|  |
| --- |
| **HAMPIRAN DAN GALAT** |
| **Pertemuan : 1&2**  TUJUAN INSTRUKSIONAL KHUSUS :   1. Mengenali perbedaan solusi analitis dengan solusi numerik 2. Mengetahui penyebab terjadinya galat 3. Menghitung galat dari hasil perhitungan numerik 4. Mengetahui fungsional dasar dalam scilab 5. Menggunakan scipad untuk membuat subprogram 6. Membangun algoritma sederhana untuk menghitung nilai hampiran fungsi |

**Materi :**

* 1. **Pengertian Metode Numerik**

Untuk memahami suatu proses pada dunia real maka proses yang terjadi dapat direpresentasikan terlebih dahulu ke dalam bentuk persamaan matematika. Adapun bentuk persamaan matematika dapat berupa sistem persamaan linear maupun persamaan taklinear, persamaan integral dengan nilai batas yang diketahui, persamaan diferensial, dan lain-lain. Representasi dari proses ini disebut dengan model matematika. Jadi model matematika menggambarkan sistem dunia nyata ke dalam bentuk persamaan matematika.

Hal yang tidak kalah penting untuk diketahui adalah bagaimana menyelesaikan solusi dari model matematika tersebut agar dapat diketahui bagaimana perilaku dari sistem yang diamati. Solusi ini disebut dengan **solusi eksak**. Salah satu metode yang digunakan untuk menyelesaikan model matematika adalah menggunakan metode analitis. Beberapa contoh metode analitis yang dapat digunakan untuk menyelesaikan model adalah Teknik Integral Parsial untuk menghitung integral dari suatu fungsi, metode Eliminasi untuk menyelesaikan sistem persamaan linear, faktorisasi untuk mencari akar dari persamaan nonlinear. Adanya kesulitan dalam menggunakan metode analitis untuk menyelesaikan model matematika yang rumit mengakibatkan munculnya metode lain yang dapat mengaproksimasi solusi eksak dari model tersebut, metode inilah yang disebut dengan metode numerik.

**Definisi 1.1** Metode numerik adalah teknik-teknik yang digunakan untuk memformulasikan masalah matematis agar dapat diselesaikan dengan menggunakan operasi perhitungan.

Karena metode numerik melibatkan operasi perhitungan yang berulang-ulang maka dalam pengembangannya metode numerik diimplemetasikan dengan memanfaatkan teknologi komputer. Bidang yang mengkaji pengembangan metode yang efisien dan akurat untuk mengaproksimasi suatu nilai eksak yang sulit diperoleh secara analitis disebut komputasi saintifik. Komputasi saintifik melibatkan perangkat keras komputer, latar belakang teori berupa teori matematika, metode dan algoritma serta perangkat lunak untuk menghampiri/mengaproksimasi solusi eksak dari model yang diteliti.

Berikut ini adalah contoh sederhana yang dapat diselesaikan dengan menggunakan metode numerik.

1. Misalkan sebuah tabung diisi penuh air dengan tinggi tabung 7 cm dan kedalamnya dimasukkan sebuah bola sehingga air dari tabung tumpah sebanyak 10 cm3. Ingin diketahui berapa ukuran diameter bola yang harus dimasukkan. Permasalahan ini diformulasikan kedalam persamaan matematika menjadi







Untuk menyelesaikan persamaan ini dapat dilakukan dengan metode yang diberikan di sekolah namun solusi dari masalah ini dapat diaproksimasi secara numerik dengan menggunakan metode numerik.

1. Misalkan berikut ini adalah data dari jarak tempuh dan kecepatan sebuah mobil

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Waktu (detik) | 0 | 3 | 6 | 8 | 12 |
| Jarak (meter) | 0 | 40 | 85 | 130 | 210 |
| Kecepatan(m/det) | 0 | 30 | 45 | 35 | 20 |

Dari data yang dimiliki dapat ditentukan posisi mobil pada detik ke-10 dengan menggunakan aproksimasi

* 1. **Sumber Kesalahan Aproksimasi**

Didalam melakukan aproksimasi munculnya kesalahan atau galat tidak dapat dihindari. Berikut ini adalah beberapa penyebab terjadinya galat:

1. Galat yang terjadi sebelum komputasi

Contoh galat yang terjadi sebelum komputasi adalah: kesalahan dalam memodelkan masalah real kedalam bentuk persamaan matematika, keterbatasan alat ukur yang digunakan, mengambil data hasil komputasi sebelumnya yang juga merupakan hasil aproksimasi.

1. Galat yang terjadi selama komputasi

Galat yang terjadi selama komputasi dapat dibagi menjadi dua yaitu:

1. Galat yang terjadi akibat pembulatan (*rounded error*) representasi bilangan hasil perhitungan pada komputer telah melalui proses pembulatan akibatnya hasil komputasinya akan disertai kesalahan.
2. Galat yang terjadi akibat pemotongan (*truncation error*)

Galat ini muncul akibat penggunaan hampiran sebagai pengganti persamaan eksak. Contohnya adalah pada fungsi yang kontinu maka penggambaran fungsi dilakukan dengan mendiskritisasi dengan sejumlah yang berhingga titik.

* 1. **Aturan Pembulatan Bilangan Desimal**

Galat yang terjadi akibat pembulatan muncul dari dua metode yang sering dilakukan yaitu:

1. Pemotongan (*chopping*)
2. Pembulatan terdekat (*rounding*)

Untuk lebih jelasnya perhatikan contoh pembulatan bilangan desimal menjadi dua digit

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bilangan** | **Pemotongan** | **Pembulatan** | **Bilangan** | **Pemotongan** | **Pembulatan** |
| 2.749 | 2.7 | 2.7 | 2.849 | 2.8 | 2.8 |
| 2.750 | 2.7 | 2.8 | 2.850 | 2.8 | 2.8 |
| 2.751 | 2.7 | 2.8 | 2.851 | 2.8 | 2.9 |
| 2.799 | 2.7 | 2.8 | 2.899 | 2.8 | 2.9 |

**Amati tabel diatas kemudian simpulkan aturan pembulatan khususnya untuk pembulatan terdekat.**

Karena banyaknya iterasi dalam algortima dapat mencapai ribuan kali maka dapat terjadi akumulasi galat yang tidak dapat dikendalikan (*blow up*) walaupun pada iterasi pertama hanya terjadi galat dengan nilai yang kecil. Oleh karena itu perlunya untuk **mengecek akurasi** dan **stabilitas** dalam menggunakan algoritma menjadi hal yang penting untuk dianalisis sebelum melakukan aproksimasi secara numerik. Jika akurasi menunjukkan pada seberapa dekat hasil aproksimasi dengan hasil sebenarnya maka stabilitas menunjukkan pada seberapa rentannya perubahan hasil aproksimasi akibat perubahan sedikit dari variabel masukkannya.

Jika akurasi digunakan untuk menunjukkan seberapa dekat suatu nilai aproksimasi dengan nilai eksak, maka presisi digunakan untuk menunjukkan seberapa dekat nilai aproksimasi dengan nilai aproksimasi lainnya. Nilai yang tidak akurat disebut dengan istilah **bias** sedangkan nilai yang tidak presisi disebut dengan **ketidakpastian** (*uncertainty*).

* 1. **Galat**

Dari cara perhitungannya galat dapat dibagi menjadi dua yaitu :

1. **Galat Eksak**

Galat eksak dihitung dari selisih antara solusi eksak dengan solusi numerik, dituliskan dalam persamaan berikut:



Dengan adalah nilai aproksimasi dan adalah nilai eksaknya

1. **Galat Relatif**

Galat relatif terbagi menjadi dua, yaitu:

1. Galat relatif sejati adalah galat yang diperoleh dengan membandingkan terhadap 

 atau 

1. Galat relatif semu adalah galat yang diperoleh dari nilai aproksimasi saat ini dikurang nilai aproksimasi sebelumnya dan dibandingkan dengan nilai aproksimasi saat ini atau dituliskan menjadi persamaan berikut ini

 atau 

Dengan nilai aproksimasi saat ini dan nilai aproksimasi sebelumnya.

Galat relatif semu sering digunakan apabila solusi eksaknya tidak diketahui.

**Contoh 1.1**:

Manakah yang lebih signifikat 10 orang gagal ujian dari 100 peserta ujian atau 1 orang gagal dari 20 orang peserta ujian?

Untuk kasus 10 orang gagal maka  sedangkan

Untuk kasus 1 orang gagal maka 

Maka terjadinya kegagalan ujian lebih signifikat pada kasus 1 orang yang gagal dibandingkan dengan 10 orang yang gagal.

* 1. **Deret Taylor**

Telah dijelaskan bahwa galat pemotongan adalah galat yang muncul akibat penggunaan hampiran sebagai pengganti persamaan eksak. Salah satu fungsi yang digunakan untuk menghampiri nilai eksak adalah menggunakan deret Taylor. Setiap fungsi yang kontinu dapat diaproksimasi dengan sebuah fungsi polinomial dengan menggunakan deret Taylor

**Definisi 1.2.**

Andaikan *f* dan *f* ΄, *f* ΄΄,… kontinu pada selang [a,b]. Misalkan  ∈ [a,b] maka untuk  disekitar, ∈ [a,b] maka *f(x)* dapat diekspansi ke dalam deret Taylor menjadi



Jika  maka deret Taylor dituliskan kembali menjadi



Jika nilai maka deret Taylor ini disebut dengan deret Mc Laurin



**Latihan 1.1**

Hampiri fungsi  ke dalam deret Taylor disekitar .



Maka dengan deret Taylor diperoleh

****

Dengan .

Jika ingin diaproksimasi nilai maka diperoleh hasilnya pada tabel berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| **Suku ke-** | **Nilai hampiran** | **Galat Eksak** | **Galat Hampiran** |
| 1 | 1 |  |  |
| 2 | 1+1=2 |  |  |
| 3 | 1+1+0.5=2.5 |  |  |
| 4 |  |  |  |
| 5 |  |  |  |

Lengkapi tabel berikut ini

* 1. **Order Konvergensi**

Melakukan aproksimasi secara numerik dari solusi eksak prinsipnya sama dengan membangun sebuah barisan  yang konvergen ke suatu nilai . Nilai  dapat dipandang sebagai hasil perhitungan numerik disetiap iterasi sedangkansuatu nilai yang diharapkan adalah nilai solusi eksak sehingga dengan semakin bertambahnya  maka barisan akan mendekati nilai . Sesuai dengan teori barisan dalam kalkulus maka berlaku



Terdapat banyak kemungkinan barisan  yang bisa konvergen menuju . Perbedaannya terletak dari kecepatan konvergensi. Untuk mengukur kecepatan konvergensi digunakan orde kekonvergenan yang dinotasikan dengan *O* (*big* – O).

**Definisi 1.3**

Misalkan konvergen ke bilangan untuk  membesar. Jika terdapat konstanta positif dan sehingga

 untuk semua  besar

Maka dikatakan konvergen ke dengan orde kekonvergenan 

Relasi antara dan  adalah



Persamaan ini menyatakan bahwa  dengan kecepatan konvergensi 

Jika nilai semakin besar maka makin cepat barisan tersebut mencapai konvergensi.

**Contoh 1.2**

Diberikan dua barisan berikut

 dan 

Keduanya akan konvergen ke 0, tetapi barisan akan lebih cepat konvergen dari pada barisan , sehingga dengan definisi dapat dituliskan

 dan 

* 1. **Pengantar Scilab**

Scilab adalah perangkat lunak yang menyerupai Matlab yang berguna untuk melakukan komputasi numerik dan visualisasi data. Aplikasi dapat diunduh pada [www.scilab.org](http://www.scilab.org) dan digunakan pada OS Linux ataupun Windows.

Seperti halnya pada Matlab setelah masuk ke jendela kerja scilab maka perintah dapat dituliskan setelah tanda -->. Untuk lebih jelasnya diberikan pada contoh 1.3

**Contoh 1.3**

-->p=2

p =

2.

-->l=3

l =

3.

-->luas=p\*l

luas =

6.

Dalam jendela kerja scilab dapat didefinisikan fungsi dengan menggunakan perintah **deff.** Penggunaan perintah ini diberikan pada contoh 1.4

**Contoh 1.4**

-->deff('L=luas(p,l)','L=p\*l') //**deff**(‘(output1,output2,...)=**namamodul**(input1,input2,...),’**persamaan**’)

-->luaspersegipanjang=luas(2,3)

luaspersegipanjang =

6.

Selama aplikasi keluar atau perintah clear belum dijalankan maka modul dengan nama luas masih dapat dipanggil untuk digunakan untuk nilai inputan yang berbeda-beda.

Scilab juga menyediakan media untuk menuliskan sekumpulan perintah yang dapat disimpan dan dipanggilan kapanpun yaitu SciNotes. Jendela editor (SciNotes) dibuka dengan memilih dari jendela Editor pada **menubar** atau dengan menuliskan scinote() pada jendela kerja scilab.

Dengan menggunakan editor scilab maka perhitungan luas persegipanjang dapat dituliskan dalam bentuk fungsi yang disimpan dengan nama file .sci.

**Contoh 1.5**

function L=luaspp(p,l);

L = p\*l;

endfunction

Pada contoh 1.5 nama fungsi adalah luaspp dan file akan otomatis disimpan sesuai dengan nama fungsinya yaitu luaspp.sci.

Untuk mengeksekusi fungsi digunakan 3 cara yaitu:

1. Pada menu bar jendela editor pilih **execute** lalu **Load into scilab**
2. Pada menu bar jendela kerja pilih **execute** lalu pilih **file yang akan dieksekusi**
3. Dengan mengetikkan secara langsung di jendela kerja exec(‘d:\metnum scilab\luaspp.sci’)artinya memanggil fungsi luaspp.sci yang berada di direktori d yang ada dalam folder bernama metnum scilab.

-->exec('d:\Metnum Scilab\luaspp.sci')

-->function L=luaspp(p,l);

--> L=p.\*l;

-->endfunction

Selanjutnya fungsi yang sudah dipanggil dapat digunakan dengan cara berikut:

-->Luas=luaspp(2,3)

Luas =

6.

Seperti halnya pada bahasa pemrograman yang lain modul atau fungsi dapat digunakan/dipanggil dalam fungsi yang lain. Dalam scilab digunakan fungsi exec(‘lokasi dan nama file’)

function V=volkotak(panjang,lebar,tinggi);

exec('d:/metnum scilab/luaspp.sci');

V=luaspp(panjang,lebar)\*tinggi;

endfunction

Beberapa variabel yang telah didefinisikan pada Scilab

|  |  |
| --- | --- |
| **Konstanta Spesial pada Scilab** | **Nilai/Keterangan** |
| %pi |  |
| %i |  |
| %e |  |
| %t dan %f | True dan false dalam konstanta boolean |

Untuk mengetahui variabel yang tersedia di Scilab gunakan perintah who.

Berikut ini adalah operator aritmatika dan operator pembanding yang di gunakan di Scilab

|  |  |  |  |
| --- | --- | --- | --- |
| **Simbol** | **Keterangan** | **Simbol** | **Keterangan** |
| + | Penjumlahan | == | Sama dengan |
| - | Pengurangan | < | Kurang dari |
| \* | Perkalian | > | Lebih dari |
| / | Pembagian | >= | Lebih dari atau sama dengan |
| \ | Pembagian kiri | <= | Kurang dari atau sama dengan |
| ^ | Pangkat | <> , ~= | Tidak sama dengan |
| ‘ | Transpose |  |  |

Untuk operator logika dalam Scilab digunakan berikut ini

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Operator** | **Keterangan** |
| & | And | Konjungsi |
| | | Or | Disjungsi |
| ~ | Nor | Ingkaran |

Penggunaan kondisi dan perulangan dalam Scilab

|  |  |
| --- | --- |
| **Perintah** | **Implementasi** |
| Penggunaan if | if komparasi (P1) then  perintah untuk (P1) benar  elseif komparasi (P2) then  perintah untuk (P2) benar  else  perintah  end |
| Penggunaan case of | select nama()  case nama1 then  Perintah\_1  case nama2 then  Perintah\_2  ....  else  Perintah\_n  end |
| Penggunaan for | inisiasi  for variabel\_penghitung = 1:langkah:n  perintah yang dikerjakan  end |
| Penggunaan While | inisialisasi  while kondisi  perintah  end |

**Latihan 1.2**

Buatlah program sederhana untuk menghitung penjumlahan dan perkalian rekursif berikut ini

1.  b. 
   1. **Visualisasi Grafik pada Scilab**

Seperti pada Matlab perintah untuk memvisualisasikan data dengan menggunakan perintah

|  |  |
| --- | --- |
| **Perintah** | **Keterangan** |
| plot2d | Menampilkan grafik 2 dimensi |
| Xtitle | Menampilkan nama dari setiap sumbu |
| suplot (m,n,p) | Menampilkan beberapa grafik dalam satu tampilan dengan m = baris, n = kolom dan p = urutan gambar |

Variasi tampilan grafik ditampilkan pada tabel berikut ini

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Perintah** | **Tampilan** | **Perintah** | **Tampilan** | **Perintah** | **Tampilan** |
| 0 | - | -5 | Oval | 1 | Hitam |
| -1 | + | -6 |  | 2 | Biru |
| -2 | X | -7 |  | 3 | Hijau |
| -3 | \* | -8 |  | 5 | Merah |
| -4 | Wajik | -9 |  | 6 | pink |

**Contoh 1.7**

|  |
| --- |
| //Program visualisasi dengan subplot  x=[0:0.1:2\*%pi]';  y1=sin(x);  y2=cos(x);  subplot(1,2,1)  plot2d(x,y1)  xtitle('Grafik Fungsi Sinus','x','y1')  subplot(1,2,2)  plot2d(x,y2)  xtitle('Grafik Fungsi Cosinus','x','y2') |



**Latihan 1.3**

1. Buatlah program untuk menghitung nilai aproksimasi e dengan menggunakan deret maclaurin untuk nilai x = -1 jika diberikan nilai e = 0.3678794414 (gunakan %e) buatlah tabel aproksimasinya sampai kesalahan relatifnya kurang dari 0.00000001, tampilkan pula perhitungan galat eksak, galat relatif dan galat relatif hampiran
2. Diketahui bahwa persamaan kuadrat adalah mempunyai akar yang diperoleh dari  atau dapat dihitung dengan persamaan . Jika dan Hitunglah perbandingan hasil perhitungan tersebut dengan nilai eksaknya dan 