MATERI PERKULIAHAN ANALISIS NUMERIK

Bab 1

Pendahuluan Matematis dan Analisis Galat

Mathematical Preliminaries and Error Analysis

Halaman buku 1 – 41 • 4 seksi

Catatan penggunaan

Dokumen ini adalah ringkasan belajar yang ditulis ulang dan disusun berdasarkan Richard L. Burden dan J. Douglas Faires, Numerical Analysis, Edisi ke-9, Cengage Learning, 2010. Dokumen ini bukan pengganti buku: bukti teorema, contoh lengkap, dan soal latihan ada di buku; gunakan nomor halaman pada tabel di bawah untuk rujukan.

Web by: Steven Paulus Siahaan

Ringkasan Bab

Bab ini menyiapkan fondasi: kalkulus yang dipakai untuk menganalisis galat, representasi bilangan di komputer, dan cara menilai kecepatan konvergensi sebuah algoritma. Hampir semua bab berikutnya memakai deret Taylor beserta suku sisa untuk menurunkan metode dan menaksir galatnya.

Komputer menyimpan bilangan real dalam jumlah digit terbatas, sehingga setiap operasi dapat menambah galat pembulatan. Pengurangan dua bilangan yang hampir sama (kehilangan angka signifikan) dan pembagian dengan bilangan sangat kecil adalah sumber galat yang paling sering merusak hasil.

Tujuan Pembelajaran

1. Menggunakan teorema nilai rata-rata, nilai antara, dan Taylor untuk menaksir galat.
2. Menghitung galat mutlak dan relatif serta menjelaskan angka signifikan.
3. Menjelaskan aritmetika titik-mengambang (pemotongan vs pembulatan) dan akibatnya.
4. Menyatakan laju konvergensi dengan notasi $O(\cdot)$.

Peta Isi Bab

Seksi	Judul	Hal.	Definisi, teorema, algoritma di buku
1.1	Review of Calculus	2	Def. 1.1; Def. 1.2; Def. 1.3; Teo. 1.4; Def. 1.5; Teo. 1.6; Teo. 1.7 (Rolle’s Theorem); Teo. 1.8 (Mean Value Theorem) …; 4 contoh
1.2	Round-off Errors and Computer Arithmetic	17	Def. 1.15; Def. 1.16; 6 contoh
1.3	Algorithms and Convergence	32	Def. 1.17; Def. 1.18; Def. 1.19; 3 contoh
1.4	Numerical Software	41	–

Rumus dan Metode Kunci

Polinom Taylor dan suku sisa

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)^{n+1}$$

Galat mutlak dan relatif

$$E_{abs} = |p - p^*|, \qquad E_{rel} = \frac{|p - p^*|}{|p|} \quad (p \neq 0)$$

Bilangan titik-mengambang k digit

$$\pm 0.\, d_1 d_2 \ldots d_k \times 10^n, \qquad d_1 \neq 0$$

Laju konvergensi

$$|\alpha_n - \alpha| \leq K\,|\beta_n| \;\Rightarrow\; \alpha_n = \alpha + O(\beta_n)$$

Web by: Steven Paulus Siahaan

1.1 Review of Calculus

Rujukan buku: halaman 2

Analisis numerik dibangun di atas kalkulus, dan seksi ini mengumpulkan alat-alat kalkulus yang akan dipakai terus-menerus untuk menurunkan metode serta menaksir galatnya. Dua gagasan paling dasar adalah limit dan kekontinuan. Fungsi f memiliki limit L di x_0 bila untuk setiap $\varepsilon > 0$ ada $\delta > 0$ sehingga $|f(x) - L| < \varepsilon$ setiap kali $0 < |x - x_0| < \delta$. Fungsi f kontinu di x_0 bila $\lim_{x \to x_0} f(x) = f(x_0)$. Ada karakterisasi yang sangat berguna untuk komputasi: f kontinu di x_0 jika dan hanya jika $f(x_n) \to f(x_0)$ untuk setiap barisan $x_n \to x_0$. Kekontinuan adalah syarat minimal agar perilaku suatu fungsi dapat diprediksi, sebab fungsi yang melompat bisa melewati titik yang kita cari. Himpunan fungsi kontinu pada selang $[a, b]$ ditulis $C[a, b]$.

Syarat yang lebih kuat adalah keterdiferensialan: $f'(x_0) = \lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}$ ada. Fungsi yang terdiferensial di suatu titik pasti kontinu di titik itu, tetapi kebalikannya salah; $|x|$ kontinu di 0 namun tidak terdiferensial di sana. Himpunan fungsi yang memiliki n turunan kontinu pada X ditulis $C^n(X)$, dan yang memiliki turunan semua orde ditulis $C^\infty(X)$. Polinomial, fungsi rasional, trigonometri, eksponensial, dan logaritma berada di C^∞ pada daerah definisinya. Secara umum, makin mulus suatu fungsi, makin baik hampiran yang dapat dibuat untuknya.

Tiga teorema tentang fungsi kontinu dan terdiferensial menjadi fondasi penaksiran galat. Teorema Rolle menyatakan: bila $f \in C[a, b]$, terdiferensial pada (a, b), dan $f(a) = f(b)$, maka ada $c \in (a, b)$ dengan $f'(c) = 0$. Teorema Nilai Rata-rata (TNR) memperumumnya: bila $f \in C[a, b]$ dan terdiferensial pada (a, b), ada $c \in (a, b)$ sehingga

$$f'(c) = \frac{f(b) - f(a)}{b - a}.$$

Secara geometris, ada titik pada grafik yang garis singgungnya sejajar dengan tali busur yang menghubungkan kedua ujung. Teorema Nilai Ekstrem menjamin bahwa fungsi kontinu pada selang tertutup terbatas mencapai nilai minimum dan maksimum; bila f juga terdiferensial pada (a, b), nilai ekstrem itu terjadi di titik ujung atau di titik kritis, yaitu titik dengan $f' = 0$. Ini memberi resep baku untuk mencari ekstrem mutlak: hitung f di kedua ujung dan di semua titik kritis, lalu bandingkan.

Untuk integral, kita memakai integral Riemann $\int_a^b f(x)\,dx$ sebagai limit jumlah $\sum f(z_i)\Delta x_i$ ketika lebar partisi menuju nol; fungsi kontinu pada $[a, b]$ selalu terintegralkan. Teorema Nilai Rata-rata untuk Integral Terboboti: bila $f \in C[a, b]$, g terintegralkan pada $[a, b]$, dan $g(x)$ tidak berganti tanda pada $[a, b]$, maka ada $c \in (a, b)$ dengan $\int_a^b f(x)g(x)\,dx = f(c)\int_a^b g(x)\,dx$. Kasus $g \equiv 1$ memberi nilai rata-rata $f(c) = \frac{1}{b-a}\int_a^b f(x)\,dx$. Syarat tidak berganti tanda pada g penting, dan akan muncul saat menurunkan galat rumus integrasi numerik. Teorema Nilai Antara menyatakan bahwa fungsi $f \in C[a, b]$ mengambil setiap nilai K di antara $f(a)$ dan $f(b)$ pada suatu titik $c \in [a, b]$. Teorema ini adalah dasar metode bisection di Bab 2.

Hasil paling penting bagi analisis numerik adalah Teorema Taylor. Misalkan $f \in C^n[a, b]$, $f^{(n+1)}$ ada pada $[a, b]$, dan $x_0 \in [a, b]$. Maka untuk setiap $x \in [a, b]$ ada $\xi(x)$ di antara x_0 dan x dengan $f(x) = P_n(x) + R_n(x)$, di mana

$$P_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k, \qquad R_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)^{n+1}.$$

P_n disebut polinomial Taylor berderajat n dan R_n suku sisa (galat pemotongan). Titik $\xi(x)$ tidak diketahui, tetapi cukup untuk menaksir galat: bila $|f^{(n+1)}| \le M$ pada selang yang

Steven Paulus Siahaan

relevan, maka $|R_n(x)| \le \frac{M}{(n+1)!}|x-x_0|^{n+1}$. Dengan batas ini kita dapat menentukan derajat yang diperlukan untuk mencapai ketelitian tertentu sebelum menghitung apa pun. Bila $n \to \infty$ dan $R_n \to 0$, diperoleh deret Taylor; kasus $x_0 = 0$ disebut deret Maclaurin. Teorema Taylor adalah kasus umum TNR ($n = 0$ memberi $f(x) = f(x_0) + f'(\xi)(x - x_0)$), dan akan dipakai kembali untuk menurunkan rumus beda hingga (Bab 4), metode Newton (Bab 2), dan metode deret Taylor untuk persamaan diferensial (Bab 5).

Satu peringatan: polinomial Taylor sangat akurat dekat x_0 tetapi dapat memburuk jauh dari x_0, sehingga untuk hampiran pada selang lebar metode lain, seperti interpolasi di Bab 3, sering lebih tepat.

Konsep kunci

Limit dan kekontinuan — f kontinu di x_0 bila $\lim_{x \to x_0} f(x) = f(x_0)$; ekuivalen dengan $f(x_n) \to f(x_0)$ untuk setiap barisan $x_n \to x_0$. Kelas $C[a,b]$ memuat semua fungsi kontinu pada $[a,b]$.

Kelas C^n — Fungsi dengan n turunan kontinu. Keterdiferensialan mengimplikasikan kekontinuan, tetapi tidak sebaliknya. Makin tinggi n, makin baik hampiran yang bisa dijamin.

Teorema Rolle & Nilai Rata-rata — Fungsi kontinu pada $[a,b]$ dan terdiferensial pada (a,b) memiliki titik c dengan $f'(c) = \frac{f(b)-f(a)}{b-a}$. Rolle adalah kasus $f(a) = f(b)$.

Teorema Nilai Ekstrem — Fungsi kontinu pada selang tertutup mencapai maksimum dan minimum mutlak, di titik ujung atau titik kritis ($f' = 0$).

Teorema Nilai Antara — Fungsi kontinu pada $[a,b]$ mengambil setiap nilai di antara $f(a)$ dan $f(b)$. Ini menjamin adanya akar bila $f(a)f(b) < 0$.

Nilai rata-rata integral terboboti — Bila f kontinu dan g terintegralkan serta tidak berganti tanda, $\int_a fg = f(c)\int_a g$ untuk suatu $c \in (a,b)$.

Polinomial Taylor dan suku sisa — $f = P_n + R_n$ dengan $R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)^{n+1}$. Suku sisa inilah galat pemotongan, ditaksir dengan batas atas turunan ke-$(n+1)$.

Teorema Nilai Rata-rata

$$f'(c) = \frac{f(b)-f(a)}{b-a}, \quad c \in (a,b)$$

Polinomial Taylor

$$P_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k$$

Suku sisa Taylor (bentuk Lagrange)

$$R_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)^{n+1}$$

Batas galat Taylor

$$|R_n(x)| \le \frac{M}{(n+1)!}|x-x_0|^{n+1}, \quad M = \max|f^{(n+1)}|$$

Steven Paulus Siahaan

Nilai rata-rata integral

$$\frac{1}{b-a}\int_a^b f(x)\,dx = f(c), \quad c \in (a,b)$$

Contoh 1: Hampiran Taylor $\sqrt{1+x}$ dan batas galatnya

Gunakan polinomial Taylor derajat 2 di $x_0 = 0$ untuk $f(x) = \sqrt{1+x}$ guna menghampiri $\sqrt{1{,}2}$, lalu bandingkan galat sebenarnya dengan batas suku sisa.

Penyelesaian.

Turunan: $f'(x) = \frac{1}{2}(1+x)^{-1/2}$, $f''(x) = -\frac{1}{4}(1+x)^{-3/2}$, $f'''(x) = \frac{3}{8}(1+x)^{-5/2}$. Di $x_0 = 0$: $f(0) = 1$, $f'(0) = \frac{1}{2}$, $f''(0) = -\frac{1}{4}$. Maka

$$P_2(x) = 1 + \frac{x}{2} - \frac{x^2}{8}.$$

Untuk $x = 0{,}2$: $P_2(0{,}2) = 1 + 0{,}1 - 0{,}005 = 1.0950$, sedangkan $\sqrt{1{,}2} = 1.095445$. Galat sebenarnya $= 0.000445$.

Suku sisa: $R_2(x) = \frac{f'''(\xi)}{6}x^3 = \frac{1}{16}(1+\xi)^{-5/2}x^3$ dengan $0 < \xi < 0{,}2$. Karena $(1+\xi)^{-5/2} < 1$ untuk $\xi > 0$, maka $|R_2| < \frac{(0{,}2)^3}{16} = 0{,}0005$. Galat sebenarnya $0{,}000445$ memang di bawah batas itu.

Contoh 2: Ekstrem mutlak pada selang tertutup

Tentukan nilai maksimum dan minimum mutlak $f(x) = x^3 - 3x$ pada $[-2, 3]$.

Penyelesaian.

Turunannya $f'(x) = 3x^2 - 3 = 0$ menghasilkan titik kritis $x = \pm 1$, keduanya berada di dalam selang. Hitung f di titik kritis dan titik ujung:

$$f(-2) = -8 + 6 = -2,\ f(-1) = -1 + 3 = 2,\ f(1) = 1 - 3 = -2,\ f(3) = 27 - 9 = 18.$$

Jadi maksimum mutlak adalah 18 di $x = 3$ (titik ujung) dan minimum mutlak adalah -2, yang tercapai di dua tempat: $x = -2$ (titik ujung) dan $x = 1$ (titik kritis). Nilai $f(-1) = 2$ hanya maksimum lokal.

Contoh 3: Titik c pada Teorema Nilai Rata-rata dan batas iterasi Taylor

(a) Cari c pada TNR untuk $f(x) = \ln x$ di $[1, 3]$. (b) Berapa derajat minimum polinomial Maclaurin e^x agar galat $< 10^{-6}$ pada $[0, 1]$?

Penyelesaian.

(a) $\frac{f(3) - f(1)}{3 - 1} = \frac{\ln 3}{2} \approx 0.5493$ dan $f'(c) = \frac{1}{c}$. Maka $c = \frac{2}{\ln 3} \approx 1.8205$, yang memang terletak di $(1, 3)$.

(b) Untuk $x_0 = 0$ dan $x \in [0, 1]$, $|R_n(x)| \le \frac{e^{\xi}}{(n+1)!}x^{n+1} \le \frac{e}{(n+1)!}$. Dengan $e \approx 2{,}718$: $n = 8$ memberi $\frac{e}{9!} \approx 7.49e-06$ (belum cukup), $n = 9$ memberi $\frac{e}{10!} \approx 7.49e-07 < 10^{-6}$. Jadi derajat minimum $n = 9$.

Kesalahan umum

• Menganggap $\xi(x)$ pada suku sisa Taylor dapat dihitung. Titik itu hanya dijamin ada; yang dipakai dalam praktik adalah batas atas M untuk $|f^{(n+1)}|$ pada selang yang memuat x_0 dan x.

• Mengira kontinu berarti terdiferensial. $f(x)=|x|$ kontinu di 0 tetapi tidak punya turunan di sana, sehingga TNR dan Rolle tidak berlaku bila titik seperti itu berada di dalam selang.

• Lupa memeriksa syarat teorema: Rolle dan TNR memerlukan kekontinuan pada selang tertutup $[a,b]$ dan keterdiferensialan pada selang buka (a,b). Contohnya $f(x)=|x|$ di $[-1,1]$ memenuhi $f(-1)=f(1)$ tetapi tidak ada titik dengan $f'=0$.

• Memakai teorema nilai rata-rata integral terboboti ketika g berganti tanda. Misalnya $g(x)=x$ pada $[-1,1]$ dan $f(x)=x$: $\int fg=\frac{2}{3}$ tetapi $\int g=0$, sehingga tidak ada c yang memenuhi.

Pertanyaan diskusi

1. Mengapa Teorema Nilai Antara menjamin keberadaan akar tetapi tidak memberi cara menemukannya? Metode numerik apa yang mengubah jaminan itu menjadi algoritma?
2. Bagaimana Teorema Taylor dengan $n=0$ dan $n=1$ berhubungan dengan TNR dan hampiran linear (garis singgung)?
3. Jika $f^{(n+1)}$ tidak terbatas pada selang yang diinginkan, apa yang terjadi pada batas galat Taylor? Beri contoh fungsi.
4. Mengapa mencari ekstrem mutlak hanya perlu memeriksa titik ujung dan titik kritis? Apa yang terjadi bila f tidak terdiferensial di suatu titik dalam selang?

Latihan mandiri (soal buatan sendiri, dengan kunci)

1. Tentukan polinomial Maclaurin derajat 4 untuk $\cos x$, hitung hampiran $\cos 1$, dan beri batas galatnya.

Kunci: $P_4(x)=1-\frac{x^2}{2}+\frac{x^4}{24}$, sehingga $P_4(1)=\frac{13}{24}\approx 0.541667$. Nilai sebenarnya $\cos 1\approx 0.540302$, galat ≈ 0.001364. Karena turunan kelima $\cos$ dibatasi oleh 1 (yaitu $|\sin|\le 1$), $|R_4(1)|\le \frac{1}{5!}=0.00833$, yang menjamin hasil di atas (galat nyata lebih kecil dari batas).

2. Tunjukkan bahwa $f(x)=x^3+x-1$ memiliki akar di $[0,1]$ dan bahwa akar itu tunggal.

Kunci: $f(0)=-1<0$ dan $f(1)=1>0$ dengan f kontinu, sehingga Teorema Nilai Antara memberi akar di $(0,1)$. Misalkan ada dua akar; menurut Teorema Rolle ada titik di antaranya dengan $f'=0$. Tetapi $f'(x)=3x^2+1\ge 1>0$, kontradiksi. Jadi akarnya tunggal.

3. Cari c pada Teorema Nilai Rata-rata untuk $f(x)=x^3$ pada $[0,2]$.

Kunci: $\frac{f(2)-f(0)}{2}=4$ dan $f'(c)=3c^2=4$, sehingga $c=\frac{2}{\sqrt{3}}\approx 1.1547\in(0,2)$ (akar negatif dibuang karena di luar selang).

4. Hitung nilai rata-rata $f(x)=x^2$ pada $[0,3]$ dan tentukan titik c yang mencapainya.

Kunci: $\frac{1}{3}\int_0^3 x^2\,dx=\frac{1}{3}\cdot 9=3$. Maka $c^2=3$, $c=\sqrt{3}\approx 1.7321\in(0,3)$.

1.2 Round-off Errors and Computer Arithmetic

Rujukan buku: halaman 17

Kalkulus dan aljabar bekerja dengan bilangan real yang boleh memiliki ekspansi desimal tak hingga, sedangkan komputer hanya menyimpan sejumlah digit berhingga. Akibatnya hanya himpunan bagian kecil dari bilangan rasional yang dapat dinyatakan secara persis, dan bilangan seperti $\sqrt{3}$ atau $0{,}1$ selalu disimpan dalam bentuk hampiran. Galat yang timbul karena representasi dan operasi aritmetika berdigit berhingga ini disebut galat pembulatan (round-off error). Pada komputasi dengan jutaan operasi, galat kecil ini dapat menumpuk, sehingga memahami asal dan perilakunya sangat penting.

Komputer modern mengikuti standar IEEE 754. Untuk format presisi ganda (64 bit), satu bit adalah tanda s, 11 bit adalah karakteristik (eksponen berbias) c, dan 52 bit adalah pecahan f. Bilangan ternormalisasi bernilai

$$(-1)^{s}\, 2^{\,c-1023}\,(1+f), \qquad 0 \le f < 1.$$

Karena 52 bit biner setara dengan sekitar 16 digit desimal, presisi ganda memberi kira-kira 16 digit signifikan. Jarak antara 1 dan bilangan mesin berikutnya adalah $2^{-52} \approx 2{,}22 \times 10^{-16}$ (epsilon mesin). Bilangan yang magnitudonya lebih besar dari sekitar $1{,}80 \times 10^{308}$ menyebabkan overflow, sedangkan bilangan ternormalisasi positif terkecil adalah $2^{-1022} \approx 2{,}23 \times 10^{-308}$; yang lebih kecil dari itu mengalami underflow dan biasanya dijadikan nol. Perhatikan pula bahwa bilangan mesin tidak tersebar merata: setiap interval $[2^{m}, 2^{m+1}]$ memuat jumlah bilangan mesin yang sama, sehingga jaraknya makin lebar untuk bilangan yang makin besar.

Untuk menganalisis pola galat dengan lebih mudah, kita memakai model desimal. Bilangan mesin k digit berbentuk $\pm 0{,}d_1 d_2 \ldots d_k \times 10^{n}$ dengan $d_1 \neq 0$. Bilangan real $y = 0{,}d_1 d_2 \ldots d_k d_{k+1} \ldots \times 10^{n}$ diubah menjadi bentuk floating-point $\mathrm{fl}(y)$ dengan dua cara. Pemotongan (chopping) membuang semua digit setelah digit ke-k. Pembulatan (rounding) menambah satu pada d_k bila $d_{k+1} \ge 5$, dan jika tidak memotong saja. Kedua cara menghasilkan galat relatif yang terbatas independen dari besar bilangannya: $\left|\frac{y - \mathrm{fl}(y)}{y}\right| \le 10^{1-k}$ untuk pemotongan dan $\le 0{,}5 \times 10^{1-k}$ untuk pembulatan.

Galat dinyatakan dengan dua ukuran. Jika p^{*} menghampiri p, galat mutlak adalah $|p - p^{*}|$ dan galat relatif adalah $\frac{|p-p^{*}|}{|p|}$ untuk $p \neq 0$. Galat relatif biasanya lebih bermakna karena memperhitungkan ukuran bilangan: galat mutlak 1 pada bilangan 10^{6} jauh lebih kecil artinya daripada pada bilangan 3. Banyak digit signifikan t didefinisikan sebagai bilangan bulat taknegatif terbesar yang memenuhi $\frac{|p-p^{*}|}{|p|} \le 5 \times 10^{-t}$.

Aritmetika mesin tidak persis aritmetika real. Dalam model buku, operasi mesin dituliskan sebagai $x \oplus y = \mathrm{fl}(\mathrm{fl}(x) + \mathrm{fl}(y))$, dan seterusnya untuk $\ominus$, $\otimes$, $\oslash$: kedua operan dibulatkan dahulu, dioperasikan secara eksak, lalu hasilnya dibulatkan lagi. Dua konsekuensi paling berbahaya adalah (i) pembatalan digit signifikan (cancellation) ketika dua bilangan hampir sama dikurangkan, sehingga hanya sedikit digit yang tersisa, dan (ii) pembesaran galat mutlak saat membagi dengan bilangan kecil atau mengalikan dengan bilangan besar. Galat relatif tidak berubah oleh pembagian itu, tetapi galat mutlaknya diperbesar.

Kabar baiknya, banyak kehilangan ketelitian dapat dihindari dengan menata ulang rumus secara matematis ekuivalen. Contoh klasik adalah rumus kuadrat: bila $b^{2} \gg 4ac$, salah satu akar melibatkan pengurangan dua bilangan hampir sama. Dengan merasionalkan pembilang diperoleh rumus alternatif $x = \frac{-2c}{b + \sqrt{b^{2} - 4ac}}$ (untuk $b > 0$) yang bebas dari pembatalan. Contoh lain adalah bentuk bersarang (nested/Horner) untuk polinomial, yang mengurangi jumlah perkalian sehingga mengurangi galat sekaligus waktu hitung. Pelajaran umumnya:

Web by: Steven Paulus Siahaan

pikirkan dahulu sebelum menghitung. Pembahasan ini menjadi dasar untuk Seksi 1.3 (stabilitas algoritma) dan terus muncul pada bab-bab berikutnya, misalnya eliminasi Gauss dan turunan numerik.

Konsep kunci

Galat pembulatan — Galat yang timbul karena komputer hanya menyimpan bilangan dengan jumlah digit berhingga, sehingga bilangan real dan hasil operasi hanya dihampiri.

Bilangan mesin IEEE presisi ganda — Bentuk $(-1)^s 2^{c-1023}(1+f)$ dengan 1 bit tanda, 11 bit karakteristik, dan 52 bit pecahan; memberi sekitar 16 digit desimal signifikan.

Overflow dan underflow — Overflow terjadi bila magnitudo hasil melebihi bilangan mesin terbesar; underflow bila lebih kecil dari yang ternormalisasi terkecil dan biasanya dijadikan nol.

Pemotongan dan pembulatan — Dua cara membentuk $\mathrm{fl}(y)$ dengan k digit: membuang digit sisa, atau menambah satu pada digit terakhir bila digit berikutnya ≥ 5. Pembulatan lebih akurat.

Galat mutlak dan relatif — $|p-p^*|$ dan $\frac{|p-p^*|}{|p|}$. Galat relatif lebih bermakna untuk menilai mutu hampiran karena memperhitungkan ukuran nilai sebenarnya.

Digit signifikan — p^* menghampiri p hingga t digit signifikan bila t adalah bilangan bulat terbesar dengan galat relatif $\leq 5\times 10^{-t}$.

Pembatalan digit signifikan — Mengurangkan dua bilangan yang hampir sama menghilangkan digit depan yang sama, sehingga hasil hanya memuat sedikit digit signifikan.

Bentuk bersarang — Menulis polinomial sebagai $((a_n x + a_{n-1})x + \cdots)x + a_0$ sehingga memerlukan n perkalian dan n penjumlahan, lebih sedikit operasi dan lebih sedikit galat.

Bilangan mesin presisi ganda

$$(-1)^s\, 2^{\,c-1023}\,(1+f)$$

Galat relatif pemotongan dan pembulatan (k digit)

$$\left|\frac{y-\mathrm{fl}(y)}{y}\right| \leq 10^{1-k}\ \text{(potong)}, \qquad \left|\frac{y-\mathrm{fl}(y)}{y}\right| \leq 0{,}5\times 10^{1-k}\ \text{(bulat)}$$

Galat mutlak dan relatif

$$E_{\text{abs}} = |p-p^*|, \qquad E_{\text{rel}} = \frac{|p-p^*|}{|p|} \quad (p\neq 0)$$

Digit signifikan

$$\frac{|p-p^*|}{|p|} \leq 5\times 10^{-t}$$

Aritmetika berdigit berhingga

$$x \oplus y = \mathrm{fl}(\mathrm{fl}(x)+\mathrm{fl}(y)), \qquad x \otimes y = \mathrm{fl}(\mathrm{fl}(x)\times \mathrm{fl}(y))$$

Rumus kuadrat alternatif (untuk $b > 0$)

$$x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}}, \qquad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

Contoh 1: Pemotongan, pembulatan, dan galatnya

Tentukan bentuk floating-point lima digit (pemotongan dan pembulatan) dari $e = 2{,}718281828\ldots$, lalu hitung galat mutlak dan relatifnya. Lakukan pula untuk $y = 0{,}00234567$ dengan tiga digit.

Penyelesaian.

Tulis $e = 0{,}2718281828\ldots \times 10^{1}$.

Pemotongan lima digit: $\mathrm{fl}(e) = 0{,}27182 \times 10^{1} = 2{,}7182$, galat mutlak $\approx 8{,}18 \times 10^{-5}$, galat relatif $\approx 3{,}01 \times 10^{-5}$.

Pembulatan lima digit: digit keenam adalah 8, jadi naik: $\mathrm{fl}(e) = 0{,}27183 \times 10^{1} = 2{,}7183$, galat mutlak $\approx 1{,}82 \times 10^{-5}$, galat relatif $\approx 6{,}68 \times 10^{-6}$.

Untuk $y = 0{,}234567 \times 10^{-2}$ dengan tiga digit: pemotongan memberi $0{,}234 \times 10^{-2}$ dengan galat relatif $\approx 2{,}42 \times 10^{-3}$, pembulatan memberi $0{,}235 \times 10^{-2}$ dengan galat relatif $\approx 1{,}85 \times 10^{-3}$.

Semua galat relatif berada di bawah batas teoretis: 10^{-4} (potong, $k = 5$), $0{,}5 \times 10^{-4}$ (bulat, $k = 5$), 10^{-2} (potong, $k = 3$), $0{,}5 \times 10^{-2}$ (bulat, $k = 3$).

Contoh 2: Pembatalan pada rumus kuadrat

Selesaikan $x^2 + 48{,}30x + 1 = 0$ dengan aritmetika pembulatan empat digit memakai rumus kuadrat biasa, lalu dengan rumus alternatif untuk akar kecil. Akar sebenarnya $x_1 \approx -0{,}020713$ dan $x_2 \approx -48{,}2793$.

Penyelesaian.

Hitung dahulu: $b^2 = 2332{,}89 \to 2333$, $4ac = 4{,}000$, selisih $2333 - 4 = 2329$, dan $\sqrt{2329} \approx 48{,}26$ (empat digit).

Rumus biasa untuk akar kecil: $x_1 = \frac{-48{,}30 + 48{,}26}{2} = \frac{-0{,}04000}{2} = -0{,}02000$. Galat relatifnya $\approx 3{,}4 \times 10^{-2}$; hanya satu digit yang benar karena $-48{,}30 + 48{,}26$ membatalkan digit depan.

Untuk akar besar: $x_2 = \frac{-48{,}30 - 48{,}26}{2} = -48{,}28$, galat relatif $\approx 1{,}5 \times 10^{-5}$, sangat baik karena tidak ada pengurangan bilangan yang hampir sama.

Rumus alternatif: $x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}} = \frac{-2{,}000}{48{,}30 + 48{,}26} = \frac{-2{,}000}{96{,}56} = -0{,}02071$, galat relatif hanya $\approx 1{,}4 \times 10^{-4}$.

Kesimpulan: untuk $b > 0$ pakai rumus biasa untuk x_2 dan rumus alternatif untuk x_1.

Contoh 3: Evaluasi polinomial: bentuk biasa lawan bersarang

Hitung $f(x) = x^3 - 4{,}2x^2 + 2{,}7x + 0{,}9$ di $x = 3{,}17$ dengan aritmetika pembulatan tiga digit, (a) bentuk biasa dan (b) bentuk bersarang. Nilai eksak $f(3{,}17) = -0{,}891367$.

Penyelesaian.

(a) Bentuk biasa: $x^2 = 10{,}0489 \to 10{,}0$; $x^3 = 10{,}0 \times 3{,}17 = 31{,}7$; $4{,}2x^2 = 42{,}0$; $2{,}7x = 8{,}559 \to 8{,}56$. Jumlah dari kiri ke kanan: $31{,}7 - 42{,}0 = -10{,}3$; $-10{,}3 + 8{,}56 = -1{,}74$; $-1{,}74 + 0{,}9 = -0{,}84$. Galat relatif $\frac{|-0{,}891367 + 0{,}84|}{0{,}891367} \approx 5{,}8 \times 10^{-2}$.

(b) Bentuk bersarang $f(x) = ((x - 4{,}2)x + 2{,}7)x + 0{,}9$: $3{,}17 - 4{,}2 = -1{,}03$; $-1{,}03 \times 3{,}17 = -3{,}2651 \to -3{,}27$; $-3{,}27 + 2{,}7 = -0{,}57$; $-0{,}57 \times 3{,}17 = -1{,}8069 \to -1{,}81$; $-1{,}81 + 0{,}9 = -0{,}91$. Galat relatif $\approx 2{,}1 \times 10^{-2}$.

Bentuk bersarang memakai 2 perkalian dan 3 penjumlahan (bentuk biasa: 4 perkalian dan 3 penjumlahan), dan pada contoh ini galatnya sekitar sepertiga kali lebih kecil. Keunggulan ini tidak dijamin pada setiap data, tetapi jumlah operasi yang lebih sedikit umumnya menolong.

Kesalahan umum

• Mengira galat mutlak yang kecil berarti hampiran bagus. Galat mutlak 10^{-3} sangat baik untuk bilangan 10^{6} tetapi buruk untuk bilangan 10^{-4}; selalu pertimbangkan galat relatif.

• Menyamakan banyak angka desimal yang cocok dengan banyak digit signifikan. Menurut definisi, digit signifikan ditentukan oleh galat relatif $\le 5 \times 10^{-t}$, bukan sekadar digit yang tampak sama.

• Mengurangkan dua bilangan yang hampir sama tanpa memikirkan ulang rumusnya. Digit depan saling meniadakan dan hasilnya hanya memuat sedikit digit signifikan; rumus sering dapat ditata ulang (merasionalkan, identitas trigonometri, dan sebagainya).

• Mengharapkan $0{,}1 + 0{,}2 = 0{,}3$ persis di komputer. Bilangan $0{,}1$ tidak punya representasi biner berhingga, sehingga pada presisi ganda $0{,}1 + 0{,}2$ menghasilkan $0{,}30000000000000004$; bandingkan bilangan real dengan toleransi, bukan dengan tanda sama.

• Menganggap pembulatan setiap langkah antara sama saja dengan membulatkan hanya hasil akhir. Pada model $\oplus$, $\otimes$ pembulatan terjadi setiap operasi, sehingga urutan dan jumlah operasi memengaruhi hasil.

Pertanyaan diskusi

1. Mengapa batas galat relatif pembulatan k digit tidak bergantung pada besar bilangan, padahal jarak antarbilangan mesin makin lebar untuk bilangan besar?
2. Mengapa bentuk bersarang lebih baik daripada bentuk biasa dari sudut pandang jumlah operasi? Adakah kasus ketika keduanya memberi hasil yang sama?
3. Dalam rumus kuadrat dengan $b > 0$ dan $b^2 \gg 4ac$, mengapa x_2 aman sedangkan x_1 tidak? Bagaimana jika $b < 0$?
4. Mengapa penjumlahan bilangan sangat besar dan sangat kecil (seperti $u \oplus v$) memberi galat mutlak besar tetapi galat relatif kecil, sedangkan pengurangan bilangan hampir sama memberi galat relatif besar?

Latihan mandiri (soal buatan sendiri, dengan kunci)

1. Bilangan mesin presisi ganda $0\ 10000000010\ 1010000\ldots0$ (tanda 0, karakteristik 10000000010, lalu pecahan 1010 diikuti nol). Tentukan nilai desimalnya.

Kunci: Karakteristik $c = 2^{10} + 2^1 = 1026$, jadi eksponen $c - 1023 = 3$. Pecahan $f = \frac{1}{2} + \frac{1}{8} = 0{,}625$. Nilainya $2^3(1 + 0{,}625) = 13$.

2. Misalkan $p = 2{,}718281828$ dihampiri oleh $p^* = 2{,}7$. Hitung galat mutlak, galat relatif, dan banyak digit signifikan t.

Kunci: Galat mutlak $\approx 0{,}018282$, galat relatif $\approx 6{,}73 \times 10^{-3}$. Karena $6{,}73 \times 10^{-3} \le 5 \times 10^{-2}$ tetapi tidak $\le 5 \times 10^{-3}$, maka $t = 2$.

3. Hitung $\sqrt{250001} - \sqrt{250000}$ dengan aritmetika pembulatan empat digit secara langsung, lalu dengan bentuk $\frac{1}{\sqrt{250001} + \sqrt{250000}}$. Bandingkan dengan nilai sebenarnya $\approx 9{,}99999 \times 10^{-4}$.

Kunci: Langsung: $\sqrt{250001} \approx 500{,}0010 \to 500{,}0$ dan $\sqrt{250000} = 500{,}0$, selisih 0 (galat relatif 100%). Bentuk alternatif: $\frac{1}{500{,}0 + 500{,}0} = \frac{1}{1000} = 0{,}001000$, galat relatif sekitar 10^{-6}.

4. Berapa perkalian dan penjumlahan yang diperlukan untuk menghitung polinomial derajat 5 dalam bentuk bersarang? Tuliskan $p(x) = 2x^5 - 3x^3 + x - 7$ dalam bentuk bersarang.

Kunci: 5 perkalian dan 5 penjumlahan. $p(x) = ((((2x + 0)x - 3)x + 0)x + 1)x - 7$, yaitu koefisien $2, 0, -3, 0, 1, -7$ diproses dengan skema Horner.

Web by: Steven Paulus Siahaan

1.3 Algorithms and Convergence

Rujukan buku: halaman 32

Metode numerik pada dasarnya adalah algoritma: prosedur yang menjelaskan tanpa ambiguitas serangkaian langkah berhingga dalam urutan tertentu untuk menyelesaikan atau menghampiri suatu masalah. Algoritma biasanya ditulis dalam pseudokode yang menyebutkan masukan (INPUT), keluaran (OUTPUT), dan langkah-langkah bernomor. Pengulangan dapat dikendalikan oleh pencacah (For $i = 1, \ldots, n$) atau oleh kondisi (While ... do), dan setiap algoritma numerik sebaiknya memiliki kriteria berhenti yang tidak bergantung pada metodenya sendiri, misalnya batas jumlah iterasi maksimum, agar tidak terjadi perulangan tak hingga ketika metode gagal mencapai toleransi.

Contoh sederhana: menjumlahkan N bilangan dengan akumulator SUM yang diberi nilai awal 0 lalu ditambah satu per satu. Contoh yang lebih bermakna adalah menentukan banyak suku minimum dari suatu deret agar galatnya di bawah toleransi tanpa memakai suku sisa Taylor. Untuk deret berganti tanda dengan suku yang magnitudonya menurun menuju nol, selisih antara jumlah deret dan jumlah parsial ke-N tidak melebihi magnitudo suku ke-$(N+1)$. Jadi algoritma cukup menjumlahkan suku demi suku sampai suku berikutnya lebih kecil dari toleransi.

Setelah algoritma ada, kita perlu menilai mutunya. Kriteria pertama adalah kestabilan: perubahan kecil pada data awal seharusnya hanya menyebabkan perubahan kecil pada hasil akhir. Algoritma yang memenuhinya disebut stabil, dan yang tidak disebut tidak stabil; sebagian algoritma stabil hanya untuk data tertentu dan disebut stabil bersyarat. Untuk membuat gagasan ini konkret, misalkan galat awal bermagnitudo $E_0 > 0$ dan setelah n operasi galatnya E_n. Pertumbuhan galat disebut linear bila $E_n \approx C\,n\,E_0$ dan eksponensial bila $E_n \approx C^n E_0$ dengan $C > 1$. Pertumbuhan linear biasanya tak terhindarkan dan masih dapat diterima bila C dan E_0 kecil, sedangkan pertumbuhan eksponensial harus dihindari karena C^n membesar dengan cepat, berapa pun kecilnya E_0. Karena itu algoritma dengan pertumbuhan galat linear dikatakan stabil, dan yang eksponensial tidak stabil.

Contoh klasik terjadi pada persamaan beda (rekurensi) linear orde dua. Solusi umum rekurensi $p_n = \alpha p_{n-1} - \beta p_{n-2}$ adalah kombinasi $c_1 r_1^{\,n} + c_2 r_2^{\,n}$ dengan r_1, r_2 akar persamaan karakteristik. Bila salah satu akar memiliki magnitudo lebih dari 1 sementara solusi yang kita inginkan hanya memakai akar yang lebih kecil, maka galat pembulatan sekecil apa pun pada nilai awal akan memunculkan komponen yang membesar secara eksponensial dan akhirnya menenggelamkan solusi yang benar. Bila akar-akarnya berpangkat netral (misalnya akar ganda 1), galat hanya tumbuh linear. Menggunakan presisi lebih tinggi (presisi ganda) memperkecil E_0 tetapi tidak menghilangkan pertumbuhan eksponensial, dan memerlukan waktu hitung lebih besar. Aritmetika selang (interval arithmetic) menyimpan batas bawah dan atas pada tiap langkah sehingga hasil akhirnya adalah selang yang pasti memuat nilai sebenarnya, tetapi selang sering menjadi terlalu lebar untuk berguna.

Bagian terakhir seksi ini memperkenalkan laju konvergensi. Misalkan $\{\beta_n\}$ konvergen ke nol dan $\{\alpha_n\}$ konvergen ke α. Bila ada konstanta $K > 0$ sehingga $|\alpha_n - \alpha| \le K|\beta_n|$ untuk n besar, kita tulis $\alpha_n = \alpha + O(\beta_n)$ (dibaca "big oh dari β_n"). Hampir selalu pembandingnya $\beta_n = 1/n^p$ dan kita mencari $p > 0$ terbesar. Definisi serupa berlaku untuk fungsi: bila $\lim_{h \to 0} G(h) = 0$, $\lim_{h \to 0} F(h) = L$, dan $|F(h) - L| \le K|G(h)|$ untuk h cukup kecil, maka $F(h) = L + O(G(h))$, dengan pembanding yang lazim $G(h) = h^{\,p}$. Teorema Taylor dari Seksi 1.1 adalah alat utama untuk menunjukkan orde seperti ini, karena suku sisa memberi pangkat h pertama yang tersisa. Makin besar p, makin cepat hampiran menuju limitnya.

Web by: Steven Paulus Siahaan

Notasi O akan dipakai sepanjang buku: orde galat pemotongan rumus beda hingga (Bab 4), orde metode integrasi dan metode persamaan diferensial (Bab 4 dan 5), serta laju konvergensi metode pencari akar (Bab 2). Perhatikan bahwa O hanya memberi batas atas, dan konstanta K tidak ditentukan; dua barisan dengan orde sama dapat memiliki konstanta yang sangat berbeda.

Konsep kunci

Algoritma — Prosedur yang menjelaskan secara tak ambigu serangkaian langkah berhingga dalam urutan tertentu untuk menyelesaikan atau menghampiri masalah; dilengkapi kriteria berhenti agar tidak berulang tak hingga.

Kestabilan algoritma — Algoritma stabil bila perubahan kecil pada data awal hanya menyebabkan perubahan kecil pada hasil. Bila hanya berlaku untuk data tertentu, disebut stabil bersyarat.

Pertumbuhan galat linear — $E_n \approx CnE_0$. Biasanya tak terhindarkan dan dapat diterima bila C dan E_0 kecil; algoritma dengan pertumbuhan ini dianggap stabil.

Pertumbuhan galat eksponensial — $E_n \approx C^n E_0$ dengan $C > 1$. Menghancurkan hasil walau E_0 sangat kecil; algoritma dengan pertumbuhan ini dikatakan tidak stabil.

Laju konvergensi barisan — $\alpha_n = \alpha + O(\beta_n)$ bila $|\alpha_n - \alpha| \le K|\beta_n|$ untuk n besar; umumnya $\beta_n = 1/n^p$ dan dicari p terbesar.

Laju konvergensi fungsi — $F(h) = L + O(G(h))$ bila $|F(h) - L| \le K|G(h)|$ untuk h cukup kecil; biasanya $G(h) = h^p$ dan dicari p terbesar.

Kriteria berhenti — Aturan yang menghentikan iterasi, misalnya suku berikutnya lebih kecil dari toleransi atau jumlah iterasi mencapai batas, supaya algoritma pasti berakhir.

Aritmetika selang — Menyimpan batas bawah dan atas pada tiap langkah sehingga hasil akhir berupa selang yang pasti memuat nilai sebenarnya, tetapi selang dapat menjadi terlalu lebar.

Sisa deret berganti tanda

$$|A - A_N| \le |a_{N+1}|, \qquad A_N = \sum_{n=1}^{N} a_n$$

Pertumbuhan galat linear

$$E_n \approx C\,n\,E_0$$

Pertumbuhan galat eksponensial

$$E_n \approx C^n E_0, \qquad C > 1$$

Laju konvergensi barisan

$$|\alpha_n - \alpha| \le K\,|\beta_n| \;\Rightarrow\; \alpha_n = \alpha + O(\beta_n), \qquad \beta_n = \tfrac{1}{n^p}$$

Laju konvergensi fungsi

$$|F(h) - L| \le K\,|G(h)| \;\Rightarrow\; F(h) = L + O(G(h)), \qquad G(h) = h^p$$

Algoritma: Banyak suku deret berganti tanda untuk toleransi TOL

1. INPUT: nilai x, toleransi TOL, batas suku maksimum M. OUTPUT: banyak suku N atau pesan gagal. Contoh deret: $\arctan x = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{2k+1}$ dengan $0 < x \leq 1$ (suku menurun menuju nol).
2. Set $k = 0$, SUM $= 0$, TERM $= x$ (suku ke-0, tanpa tanda).
3. Selama $k < M$ kerjakan langkah 4 sampai 6.
4. Set SUM = SUM $+(-1)^k \cdot$ TERM (tambahkan suku berganti tanda); hitung suku berikutnya NEXT $= \frac{x^{2k+3}}{2k+3}$.
5. Jika NEXT $<$ TOL maka OUTPUT $N = k+1$ suku dan SUM; berhenti (galat $\leq$ NEXT $<$ TOL).
6. Set TERM = NEXT dan $k = k+1$.
7. Bila $k = M$ tanpa memenuhi toleransi, OUTPUT 'Metode gagal'; berhenti.

Contoh 1: Banyak suku deret arctan

Gunakan algoritma di atas untuk $x = \frac{1}{2}$ dan TOL $= 10^{-4}$. Berapa suku yang diperlukan, dan berapa galat sebenarnya terhadap $\arctan \frac{1}{2} = 0{,}4636476$?

Penyelesaian.

Suku ke-k (sebelum tanda) adalah $\frac{(1/2)^{2k+1}}{2k+1}$:
$0{,}500000;\ 0{,}041667;\ 0{,}006250;\ 0{,}001116;\ 0{,}000217;\ 0{,}0000444;\ \ldots$

Jumlah parsial: $S_0 = 0{,}500000$, $S_1 = 0{,}458333$, $S_2 = 0{,}464583$, $S_3 = 0{,}463467$, $S_4 = 0{,}463684$.

Suku NEXT setelah $k = 3$ adalah $2{,}17 \times 10^{-4} > 10^{-4}$, belum berhenti. Setelah $k = 4$ NEXT $= 4{,}44 \times 10^{-5} < 10^{-4}$, algoritma berhenti dengan $N = 5$ suku dan hampiran $0{,}463684$.

Galat sebenarnya $|0{,}4636476 - 0{,}463684| \approx 3{,}67 \times 10^{-5}$, di bawah NEXT $= 4{,}44 \times 10^{-5}$ sesuai $|A - A_N| \leq |a_{N+1}|$. Dengan TOL $= 10^{-5}$ diperlukan $N = 6$ suku (jumlah $0{,}463640$, galat $7{,}7 \times 10^{-6}$).

Contoh 2: Rekurensi tidak stabil lawan stabil

Rekurensi $p_n = \frac{37}{6}p_{n-1} - p_{n-2}$ memiliki solusi $p_n = 6^{-n}$ untuk $p_0 = 1$, $p_1 = \frac{1}{6}$. Misalkan p_1 salah sebesar $\delta = 10^{-6}$ (galat awal), perhitungan selanjutnya eksak. Hitung galat e_n untuk $n = 1, \ldots, 8$ dan bandingkan dengan rekurensi $p_n = 2p_{n-1} - p_{n-2}$ dengan galat awal δ yang sama pada p_1.

Penyelesaian.

Akar karakteristik $r^2 - \frac{37}{6}r + 1 = 0$ adalah $r = 6$ dan $r = \frac{1}{6}$. Galat memenuhi rekurensi yang sama dengan $e_0 = 0$, $e_1 = \delta$, sehingga $e_n = \delta\,\frac{6^n - 6^{-n}}{6 - \frac{1}{6}}$.

Nilai: $e_1 = 1{,}0 \times 10^{-6}$, $e_2 = 6{,}2 \times 10^{-6}$, $e_3 = 3{,}7 \times 10^{-5}$, $e_4 = 2{,}2 \times 10^{-4}$, $e_5 = 1{,}3 \times 10^{-3}$, $e_6 = 8{,}0 \times 10^{-3}$, $e_7 = 4{,}8 \times 10^{-2}$, $e_8 = 0{,}29$.

Bandingkan dengan nilai benar $p_8 = 6^{-8} \approx 6 \times 10^{-7}$: galat $0{,}29$ jauh melampaui nilai yang dihitung; hasil tidak berarti. Galat membesar kira-kira 6 kali lipat tiap langkah, yaitu eksponensial dengan $C = 6$: algoritma tidak stabil.

Untuk $p_n = 2p_{n-1} - p_{n-2}$ akarnya ganda $r = 1$, sehingga $e_n = n\delta$: $e_8 = 8 \times 10^{-6}$. Pertumbuhan linear, algoritma stabil.

Contoh 3: Orde konvergensi barisan dan fungsi

(a) Tentukan orde konvergensi $\alpha_n = \frac{2n+1}{n^2+n}$ dan $\hat{\alpha}_n = \frac{5n-2}{n^4}$ menuju nol. (b) Tunjukkan $\sin h = h + O(h^3)$.

Penyelesaian.

(a) Karena $2n + 1 \le 3n$ dan $n^2 + n \ge n^2$, maka $|\alpha_n| \le \frac{3n}{n^2} = \frac{3}{n}$, jadi $\alpha_n = 0 + O(1/n)$. Ini tidak dapat diperbaiki: $n\alpha_n \to 2$ ($n\alpha_n$ = 1,5; 1,67; 1,75; 1,91; 1,99 untuk $n = 1, 2, 3, 10, 100$). Sedangkan $5n - 2 \le 5n$ memberi $|\hat{\alpha}_n| \le \frac{5n}{n^4} = \frac{5}{n^3}$, jadi $\hat{\alpha}_n = 0 + O(1/n^3)$ ($n^3\hat{\alpha}_n \to 5$). Barisan kedua jauh lebih cepat: pada $n = 100$, $\alpha_n \approx 0{,}0199$ sedangkan $\hat{\alpha}_n \approx 5 \times 10^{-6}$.

(b) Teorema Taylor untuk $\sin h$ derajat 2 di sekitar 0: $\sin h = h - \frac{\cos\xi}{6}h^3$ untuk suatu ξ di antara 0 dan h. Maka $|\sin h - h| \le \frac{1}{6}|h|^3$, dengan $K = \frac{1}{6}$, sehingga $\sin h = h + O(h^3)$. Numerik: $\frac{\sin h - h}{h^3}$ bernilai $-0{,}1666$ di $h = 0{,}1$ dan $-0{,}16667$ di $h = 0{,}01$, mendekati $-\frac{1}{6}$.

Kesalahan umum

• Menganggap algoritma yang stabil pasti memberi hasil akurat. Kestabilan hanya berarti galat tidak membesar berlebihan; hasil tetap bisa kurang akurat bila masalahnya sendiri peka atau galat awalnya besar.

• Mengira presisi ganda menyelesaikan ketidakstabilan. Presisi lebih tinggi hanya memperkecil E_0; pertumbuhan C^n tetap terjadi, hanya tertunda beberapa langkah.

• Menafsirkan $O(h^p)$ sebagai kesamaan atau menyamakan konstanta K. Notasi O hanya memberi batas atas dengan K tak ditentukan; dua metode berorde sama dapat berbeda jauh dalam ketelitian praktis.

• Mencari orde konvergensi dengan menyatakannya terlalu kecil, misalnya menyebut $\frac{5n-2}{n^4} = O(1/n)$. Pernyataan itu benar tetapi tidak informatif; yang diminta adalah p terbesar.

• Menghentikan deret berganti tanda dengan membandingkan sembarang suku dengan toleransi. Batas $|A - A_N| \le |a_{N+1}|$ hanya berlaku bila suku-suku berganti tanda dan magnitudonya menurun menuju nol.

Pertanyaan diskusi

1. Mengapa pertumbuhan galat linear diterima sebagai tak terhindarkan, sedangkan eksponensial tidak? Kapan galat linear pun dapat menjadi masalah?
2. Pada rekurensi orde dua, bagaimana menentukan apakah suatu solusi tertentu dapat dihitung secara stabil dari nilai awalnya? Peran apa yang dimainkan akar karakteristik?
3. Apa arti $\alpha_n = \alpha + O(1/n^2)$ dibandingkan $\alpha_n = \alpha + O(1/n)$ bagi jumlah iterasi yang diperlukan untuk ketelitian tertentu?
4. Mengapa teorema Taylor adalah alat alami untuk menentukan orde $O(h^p)$ dari suatu hampiran fungsi? Apa yang menentukan p terbesar?

Latihan mandiri (soal buatan sendiri, dengan kunci)

1. Tentukan p terbesar sehingga $\frac{3n+1}{n^3} = 0 + O(1/n^p)$ dan beri konstanta K yang berlaku untuk semua $n \ge 1$.

Kunci: $3n+1 \le 4n$ untuk $n \ge 1$, sehingga $\frac{3n+1}{n^3} \le \frac{4}{n^2}$: $p = 2$ dengan $K = 4$. Tidak bisa $p = 3$ karena $n^2 \cdot \frac{3n+1}{n^3} \to 3 \ne 0$ (jadi barisan tidak lebih kecil dari K/n^3 untuk K tetap).

2. Rekurensi $p_n = \frac{5}{2}p_{n-1} - p_{n-2}$ (akar 2 dan $\frac{1}{2}$) dipakai untuk menghitung $p_n = 2^{-n}$ dari $p_0 = 1$, $p_1 = \frac{1}{2}$. Apakah stabil? Jelaskan.

Kunci: Tidak stabil. Galat awal δ pada p_1 menghasilkan $e_n = \delta\frac{2^n - 2^{-n}}{2 - \frac{1}{2}}$, tumbuh eksponensial dengan $C = 2$ (berlipat dua tiap langkah), sedangkan solusi yang diinginkan 2^{-n} mengecil, sehingga cepat tertelan galat.

3. Gunakan algoritma suku-berganti-tanda untuk $e^{-1} = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!}$ dengan TOL $= 10^{-3}$. Berapa suku yang diperlukan dan berapa jumlahnya?

Kunci: Suku: $1,\ 1,\ \frac{1}{2},\ \frac{1}{6},\ \frac{1}{24},\ \frac{1}{120},\ \frac{1}{720} \approx 0{,}00139,\ \frac{1}{5040} \approx 0{,}000198$. Suku berikutnya pertama yang $< 10^{-3}$ adalah $\frac{1}{5040}$ ($k = 7$), jadi jumlahkan $k = 0, \ldots, 6$: $N = 7$ suku, jumlah $\approx 0{,}368056$. Nilai sebenarnya $0{,}367879$, galat $1{,}8 \times 10^{-4} < 1{,}98 \times 10^{-4}$.

4. Tunjukkan $e^h - 1 - h = O(h^2)$ untuk $|h| \le 1$.

Kunci: Teorema Taylor derajat 1: $e^h = 1 + h + \frac{e^\xi}{2}h^2$ dengan ξ di antara 0 dan h. Karena $|\xi| \le 1$, $e^\xi \le e$, jadi $|e^h - 1 - h| \le \frac{e}{2}h^2$: $K = \frac{e}{2}$ dan $p = 2$. Numerik: $\frac{e^h - 1 - h}{h^2} \approx 0{,}517$ di $h = 0{,}1$ dan $0{,}5017$ di $h = 0{,}01$, menuju $\frac{1}{2}$.

Web by: Steven Paulus Siahaan

1.4 Numerical Software

Rujukan buku: halaman 41

Program yang kita tulis untuk latihan di kuliah ini disebut program khusus (special-purpose): ia cukup untuk menyelesaikan contoh dan soal, tetapi tidak dirancang menghadapi segala kondisi buruk yang mungkin muncul. Perangkat lunak numerik yang dipakai di dunia nyata, misalnya pustaka LAPACK, IMSL, NAG, atau fungsi bawaan MATLAB, bersifat serba guna (general-purpose). Perbedaannya bukan pada rumus dasar, melainkan pada kehati-hatian: program serba guna memperhitungkan galat pembulatan, luapan (overflow), dan kehabisan (underflow), serta menyatakan dengan jelas rentang masukan yang masih menghasilkan ketelitian yang dijanjikan.

Contoh yang paling mudah dipahami adalah norma Euclid vektor, $\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}$. Menurut rumus, soalnya sederhana: kuadratkan, jumlahkan, tarik akar. Namun pada komputer dengan rentang bilangan terbatas, kuadrat sebuah komponen bisa melampaui bilangan terbesar yang dapat disimpan (overflow) padahal normanya sendiri masih wajar, atau menjadi lebih kecil daripada bilangan positif terkecil sehingga dibulatkan menjadi nol (underflow) dan seluruh hasil menjadi 0. Overflow biasanya menghentikan program atau menghasilkan tak hingga, sedangkan underflow biasanya diam-diam menghasilkan nol, sehingga lebih berbahaya karena tidak terlihat.

Program serba guna mengatasinya dengan dua gagasan. Pertama, ia mengenal karakteristik mesin lewat parameter, misalnya bilangan terbesar λ dan bilangan positif terkecil σ pada mesin hipotetis berbasis 10 dengan t digit dan eksponen antara $e_{\min}$ dan $e_{\max}$. Pada mesin seperti itu $\sigma = 10^{e_{\min}-1}$ dan $\lambda = (1-10^{-t})10^{e_{\max}}$. Kedua, komponen dikelompokkan menurut besarnya (kecil, sedang, besar) dan komponen kecil atau besar diberi faktor skala sebelum dikuadratkan, lalu hasilnya dikembalikan ke skala semula. Cara paling sederhana yang berprinsip sama adalah membagi semua komponen dengan komponen terbesar $m = \max_i |x_i|$, sehingga $\|\mathbf{x}\|_2 = m\left(\sum_i (x_i/m)^2\right)^{1/2}$. Sekarang semua suku berada di $[0,1]$ dan jumlahnya paling besar n, sehingga tidak ada overflow, dan komponen terbesar menyumbang tepat 1 sehingga tidak ada underflow yang merusak hasil.

Harga yang dibayar adalah kerja tambahan (pembagian, dua kali lintasan data, percabangan) dan program yang jauh lebih panjang. Karena itu program serba guna terasa "berlebihan" bila dipakai untuk soal kecil, tetapi bagi pengguna yang hanya ingin hasil yang dapat dipercaya, kehati-hatian itu justru nilai utamanya. Perangkat lunak semacam ini juga diuji ketat, didokumentasikan lengkap, dan biasanya portabel, meskipun pengguna tetap perlu membaca dokumentasi karena ketergantungan pada mesin tidak hilang seluruhnya.

Ada dua jenis sumber perangkat lunak. Yang publik (gratis) mencakup EISPACK (nilai dan vektor eigen), LINPACK (sistem linear dan kuadrat terkecil), BLAS (operasi dasar vektor dan matriks), dan LAPACK yang menggantikan EISPACK dan LINPACK dalam satu pustaka yang dioptimalkan untuk prosesor modern; semuanya dapat dicari lewat repositori seperti netlib. Yang komersial mencakup IMSL dan NAG, yaitu pustaka subrutin yang dipanggil dari program Fortran, C, atau Java. Selain itu ada lingkungan mandiri tempat pengguna mengetikkan perintah: MATLAB (awalnya antarmuka mudah ke LINPACK dan EISPACK), Maple, dan Mathematica yang juga mampu menghitung secara simbolik. Python dengan NumPy dan SciPy kini menempati peran serupa dan di bawahnya memanggil BLAS dan LAPACK.

Pesan praktisnya: untuk belajar, tulis sendiri algoritmanya agar paham cara kerjanya; untuk pekerjaan nyata, gunakan rutin pustaka yang sudah teruji, tetapi tetap pahami asumsi, rentang masukan, dan arti peringatan yang dikeluarkannya. Seksi ini menjadi jembatan antara Bab 1 (galat, aritmetika komputer) dan metode-metode Bab 2 dan seterusnya, yang

Web by: Steven Paulus Siahaan

pada akhir tiap bab akan dikaitkan dengan rutin pustaka yang relevan.

Konsep kunci

Program khusus vs serba guna — Program khusus cukup untuk belajar dan soal biasa; program serba guna mengantisipasi galat pembulatan, overflow, underflow, dan menyatakan rentang masukan yang menjamin ketelitian tertentu.

Overflow — Hasil antara melebihi bilangan terbesar λ yang dapat disimpan. Program biasanya gagal atau menghasilkan tak hingga, walau hasil akhirnya mungkin wajar.

Underflow — Hasil antara bermagnitudo lebih kecil dari bilangan positif terkecil σ sehingga dibulatkan menjadi 0. Program biasanya terus berjalan, sehingga galatnya mudah tak terlihat.

Parameter mesin — Besaran seperti t, $e_{\min}$, $e_{\max}$, σ, λ yang menggambarkan sistem floating-point; perangkat lunak serba guna memakainya agar tetap benar di berbagai komputer.

Penskalaan — Mengalikan atau membagi komponen dengan faktor tertentu sebelum dikuadratkan agar tidak terjadi overflow atau underflow, lalu mengembalikan skala pada hasil.

BLAS, LINPACK, EISPACK, LAPACK — Rangkaian pustaka publik untuk aljabar linear numerik. LAPACK menggabungkan dan memperbarui LINPACK serta EISPACK, dan memakai BLAS sebagai operasi dasarnya.

IMSL dan NAG — Pustaka komersial berisi ratusan hingga ribuan subrutin untuk hampir semua jenis masalah numerik, lengkap dengan dokumentasi dan program contoh.

Lingkungan mandiri — MATLAB, Maple, Mathematica (dan serupa) tempat pengguna memberi perintah interaktif dan dapat pula memprogram dalam bahasa perintahnya.

Norma Euclid

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2}$$

Norma dengan penskalaan

$$m = \max_i |x_i|, \qquad \|\mathbf{x}\|_2 = m \left(\sum_{i=1}^{n} \left(\frac{x_i}{m}\right)^2 \right)^{1/2}$$

Bilangan positif terkecil mesin hipotetis

$$\sigma = 10^{e_{\min} - 1}$$

Bilangan terbesar mesin hipotetis

$$\lambda = (1 - 10^{-t})\, 10^{e_{\max}}$$

Algoritma: Norma Euclid dengan penskalaan sederhana

1. Masukkan vektor $\mathbf{x} = (x_1, \ \ldots, x_n)$. Jika semua komponen nol, keluarkan 0 dan berhenti.
2. Hitung $m = \max_i |x_i|$ (satu lintasan data).

3. Hitung $s = \sum_{i=1}^{n} (x_i/m)^2$. Karena $0 \le |x_i/m| \le 1$, setiap suku paling besar 1 sehingga $s \le n$ dan tidak terjadi overflow.
4. Keluarkan $\|\mathbf{x}\|_2 = m\sqrt{s}$.
5. Catatan: rutin pustaka sungguhan (misalnya BLAS) mengelompokkan komponen menjadi kecil, sedang, besar dan hanya menskalakan yang perlu, supaya tidak mengeluarkan biaya pembagian untuk setiap komponen.

Contoh 1: Overflow pada mesin hipotetis

Sebuah mesin berbasis 10 memakai $t = 4$ digit, $e_{\min} = -5$, $e_{\max} = 5$. Hitung σ dan λ, lalu tunjukkan mengapa menghitung $\|\mathbf{x}\|_2$ untuk $\mathbf{x} = (300, 400)$ secara langsung gagal, dan perbaiki dengan penskalaan.

Penyelesaian.

$$\sigma = 10^{-6}, \qquad \lambda = (1 - 10^{-4})\,10^{5} = 99990.$$

Komponen 300 dan 400 sendiri berada dalam rentang. Kuadratnya: $300^2 = 90000 < \lambda$ aman, tetapi $400^2 = 160000 > \lambda$ sehingga terjadi overflow, padahal norma sebenarnya $\sqrt{90000 + 160000} = 500 < \lambda$.

Dengan penskalaan: $m = 400$, komponen terskala $(0{,}75;\ 1)$, jumlah kuadrat $0{,}5625 + 1 = 1{,}5625$, akar $= 1{,}25$, lalu $400 \times 1{,}25 = 500$. Tidak ada hasil antara yang melampaui λ.

Contoh 2: Overflow dan underflow pada bilangan ganda (double)

Pada aritmetika presisi ganda IEEE (bilangan terbesar sekitar $1{,}8 \times 10^{308}$), hitung norma $\mathbf{u} = (3 \times 10^{200},\ 4 \times 10^{200})$ dan $\mathbf{v} = (3 \times 10^{-200},\ 4 \times 10^{-200})$ secara langsung dan dengan penskalaan.

Penyelesaian.

Untuk $\mathbf{u}$: $(3 \times 10^{200})^2 = 9 \times 10^{400}$ melampaui $1{,}8 \times 10^{308}$, sehingga jumlah kuadrat menjadi tak hingga (diverifikasi dengan kode), dan norma langsung $= \infty$. Dengan penskalaan, $m = 4 \times 10^{200}$, suku terskala $0{,}75$ dan 1, jumlah kuadrat $1{,}5625$, akar $1{,}25$, hasil 5×10^{200} (benar).

Untuk $\mathbf{v}$: $(3 \times 10^{-200})^2 = 9 \times 10^{-400}$ lebih kecil daripada bilangan positif terkecil yang dapat disimpan (sekitar 5×10^{-324}), sehingga dibulatkan menjadi 0 dan norma langsung $= 0$, jelas salah. Dengan penskalaan diperoleh 5×10^{-200} (benar).

Pelajaran: galat underflow tidak mengeluarkan peringatan apa pun, hasilnya hanya salah secara diam-diam.

Contoh 3: Menghitung norma vektor biasa dan batas aman

Hitung $\|\mathbf{x}\|_2$ untuk $\mathbf{x} = (-6, 2, 9, -4, 12)$ dan jelaskan kapan perlu waspada terhadap overflow pada presisi ganda.

Penyelesaian.

$$\|\mathbf{x}\|_2 = \sqrt{36 + 4 + 81 + 16 + 144} = \sqrt{281} \approx 16{,}763.$$

Untuk vektor sebesar ini penskalaan tidak diperlukan. Pada presisi ganda, kuadrat komponen overflow bila $|x_i|$ lebih besar dari sekitar $1{,}3 \times 10^{154}$ dan kuadrat menjadi 0 (underflow) bila $|x_i|$ lebih kecil dari sekitar 10^{-162}. Selama semua komponen berada jauh di dalam rentang itu, rumus langsung cukup; di luar itu gunakan rutin pustaka (mis. fungsi norma NumPy atau BLAS).

Kesalahan umum

• Mengira rumus yang benar secara matematis otomatis benar secara numerik. Urutan dan bentuk perhitungan menentukan apakah hasil antara melampaui rentang mesin.

• Menganggap underflow tidak berbahaya karena tidak ada pesan galat. Jika semua suku underflow, hasilnya 0 yang salah total.

• Menyalin program contoh buku/kuliah ke aplikasi nyata tanpa memeriksa rentang masukan dan penanganan kasus khusus, padahal program itu dirancang untuk belajar.

• Memakai rutin pustaka sebagai kotak hitam tanpa membaca dokumentasinya (syarat masukan, arti kode galat, peringatan ketelitian).

Pertanyaan diskusi

1. Mengapa overflow dianggap lebih mudah dideteksi daripada underflow? Bagaimana perbedaan itu memengaruhi cara kita menguji program?
2. Penskalaan dengan membagi komponen terbesar membutuhkan dua lintasan data dan satu pembagian per komponen. Kapan biaya itu layak dibayar, dan bagaimana rutin pustaka mengurangi biayanya?
3. Apa saja keuntungan dan kerugian memakai pustaka komersial dibanding pustaka publik atau program tulisan sendiri untuk proyek penelitian?
4. Mengapa BLAS dipisahkan sebagai lapisan tersendiri dari LAPACK? Apa manfaatnya bagi portabilitas dan kecepatan?

Latihan mandiri (soal buatan sendiri, dengan kunci)

1. Mesin hipotetis berbasis 10 memakai $t = 4$, $e_{\min} = -4$, $e_{\max} = 4$. Hitung λ. Tunjukkan bahwa untuk $\mathbf{x} = (60, 80)$ tidak ada kuadrat yang overflow, tetapi jumlah kuadratnya overflow.

Kunci: $\lambda = (1 - 10^{-4})10^4 = 9999$. Kuadrat: $3600 < \lambda$ dan $6400 < \lambda$, tetapi jumlahnya $10000 > 9999$ sehingga overflow, padahal norma sebenarnya 100. Dengan penskalaan ($m = 80$): $0{,}75^2 + 1 = 1{,}5625$, akar $1{,}25$, hasil 100.

2. Pada presisi ganda, hitung secara langsung dan dengan penskalaan norma $\mathbf{x} = (10^{-170}, 10^{-170})$. Berapa hasil yang benar?

Kunci: Langsung: $(10^{-170})^2 = 10^{-340}$ underflow menjadi 0, norma $= 0$ (salah). Penskalaan: $m = 10^{-170}$, suku terskala $1, 1$, jumlah 2, hasil $\sqrt{2} \cdot 10^{-170} \approx 1{,}414 \times 10^{-170}$.

3. Buktikan bahwa dengan penskalaan $m = \max_i |x_i|$ (dengan $m > 0$) jumlah $\sum_i (x_i/m)^2$ selalu berada di antara 1 dan n.

Kunci: Setiap $|x_i/m| \leq 1$ sehingga tiap suku ≤ 1 dan jumlahnya $\leq n$. Untuk indeks k tempat $|x_k| = m$ suku bernilai tepat 1, dan semua suku tak negatif, sehingga jumlah ≥ 1.

4. Anda perlu menghitung semua nilai eigen sebuah matriks simetris berukuran 500×500. Tuliskan alasan mengapa memakai rutin LAPACK (atau fungsi eig di MATLAB/NumPy) lebih tepat daripada menulis sendiri metode pangkat.

Kunci: Rutin pustaka sudah teruji, tahan terhadap galat pembulatan/overflow, memanfaatkan struktur simetris dan operasi BLAS yang dioptimalkan, menghitung semua nilai eigen sekaligus, dan menyertakan penanganan kasus khusus; metode pangkat hanya memberi satu nilai eigen dominan dan bisa lambat atau gagal.

Pesan Utama

Ingat

Galat tidak dapat dihilangkan, tetapi dapat dikendalikan: pilih rumus yang menghindari pengurangan hampir-sama dan susun ulang perhitungan (misalnya perkalian bersarang).

Langkah Belajar Berikutnya

1. Baca ulang tiap seksi di buku menurut halaman pada Peta Isi Bab, lalu kerjakan contoh secara manual.
2. Kerjakan soal Exercise Set tiap seksi, mulai dari soal bernomor ganjil (jawaban ada di bagian akhir buku).
3. Implementasikan satu metode dari bab ini (Python/MATLAB) dan bandingkan hasilnya dengan hitungan tangan.

Sumber: Richard L. Burden dan J. Douglas Faires, Numerical Analysis, Edisi ke-9, Cengage Learning, 2010.
Dibuat otomatis 07-11-2026.

Web by: Steven Paulus Siahaan