup besar adalah dengan menggunakan pendekatan rumus Stirling, yaitu

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

Misalkan $7! \approx \sqrt{14\pi}\left(\frac{7}{e}\right)^7$. Pada implementasi ini kita akan melihat akurasi aproksimasi ini dengan menhitung kesalahan mutlak dan kesalahan relatif untuk beberapa nilai n. Coba lakukan perhitungan pada MATLAB langsung pada command windows. Misalnya kita ingin menghitung 7! dengan rumus Stirling dan melihat kedua jenis kesalahan tersebut.

Tulis dulu perintah berikut

```
>> format long
```

untuk menampilkan bilangan dalam format panjang (long). Kemudian untuk mengitung 7! secara eksak, tulis

```
>> a = factorial(7)
a =
        5040
```

Ini berarti a = 5040 adalah nilai eksak untuk 7!. Untuk menghitung aproksimasinya dengan rumus Stirling, kita berikan perintah berikut

```
>>n=7;stirling7 = sqrt(2*pi*n)*(n/exp(1))^n

stirling7 =

     4.980395831612455e+003
```

Output ini berarti $4.980395831612455 \times 10^3 = 4980.395831612455$ atau dibulatkan menjadi 4980. Untuk mengetahui kesalahan mutlak dan kesalahan relatif, tuliskan

```
>> >> err_mutlak=abs(a-stirling7),err_rel=err_mutlak/stirling7

err_mutlak =


  59.604168387544632
```

```
err_rel =

   0.011967757263231
```

Ini berarti: walaupun kesalahan mutlak cukup besar yaitu mencapai 60 namun kesalahan relatifnya sangat kecil yaitu sekitar 0.012 atau 1.2% saja. Bandingkan hasilnya dengan Tabel 2.2 pada buku teks.

Bagaimana cara membangkitkan data aproksimasi dan kesalahan untuk bilangan bulat positif lainnya. Tentu cara manual ini sangat tidak efektif. Untuk itu kita susun m-file berikut (lihat buku teks hal 65): Buka dulu m-file editor atau new script, tulis perintah berikut.

```
n = 15;%bilangan bulat terbesar yang mau dihitung faktorialnya
eks = zeros(n,1);% menyiapkan matriks kolom untuk nilai faktorial eksak
ap=zeros(n,1);% sama seperti sebelumnya tapi untuk aproksimasi
err=zeros(n,1);% sama dengan sebelumnya untuk error mutlak
err_re=zeros(n,1);% sama dengan sebelumnya untuk error relatif
      for k = 1:n %k berjalan dari 1,2,3,...,n
            eks(k)  = factorial(k);%menghitung nilai faktorial secara eksak
            ap(k)  = sqrt(2*pi*k)*(k/exp(1))^k;%implementasi rumus Stirling
            err(k)=abs(ap(k)-eks(k));%pendefinisian error mutlak
            err_re(k)=err(k)/eks(k);%pendefinisian error relatif.
      end
[eks ap err err_re]%menampilkan ke empat vektor di atas dalam bentuk table.
```

Ingat perintah yang diawali oleh tanda % bukan bagian program tetapi hanya komen atau keterangan. Selanjutnya simpan file ini dengan nama **stirling.m**. Selanjutnya lakukan pada command window perintah berikut, kemudian enter.

```
>> stirling
```

Apa yang Anda peroleh! Bandingkan dengan data pada Tabel. Observasilah data tersebut, jawablah pertanyaan berikut

Pertanyaan untuk responsi

1. Bagaimana kecenderungan pola kesalahan mutlaknya?
2. Bagaimana kecenderungan pola kesalahan relatifnya?
3. Dapatkan Anda melengkapi tabel tersebut sampai $n = 20$. Coba lakukan dan berikan hasilnya. Apakah kesalahannya masih tetap konsisten?
4. Gambarkan pada grafik pola kedua jenis error tersebut.

2. Aproksimasi nilai e melalui formula limit

Pada kalkulus Anda tahu bahwa nilai e didefinisikan sebagai limit berikut

$$e = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n.$$

Ini berarti semakin besar n maka nilai $\left(1 + \frac{1}{n}\right)^n$ semakin mendekati nilai e. Untuk mengetahui nilai eksak e pada MATLAB, berikan perintah berikut

```
>> e=exp(1)

e =
```

```
   2.718281828459046
```

Coba Anda lakukan secara manual sebagai berikut:

```
> e100=(1+1/100)^100
e100 =
   2.704813829421529
```

Ini adalah aproksimasi nilai e untuk $n = 100$. Error mutlak dan error relatifnya diperoleh sebagai berikut.

```
>> err_mutlak = abs(e100-e),err_rel=err_mutlak/e
err_mutlak =
   0.013467999037517
err_rel =
   0.004954599959619
```

Kelihatannya hasil ini cukup baik. Kita lakukan eksperimen dengan memperbesar nilai n. Kita akan mengambil $n = 10^k$ di mana k berjalan dari 1, 2, . . . , 17. Agar efektif kita susun m-file seperti terdapat dalam buku teks hal 67. Kali ini kita tambahkan satu baris untuk menghitung kesalahan relatifnya.

```
m = 17;
ap = zeros(m,1);
err = zeros(m,1);
     for k = 1: m
          n(k) = 10^k;
          ap(k) = (1+1/n(k))^n(k);
          err(k)=abs(ap(k)-exp(1));
          err_rel(k)=err(k)/exp(1);
     end
[ap err]
```

Coba simpan m-file ini dengan nama **eksponen1.m.** Melalui command window, eksekusi m-file ini dengan mengetikan perintah berikut, lalu enter.

```
>>eksponen1
```

Pertanyaan untuk responsi

1. Berikan ulasan apa maksud masing-masing baris pada m-file di atas. Anda dapat menulis kembali m-file ini dengan memberikan nomor untuk tiap-tiap baris.
2. Bagaimana pola dua macam kesalahan yang Anda dapatkan, menurun atau meningkat?
3. Apakah hasil yang diperoleh dari simulasi ini sesuai dengan teori matematika yang ada khususnya konsep limit.
4. Kalau ada keganjilan dengan teori, kapan fenomena ini mulai terjadi?
5. Kalau diteruskan untuk n yang lebih besar lagi, apa kira-kira yang akan terjadi.

3. Aproksimasi nilai e melalui deret Taylor

Selain bentuk limit di atas, nilai e mempunyai representasi dalam bentuk deret takhingga berikut

$$e = \lim_{n\to\infty}\left(1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{n!}\right).$$

Anda dapat menghitung nilai aproksimasi ini langsung pada command window dengan perintah berikut, yaitu khusus untuk $n = 10$.

```
>> n=10;k=1:n;d=sum(1./factorial(k));e10=1+d

e10 =

   2.718281801146385
```

Perhatikan ini adalah cara smart untuk melakukan penjumlahan suku-suku deret dengan menggunakan operasi array. (lihat kembali Bab I). Perintah `n=10;` `k=1:10` akan menghasilkan array 1, 2, 3, . . . , 10 dan `d=sum(1./factorial(k));` akan menghasilkan jumlahan

$$\frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{10!}$$

Sehingga perintah `e10=1+d` akan menghasilkan aproksimasi nilai e sampai $n = 10$. Cara lain adalah dengan menggunakan metoda iteratif sepertim-file pada buku hal 68.

Pertanyaan untuk responsi

1. Dengan MATLAB, hitunglah nilai aproksimasi e dengan deret Taylor ini untuk n = 1 s.d. 20. Untuk masing-masing, hitunglah error multak dan error relatifnya. Anda dapat memodifikasi sedikit m-file pada hal 68.
2. Berapa dibutuhkan banyak suku n agar error mutlaknya kurang dari 10^{-16}. Kali ini m-file hal 68 perlu dimodifikasi agak banyak.
3. Bandingkan dengan aproksimasi dengan menggunakan limit sebelumnya. Formula mana yang lebih stabil terhadap rounding error. Berikan alasannya.

Penyelesaian untuk semua pertanyaan responsi harus dikerjakan dan **dikumpulkan 2 pekan** sejak tanggal diberikan. Keseriusan mengerjakan responsi mempunyai pengaruh besar terhadap kinerja Praktikum Anda. Tidak ada yang sulit asalkan mau “telaten” saja.

PENGEMBANGAN

Cari tahu bentuk representasi deret bilangan π; misalnya melalui buku kalkulus atau artikel pada internet. Coba aproksimasi nilainya seperti yang Anda lakukan pada bilangan alam e di atas. Hanya mahasiswa yang visioner saja yang akan tertarik dan penasaran dengan masalah pengembangan ini.