Rangkuman Metode Numerik

Informatika, Universitas Diponegoro

4 April 2025

Contents

1 Galat dan Konsep Aritmatika Komputer

1.1 Pengertian Galat (Error) dalam Metode Numerik

Galat atau error dalam metode numerik merupakan perbedaan antara hasil eksak dan hasil aproksimasi dalam perhitungan numerik. Galat muncul akibat keterbatasan dalam representasi angka dan metode komputasi yang digunakan.

Secara umum, galat dalam metode numerik dapat dikategorikan menjadi dua jenis utama:

- Galat Pembulatan (Roundoff Error)
- Galat Pemotongan (Truncation Error)

Selain itu, terdapat dua bentuk pengukuran galat yang penting:

- Galat Absolut
- Galat Relatif

1.2 Galat Absolut dan Galat Relatif

1.2.1 Galat Absolut

Galat absolut adalah perbedaan antara nilai eksak (x_{eksak}) dan nilai aproksimasi ($x_{aproksimasi}$). Didefinisikan sebagai:

$$E_{absolut} = |x_{eksak} - x_{aproksimasi}|$$

Galat ini mengukur seberapa jauh hasil aproksimasi dari nilai sebenarnya.

1.2.2 Galat Relatif

Galat relatif adalah perbandingan antara galat absolut dengan nilai eksak, yang dihitung sebagai:

$$E_{relatif} = \frac{|x_{eksak} - x_{aproksimasi}|}{|x_{eksak}|}$$

$$Atau$$

$$E_{relatif} = \frac{|x_{sekarang} - x_{sebelumnya}|}{|x_{sekarang}|}$$

Galat relatif lebih berguna dalam membandingkan error antara nilai yang memiliki skala berbeda.

Kita bisa menyimpulkan hasil estimasi benar hingga n digit desimal jika

$$|Error| \leq 10^{-n}$$

Kita bisa menyimpulkan hasil estimasi benar hingga pembulatan n desimal digit jika

$$|Error| \leq \frac{1}{2} \cdot 10^{-n}$$

Contoh Perhitungan Galat Absolut dan Relatif

Jika nilai eksak suatu perhitungan adalah ($x_{eksak} = 3.14159$) dan hasil aproksimasi ($x_{aproksimasi} = 3.14$), maka:

$$E_{absolut} = |3.14159 - 3.14| = 0.00159$$

$$E_{relatif} = \frac{0.00159}{3.14159} \approx 0.000506$$

1.3 Galat Pembulatan (Roundoff Error)

Galat pembulatan terjadi karena keterbatasan representasi bilangan dalam komputer. Komputer hanya dapat menyimpan angka dalam jumlah digit tertentu, sehingga nilai yang sangat kecil atau sangat besar bisa mengalami pembulatan.

Contoh Roundoff Error

Jika kita ingin merepresentasikan 1/3 dalam bentuk desimal:

$$\frac{1}{3} = 0.3333333...\text{tak hingga desimal}$$

Dalam komputer, angka ini hanya bisa disimpan dalam keterbatasan jumlah digit, misalnya hanya 5 digit:

$$0.33333$$

Sehingga, ada selisih antara nilai eksak dan nilai yang disimpan, yang disebut roundoff error.

Penyebab Roundoff Error

- Representasi floating-point yang terbatas
- Operasi aritmatika yang mengakumulasi error
- Pembulatan otomatis dalam penyimpanan angka

1.4 Galat Pemotongan (Truncation Error)

Galat pemotongan terjadi ketika suatu perhitungan numerik menghentikan deret tak hingga atau pendekatan pada titik tertentu, sehingga hasilnya tidak sepenuhnya akurat.

Contoh Truncation Error

Pendekatan fungsi eksponensial menggunakan deret Taylor:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Jika kita hanya menghitung sampai suku ke-3, maka ada kesalahan karena kita tidak menyertakan suku-suku selanjutnya.

Rumus Deret Taylor

$$f(x) = \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(a)(x-a)^k$$

Rumus Deret Maclaurin

Maclaurin series adalah bentuk spesial dari deret taylor dengan perluasan pada $a = 0$

$$f(x) = \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(0)(x)^k$$

Cara Mengurangi Truncation Error

- Menggunakan lebih banyak suku dalam deret aproksimasi
- Menggunakan metode numerik yang lebih presisi

1.5 Konsep Aritmatika Komputer dalam Metode Numerik

Komputer menggunakan representasi bilangan floating-point untuk menyimpan angka desimal dalam bentuk:

$$x = \pm m \times b^e$$

Di mana:

- (m) adalah mantisa
- (b) adalah basis (biasanya 2 dalam komputer)
- (e) adalah eksponen

Karena keterbatasan penyimpanan, komputer mengalami kesalahan representasi yang bisa menyebabkan akumulasi error dalam perhitungan numerik.

Contoh Mantissa

Dalam sistem biner (basis 2), contoh representasi nya adalah sebagai berikut

$$\pm 1.8503717077085943 \quad e - 16$$

- $\pm$ = tanda
- 1.8503717077085943 = mantissa
- e-16 = eksponen

Operasi Floating-Point dalam Komputer

- Penjumlahan dan Pengurangan - Dapat menyebabkan kehilangan presisi jika angka memiliki eksponen yang sangat berbeda.
- Perkalian dan Pembagian - Dapat menghasilkan overflow atau underflow jika angka terlalu besar atau kecil.
- Truncation dan Pembulatan - Angka bisa dibulatkan secara otomatis oleh sistem, menghasilkan galat kecil yang terakumulasi.

1.6 Tambahan Informasi

- Angka yang bisa direpresentasikan secara akurat disebut angka mesin
- Jumlahan dari angka mesin tidak selalu menghasilkan angka mesin
- Akurasi berkaitan dengan kedekatan ke nilai asli
- Presisi berkaitan dengan kedekatan ke nilai perkiraan
- Pembulatan adalah mengganti angka dengan angka mesin terdekat
- Pemotongan adalah membuang seluruh digit tambahan
- Angka yang memiliki ekspansi terbatas dalam suatu sistem penomoran mungkin memiliki ekspansi tak hingga dalam sistem penomoran yang lain contoh nya representasi 1.1 dalam desimal tidak akan bisa direpresentasikan secara akurat dalam sistem biner

1.7 Kesimpulan

- Galat dalam metode numerik terdiri dari galat pembulatan dan galat pemotongan.
- Galat pembulatan disebabkan oleh keterbatasan penyimpanan floating-point dalam komputer.
- Galat pemotongan terjadi saat deret atau pendekatan dihentikan pada titik tertentu.
- Galat absolut mengukur perbedaan langsung antara nilai eksak dan aproksimasi.
- Galat relatif mengukur kesalahan dalam skala relatif terhadap nilai eksak.

2 Pencarian Solusi Persamaan Satu Variabel

Pencarian akar atau solusi dari persamaan satu variabel adalah salah satu topik fundamental dalam metode numerik. Dalam banyak kasus, persamaan tidak dapat diselesaikan secara analitis, sehingga pendekatan numerik diperlukan untuk menemukan solusi yang mendekati akar sebenarnya.

Akar persamaan satu variabel umumnya berbentuk:

$$f(x) = 0$$

Di mana kita mencari nilai (x) yang membuat $(f(x))$ bernilai nol.

Metode-Metode dalam Pencarian Akar

Berbagai metode numerik digunakan untuk mencari akar suatu fungsi, di antaranya:

- Metode Biseksi (Bisection Method)
- Metode Regula Falsi (False Position Method)
- Metode Secant
- Metode Newton-Raphson (Newton Forward, Central, Backward)

2.1 Metode Biseksi

Metode biseksi adalah metode numerik sederhana untuk mencari akar dengan membagi interval secara berulang.

Langkah-langkah Metode Biseksi

- Pilih dua titik awal (a) dan (b) yang mengapit akar, sehingga $(f(a) \times f(b) < 0)$.
- Hitung titik tengah $(c = \frac{a+b}{2})$.
- Evaluasi $(f(c))$:
  - Jika $(f(c) = 0)$, maka (c) adalah akar.
  - Jika $(f(a) \times f(c) < 0)$, perbarui $(b = c)$.
  - Jika $(f(c) \times f(b) < 0)$, perbarui $(a = c)$.
- Ulangi proses hingga konvergen terhadap akar.

2.2 Metode Regula Falsi

Metode Regula Falsi mirip dengan metode biseksi, tetapi menggunakan pendekatan garis lurus untuk mencari akar.

Langkah-langkah Metode Regula Falsi

- Pilih dua titik awal a dan b yang mengapit akar.
- Gunakan rumus:
$$c = b - \frac{f(b)(b-a)}{f(b) - f(a)}$$
- Evaluasi $(f(c))$ dan perbarui interval.
- Ulangi hingga konvergen terhadap akar.

2.3 Metode Secant

Metode secant adalah metode yang sama dengan metode regula falsi namun metode secant bukan termasuk metode bracket tertutup sehingga ada kemungkinan hasil nya divergen.

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}$$

2.4 Metode Newton-Raphson

Metode Newton-Raphson menggunakan turunan pertama fungsi untuk menemukan akar dengan iterasi:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

2.5 Kesimpulan

- Metode biseksi stabil tetapi konvergen lambat.
- Metode regula falsi mirip dengan biseksi tetapi lebih cepat.
- Metode secant tidak memerlukan turunan tetapi bisa kurang stabil.
- Metode Newton-Raphson cepat tetapi membutuhkan turunan fungsi.

3 Interpolasi

Interpolasi merupakan teknik dalam metode numerik yang digunakan untuk menentukan nilai suatu fungsi pada titik yang tidak diketahui dengan menggunakan serangkaian titik data yang diketahui. Metode interpolasi banyak digunakan dalam berbagai aplikasi seperti pemrosesan sinyal, analisis data, dan simulasi numerik.

Interpolasi berbeda dengan regresi. Jika regresi bertujuan mencari hubungan terbaik untuk mendekati data, interpolasi bertujuan mendapatkan nilai yang tepat di antara titik-titik data yang ada.

Dalam rangkuman ini, kita akan membahas tiga metode interpolasi utama:

- Metode Newton (beda maju, sentral, beda mundur)
- Metode Lagrange
- Metode Neville

3.1 Metode Newton

Metode interpolasi Newton menggunakan konsep beda terbagi untuk membentuk polinom interpolasi. Metode ini memiliki tiga bentuk utama:

- Beda Maju (Forward Difference)
- Beda Sentral (Central Difference)
- Beda Mundur (Backward Difference)

Rumus Umum Interpolasi Newton Polinom interpolasi Newton berbasis beda terbagi didefinisikan sebagai berikut:

$$P_n(x) = f(x_0) + \sum_{k=1}^{n} \left(\prod_{i=0}^{k-1} (x - x_i) \right) \Delta^k f(x_0)$$

$$\Delta^k f(x_0) = \text{beda terbagi orde ke-k}$$
$$x_0, x_1, \ldots, x_n = \text{ titik data}$$

$$\Delta f(x_n) = f[x_n, \cdots, x_1, x_0] = \frac{f[x_n, \cdots, x_1, x_0] - f[x_{n-1}, x_{n-2}, \cdots, x_1, x_0]}{x_n - x_0}$$

Keunggulan:

- Cocok untuk data dengan interval yang tidak merata.
- Bisa diperbarui secara rekursif tanpa harus menghitung ulang dari awal.

3.2 Metode Lagrange

Interpolasi Lagrange menggunakan basis polinom untuk menemukan fungsi interpolasi tanpa perlu menghitung beda terbagi.

Rumus Interpolasi Lagrange

$$P(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

Di mana ($L_i(x)$) adalah polinom Lagrange:

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

Keunggulan:

- Tidak memerlukan perhitungan iteratif tambahan seperti metode Newton.
- Bisa digunakan langsung tanpa perlu menyusun tabel beda terbagi.

3.3 Metode Neville

Metode Neville menggunakan pendekatan rekursif untuk menghitung nilai interpolasi.

Rumus Interpolasi Neville

$$P_{i,j} = \frac{(x - x_j)P_{i,j-1} - (x - x_i)P_{i+1,j}}{x_i - x_j}$$

Keunggulan:

- Memberikan solusi dengan perhitungan langsung tanpa perlu membentuk polinom eksplisit.
- Cocok untuk interpolasi di titik tertentu tanpa perlu membangun seluruh polinom.

Metode ini biasanya digunakan dalam kasus dengan banyak titik data atau ketika diperlukan interpolasi yang lebih akurat.

4 Regresi

Regresi adalah salah satu metode dalam statistika yang digunakan untuk memodelkan hubungan antara variabel independen (x) dan variabel dependen (y). Salah satu tujuan utama dari regresi adalah untuk menemukan persamaan yang dapat digunakan untuk melakukan prediksi berdasarkan data yang tersedia.

Dalam regresi linier, hubungan antara x dan y diasumsikan berbentuk linier, sehingga model yang digunakan umumnya berupa:

$$y = a_0 + a_1 x + \epsilon$$

di mana:

- y adalah variabel dependen,
- x adalah variabel independen,
- a_0 adalah konstanta (intersection/titik potong y),
- a_1 adalah koefisien regresi (kemiringan garis regresi),
- ϵ adalah galat atau error yang diharapkan minimal.

Selain regresi linier, terdapat juga metode regresi polinomial yang lebih kompleks, di mana modelnya memiliki derajat lebih tinggi untuk menangkap pola data yang lebih kompleks.

Metode paling umum dalam menemukan parameter regresi adalah Least Squares Method (Metode Kuadrat Terkecil), yang bertujuan untuk meminimalkan jumlah kuadrat selisih antara nilai yang diamati dan nilai yang diprediksi oleh model regresi.

Pada tugas ini, pendekatan regresi akan dilakukan dengan metode komputasi matriks untuk mendapatkan solusi yang lebih efisien dan akurat.

Regresi linear memiliki model umum:

$$f(x) = a + bx$$

Dengan mencari selisih kuadrat bentuk umum tersebut dengan data aktual maka didapatkan

$$\sum_{i=1}^{n} (y_i - f(x_i))^2 = \sum_{i=1}^{n} (y_i - a - bx_i)^2$$

Dengan mengambil turunan parsial dari deret tersebut maka didapatkan beberapa persamaan linear yang bisa diselesaikan dengan menggunakan metode pencarian solusi matriks

$$an + \hat{x}b = \hat{y}$$
$$\hat{x}a + (\sum_{i=1}^{n} x_i^2)b = \sum_{i=1}^{n} x_i y_i$$

Dengan

$$\hat{x} = \sum_{i=1}^{n} x_i \qquad \hat{y} = \sum_{i=1}^{n} y_i$$

Kedua persamaan tersebut bisa dimodelkan dengan matriks, sehingga solusi nya bisa ditemukan dengan berbagai metode solusi sistem persamaan linier matriks seperti metode cramer, Gauss, dan Gauss-Jordan.

$$\begin{bmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix}$$

Model tersebut juga bisa dihampiri dengan persamaan normal matriks

$$\mathbf{XB} = \mathbf{Y}$$

Dapat disusun persamaan normal dalam regresi linear adalah:

$$(\mathbf{X}^T\mathbf{X})\mathbf{B} = \mathbf{X}^T\mathbf{Y}$$

Dengan,

$$\mathbf{X}^T\mathbf{X} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \cdots & \cdots \\ 1 & x_n \end{bmatrix} = \begin{bmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{bmatrix}$$

$$\mathbf{X}^T\mathbf{Y} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix}$$

Maka b dapat dihitung dengan cara:

$$\mathbf{B} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y}$$

Sehingga didapatkan model yang sama,

$$\begin{bmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix}$$

Model ini bisa diekspansi tergantung dengan jumlah variabel dependen atau prediktor yang digunakan. Model ini juga bisa digunakan untuk regresi polinomial berderajat m.

$$f_m(x) = a_0 + a_1 x + a_2 x^2 \cdots a_m x^m + \epsilon$$

Regresi untuk model tersebut bisa dinyatakan dengan,

$$\begin{bmatrix} n & \sum x_i & \sum x_i^2 & \sum x_i^3 & \cdots & \sum x_i^m \\ \sum x_i & \sum x_i^2 & \sum x_i^3 & \sum x_i^4 & \cdots & \sum x_i^{m+1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum x_i^{m-1} & \sum x_i^m & \sum x_i^{m+1} & \sum x_i^{m+2} & \cdots & \sum x_i^{2m} \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_i y_i \\ \vdots \\ \sum x_i^m y_i \end{bmatrix}$$

Untuk kasus regresi eksponensial membutuhkan manipulasi aljabar sebelum menggunakan model di atas

$$\begin{aligned} y &= ae^{bx} \\ \ln(y) &= \ln(ae^{bx}) \\ \ln(y) &= \ln(a) + \ln(e^{bx}) \\ \ln(y) &= \ln(a) + bx \end{aligned}$$

Setelah mendapatkan bentuk tersebut maka regresi bisa dihitung dengan pola matriks yang sama. Setelah mendapatkan nilai a dan b maka bentuk persamaan akan dikembalikan ke bentuk eksponensial.

$$\begin{aligned} \ln(y) &= \ln(a) + bx \\ e^{ln(y)} &= e^{ln(a)+bx} \\ y &= ae^{bx} \end{aligned}$$

5 Turunan Numerik

Formula untuk turunan adalah

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Namun ketika fungsi nya tidak diketahui secara eksplisit, hanya titik titik data yang diketahui atau fungsi terlalu rumit, maka diperlukan metode alternatif untuk menghitung nilai dari turunan. Hal ini disebut turunan numerik. 3 Metode yang digunakan dalam tugas ini adalah :

- Newton beda hingga maju
- Newton beda hingga mundur
- Newton beda hingga sentral

Pada rangkuman ini juga akan dijelaskan perhitungan ekstrapolasi menggunakan metode Richardson. Ekstrapolasi adalah proses memperkirakan atau memperluas nilai suatu variabel di luar data atau interval pengamatan yang tersedia, namun tetap mengikuti pola atau tren yang ada.

5.1 Newton Beda Hingga Maju

Beda Maju Ordo $O(h)$

Uraikan $f(x_{i+1})$ di sekitar x_i :

$$\begin{aligned}
f(x_{i+1}) &= f(x_i) + \frac{(x_{i+1} - x_i)}{1!} f'(x_i) + \frac{(x_{i+1} - x_i)^2}{2!} f''(x_i) + \dots \\
f_{i+1} &= f_i + h f_i' + \frac{h^2}{2} f_i'' + \dots \\
h f_i' &= f_{i+1} - f_i - \frac{h^2}{2} f_i'' + \dots \\
f_i' &= \frac{f_{i+1} - f_i}{h} - \frac{h}{2} f_i'' + O(h) \\
f_i' &= \frac{f_{i+1} - f_i}{h} + O(h)
\end{aligned}$$

yang dalam hal ini, $O(h) = h\frac{1}{2}f''(t), \quad x_i < t < x_{i+1}$, dengan galat pemotongan $O(h)$.

Newton Beda Hingga Maju untuk Turunan Pertama Orde $O(h^2)$

Sebelumnya, telah diturunkan rumus beda maju dengan orde $O(h)$:

$$f_i' = \frac{f_{i+1} - f_i}{h} + O(h)$$

Untuk meningkatkan akurasi hingga orde $O(h^2)$, deret Taylor dapat dimanfaatkan hingga suku orde lebih tinggi.

Diketahui ekspansi Taylor untuk $f(x)$ di sekitar x_i:

$$f(x_{i+1}) = f(x_i) + hf'(x_i) + \frac{h^2}{2}f''(x_i) + \frac{h^3}{6}f'''(x_i) + O(h^4)$$
$$f(x_{i+2}) = f(x_i) + 2hf'(x_i) + 2h^2f''(x_i) + \frac{8h^3}{6}f'''(x_i) + O(h^4)$$

Eliminasi $f''(x_i)$ dengan mengalikan persamaan pertama dengan 4 dan menguranginya dari persamaan kedua:

$$f(x_{i+2}) - 4f(x_{i+1}) + 3f(x_i) = -2hf'(x_i) + O(h^3)$$

Sehingga, diperoleh rumus beda maju dengan orde lebih tinggi:

$$f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_i)}{2h} + O(h^2)$$

Rumus ini memiliki galat orde $O(h^2)$, yang lebih akurat dibandingkan dengan formula beda hingga maju orde pertama.

Newton Beda Hingga Maju untuk Turunan Kedua Orde $O(h)$

$$f_0'' = \frac{f_{i+2} - 2f_{i+1} + f_i}{h^2} + O(h)$$

Newton Beda Hingga Maju untuk Turunan Kedua Orde $O(h^2)$

$$f_0'' = \frac{-f_{i+3} + 4f_{i+2} - 5f_{i+1} + 2f_i}{12h} + O(h)$$

5.2 Newton Beda Hingga Mundur

Beda Mundur untuk ordo $O(h)$

Uraikan $f(x_{i-1})$ di sekitar x_i :

$$f(x_{i-1}) = f(x_i) + \frac{(x_{i+1} - x_i)}{1!}f'(x_i) + \frac{(x_{i+1} - x_i)^2}{2!}f''(x_i) + \dots$$
$$f_{i-1} = f_i - hf_i' + \frac{h^2}{2}f_i'' + \dots$$
$$hf_i' = f_i - f_{i-1} + \frac{h^2}{2}f_i'' + \dots$$
$$f_i' = \frac{f_i - f_{i-1}}{h} - \frac{h}{2}f_i'' + \dots$$
$$f_i' = \frac{f_i - f_{i-1}}{h} + O(h)$$

yang dalam hal ini, $O(h) = -h\frac{1}{2}f''(t), \quad x_{i-1} < t < x_i$. $O(h)$ adalah galat pemotongan

Newton Beda Hingga Mundur untuk Turunan Pertama Orde $O(h^2)$

Gunakan ekspansi Taylor di sekitar x_i:

$$f(x_{i-1}) = f(x_i) - hf'(x_i) + \frac{h^2}{2}f''(x_i) - \frac{h^3}{6}f'''(x_i) + O(h^4)$$
$$f(x_{i-2}) = f(x_i) - 2hf'(x_i) + \frac{4h^2}{2}f''(x_i) - \frac{8h^3}{6}f'''(x_i) + O(h^4)$$

Eliminasi $f''(x_i)$ dengan mengalikan persamaan pertama dengan 4 dan menguranginya dengan persamaan kedua:

$$f(x_{i-2}) - 4f(x_{i-1}) + 3f(x_i) = -2hf'(x_i) + O(h^3)$$

Sehingga diperoleh formula beda mundur untuk turunan pertama dengan akurasi lebih tinggi:

$$f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + f(x_{i-2})}{2h} + O(h^2)$$

Newton Beda Hingga Mundur untuk Turunan Kedua Orde $O(h)$

Untuk mendapatkan rumus turunan kedua, tetap menggunakan ekspansi Taylor lebih lanjut:

$$f(x_{i-1}) = f(x_i) - hf'(x_i) + \frac{h^2}{2}f''(x_i) - \frac{h^3}{6}f'''(x_i) + O(h^4)$$

$$f(x_{i-2}) = f(x_i) - 2hf'(x_i) + 2h^2f''(x_i) - \frac{4h^3}{6}f'''(x_i) + O(h^4)$$

Untuk menghilangkan $f'(x_i)$, gunakan kombinasi linear:

$$f(x_{i-2}) - 2f(x_{i-1}) + f(x_i) = (-2hf'(x_i) + 2h^2f''(x_i)) - 2(-hf'(x_i) + \frac{h^2}{2}f''(x_i)) + O(h^4)$$

Menyusun ulang:

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

Sehingga diperoleh rumus beda mundur orde $O(h)$ untuk turunan kedua:

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

5.3 Newton Beda Hingga Sentral

Kurangkan persamaan beda maju dan beda mundur

$$\begin{aligned} f_{i+1} - f_{i-1} &= 2hf_i' + \frac{h^3}{3}f_i''' + \cdots \\ 2hf_i' &= f_{i+1} - f_{i-1} - \frac{h^3}{3}f_i''' + \cdots \\ f_i' &= \frac{f_{i+1} - f_{i-1}}{2h} - \frac{h^2}{6}f_i''' + \cdots \\ f_i' &= \frac{f_{i+1} - f_{i-1}}{2h} + O(h^2) \end{aligned}$$

yang dalam hal ini, $O(h^2) = -h\frac{1}{6}f'''(t), \quad x_{i-1} < t < x_{i+1}$. $O(h)$ adalah galat pemotongan.

Newton Beda Hingga Sentral untuk Turunan Kedua ($O(h^2)$)

Pendekatan beda hingga sentral memanfaatkan titik di sekitar x_i, yaitu x_{i+1} dan x_{i-1}, untuk mencapai akurasi lebih tinggi.

Ekspansi fungsi $f(x)$ di sekitar x_i:

$$\begin{aligned} f(x_{i+1}) &= f(x_i) + hf'(x_i) + \frac{h^2}{2}f''(x_i) + \frac{h^3}{6}f'''(x_i) + \frac{h^4}{24}f^{(4)}(x_i) + O(h^5) \\ f(x_{i-1}) &= f(x_i) - hf'(x_i) + \frac{h^2}{2}f''(x_i) - \frac{h^3}{6}f'''(x_i) + \frac{h^4}{24}f^{(4)}(x_i) + O(h^5) \end{aligned}$$

Menghilangkan Turunan Pertama $f'(x_i)$ dengan menjumlahkan kedua ekspansi:

$$f(x_{i+1}) + f(x_{i-1}) = 2f(x_i) + h^2f''(x_i) + \frac{h^4}{12}f^{(4)}(x_i) + O(h^5)$$

Kemudian, susun ulang untuk mendapatkan $f''(x_i)$:

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

Jadi, formula Newton Beda Hingga Sentral untuk turunan kedua dengan akurasi $O(h^2)$ adalah:

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

5.4 Ekstrapolasi Richardson

Ekstrapolasi Richardson adalah metode sederhana yang digunakan untuk meningkatkan akurasi dalam berbagai prosedur numerik, termasuk aproksimasi beda hingga. Teknik ini bekerja dengan menghilangkan suku galat utama dari suatu pendekatan numerik, sehingga menghasilkan estimasi yang lebih akurat.

Misalkan kita ingin menghitung turunan numerik $f'(x)$ menggunakan metode beda hingga maju dengan dua nilai langkah h dan $h_2 = \frac{h}{2}$. Dengan pendekatan orde kedua, kita memiliki:

$$D(h) = f'(x) + Ch^2 + O(h^4)$$

dan untuk langkah yang lebih kecil:

$$D(h_2) = f'(x) + Ch_2^2 + O(h^4)$$

Dengan menggunakan teknik ekstrapolasi Richardson, kita dapat mengeliminasi suku galat orde $O(h^2)$ dengan kombinasi linier sebagai berikut:

$$D_{\text{Richardson}} = \frac{2^p D(h_2) - D(h)}{2^p - 1}$$

di mana p adalah orde dari suku galat utama yang ingin dieliminasi. Dalam kasus ini, $p = 2$, sehingga rumus menjadi:

$$D_{\text{Richardson}} = \frac{4D(h_2) - D(h)}{3}$$

Dengan cara ini, hasil yang diperoleh memiliki orde galat yang lebih kecil, yaitu $O(h^4)$, yang berarti tingkat akurasi meningkat dibandingkan metode awal.

6 Integral Numerik

Integrasi numerik adalah pendekatan untuk menghitung integral tak tentu atau tentu menggunakan metode numerik. Dalam tugas ini, metode yang digunakan adalah sebagai berikut:

- Metode Trapesium
- Metode Simpson 1/3
- Metode Simpson 3/8

6.1 Metode Trapesium

Metode ini bekerja dengan mendekati daerah integral sebagai sejumlah trapesium dan menghitung luas totalnya. Metode Trapesium cocok digunakan ketika diperlukan metode yang sederhana dan efisien dalam perhitungan. Namun, metode ini kurang akurat untuk fungsi yang sangat non-linear atau memiliki osilasi tinggi.

Karakteristik Data yang Cocok

Metode ini bekerja dengan baik untuk fungsi yang:

- Relatif halus atau tidak terlalu berfluktuasi tajam.
- Memiliki interval pembagian yang cukup kecil untuk meningkatkan akurasi.

Metode Trapesium mendekati integral I sebagai jumlah luas trapesium dengan batas a hingga b. Rumus dasarnya adalah:

$$I \approx \frac{b-a}{2}\left(f(a)+f(b)\right)$$

Jika terdapat n subinterval dengan lebar $h = \frac{b-a}{n}$, maka rumus diperluas menjadi:

$$I \approx \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right]$$

6.2 Metode Simpson 1/3

Metode Simpson 1/3 adalah salah satu metode dalam integrasi numerik yang lebih akurat dibandingkan metode trapesium. Metode ini menggunakan pendekatan parabola untuk menghampiri fungsi yang diintegralkan.

Karakteristik Data yang Cocok Metode ini cocok digunakan untuk fungsi yang:

- Relatif halus dan kontinu.
- Memiliki pola yang tidak terlalu kompleks dalam interval integrasi.
- Jumlah subinterval harus genap agar metode ini dapat diterapkan dengan benar.

Formula Umum Metode Simpson 1/3 Metode ini membagi interval integrasi menjadi n subinterval genap dengan lebar $h = \frac{b-a}{n}$, dan mendekati fungsi menggunakan interpolasi kuadratik. Rumus dasarnya adalah:

$$I \approx \frac{h}{3}\left[f(x_0) + 4\sum_{i=1,\text{ganjil}}^{n-1} f(x_i) + 2\sum_{i=2,\text{genap}}^{n-2} f(x_i) + f(x_n)\right]$$

Metode ini memberikan hasil yang lebih akurat dibandingkan metode trapesium, terutama untuk fungsi yang dapat dihampiri oleh polinomial derajat dua.

6.3 Metode Simpson 3/8

Metode ini membagi interval integrasi menjadi kelipatan tiga subinterval dengan lebar $h = \frac{b-a}{n}$, dan mendekati fungsi menggunakan interpolasi kubik. Metode ini cocok jika titik-titik sampel yang digunakan tersebar secara merata dalam interval. Metode Simpson 3/8 sering digunakan bersama metode Simpson 1/3 dalam kasus di mana jumlah subinterval tidak sepenuhnya sesuai dengan kelipatan tiga, sehingga kombinasi keduanya dapat meningkatkan akurasi integrasi numerik. Rumus dasarnya adalah:

$$I \approx \frac{3h}{8}\left[f(x_0) + 3\sum_{i=1,\text{mod } 3\neq 0}^{n-1} f(x_i) + 2\sum_{i=3,\text{kelipatan } 3}^{n-3} f(x_i) + f(x_n)\right]$$

Metode ini lebih akurat dibandingkan metode Simpson 1/3 untuk fungsi yang lebih kompleks dan memungkinkan integrasi pada interval yang lebih luas.