P E N D A H U L U A N

Q-taro, Prototipe robot bola yang bereaksi atas panas, cahaya, suara, & musik dengan mengeluarkan bunyi dan permainan cahaya. Perpaduan aspek hardware & software untuk merespon variabel fisik dengan sangat akurat & presisi. Sony, Sang Perusahaan Penemu, yang memiliki motto 'Be Creative', menyebut Q-taro sebagai 'Healing Creature.'
Foto: B. Müller; dpa CHIP,2002

Metode numerik secara harafiah berarti suatu cara berhitung dengan menggunakan angka-angka, sedangkan secara istilah metode numerik adalah teknik yang digunakan untuk memformulasikan persoalan matematik sehingga dapat diselesaikan dengan operasi aritmatika biasa.

Perjalanan seribu mil dimulai dari satu langkah
(pepatah)

1.1. Pemodelan Sistem Fisis: Analitik dan Numerik

Persoalan yang melibatkan model matematika banyak muncul dalam berbagai disiplin ilmu pengetahuan, seperti dalam bidang Fisika, Kimia, Ekonomi, atau pada persoalan rekayasa, seperti Teknik Sipil, Teknik Mesin, Elektro, dan sebagainya. Dalam hal ini, pemodelan pada sistem fisis bisa didefinisikan sebagai sebuah formula atau persamaan yang menyatakan perilaku dari suatu fenomena fisika dalam bentuk pernyataan matematika. Pernyataan ini secara aktual bisa meliputi bentuk hubungan aljabar sederhana sampai persamaan diferensial yang begitu rumit.

Hal yang harus dikerjakan antara lain dengan mencermati lingkup persoalan, melakukan pemilahan atas variabel serta parameter yang primer dan sekunder, serta menetapkan suatu model, yang dinilai cukup sederhana untuk analisis selanjutnya, tetapi sekaligus cukup realistis untuk menggambarkan keadaan dalam realitas. "*Great engineering is simple engineering*".

Tiga aspek yang harus diperhatikan dalam pemodelan sistem suatu fenomena fisika adalah :

- Hukum-hukum alam yang berlaku
- Informasi serta pengalaman di lapangan
- Sasaran akhir yang ingin dicapai

Mengingat hukum-hukum alam umumnya diungkapkan dalam pernyataan yang bersifat pasti serta tidak mengandung keragu-raguan akan sebab dan akibatnya, maka model yang diperoleh bersifat *deterministik*. Model ini sering berupa pernyataan matematika yang dijabarkan dari azas-azas kekekalan energi, massa dan momentum.

Sebaliknya, ketidaklengkapan informasi mengenai aspek-aspek tertentu dari realitas, atau tidak tersedianya rumusan yang memadai untuk menyatakan hukum alam yang berlaku sering mendorong pembentukan model yang bersifat *non-deterministik*. Model jenis ini sering dikembangkan dengan menggunakan konsep peluang atau probabilitas, namun tidak tertutup kemungkinan model itu semata-mata bersifat *heuristik* atau *ad-hoc*.

Seringkali model matematika tersebut muncul dalam bentuk yang tidak ideal alias rumit. Model matematika yang rumit ini adakalanya tidak dapat diselesaikan dengan metode analitik yang sudah umum untuk mendapatkan solusi sejatinya (*exact solution*). Yang dimaksud dengan metode analitik adalah metode penyelesaian model matematika dengan rumus-rumus aljabar yang sudah baku (lazim).

Sebagai contoh ilustrasi, kita tinjau beberapa kasus dalam model matematik di bawah ini, yang kemudian membuat anda mungkin menyerah, atau mungkin mengatakan bahwa soal-soal tersebut tidak dapat diselesaikan dengan metode analitik yang biasa anda kenal. Bagaimana cara menyelesaikannya?

➯ *Bisakah anda menentukan akar-akar persamaan polinom ini?*

$$23.4x^7 - 1.25x^6 + 120x^4 + 15x^3 - 120x^2 - x + 100 = 0 \qquad (1.1)$$

→ *Komentar:*

Biasanya untuk polinom derajat 2 orang masih dapat mencari akar-akar polinom dengan rumus *abc* yang terkenal itu, yaitu:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad (1.2)$$

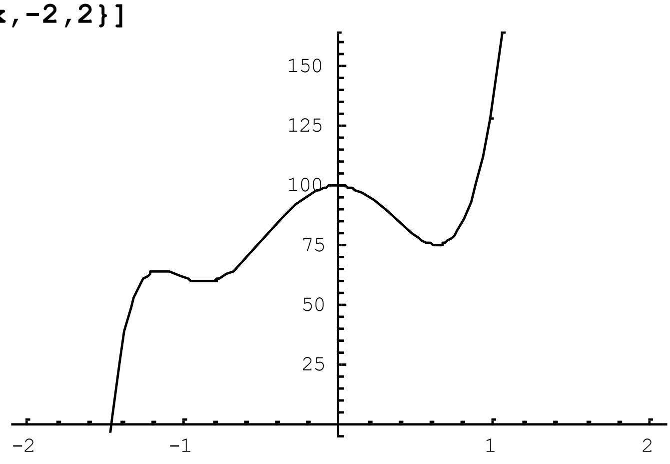
namun, untuk polinom derajat > 2, seperti pada soal (i), tidak terdapat rumus aljabar untuk menghitung akar polinom. Yang mungkin kita lakukan adalah dengan memanipulasi polinom, misalnya dengan memfaktorkan (atau menguraikan) polinom tersebut menjadi perkalian beberapa suku. Semakin tinggi derajat polinom, jelas semakin sukar memfaktorkannya. Ada juga beberapa alternatif lain.

Yang pertama dengan cara coba-coba seperti *metode pembagian sintetis Horner*. Dengan metode ini, polinom dibagi dengan sebuah bilangan. Jika sisa pembagiannya nol, maka bilangan tersebut adalah akar polinom.

Cara kedua adalah secara grafik, yaitu dengan merajah kurva fungsi di atas kertas grafik, kemudian berdasarkan gambar kurva, kita mengambil tarikan akar secara kasar, yaitu titik poyong kurva dengan sumbu-*x*. Cara ini, selain kaku dan tidak praktis, ketelitian akar yang diperoleh sangat bergantung pada ketelitian penggambaran kurva. Lagipula, merajah kurva pada kertas grafik hanya terbatas pada fungsi yang dapat digambarkan pada bidang dua dimensi atau tiga dimensi.

Untuk fungsi dengan variabel lebih besar dari 3 jelas tidak dapat (malah tidak mungkin) kita gambar kurvanya.

```
Plot[{23.4*x^7 -1.25*x^6+120*x^4 +15*x^3 - 120*x^2 -
x+100},{x,-2,2}]
```



□Graphics□

Tentukan harga x yang memenuhi persamaan:

$$\sqrt{27.8e^{5x} - \frac{1}{x}} = \cos^{-1} \frac{(120x^2 + \sqrt{2x}}{17x - 65} \tag{1.3}$$

→ *komentar:*
Soal kedua ini masih sejenis dengan soal pertama, yaitu menentukan nilai x yang memenuhi kedua persamaan.

Selesaikan sistem persamaaan linier (linear):

$$\begin{array}{llllllll}
1.2a & -3b & -12c & +12d & +4.8e & -5.5f & +100g & =18 \\
0.9a & +3b & -c & +16d & +8e & -5f & -10g & =17 \\
4.6a & +3b & -6c & -2d & +4e & +6.5f & -13g & =19 \\
3.7a & -3b & +8c & -7d & +14e & 8.4f & +16g & =6 \\
2.2a & +3b & +17c & +6d & +12e & -7.5f & +18g & =9 \\
5.9a & +3b & +11c & +9d & -5e & -25f & -10g & =0 \\
1.6a & +3b & +1.8c & +12d & =7e & 2.5f & +g & =-5
\end{array} \tag{1.4}$$

→*komentar:*
Tidak ada rumus yang baku untuk menemukan solusi sistem persamaan linier. Apabila sistem persamaannya hanya berupa dua garis lurus dengan dua variabel, kita masih dapat menemukan solusinya (dalam hal ini titik potong kedua garis) dengan menggunakan rumus titik potong dua buah garis atau dengan aturan Cramer.
Kita juga dapat menemukan titik potong tersebut dengan menggambar kedua garis pada kertas grafik. Untuk sistem yang terdiri dari tiga buah persamaan linier dengan tiga variabel, aturan Cramer masih dapat digunakan untuk memecahkan sistem. Tetapi untuk sistem dengan jumlah persamaan dan jumlah variabel lebih besar dari tiga, tidak ada rumus yang dapat dipakai untuk memecahkannya.

Tentukan nilai maksimum fungsi tiga dimensi:

$$f(x,y) = \cos\frac{x-\sqrt{\sin(x)}+3}{4+(xy)^2} + \sin(3xy-1) - \tan(\frac{x(0.08+\cos(x))}{y}) \quad (1.5)$$

→*komentar:*
Relatif sukar mencari titik optimum fungsi yang memiliki banyak variabel. Untuk menentukan titik optimum (titik ekstrim fungsi), pertamatama orang harus menentukan turunan fungsi, menjadikan ruas kanannya sama dengan nol, lalu memeriksa jenis titik ekstrimnya. Bila fungsinya cukup rumit dan disusun oleh banyak variabel, menghitung turunan fungsi menjadi pekerjaan yang sukar atau bahkan tidak mungkin dilakukan.

➯ *Hitung nilai integral-tentu berikut:*

$$\int_{1.2}^{2.5} (\sqrt{(45.3e^{7x} + \frac{100}{x})^4} + \frac{4}{(x^2+1)})dx \quad (1.6)$$

→*komentar:*
Pada soal ini, tidak ada teknik integrasi yang dapat digunakan untuk fungsi yang bentuknya rumit itu.

➯ *Diberikan persamaan differensial biasa (PDB) dengan nilai awal:*

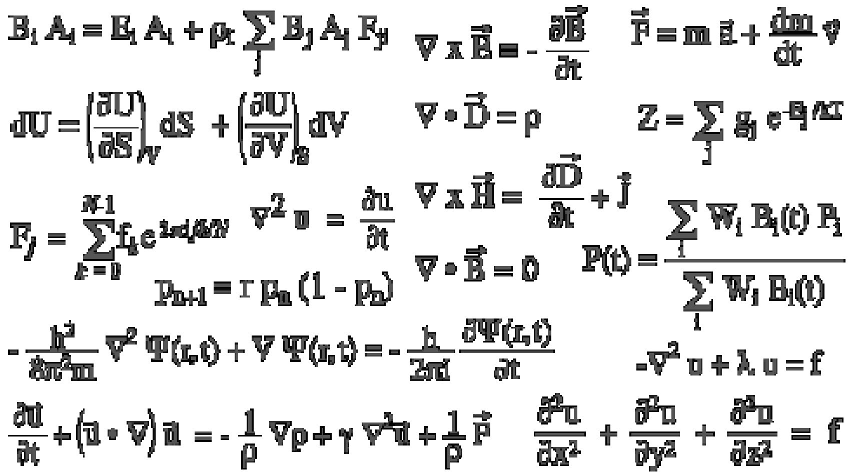
$$150y'' + 2y't = \frac{\sqrt{\ln(21t+40)y}}{t^2} + 120 \qquad ; y'(0) = 0, y(0) = 1.2 \qquad (1.7)$$

hitung nilai y pada t=1.8 !

→*komentar:*
Begitu juga pada soal ini, tidak terdapat metode persamaan diferensial untuk menyelesaikannya. Dengan kata lain, persoalan (1.6) dan (1.7) tidak mempunyai solusi analitik.

1.2. Model Fisika & Ruang Lingkupnya

Sebelum kita kaji contoh kasus fisika, kita tengok beberapa formulasi fisika yang model matematisnya termasuk paling menantang sepanjang masa, seperti pada ilustrasi gambar 1.2 (ilustrasi dari Ridwan & Fauzi, ITB).



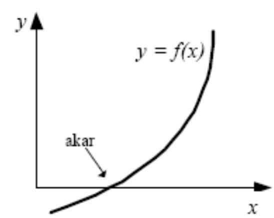
Gambar 1.2. Formulasi Paling Menantang sepanjang Masa

Didalamnya terdapat Persamaan Newton, Persamaan Schrodinger, Persamaan Navier-Stokes, Persamaan Poisson, Laju Aliran Kalor, Persamaan Helmholtz, Transformasi Fourier, Persamaan Maxwell, Fungsi Partisi, Dinamika Populasi, Hukum Termodinamika, Radioaktivitas Nuklir, dan Hukum Ekonomi.

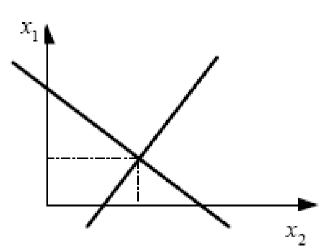
Terlihat bahwa model matematisnya terbentuk tidak sederhana secara analitik, dan solusinya terkait dengan bentuk-bentuk persamaan diferensial biasa berbagai orde dengan keterkaitan berbagai keadaan awal dan keadaan batas, persamaan diferensial parsial, integrasi, persamaan linear simultan atau matriks, atau finding roots, dll.

Ada empat pokok bahasan yang ditulis di dalam buku ini, untuk lebih memahami perilaku model kasus fisisnya dan karakteristik metode pendekatan komputasinya, dan berikut ringkasannya:

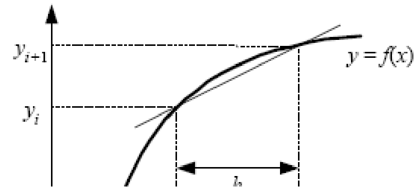
1. Solusi Persamaan Non Linear.
   → Selesaikan $f(x) = 0$ untuk x,



2. Matriks & Solusi Persamaan Linear.
   → Selesaikan sistem persamaan linear
   $a_{11}x_1 + a_{12}x_2 = c_1$
   $a_{21}x_1 + a_{22}x_2 = c_2$
   untuk harga-harga x_1 dan x_2.

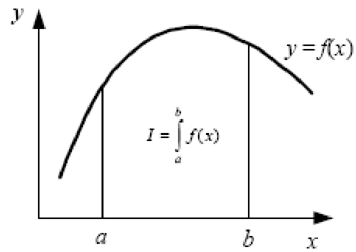


3. Diferensiasi & Integrasi numerik.
→Diberikan titik (x_i, y_i) dan titik (x_{i+1}, y_{i+1}). Tentukan $f\,'(x_i)$.



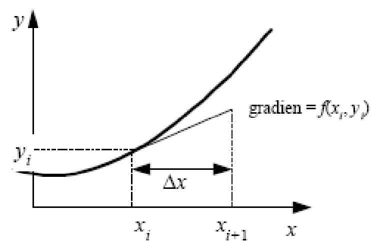
→Hitung integral berikut

$$I = \int_a^b f(x)\,dx$$



4. Solusi Persamaan Diferensial Biasa dengan nilai awal.

Diberikan $dy/dx = f(x,y)$ dan
dengan nilai awal $y_0 = y(x_0)$
Tentukan nilai $y(x_t)$ untuk $x_t \in R$



Bagaimana contoh kasus fisika-nya?

Berikut ini adalah satu contoh kasus fisis yang akan kita selesaikan dengan metode analitik dan metode metode numerik (Chapra dkk,1998). Kita tinjau Hukum Newton II tentang gerak, dalam model atau pernyataan matematika sederhana dinyatakan

$$F = ma \tag{1.8}$$

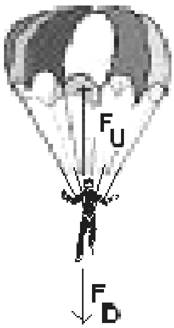
Atau dituliskan menjadi

$$a = \frac{F}{m} \tag{1.9}$$

Dimana F = besar gaya yang bekerja pada benda (N, atau kg m/dt^2), m=massa benda (kg) dan a = percepatan (m/dt^2). Karena bentuknya aljabar sederhana, solusi persamaan (1.8) bisa didapat dengan mudah.

Model yang lebih kompleks bisa ditinjau saat menentukan kecepatan akhir benda jatuh bebas dekat permukaan bumi. Contoh adalah penerjun payung yang menggunakan parasut. Model kasus ini diturunkan dari pernyataan percepatan sebagai perubahan kecepatan pada selang waktu (*dv/dt*)

$$\frac{dv}{dt} = \frac{F}{m} \qquad (1.10)$$



Gaya (*F*) terdiri dari dua gaya berlawanan, yaitu yang cenderung kebawah karena tarikan gravitasi F_D dan gaya tarik keatas oleh resistansi udara F_U :

$$F = F_D + F_U \qquad (1.11)$$

Jika gaya kebawah ditandai positif, maka $F_D = mg$, dimana *g*=konstanta gravitasi dengan nilai pendekatan 9,8 m/dt^2. resistansi udara bisa diformulasikan dengan bermacam variasi. Pendekatan sederhana adalah diasumsikan bahwa gaya ini berbanding linier dengan kecepatan, dan arahnya keatas, sehingga $F_U = -cv$, dimana *c*= konstanta pembanding disebut koefisien tarik (kg/dt).

Dengan subtitusi (1.11) dan formulasi F_D dan F_U, persamaan (1.10) berubah menjadi

$$\frac{dv}{dt} = \frac{mg - cv}{m} \qquad (1.12)$$

Atau disederhanakan menjadi

$$\frac{dv}{dt} = g - \frac{c}{m}v \qquad (1.13)$$

Persamaan (1.13) adalah persamaan diferensial, dimana solusi eksak untuk kecepatan jatuh penerjun tidak bisa ditentukan dengan manipulasi aljabar sederhana. Selebihnya, teknik yang lebih lanjut pada kalkulus harus digunakan untuk menentukan solusi analitik atau eksaknya.

Jika keadaan awal penerjun diam (*v=0* pada *t=0*) persamaan (1.13) menjadi

$$v(t) = \frac{gm}{c}\left(1 - e^{-(c/m)t}\right) \qquad (1.14)$$

Persamaan (1.14) dinamakan solusi analitik atau eksak karena secara eksak memenuhi persamaan diferensial biasa.

Solusi metode numerik untuk kasus ini bisa diperoleh dengan memformulasikan kembali bentuk (*dv/dt*) pada persamaan (1.13) dengan catatan bahwa $dv/dt \cong \Delta v/\Delta t$ adalah pendekatan karena Δt terbatas, sebagai berikut :

$$\frac{dv}{dt} \cong \frac{\Delta v}{\Delta t} = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} \qquad (1.15)$$

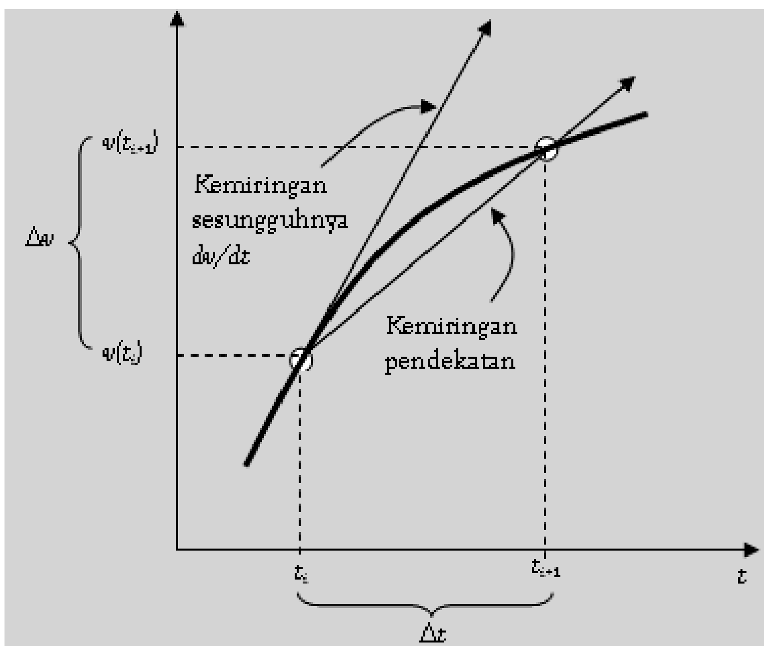
Dimana Δv dan Δt= selisih dalam kecepatan dan waktu dihitung pada interval terbatas. $v(t_i)$=kecepatan pada waktu awal t_i, dan $v(t_i+1)$=kecepatan setelah sekian waktu t_i+1. Asumsi diatas dapat diperjelas dengan ilustrasi pada gambar 1.3.

Persamaan (1.15) disubtitusikan ke (1.13) akan memberikan :

$$\frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} = g - \frac{c}{m} v(t_i) \qquad (1.16)$$

Dan bisa diubah menjadi

$$v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i) \right] (t_{i+1} - t_i) \qquad (1.17)$$



Gambar 1.3 Penggunaan selisih hingga untuk pendekatan turunan pertama v terhadap t

Banyak model matematika dari fenomena fisika yang lebih kompleks, yang tidak bisa diselesaikan secara analitik atau eksak, sehingga memerlukan teknik fisika matematika dari sekedar solusi aljabar sederhana. Dalam banyak kasus, alternatif satu-satunya adalah mengembangkan solusi numerik.

Contoh 1a

Seorang penerjun payung bermassa 68,1 kg meloncat dari balon udara yang diam. Tentukan kecepatan sebelum meluncur di tempat pendaratan dengan metode analitik dan metode numerik ! (Koefisien tarik= 12,5 kg/dt. Lebar langkah komputasi 2 detik)

<u>Solusi Analitik</u>

Masukkan parameter-parameter ke persamaan (1.14)

$$v(t) = \frac{9{,}8(68{,}1)}{12{,}5}(1 - e^{-(12{,}5/68{,}1)t}) = 53{,}39(1 - e^{-0{,}18355\,.t})$$

dan kecepatan *v(t)* bisa dihitung.

<u>Solusi Numerik</u>

Pada perhitungan awal ($t_i=0$), kecepatan penerjun adalah nol. Dengan memasukkan nilai parameter ke persamaan (1.10) bisa dihitung kecepatan pada t_{i+1}=2 detik :

$$v = 0 + \left[9{,}8 - \frac{12{,}5}{68{,}1}(0)\right]2 = 19{,}60 m/dt$$

Untuk interval berikutnya (dari t=2 ke 4 s), perhitungan diulang didapatkan:

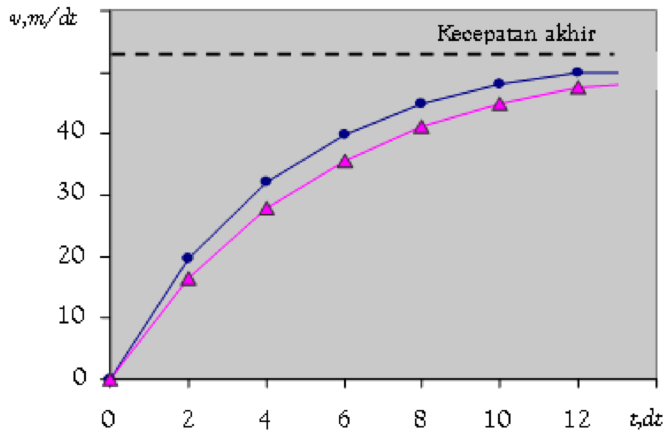
$$v = 19{,}60 + \left[9{,}8 - \frac{12{,}5}{68{,}1}(19{,}60)\right]2 = 32{,}00 m/dt$$

Perhitungan berlanjut untuk interval-interval berikutnya. Perbandingan solusi analitik dan numerik disajikan dalam tabel dan diilustrasikan pada gambar 1.4.

Solusi numerik yang diplot berdekatan dengan solusi analitik atau eksak, walaupun ada ketidakcocokan. Hal ini menunjukkan bahwa metode numerik menangkap fitur dasar dari solusi eksak. Satu cara untuk meminimalkan ketidakcocokan adalah menggunakan lebar langkah yang cukup kecil.

Jika diaplikasikan persamaan (1.17) pada interval 1 detik akan menghasilkan error yang lebih kecil, sehingga lintasannya berada pada solusi yang benar. Penggunaan lebar langkah yang semakin kecil pada solusi numerik menggunakan perhitungan tangan tentu tidak praktis, tetapi dengan memanfaatkan komputer, perhitungan bilangan yang besarpun bisa diselesaikan dengan mudah.

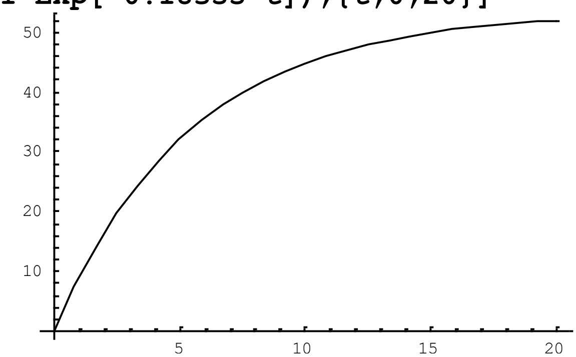
t,dt	V,m/dt	
	analitik	*numerik*
	0,00	0,00
2	16,40	19,60
4	27,77	32,00
6	35,64	39,85
8	41,10	44,82
10	44,87	47,97
12	47,49	49,96
∞	53,39	53,39



Gambar 1.4 Perbandingan solusi analitik dan numerik untuk kasus penerjun payung

Dari kasus sederhana ini, sebuah model dari kasus penerjun payung bisa dibuat secara akurat tanpa menyelesaikan persamaan diferensial secara eksak.

```
Plot[53.39(1-Exp[-0.18355*t]),{t,0,20}]
```



-Graphics-

1.3. Kedudukan Analitik & Metode numerik di bidang Terapan?

Di bidang rekayasa pada sektor industri, akademis dari berbagai bidang seperti teknik, sains murni, dan aplikasi-aplikasi lainnya jelas memiliki kebutuhan untuk menemukan solusi persoalan secara praktis. Bagi insinyur atau saintis terapan yang terjun didalamnya tentunya harus memiliki banyak cara penyelesaian persoalan model matematik yang dirasa terlalu sulit atau dalam bentuk yang

kurang konkrit. Satu sisi, penyelesaian analitik yang sering diberikan oleh matematika analitik kurang berguna. Solusi hampiran biasanya sudah memenuhi persyaratan rekayasa dan dapat diterima sebagai solusi. Lebih sering metode analitik hanya dijadikan jaminan keberadaan (atau hanya mengkarakterisasikan beberapa properti umum) solusi, tetapi tidak memberikan cara menemukan solusi tersebut.

Persoalan rekayasa dalam prakteknya tidak selalu membutuhkan solusi dalam bentuk fungsi matematika simbolik atau solusi kontinu. Seringkali solusi yang diinginkan malah dalam bentuk numerik, misalnya persoalan integral tentu dan persamaan diferensial.

Contoh 1b.

Sebuah bola logam dipanaskan sampai pada suhu 100°C. Kemudian, pada saat t = 0, bola itu dimasukkan ke dalam air yang bersuhu 30°C. Setelah 3 menit, suhu bola berkurang menjadi 70°C. Tentukan suhu bola setelah 22.78 menit menit. Diketahui tetapan pendinginan bola logam itu adalah 0.1865 (Sartono,2005).

Solusi:

Dengan menggunakan hukum pendinginan Newton, laju pendinginan bola setiap detiknya adalah $dT/dt = -k(T - 30)$ yang dalam hal ini k adalah tetapan pendinginan bola logam yang harganya 0.1865. Bagi matematikawan, untuk menentukan suhu bola pada t = 22.78 menit, persamaan diferensial tersebut harus diselesaikan terlebih dahulu agar suhu T sebagai fungsi dari waktu t ditemukan. Persamaan diferensial ini dapat diselesaikan dengan metode kalkulus diferensial.
Solusi umumnya adalah

$$T(t) = ce^{-kt} + 30$$

Nilai awal yang diberikan adalah $T(0)=100$. Dengan menggunakan nilai awal ini, solusi khusus persamaan diferensial adalah

$$T(t) = 70e^{-0.1865\,t} + 30$$

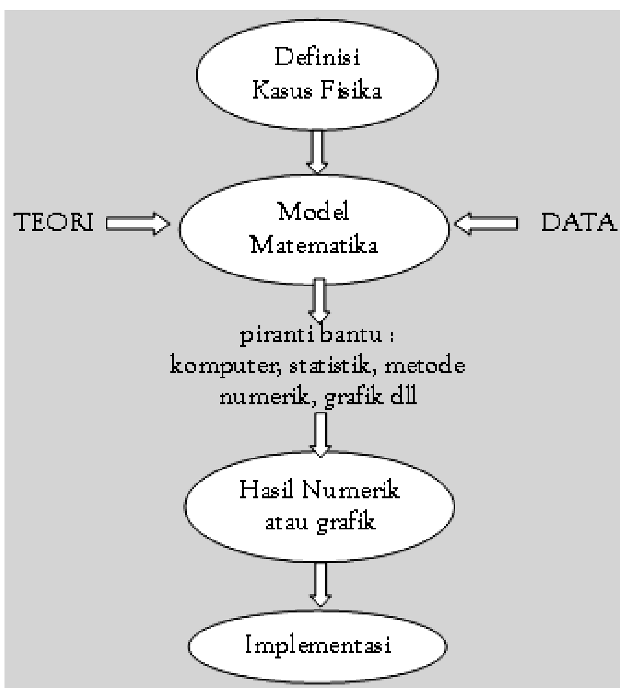
Dengan menyulihkan t = 22.78 ke dalam persamaan T, diperoleh
$T(22.78) = 70e^{-0.1865 \times 22.78} + 30 = 31°C$.
Jadi, suhu bola setelah 22.78 menit adalah 31°C.

Bagi saintis terapan, solusi persamaan diferensial yang berbentuk fungsi menerus ini tidak terlalu penting (bahkan beberapa persamaan diferensial tidak dapat dicari solusi khususnya karena memang tidak ada teknik yang baku untuk menyelesaikannya). Dalam praktek di lapangan, seringkali saintis terapan hanya ingin mengetahui berapa suhu bola logam setelah t tertentu misalnya setelah 30 menit tanpa perlu mencari solusi khususnya dalam bentuk fungsi terlebih dahulu. Fokusnya, cukup memodelkan sistem ke dalam persamaan diferensial, lalu solusi untuk *t* tertentu dicari secara numerik.

Pada aplikasi sistem kontrol industri dan laboratorium, bagian mendasar dari perhitungan rekayasa yang dilakukan adalah perhitungan "waktu nyata" (*real time computing*), yaitu perhitungan keluaran (hasil) dari data yang diberikan dilakukan secara simultan dengan *event* pembangkitan data tersebut, sebagaimana yang dibutuhkan dalam proses kontrol seperti sistem kontrol proses kimia atau reaksi nuklir, memandu pesawat udara atau roket dan sebagainya.

Karena itu, kecepatan perhitungan dan kebutuhan memori komputer adalah pertimbangan yang sangat penting. Jelaslah bahwa kecepatan tinggi, keandalan, dan fleksibilitas komputer memberikan akses untuk penyelesaian masalah terapan.

Sebagai contoh, solusi sistem persamaan linear yang besar menjadi lebih mudah dan lebih cepat diselesaikan dengan komputer. Perkembangan yang cepat dalam metode numerik antara lain ialah penemuan metode baru, modifikasi metode yang sudah ada agar lebih mangkus, analisis teoritis dan praktis algoritma untuk proses perhitungan baku, pengkajian galat, dan penghilangan jebakan yang ada pada metode. Gambar 1.5 menunjukkan proses penyelesaian kasus fisis dengan pendekatan metode numerik.



Gambar 1.5 Proses penyelesaian masalah fisis dengan metode numerik

1.4. Sumber Utama Kesalahan

Analisis kesalahan di dalam hasil metode numerik adalah dasar perhitungan yang bijak, baik dilakukan secara manual ataupun dengan komputer. Harga masukan jarang yang mempunyai harga pasti (eksak), karena seringkali didasarkan pada percobaan atau taksiran, dan proses metode numerik itu sendiri mempunyai berbagai macam kesalahan (dalam beberapa referensi berbeda digunakan istilah *error*, galat).

Umumnya didefinisikan dua macam cara untuk menyatakan kesalahan suatu pengukuran atau perhitungan, yaitu kesalahan absolut (*absolute error*) dan kesalahan relatif (*relative error*).

Kesalahan absolut[‡]adalah *nilai sebenarnya* dikurangi *nilai pendekatan*, dinyatakan dengan:

$$e_x = x - \bar{x} \tag{1.18}$$

dan *kesalahan relatif* adalah *kesalahan absolut* dibagi dengan *nilai sebenarnya*:

$$e_r = \frac{e_x}{x} = \frac{x - \bar{x}}{x} \tag{1.19}$$

Contoh 1c

Tinjau suatu pengukuran panjang jembatan dan panjang paku keling masing-masing didapatkan 9999 dan 9 cm. Jika nilai sebenarnya masing-masing 10.000 dan 10 cm, hitung (*a*) kesalahan absolut (*b*) kesalahan relatif

Solusi

Kesalahan absolut:

$$e_{x-jembatan} = 10.000 - 9999 = 1 \text{ cm}$$
$$e_{x-paku} = 10 - 9 = 1 \text{ cm}$$

Kesalahan relatif:

$$e_{r-jembatan} = \frac{1}{10.000} = 0{,}0001 = 0{,}01\%$$
$$e_{x-paku} = \frac{1}{10} = 0{,}1 = 10\%$$

kedua pengukuran memberikan kesalahan 1 cm, tetapi kesalahan relatif untuk paku jauh lebih besar. Sebagai perbandingan, jika nilai sebenarnya 0,00006 dan nilai pendekatannya 0,00005, kesalahan absolutnya hanya 10^{-5}, tapi kesalahan relatifnya

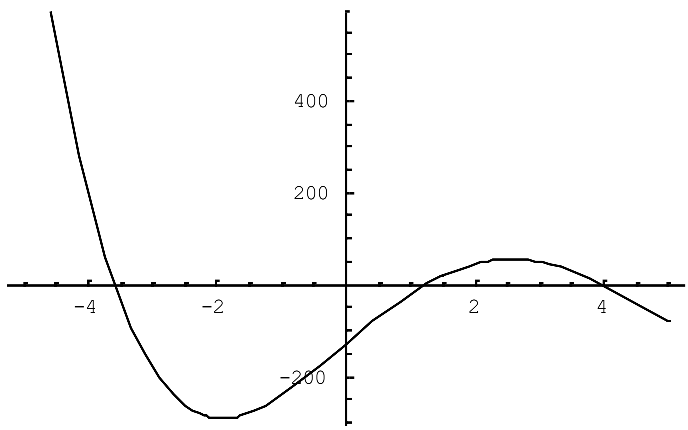
---

[‡] Beberapa teks lain mungkin mendefinisikan $e_x = \bar{x} - x$, formula bebas mana yang dipilih, asal konsisten.

0,2 atau 20%. Di lain pihak, bila nilai sebenarnya 100,500 dan nilai pendekatan 100,000, kesalahan absolut 500 tetapi kesalahan relatifnya hanya 0,005 atau 0,5%.

Kalau diperhatikan disini e_x dan e_r adalah kesalahan yang dinormalisasi terhadap nilai sebenarnya. Padahal, secara aktual situasi seperti itu jarang didapatkan.

Pada metode numerik, nilai sebenarnya akan diketahui hanya berkaitan dengan fungsi yang bisa diselesaikan secara analitik. Padahal banyak model matematika dari fenomena fisis yang tidak bisa diselesaikan secara analitik. Dalam situasi ini, yang benar-benar kita tahu adalah harga pendekatan dan taksiran kesalahan, atau suatu batasan ukuran maksimum dari kesalahan. Contohnya, kita tahu salah satu akar negatif dari polinomial $x^4 - 9x^3 - 2x^2 + 120x - 130$ (Lihat gambar 1.14) berada diantara nilai –3,60016 dan –3,60013.



Gambar 1.14. Ploting Mathematica untuk fungsi polinomial
$x^4 - 9x^3 - 2x^2 + 120x - 130$

Maka, dapat dikatakan bahwa akarnya adalah –3,600145 ± 0,000015. Harga pendekatannya $\bar{x} = -3{,}600145$, mengandung beberapa kesalahan. Meskipun kita tidak mengetahui besarnya kesalahan tetapi kita tahu bahwa kesalahan ini tidak lebih dari 1,5 x 10^{-5}.

```
NSolve[x^4-9 x^3-2 x^2+120x-130□0,x]
 {{x→-3.60014},{x→1.22859},{x→3.97207},{x→7.39948}}
```

Alternatif metode numerik adalah kesalahan dinormalisasi menggunakan estimasi nilai terbaik yang didapat terhadap nilai sebenarnya yaitu *nilai pendekatan* itu sendiri.

Hal yang harus diingat, metode numerik menggunakan pendekatan iteratif untuk mencari solusi komputasi. Prosesnya adalah berulang-ulang (*repeatdly*), atau secara iteratif untuk mencari nilai pendekatan yang baik, lebih baik, dan paling baik, lebih lanjut terkait definisi nilai pendekatan sekarang adalah dibuat berdasarkan nilai pendekatan sebelumnya, sehingga formula (1.19) menjadi

$$e_{ra} = \frac{e_{\bar{x}}}{\bar{x}_c} = \frac{\bar{x}_c - \bar{x}_p}{\bar{x}_c} \qquad (1.20)$$

dimana e_{ra} =kesalahan pendekatan relatif $e_{\bar{x}}$ =kesalahan pendekatan, $\bar{x}_p$ =nilai pendekatan sebelumnya, $\bar{x}_c$ =nilai pendekatan sekarang.

Seringkali ketika melakukan komputasi, kita tidak mempersoalkan tanda (positif atau negatif) dari kesalahan, tetapi lebih tertarik apakah prosentase nilai absolut lebih rendah dari toleransi prosentase yang telah di-spesifikasi, e_s . Proses komputasi akan diulang sampai

$$\left| e_{ra} \right| < e_s \qquad (1.21)$$

Toleransi kesalahan e_s juga berkaitan erat dengan jumlah bilangan berarti dalam pendekatan. Kaitan ini bisa ditunjukkan (Scarborough, 1966) sebagai berikut :

$$e_s = (0{,}5x10^{2-n})\% \qquad (1.22)$$

Contoh 1d

Dalam matematika, fungsi seringkali dapat dinyatakan dalam deret takhingga. Dalam hal ini fungsi eksponensial bisa dihitung dengan

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!} \qquad (1.23)$$

Sehingga semakin suku yang ditambahkan dalam deretan, pendekatan menjadi semakin baik dan estimasi nilai kebenaran e^x menjadi semakin baik juga. Persamaan (1.23) adalah perluasan deret Taylor disekitar titik x=0 yang disebut *deret Maclaurin*§. Mulai dengan suku paling awal, e^x=1, jumlahkan dengan suku

---

§ deret Maclaurin lainnya, diantaranya: $\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$,

$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$ dan $\ln(x+1) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots$

selanjutnya untuk memperkirakan nilai $e^{0,5}$. Setiap kali menambahkan satu suku, hitung prosentase kesalahan relatif (e_r) dan kesalahan pendekatan relatif (e_{ra}) masing-masing dengan persamaan (1.19) dan (1.20). Catatan bahwa nilai kebenaran $e^{0,5}$=1,648721… Tambahkan suku-suku sampai nilai kesalahan absolut dari estimasi pendekatan e_{ra} kurang dari toleransi kesalahan e_s sejumlah tiga bilangan berarti.

Solusi

Pertama, gunakan persamaan (1.22) untuk menentukan standar kesalahan yang ditentukan, tepatnya paling sedikit tiga bilangan berarti:

$$e_s = (0{,}5x10^{2-3})\% = 0{,}05\%$$

Lalu kita akan tambahkan suku-suku dalam deret sampai e_{ra} kurang dari level ini.

Estimasi pertama, sama dengan 1. Estimasi kedua diberikan oleh penjumlahan kedua suku
berikut:

$$e^x = 1 + x \,,$$

untuk x=0,5

$$e^{0,5} = 1 + 0{,}5 = 1{,}5$$

Kesalahan relatif (e_r) = $\dfrac{1{,}648721 - 1{,}5}{1{,}648721} 100\% = 9{,}02\%$

Kesalahan pendekatan relatif (e_{ra}) = $\dfrac{1{,}5 - 1}{1{,}5} 100\% = 33{,}3\%$

Karena e_{ra} lebih dari nilai e_s, dilanjutkan dengan menjumlahkan suku berikutnya, $x^2/2!$, dan diulang perhitungan kesalahannya. Proses berlanjut sampai $e_{ra} < e_s$. Komputasi keseluruhan diringkas sebagai berikut:

Suku	*Hasil*	e_r (%)	e_{ra} (%)
1	1	39,3	
2	1.5	9,02	33,3
3	1,625	1,44	7,69
4	1,645833333	0,175	1,27
5	1,648437500	0,0172	0,158
6	1,648697917	0,00142	0,0158

Setelah suku keenam disertakan, kesalahan pendekatan dibawah $e_s = 0{,}05\%$ dan komputasi dihentikan.

Sebenarnya terlihat bahwa, dari jumlah tiga bilangan berarti, hasilnya tepat sampai suku kelima. Hal ini karena, untuk kasus ini, kedua persamaan (1.19) dan (1.20) kaku. Artinya, keduanya memastikan bahwa hasilnya sekurang-kurangnya sebaik spesifikasinya.

Dengan mengenali dan memahami masalah-masalah yang tidak bisa diabaikan dalam pemrograman metode numerik, proses penyelesaiannya menjadi tidak terasa sulit ☺.

```
f(x)=e^x → deret pangkat dalam x di sekitar x=0, hingga 6 suku
Series[Exp[x],{x,0,6}]
```

$$1+x+\frac{x^2}{2}+\frac{x^3}{6}+\frac{x^4}{24}+\frac{x^5}{120}+\frac{x^6}{720}+O$$

Setelah memahami pengertian dasar diatas, selanjutnya akan dibahas tiga macam kesalahan dasar dalam metode numerik, yaitu:

[1] kesalahan *inherent*, [2] kesalahan pemendekan (*truncation error*), dan [3] kesalahan pembulatan (*roundoff errors*).

1.4.1. Kesalahan Yang Tidak Dapat Dipisahkan (*Inherent Errors*)

Kesalahan *Inherent* adalah kesalahan didalam besaran data, disebabkan oleh ketidakpastian pengukuran, baik oleh kekeliruan langsung atau oleh harga pendekatan yang diperlukan untuk menyatakan suatu bilangan dengan jumlah digit tak terbatas, dimana tentunya tidak dapat dinyatakan secara tepat oleh jumlah digit yang tersedia.

Suatu pengukuran fisis, seperti pengukuran jarak, tegangan atau periode waktu tidak bisa sangat eksak. Bila pengukuran diberikan dalam jumlah digit yang banyak, seperti bila tegangan 6,4837569 volt, dapat kita pastikan paling sedikit beberapa digit bagian kanan tidak mempunyai arti, karena tegangan tidak terukur sampai dengan ketelitian itu.

Pengukuran yang hanya diberikan dalam beberapa digit, seperti interval waktu 2,3 detik, dapat dipastikan ada kesalahan *inherent* didalamnya., karena hanya akan merupakan suatu kebetulan interval waktu tersebut benar-benar 2,3 detik.[**] Dalam kasus semacam ini dapat diketahui beberapa batasan yang masuk akal, misalnya interval waktu adalah 2,3 ± 0,1 detik.

Bila data pengukuran fisis dinyatakan tanpa kualifikasi digit yang mempunyai arti (*significant*), kadang-kadang dianggap data ini teliti untuk

---

[**] Pada kenyataannya, sering tidak ada artinya mengatakan hasil pengukuran yang "tepat" (eksak); tidak ada definisi yang tepat dapat berlaku untuk hal ini

setengah angka dari digit terakhir. Jadi bila jarak dinyatakan sebagai 5,63 cm dapat dimengerti bahwa bilangan tersebut tidak akan kurang dari 5,625 dan tidak akan lebih dari 5,635. Tetapi perjanjian (*konvensi*) ini tidak selalu terlihat.

Tanpa memandang jumlah digit yang digunakan untuk menyatakan suatu besaran[††], besaran ini dapat mengandung beberapa macam kesalahan langsung. Kesalahan tersebut berkisar antara kekeliruan yang sederhana seperti kesalahan menyalin data atau kesalahan membaca skala sampai dengan kesalahan "canggih" (*sophisticated*) didasarkan pada ketidak-lengkapan pemahaman hukum-hukum fisika.

Banyak bilangan yang tidak dapat dinyatakan secara tepat dalam sejumlah digit tertentu. Bila kita perlukan π dalam suatu perhitungan, kita dapat menuliskan sebagai 3,14. 3,14159265 atau 3,141592653589793 atau Mathematica bahkan bisa membangkitkan hingga beberapa ratus ribu tempat desimal ☺.

```
pi=N[Pi,100]
3.14159265358979323846264338327950288419716939937510582
097494459230781640628620899862803482534211706 8
```

Hal ini juga terjadi pada bilangan pecahan dengan bilangan dasar tertentu yang tidak dapat ditampilkan secara tepat bila menggunakan bilangan dasar lainnya. Bilangan $\frac{1}{10}$ jelas dapat dituliskan secara sederhana dalam desimal 0,1. tetapi penulisannya dalam bilangan biner adalah 0,000110011001100…, suatu bilangan biner yang tidak berhenti-henti. Jadi, menjumlahkan 10 bilangan yang masing-masing merupakan pendekatan biner dari bilangan desimal 0,1 tidak akan memberikan dengan tepat angka 1,0. Keanehan seperti ini pasti dijumpai saat pertama kali bekerja dalam pemrograman dengan komputer biner.

1.4.2. Kesalahan Pemendekan (*Truncation Errors*)

Kesalahan *inherent* diacu sebagai kesalahan dalam data, lalu data diolah oleh komputer dengan menggunakan prosedur metode numerik. Kedua jenis kesalahan lainnya, kesalahan pemendekan (*truncation*) dan kesalahan pembulatan (*roundoff*), adalah kesalahan yang muncul oleh prosedur metode numerik sendiri.

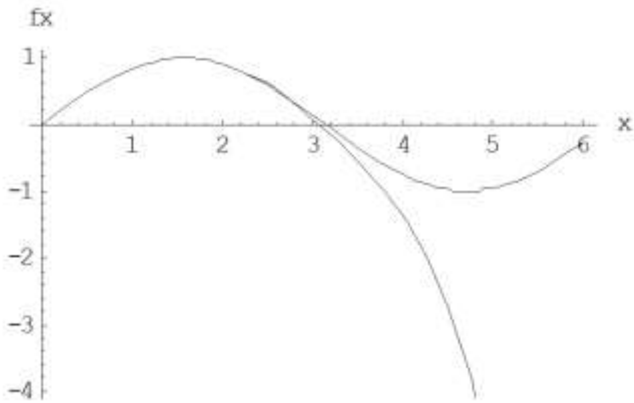
Deret Taylor tak terhingga, yang sering dikenal

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots \tag{1.24}$$

[††] Rumusan yang dinilai mendekati kebenaran, Besaran :"sifat melekat pada sebuah obyek atau benda (konkrit atau abstrak), yaitu sifat yang terdapat dalam, atau yang tidak dapat dipisahkan dari obyek atau benda tersebut sehingga dapat difahami sebagai salah satu ciri, atribut atau jatidiri obyek atau benda tersebut".

dapat digunakan untuk menghitung sinus dari suatu sudut x dalam radian. Kita tidak dapat menggunakan semua suku deret tersebut dalam perhitungan, karena deret berjumlah takhingga, maka kita harus putuskan setelah menghitung sejumlah suku tertentu, misal sampai x^7.

```
Plot[{Sin[x],(x-(x^3/3!)+(x^5/5!)-(x^7/7!))},{x,0,6},
AxesLabel ->{x[rad],fx}]
```




```
□Graphics□
```

Gambar 1.15. Plot kedua fungsi pada persamaan (1.24), ada simpangan??

Suku yang diabaikan (yang jumlah sukunya tak terbatas) memberikan kesalahan pada hasil perhitungan. Kesalahan ini disebut *kesalahan pemendekan*, disebabkan oleh pemotongan dari proses matematika tak terhingga (*infinite*).

Banyak prosedur yang digunakan pada metode numerik adalah takhingga, sehingga kesalahan pemendekan ini merupakan bagian kesalahan utama. Kita akan meninjau formula matematika yang dipakai secara luas di metode numerik untuk menyatakan fungsi-fungsi dalam membuat pendekatan, yaitu deret Taylor:

$$f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2$$

$$+ \frac{f^{(3)}(x_i)}{3!}(x_{i+1} - x_i)^3 + \ldots + \frac{f^{(n)}(x_i)}{n!}(x_{i+1} - x_i)^n + R_n \qquad (1.25)$$

Catatan bahwa persamaan (1.25) adalah sebuah deret takhingga. Suku terakhir adalah dimasukkan untuk perhitungan penjumlahan semua suku dari n+1 sampai takhingga:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} h^{n+1} \qquad (1.26)$$

Dimana n berkonotasi untuk pendekatan pangkat-n dan ξ adalah sebuah nilai x berada diantara x_i dan x_{i+1}.

Dalam implementasi pada analisa numerik, sebagai contoh kadang-kadang dilakukan pendekatan deret taylor orde kedua, sehingga rumusannya menjadi:

$$f(x_{i+1}) \cong f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2 \qquad (1.27)$$

atau untuk pendekatan orde pertama diberikan oleh:

$$f(x_{i+1}) \cong f(x_i) + f'(x_i)(x_{i+1} - x_i) \qquad (1.28)$$

Contoh 1e

Gunakan deret Taylor orde nol sampai keempat untuk melakukan pendekatan terhadap fungsi

$$f(x) = -0{,}1x^4 - 0{,}15x^3 - 0{,}5x^2 - 0{,}25x + 1{,}2$$

dari x_i=0 dengan h=1. Artinya, tebak nilai dari fungsi pada x_{i+1}=1

Solusi

Berkaitan dengan fungsi yang telah diketahui, kita dapat menghitung nilai *f(x)* antara 0 dan 1. Hasil (gambar 1.16) menunjukkan bahwa fungsi dimulai pada f(0)=1,2 dan selanjutnya kurva melengkung turun sampai f(1)=0,2. Sehingga, nilai sebenarnya yang kita coba tebak adalah 0,2.

Pendekatan deret Taylor dengan n=0 adalah $f(x_{i+1}) \cong 1{,}2$, sehingga seperti dalam gambar 1.16, pendekatan orde nol adalah nol. Maka *truncation error* adalah

$e_x = 0{,}2 - 1{,}2 = -1{,}0$ pada x=1.

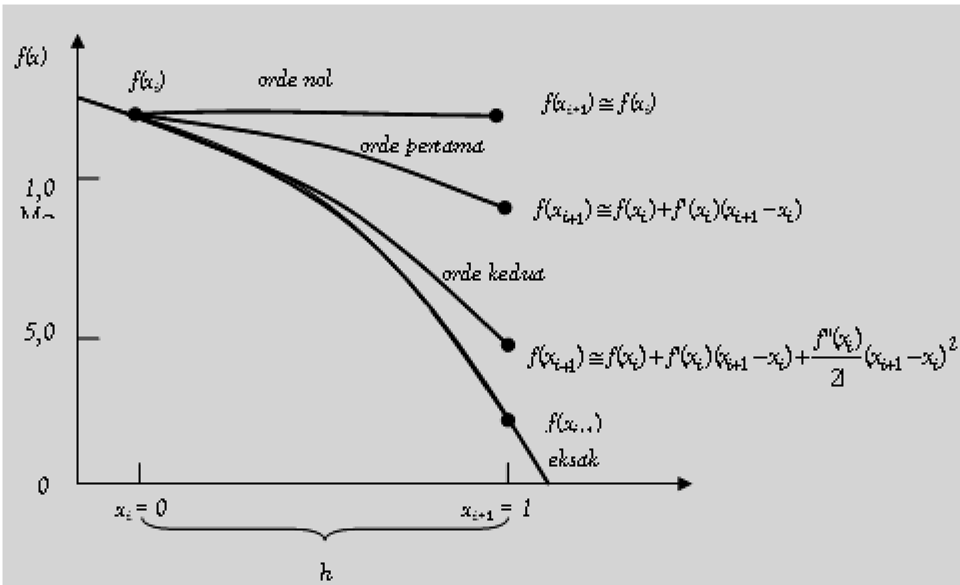
Pada n=1, turunan pertama harus dievaluasi dulu pada x=0,

$$f'(0) = -0{,}4(0{,}0)^3 - 0{,}45(0{,}0)^2 - 1{,}0(0{,}0) - 0{,}25 = -0{,}25,$$

sehingga pendekatan orde pertama didapatkan: $f(x_{i+1}) \cong 1{,}2 - 0{,}25h$ sehingga f(1)=0,95. Konsekuensinya, pendekatan mulai menangkap fungsi lintasan lengkung berupa garis lurus menurun.

Hasil ini memberikan reduksi pada *truncation error*:

$$e_x = 0{,}2 - 0{,}95 = -0{,}75 \,.$$



Gambar 1.16. Pendekatan $f(x) = -0{,}1x^4 - 0{,}15x^3 - 0{,}5x^2 - 0{,}25x + 1{,}2$ pada *x*=1 dengan ekspansi deret Taylor orde nol, orde pertama, dan orde kedua.

Untuk n=2, turunan kedua dihitung pada x=0, memberikan:

$$f''(0) = -1{,}2(0{,}0)^2 - 0{,}9(0{,}0) - 1{,}0 = -1{,}0$$

selanjutnya, $f(x_{i+1}) \cong 1{,}2 - 0{,}25h - 0{,}5h^2$.

Subtitusi h=1, memberikan *f(1)*=0,45. Terlihat bahwa penambahan turunan kedua, menyebabkan kelengkungan kurva semakin baik dan truncation error berkurang jauh menjadi: $e_x = 0{,}2 - 0{,}45 = -0{,}25$.

Penambahan suku deret akan memberikan nilai pendekatan yang semakin baik. Ketika turunan ketiga dan keempat dimasukkan, didapatkan hasil yang sama persis dengan persamaan soal, sebagai berikut:

$$f(x_{i+1}) \cong 1{,}2 - 0{,}25h - 0{,}5h^2 - 0{,}15h^3 - 0{,}1h^4$$

dimana suku sisa: $R_n = \dfrac{f^{(5)}(\xi)}{5!} h^5 = 0$

karena turunan kelima dari polinomial orde empat adalah nol.

Konsekuensinya, deret Taylor sampai turunan keempat menghasilkan perkiraan eksak pada x_{i+1}=1:

$$f(1) \cong 1{,}2 - 0{,}25(1) - 0{,}5(1)^2 - 0{,}15(1)^3 - 0{,}1(1)^4 = 0{,}2$$

Secara umum, deret Taylor orde ke-n, akan memberikan nilai eksak polinomial orde ke-n. Untuk fungsi kontinu dan bisa dideferensialkan lainnya, seperti eksponensial dan sinusoidal, sejumlah suku terbatas tidak akan memberikan perkiraan nilai eksak.

Penulisan (1.26) dalam bab-bab selanjutnya, dituliskan sebagai $R_n = O(h^{n+1})$, dimana *truncation error* adalah pada orde h^{n+1}, dan kesalahan akan proporsional dengan lebar langkah *h* pangkat *(n+1)*.

1.4.3. Kesalahan Pembulatan (*roundoff errors*)

Dalam notasi desimal, setiap bilangan real dinyatakan ole h barisan berhingga atau takhingga angka desimal. Untuk komputasi mesin bilangan harus digantikan oleh sejumlah angka berhingga.

Komputer memiliki dua cara untuk menyatakan bilangan, yakni bilangan titik tetap atau bilangan bulat (*fix point*) dan bilangan titik kambang atau titik mengambang (*floating point*) ‡‡. Dalam sistem titik tetap, bilangan dinyatakan dalam sejumlah tetap posisi desimal, misalnya 62358; 13; 1000. Sistem bilangan ini pemakaiannya tidak praktis dalam pekerjaan ilmiah karena keterbatasan rentang. Perhitungan ilmiah dalam terapan sains dan rekayasa biasanya dilakukan dalam aritmetika *floating point*.

Suatu bilangan *floating point* dari n angka dalam dasar β mempunyai bentuk $x = \pm(.d_1 d_2 \ldots d_n)_\beta \beta^e$, dimana $(.d_1 d_2 \ldots d_n)_\beta$ adalah suatu pecahan $-\beta$ yang disebut mantissa, dan e adalah suatu bilangan bulat yang disebut eksponen. Bilangan *floating point* semacam itu dikatakan ternormalisasi dalam hal $d_1 \# 0$, *atau* $d_1=d_2=\ldots=d_n=0$. Contohnya bilangan 0,6238x10^3; 0,1714x10^{-13}; –0,2x10^1, dapat dituliskan sebagai: 0,6238E03; 0,1714E–13 dan –0,2E01. Implementasi komputer bentuk $x = \pm(.d_1 d_2 \ldots d_n)_\beta \beta^e$ menetapkan pembatasan pada banyaknya yang digunakan dalam mantissa dan bahwa rentang eksponen yang mungkin juga terbatas.

Komputer sekarang dominan pada β=32, walaupun teknologi terbaru memiliki β=64, sedangkan komputer-komputer lama memiliki β=2,8,10 atau 16 dan pada kebanyakan kalkulator meja atau kalkulator saku memiliki β=10. Pada β=32 berarti komputer menggunakan 32 bit (angka biner) untuk menyatakan

---

‡‡ beberapa literatur indonesia seperti dasar-dasar metode numerik, Nyoman Susila ITB, menyebut titik kambang, terjemah Elementary Numeric Analysis, Conte–de Boor oleh Mursaid, Erlangga menyebut titik mengambang, disini dipakai istilah baku-nya *floating point*

bilangan real presisi-tunggal memakai 8 bit untuk eksponen dan 24 bit untuk mantissa. Jadi nilai mutlak bilangan real berada dalam rentang 0,2938736E–38 sampai 0,1701412E+39 (yakni 2^{-128} sampai 2^{127}) dengan enam angka desimal presisi numerik (yakni $2^{-23}=1,2x10^{-7}$).

Komputer yang menggunakan 48 bit untuk menyatakan bilangan-bilangan real presisi tunggal mungkin memakai 8 bit untuk eksponen dan 40 bit untuk mantissa. Komputer ini dapat menyatakan bilangan real yang nilai mutlaknya berada dalam rentang 0,29387358771E–39 sampai 0,17014118346E+39, degan 11 angka desimal presisi numerik (yakni $2^{-29}=1,8x10^{-12}$).

Ada dua jalan yang ditempuh untuk menterjemahkan suatu bilangan nyata x tertentu ke dalam suatu bilangan *floating point fl(x)* yang tersusun dari sejumlah n angka $-\beta$, yaitu dengan jalan pembulatan (*rounding*) dan pemenggalan (*chopping*). Pada pembulatan, *fl(x)* dipilih sebagai bilangan *floating point* ternormalisasi yang terdekat dengan x; suatu aturan khusus, seperti pembulatan simetris (pembulatan menjadi angka genap) digunakan dalam hal terjadinya persamaan angka (tie). Pada pemenggalan, *fl(x)* dipilih sebagai bilangan *floating point* ternormalisasi yang terdekat antara x dan 0.

Jika misalnya, digunakan bilangan-bilangan *floating point* desimal dari dua angka, maka:

$$fl\left(\tfrac{2}{3}\right)=\begin{cases}(0,67)10^{0} & \quad dibulatkan(rounded)\\ (0,66)10^{0} & \quad dipenggal(chopped)\end{cases}$$

$$fl(-838)=\begin{cases}-(0,84)10^{3} & \quad dibulatkan(rounded)\\ -(0,83)10^{3} & \quad dipenggal(chopped)\end{cases}$$

Aturan pembulatan yang kita kenal adalah membuang angka desimal yang ke (k+1) dan sesudahnya, jika :

[a] Bilangan yang dibuang lebih kecil dari setengah satuan dalam posisi ke-k, biarkan angka desimal ke-k tidak diubah (*Pembulatan ke bawah*). Pembulatan 1,263 sampai 2 angka dibelakang koma, memberikan 1,26

[b] Bilangan yang dibuang lebih besar dari setengah satuan dalam posisi ke-k, tambahkan satu pada angka desimal ke-k (*Pembulatan ke atas*). Pembulatan 1,263 sampai 1 angka dibelakang koma, memberikan 1,3

[c] Bilangan tersebut tepat setengah satuan, bulatkan ke angka desimal genap yang terdekat. Pembulatan 3,45 dan 3,55 ke 1 angka dibelakang koma masing-masing memberikan 3,4 dan 3,6.

Aturan bagian terakhir dimaksudkan untuk menjamin bahwa dalam pembuangan tepat setengah desimal, pembulatan ke atas dan ke bawah secara rata-rata berlangsung hampir sama seringnya. Jika kita membulatkan 1,2535 sampai 3,2,1 angka dibelakang koma, diperoleh 1,254; 1,25; dan 1,3 tetapi jika 1,25 dibulatkan sampai satu angka dibelakang koma, tanpa informasi lebih jauh akan memberikan 1,2.

Pemenggalan tidak direkomendasikan karena memperkenalkan kesalahan yang bersistem dan dapat lebih besar daripada kesalahan pembulatan. Meskipun demikian, agak mengejutkan bahwa banyak komputer memakai pemenggalan! Kebanyakan komputer yang memakai pembulatan selalu membulatkan ke atas dalam item [c], karena secara teknis ini lebih mudah direalisasikan.

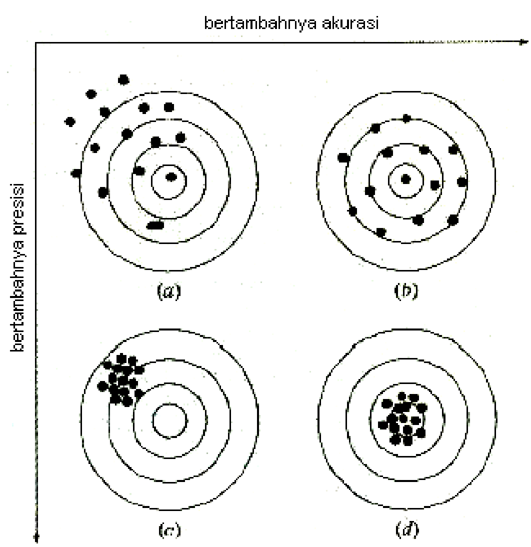
Pembulatan dapat menghancurkan suatu komputasi secara lengkap, walaupun komputasinya kecil. Umumnya, pembulatan semakin berbahaya dengan semakin banyaknya operasi hitungan yang harus dilakukan (mungkin beberapa juta). Karenanya adalah penting untuk menganalisis program komputasi untuk menaksir kesalahan pembulatan yang diharapkan dan menemukan penyusunan komputasi yang mengakibatkan pengaruh kesalahan pembulatan sekecil mungkin.

Sejumlah kesalahan pembulatan yang serius terjadi pada saat:

1) Menjumlahkan (atau mengurangkan) bilangan yang sangat kecil dengan bilangan yang cukup besar, misalnya 123,5 dengan 0,0000043.
2) Mengurangkan satu bilangan dengan bilangan yang lain, yang besarnya hampir sama, misalnya 3,141592653 dengan 3,141595734.

1.5. Presisi dan Akurasi

Dalam proses komputasi, berkurangnya kecermatan dari hasil akhir, seperti pada kasus kesalahan inherent, truncation dan round off, dapat menyelinap masuk pada saat yang tidak diperkirakan.



Gambar 1.17. Ilustrasi presisi dan akurasi

Sehingga aspek penting yang harus selalu dijadikan perhatian terhadap data komputasi adalah presisi (*precision*)dan akurasi (*accuracy*). Capra-Canale (1998), memberikan ilustrasi terkait dengan aspek presisi dan akurasi seperti pada gambar 1.17. Akurasi mengacu pada seberapa dekat angka pendekatan terhadap harga sebenarnya dan presisi mengacu pada jumlah angka signifikan yang menyatakan suatu besaran dan penyebaran dari nilai-nilai yang didapatkan.

Contoh 1f

Jumlahkan bilangan yang cukup kecil 0,0010 dengan bilangan yang besar 4000 menggunakan komputer hipotetik dengan 4 digit mantissa dan 1 digit eksponen. Apakah terjadi kesalahan?

<u>Solusi</u>

Bilangan yang cukup kecil dimodifikasi sehingga eksponennya cocok dengan bilangan yang besar,

$$\begin{array}{ll} 0{,}4000 & .10^4 \\ \underline{0{,}0000001} & \underline{.10^4} \\ 0{,}4000001 & .10^4 \end{array}$$

lalu dipenggal (*chopped*) menjadi $0{,}4000\ .10^4$ oleh komputer hipotetik. Pada komputasi telah terjadi kesalahan, karena setelah penjumlahan, memberikan hasil yang sama dengan bilangan yang besar.

Contoh 1g

Lakukan test program menggunakan quickC atau Basic pada komputer IBM PC penjumlahan bilangan 1 dan 0,00001 yang diulang sebanyak 10.000 kali dan hitunglah besar kesalahan relatifnya !

<u>Solusi</u>

Program dibuat untuk menghitung bilangan `1 + 0,00001+ 0,00001 + 0,00001 +…` , dengan `0,00001` sebanyak `10.000` kali

Program dalam C

```
/*Penjumlahan dengan Presisi Tunggal    sum_singl.c */
#include <stdio.h>
main ()
{
float x, jum = 1.0;
int i = 0;
      for (i =1; i<= 10000; i++)
      {
            jum = jum + 0.00001;
```

```
      }
      printf("\nJum = %f\n", jum);
}
```

Hasil dari program pada IBM PC adalah

Jum = 1.100136

Kesalahan relatif hasil komputasi adalah

$$\frac{1{,}1-1{,}100136}{1{,}1}=-0{,}000124 \text{ atau } -0{,}0124\%$$

Contoh 1h

Bandingkan hasil perhitungan f(500) dan g(500) dengan menggunakan enam angka dan pembulatan, Fungsi-fungsinya adalah:

$$f(x)=x\left(\sqrt{x+1}-\sqrt{x}\right) \text{ dan } g(x)=\frac{x}{\sqrt{x+1}+\sqrt{x}}$$

<u>Solusi</u>

Untuk fungsi pertama, f(x) pada x=500 memberikan

$$f(500)=500\left(\sqrt{501}-\sqrt{500}\right)=500(22{,}3830-22{,}3607)=500(0{,}0223)=11{,}1500$$

Untuk fungsi kedua, g(x) pada x=500 memberikan

$$g(500)=\frac{500}{\sqrt{501}+\sqrt{500}}=\frac{500}{22{,}3830+22{,}3607}=\frac{500}{44{,}7437}=11{,}1748$$

fungsi kedua g(x), secara aljabar ekivalen f(x), dan terlihat bahwa jawaban g(500)=11,1748 melibatkan kesalahan lebih kecil daripada f(x) dan sama seperti yang diperoleh dengan membulatkan jawaban sebenarnya 11,174755300747198… sampai enam angka. Ekivalensi f(x) dan g(x) ditunjukkan pada persamaan berikut:

$$f(x)=x\left(\sqrt{x+1}-\sqrt{x}\right)\frac{\sqrt{x+1}+\sqrt{x}}{\sqrt{x+1}+\sqrt{x}}=\frac{x\left[\left(\sqrt{x+1}\right)^2-\left(\sqrt{x}\right)^2\right]}{\sqrt{x+1}+\sqrt{x}}=\frac{x}{\sqrt{x+1}+\sqrt{x}}=g(x)$$

```
x((x+1)^0.5-(x)^0.5) /. x→500
11.1748
x/((x+1)^0.5+(x)^0.5) /. x→500
11.1748
```

Efek dari pembulatan(*round off*) bisa diminimalisir dengan merubah algoritma komputasi walaupun hal ini harus dipikirkan dan direncanakan kasus demi kasus. Beberapa strategi yang digunakan termasuk:

1) Menggunakan presisi ganda (*Double precision [McCracken]*)
2) Melakukan pengelompokan (*Grouping*)

3) Ekspansi Taylor (*Taylor expansions*)
4) Merubah definisi variabel-variabel
5) Menuliskan kembali persamaan untuk menghindari pengurangan

Strategi penggunaan presisi ganda (*double precision*) dan pengelompokan (*grouping*) dapat diterapkan pada contoh 1f. Dengan mengubah variabel dari presisi tunggal ke presisi ganda dan di-inisialisasi pada 1, seperti pada list program dibawah, memberikan hasil yang lebih baik.

```
/*Penjumlahan dengan Presisi Ganda      sum_dbl.c */
#include <stdio.h>
main ()
{
double x, jum = 1.0;
int i = 0;
      for (i =1; i <= 10000; i++)
      {
            jum = jum + 0.00001;
      }
      printf("\nJum = %f\n", jum);
}
```

Hasil pada versi ini
Jum=1.10000000000

sedangkan strategi *grouping*, membantu menghitung jumlah sekian bilangan kecil, untuk mengurangi kesalahan pembulatan.

```
/*Penjumlahan dengan Grouping      sum_grup.c */
#include <stdio.h>
main ()
{
float x, grup_total, jum = 1;
int i, k = 0;
      for (i =1; i <= 100; i++)
      {
             grup_total = 0;
                    for (k=1; k <= 100; k++)
            {
            grup_total = grup_total +  0.00001;
            }
      jum = jum + grup_total;
      }
      printf("\nJum = %13.8e\n", jum);
}
```

Hasil pada versi ini
Jum=1.10000467

Pada program terlihat, setiap 100 bilangan kecil dikumpulkan, baru total kelompok dihitung, selanjutnya total kelompok-kelompok diakumulasi. Pada kedua strategi pendekatan yang ditunjukkan, akurasi meningkat secara signifikan, walaupun pada *double precision* lebih baik dibanding *grouping*.

Strategi ekspansi Taylor (*Taylor expansions*), merubah definisi variabel-variabel dan menuliskan kembali persamaan untuk menghindari pengurangan, dapat difahami dari analisa kasus penilaian fungsi, $f(x)=1-\cos x$ dalam aritmetika 6-angka desimal dan masalah mencari akar dari persamaan kuadrat $ax^2+bx+c=0$

Pada kasus penilaian fungsi, $f(x)=1-\cos x$ memiliki potensi kehilangan angka-angka signifikan jika dihitung *f(x)* dengan mencari cos x lebih dahulu, kemudian mengurangkan nilai yang diperoleh dengan 1, karena cos x ≅ 1 untuk x yang mendekati nol. Karena tidak bisa menghitung cos x sampai lebih dari 6 angka, sehingga kesalahan dalam nilai yang dihasilkan dari hitungan bisa lebih besar dari *f(x)* untuk x mendekati nol. Untuk menghitung nilai *f(x)* dekat nol sampai kira-kira 6 angka signifikan dengan menggunakan aritmetika 6 angka, maka harus digunakan rumus lain untuk *f(x)*.

$$f(x)=1-\cos x=\frac{1-\cos^2 x}{1+\cos x}=\frac{\sin^2 x}{1+\cos x} \qquad (1.29)$$

yang dapat dicari nilainya dengan sangat cermat untuk x yang kecil, atau juga dapat digunakan ekspansi Taylor untuk *f(x),* yakni:

$$f(x)=\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\ldots \qquad (1.30)$$

yang menunjukkan, misalnya untuk $|x|\le 10^{-6}$, $\frac{x^2}{2}$ sesuai dengan *f(x)* sampai paling sedikit enam angka signifikan.

Persamaan (1.2) pada awal bab ini, dikenal sebagai formula abc yang akrab digunakan untuk menentukan akar-akar persamaan kuadrat $ax^2+bx+c=0$, secara aljabar. Diandaikan bahwa b^2-4ac > 0 dimana b>0, maka akar pada nilai mutlak terbesar , adalah:

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a} \qquad (1.31)$$

Aspek metode numerik dapat memberikan pandangan lain atas akurasi formula abc tersebut pada beberapa variasi konstantanya. Misalnya ditinjau persamaan $0.01\,x^2 + 111{,}11x + 1{,}2121 = 0$. Dengan menggunakan persamaan (1.31) dan aritmetika *floating point* pemenggalan sampai lima angka desimal dan sepuluh angka desimal, dapat dihitung:

	5 angka desimal	**10 angka desimal**
$\mathbf{b^2}$	12345.43210	12345.4321000000
$\mathbf{b^2 - 4ac}$	12345.38362	12345.3836160000
$\sqrt{b2 - 4ac}$	111.10978	111.1097818196
$x_1 = \dfrac{-b + \sqrt{b^2 - 4ac}}{2a}$	– 0.00000	– 0.0000010909

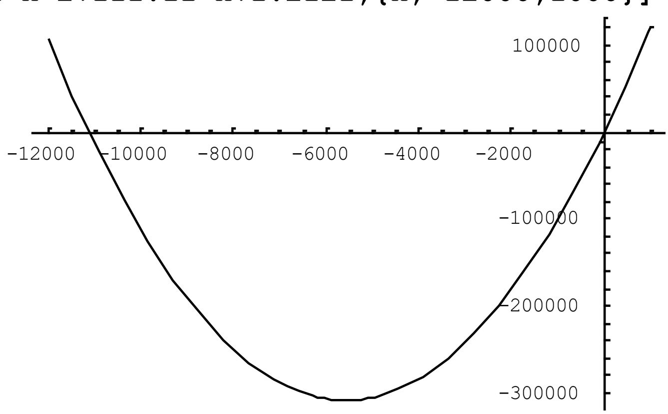
padahal dalam kenyataannya x_1=–0,01091 (5 angka desimal) atau – 0.0109090198 (10 angka desimal) sebagai akar yang tepat.

Hilangnya angka-angka signifikan pada kasus ini bisa dihindari dengan menggunakan rumus lain untuk menghitung akar pada nilai mutlak terbesar, yakni:

$$x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}} \tag{1.32}$$

yang akan memberikan hasil dengan cermat.

```
Plot[0.01 x^2+111.11 x+1.2121,{x,-12000,1000}]
```




```
□Graphics□

NSolve[0.01 x^2+111.11 x+1.2121□0,x]
 {{x→-11111.},{x→-0.010909}}
```

Kalau kita lakukan uji lebih lanjut, misalnya dilakukan empat uji dengan melakukan variasi pada konstanta a, yaitu a<1 (a= 0.01), a=1, a>1 (a=3), dan a>>1 (a=100), ternyata memberikan hasil yang akurat pada formula (1.32) untuk semua uji, sedangkan formula abc pada persamaan (1.31) hanya akurat pada uji kedua, yaitu saat a=1.

Lihat perbandingan akurasi kedua persamaan diatas pada tabel Microsoft Excell dibawah. Cobalah untuk variasi yang lain !.

Kasus 1:	a <1		Cek subtitusi Balik	Formula
aX^2	bX	c		
0.01	111.11	1.2121		
	x1 formula1	-0.0000010909	1.2119787899	(1.31)
	x1 formula2	-0.0109090198	0.0000000000	(1.32)
Kasus 2:	a=1			
aX^2	bX	c		
1	111.11	1.2121		
	x1 formula1	-0.0109100804	0.0000000000	(1.31)
	x1 formula2	-0.0109100804	0.0000000000	(1.32)
Kasus 3:	a>1			
aX^2	bX	c		
3	111.11	1.2121		
	x1 formula1	-0.0982100177	-9.6710794465	(1.31)
	x1 formula2	-0.0109122242	0.0000000000	(1.32)
Kasus 4:	a>>1			
aX^2	bX	c		
100	111.11	1.2121		
	x1 formula1	-110.1827227424	1201782.0488691900	(1.31)
	x1 formula2	-0.0110182723	0.0000000000	(1.32)

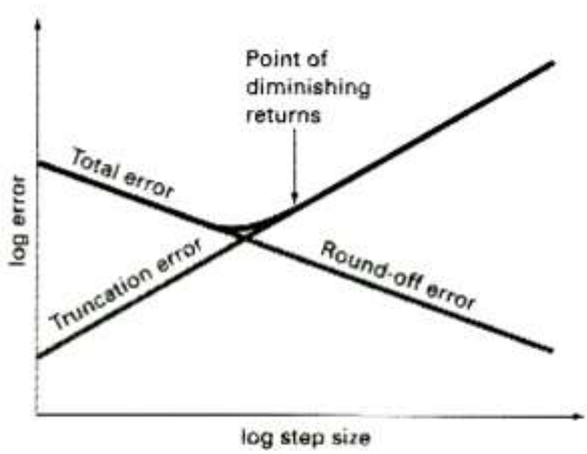
```
NSolve[1 x^2+111.11 x+1.2121□0,x]
 {{x→-111.099},{x→-0.0109101}}
 NSolve[3 x^2+111.11 x+1.2121□0,x]
 {{x→-37.0258},{x→-0.0109122}}
 NSolve[100 x^2+111.11 x+1.2121□0,x]
 {{x→-1.10008},{x→-0.0110183}}
```

1.10. Kajian Kesalahan Metode numerik

Kesalahan metode numerik dapat dikatakan sebagai penjumlahan kesalahan pemenggalan dan kesalahan pembulatan. Dari kasus penerjun payung, Kesalahan pembulatan akan bertambah ketika jumlah komputasi dalam analisis dinaikkan. Cara yang efektif untuk mereduksi kesalahan pembulatan adalah dengan meningkatkan angka signifikan.

Komputasi dapat meminimalkan kesalahan pemenggalan dengan cara memperkecil lebar langkah, tetapi pada saat itu juga porsi kesalahan pembulatan mulai mendominasi dan memungkinkan kesalahan komputasi bertambah. Lihat ilustrasi Capra-Canale yang menggambarkan keadaan dilematis ini, tampak pada gambar 1.18.

Yang diinginkan adalah memperkecil lebar langkah untuk mereduksi kesalahan pemenggalan, sekaligus adanya kenyataan bahwa tidak diikuti oleh peningkatan kesalahan pembulatan. Berarti harus ditemukan titik balik pengurangan (*diminishing return*) dimana kesalahan pembulatan mulai meniadakan efektivitas lebar langkah yang diperkecil.



Gambar 1.18. Dilematis antara kesalahan pemenggalan dan kesalahan pembulatan terkait penentuan lebar langkah

Dalam realitas, perangkat komputer saat ini sudah mendukung penanganan angka signifikan yang cukup besar, sehingga angka pembulatan tidak lebih dominan. Hal yang akhirnya harus disadari adalah kita harus selalu mencari, mencari dan mencari minimal dengan metode *trial and error* dalam memperkirakan akurasi dan presisi, yang pada titik tersebut melibatkan juga ’intuisi’ dan ’pengalaman’.

1.6. SOAL-SOAL

1) [a]. Ulangi pada contoh 1a. Hitung kecepatan parasut dengan metode metode numerik untuk t=12 detik, dengan lebar langkah 1 dan 0,5 detik. Buatlah pernyataan berkaitan dengan kesalahan perhitungan berdasarkan hasil. [b] Dibanding hubungan linear pada persamaan $F_u = -cv$, untuk gaya tarik keatas pada kasus parasut, secara aktual adalah nonlinear dan lebih baik dinyatakan oleh hubungan kuadrat, $F_u = -c'v^2$. Dengan menggunakan hubungan ini, hitung ulang pada contoh 1a. dengan keadaan awal dan nilai parameter yang sama. Gunakan nilai 0,23 kg/m untuk *c'*.
2) Jumlah bahan radioaktif yang berada di dalam reaktor tertutup diukur dalam konsentrasi *c* (becquerel/liter atau Bq/L). Laju peluruhan bahan=$-kc$ dimana k=konstanta dengan satuan hari^{-1}, dengan hubungan: $dc = -kc.dt$. Dengan metode numerik selesaikan persamaan ini dari t=0 sampai 1 hari, dengan k=0,1 hari^{-1}. Gunakan lebar langkah Δt=0,1. Konsentrasi bahan pada radioaktif pada t=0 adalah 10 Bq/L.
3) Bilangan-bilangan berikut ini diberikan pada komputer desimal dengan mantissa ternormalisasi empat angka
   a. $0{,}4523\ .10^4$ b. $0{,}2115\ .10^{-3}$ c. $0{,}2583\ .10^1$
4) Lakukan operasi berikut dan tunjukkan kesalahannya dari hasil hitungan, dengan asumsi pembulatan simetris:
   [i]. a + b + c [ii]. a – b [iii]. a.b/c
5) Andaikan $P(x)=((x^3-3x^2)+3x)-1$ dan $Q(x)=((x-3)\ x + 3)\ x - 1$, gunakan pembulatan 3 angka untuk menghitung pendekatan pada x=2,19. Bandingkan dengan nilai sebenarnya P(2,19)= Q(2,19) = 1,685159
6) Carilah akar yang terkecil dari persamaan berikut:
   $$x^2 + 0{,}4002.10^0\, x + 0{,}8.10^{-4} = 0$$
   dengan menggunakan rumus (1.24) dan (1.25). Kerjakan dalam aritmetika *floating point* dengan menggunakan mantissa 4 tempat desimal.

DAFTAR PUSTAKA

Chapra, S.C., and Canale, R.P., *Numerical Methods for Engineers*, McGraw-Hill, 1998

dan Simulasi Sistem Fisis, ITB, 2007

Mathews, J.H., *Numerical Methods for Mathematics, Science and Engineering*, Prentice-Hall Inc., 1992

Nakamura, S., *Applied Numerical Methods in C*, Prentice-Hall Inc. 1993

Subakti, Irfan, *Metode Numerik*, FTI ITS, Surabaya, 2003