

# METODE NUMERIK

## dan Pemrogramannya Menggunakan Julia

---

Divisi Matematika Komputasi

Departemen Matematika  
Fakultas Matematika dan Ilmu Pengetahuan Alam  
Institut Pertanian Bogor

Terakhir diperbarui pada: February 11, 2022



**Math IPB**

[www.math.ipb.ac.id](http://www.math.ipb.ac.id)



---

# DAFTAR ISI

<b>Sampul</b>	<b>i</b>
<b>Daftar Isi</b>	<b>iii</b>
<b>Prakata</b>	<b>viii</b>
<b>1 Metode Numerik dan Analisis Galat</b>	<b>1</b>
1.1 Metode Numerik . . . . .	2
1.1.1 Perbedaan Metode Analitik dan Numerik . . . . .	2
1.1.2 Ide Dasar dan Tahapan Metode Numerik . . . . .	3
1.2 Analisis Galat . . . . .	6
1.2.1 Sumber-Sumber Galat . . . . .	6
1.2.2 Galat Mutlak dan Galat Relatif . . . . .	7
1.2.3 Galat Maju dan Galat Mundur . . . . .	9
1.2.4 Sensitifitas . . . . .	10
1.2.5 Stabilitas dan Akurasi . . . . .	12
1.3 Praktikum: Pengenalan Julia . . . . .	12
1.3.1 Cara Memasang Julia . . . . .	13
1.3.2 Perintah Dasar . . . . .	14
1.3.3 <i>String</i> . . . . .	15
1.3.4 Vektor, Matriks dan Operasinya . . . . .	16
1.3.5 Fungsi . . . . .	18
1.3.6 Paket . . . . .	20
1.3.7 Plot Grafik Fungsi . . . . .	20
1.3.8 <i>Looping</i> dan <i>Conditioning</i> . . . . .	21
1.4 Latihan-Latihan . . . . .	23
1.4.1 Ulasan Materi . . . . .	23
1.4.2 Soal Pemrograman . . . . .	24
<b>2 Loss of Significant Digit dan Representasi Bilangan</b>	<b>27</b>
2.1 <i>Loss of Significance Digit</i> . . . . .	27
2.2 Representasi Bilangan . . . . .	29
2.2.1 Bilangan Desimal dan Biner . . . . .	29
2.2.2 <i>Floating-Point Number</i> . . . . .	33
2.3 Praktikum . . . . .	34
2.3.1 Kepresisionan Terbatas . . . . .	34

2.3.2	<i>Loss of Significant Digit</i>	35
2.3.3	Akurasi Komputer	36
2.4	Latihan-Latihan	37
2.4.1	Ulasan Materi	37
2.4.2	Soal Pemrograman	38
<b>3</b>	<b>Solusi Persamaan Tak-Linear 1</b>	<b>39</b>
3.1	Metode <i>Bisection</i>	40
3.2	Metode <i>Regula-Falsi</i>	42
3.3	Metode Iterasi Titik Tetap	44
3.3.1	Mencari Titik Tetap	45
3.3.2	Eksistensi dan Kekonvergenan	46
3.3.3	Galat Mutlak dan Relatif	47
3.4	Praktikum	48
3.4.1	Metode <i>Bisection</i>	50
3.4.2	Metode <i>Regula Falsi</i>	52
3.4.3	Iterasi Titik Tetap	53
3.5	Latihan-latihan	55
3.5.1	Ulasan Materi	55
3.5.2	Soal Pemrograman	56
<b>4</b>	<b>Solusi Persamaan Tak-Linear 2</b>	<b>59</b>
4.1	Ordo Akar dan Kecepatan Kekonvergenan	59
4.2	Metode Newton-Raphson	60
4.3	Metode <i>Secant</i>	63
4.4	Praktikum	64
4.4.1	Metode Newton-Raphson	66
4.4.2	Metode <i>Secant</i>	68
4.5	Latihan-latihan	70
4.5.1	Ulasan Materi	70
4.5.2	Soal Pemrograman	70
<b>5</b>	<b>Solusi Sistem Persamaan Linear: Metode Langsung</b>	<b>73</b>
5.1	Substitusi Mundur	74
5.2	Eliminasi Gauss	76
5.2.1	Eliminasi Gauss tanpa <i>Pivoting</i>	76
5.2.2	Eliminasi Gauss dengan <i>Pivoting</i>	78
5.3	Faktorisasi <i>LU</i>	79
5.3.1	Faktorisasi <i>LU</i> tanpa <i>pivoting</i>	80
5.3.2	Faktorisasi <i>LU</i> dengan <i>pivoting</i>	83
5.4	Praktikum	87
5.4.1	Metode Substitusi Mundur	90
5.4.2	Metode Eliminasi Gauss	90
5.4.3	Metode Faktorisasi <i>LU</i>	92
5.5	Latihan-latihan	93
5.5.1	Ulasan Materi	93
5.5.2	Soal Pemrograman	94

<b>6 Solusi Sistem Persamaan Linear: Metode Iteratif</b>	<b>97</b>
6.1 Metode Jacobi . . . . .	98
6.2 Metode Gauss-Seidel . . . . .	100
6.3 <b>Metode Rekonstruksi Aljabar</b> . . . . .	103
6.4 <b>Metode Conjugate Gradient</b> . . . . .	105
6.5 Praktikum . . . . .	107
6.5.1 Metode Jacobi . . . . .	109
6.5.2 Metode Gauss-Seidel . . . . .	111
6.5.3 <b>Metode Rekonstruksi Aljabar</b> . . . . .	113
6.5.4 <b>Conjugate Gradient</b> . . . . .	115
6.6 Latihan-latihan . . . . .	116
6.6.1 Ulasan Materi . . . . .	116
6.6.2 Soal Pemrograman . . . . .	117
<b>7 Hampiran Polinomial: Deret Taylor dan Interpolasi</b>	<b>121</b>
7.1 Deret Taylor . . . . .	122
7.2 Interpolasi . . . . .	125
7.2.1 Interpolasi Lagrange . . . . .	128
7.2.2 Interpolasi Newton . . . . .	131
7.3 Praktikum . . . . .	135
7.3.1 Hampiran Deret Taylor . . . . .	136
7.3.2 Interpolasi Linear . . . . .	138
7.3.3 Interpolasi Lagrange . . . . .	139
7.3.4 Interpolasi Newton . . . . .	141
7.4 Latihan-latihan . . . . .	144
7.4.1 Ulasan Materi . . . . .	144
7.4.2 Soal Pemrograman . . . . .	145
<b>8 Pencocokan Kurva (Regresi)</b>	<b>149</b>
8.1 Regresi Linear . . . . .	151
8.2 Regresi Pangkat . . . . .	153
8.3 Linearisasi Data . . . . .	154
8.4 Regresi Polinomial . . . . .	156
8.5 Praktikum . . . . .	159
8.5.1 Regresi Linear . . . . .	160
8.5.2 Regresi Pangkat . . . . .	161
8.5.3 Linearisasi Data . . . . .	162
8.5.4 Regresi Polinomial . . . . .	163
8.6 Latihan-latihan . . . . .	164
8.6.1 Ulasan Materi . . . . .	164
8.6.2 Soal Pemrograman . . . . .	165
<b>9 Turunan Numerik</b>	<b>169</b>
9.1 Hampiran Turunan . . . . .	169
9.2 Formula Beda-Pusat . . . . .	170
9.3 Ekstrapolasi Richardson . . . . .	173
9.4 Praktikum . . . . .	175
9.4.1 Hampiran dari Turunan . . . . .	176
9.4.2 Formula Beda Pusat . . . . .	177

9.4.3	Ekstrapolasi Richardson . . . . .	179
9.5	Latihan-Latihan . . . . .	180
9.5.1	Ulasan Materi . . . . .	180
9.5.2	Soal Pemrograman . . . . .	181
<b>10</b>	<b>Integral Numerik 1</b>	<b>183</b>
10.1	Formula Kuadratur . . . . .	184
10.2	Aturan Komposit . . . . .	191
10.3	Praktikum . . . . .	193
10.3.1	Formula Kuadratur . . . . .	194
10.3.2	Aturan Komposit . . . . .	195
10.3.3	Analisis Galat Aturan Komposit . . . . .	195
10.4	Latihan-latihan . . . . .	197
10.4.1	Ulasan Materi . . . . .	197
10.4.2	Soal Pemrograman . . . . .	198
<b>11</b>	<b>Integral Numerik 2</b>	<b>201</b>
11.1	Aturan Rekursif dan Integral Romberg . . . . .	202
11.2	Kuadratur Adaptif . . . . .	208
11.3	Praktikum . . . . .	213
11.3.1	Aturan Rekursif . . . . .	215
11.3.2	Aturan Romberg . . . . .	216
11.3.3	Aturan Kuadratur Adaptif . . . . .	217
11.4	Latihan-latihan . . . . .	219
11.4.1	Ulasan Materi . . . . .	219
11.4.2	Soal Pemrograman . . . . .	220
<b>12</b>	<b>Persamaan Differensial Biasa 1</b>	<b>223</b>
12.1	Pendahuluan Persamaan Differensial . . . . .	224
12.2	Metode Euler . . . . .	225
12.3	Metode Heun . . . . .	228
12.4	Metode Deret Taylor . . . . .	230
12.5	Metode Runge-Kutta . . . . .	232
12.6	Praktikum . . . . .	234
12.6.1	Metode Euler . . . . .	236
12.6.2	Metode Heun . . . . .	238
12.6.3	Metode Deret Taylor Orde-4 . . . . .	239
12.6.4	Metode Runge-Kutta Orde-4 . . . . .	241
12.7	Latihan-latihan . . . . .	243
12.7.1	Ulasan Materi . . . . .	243
12.7.2	Soal Pemrograman . . . . .	244
<b>13</b>	<b>Persamaan Differensial Biasa 2</b>	<b>247</b>
13.1	Metode Runge-Kutta-Fehlberg 4/5 . . . . .	248
13.2	Sistem Persamaan Differensial . . . . .	250
13.3	Masalah Nilai Batas . . . . .	253
13.3.1	Metode <i>Linear Shooting</i> . . . . .	254
13.3.2	Metode <i>Finite-Difference</i> . . . . .	256
13.4	Praktikum . . . . .	258

13.4.1 Metode RKF45 . . . . .	261
13.4.2 Solusi Sistem Persamaan Diferensial . . . . .	262
13.4.3 Solusi Masalah Nilai Awal Orde Tinggi . . . . .	264
13.4.4 Solusi Masalah Nilai Batas . . . . .	265
13.5 Latihan-latihan . . . . .	268
13.5.1 Ulasan Materi . . . . .	268
13.5.2 Soal Pemrograman . . . . .	269
<b>14 Persamaan Diferensial Parsial</b>	<b>273</b>
14.1 Persamaan Hiperbolik . . . . .	273
14.2 Persamaan Parabolik . . . . .	277
14.3 Persamaan Eliptik . . . . .	280
14.4 Praktikum . . . . .	285
14.4.1 Persamaan Hiperbolik . . . . .	287
14.4.2 Persamaan Parabolik . . . . .	289
14.4.3 Persamaan Eliptik . . . . .	290
14.5 Latihan-Latihan . . . . .	296
14.5.1 Ulasan Materi . . . . .	296
14.5.2 Soal Pemrograman . . . . .	296
<b>Daftar Pustaka</b>	<b>299</b>

---

## PRAKATA

Semua skrip fungsi algoritma dan notebook pendamping praktikum dapat diunduh atau menambahkan *package* Julia dari laman <https://github.com/mkhoirun-najiboi/metnum.jl>

---

---

## BAB 1

---

# METODE NUMERIK DAN ANALISIS GALAT

Pada berbagai cabang disiplin ilmu, suatu permasalahan sering kali melibatkan model matematika, seperti persamaan tak linear, sistem persamaan linear, hingga persamaan differensial. Meskipun solusi dari beberapa permasalahan tersebut dapat dicari menggunakan metode analitik seperti aturan aljabar, namun sering kali permasalahan terlalu rumit dan kompleks, sehingga sulit untuk dicari solusi eksaknya menggunakan metode analitik. Sebagai contoh, suatu persamaan tak-linear

$$f(x) = ax^2 + bx + c$$

dapat dicari nilai akar persamaannya ketika  $f(x) = 0$  menggunakan rumus  $abc$  yaitu

$$x_{12} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Akan tetapi, persamaan yang memiliki pangkat tinggi, seperti

$$g(x) = 2x^7 - 5x^6 + 11x^4 + 3x^3 - 17x^2 - 107$$

atau persamaan dengan fungsi matematika yang rumit, seperti

$$h(x) = \exp(2x) \sin(3x)$$

akan sangat sulit untuk dicari solusi akar persamaannya secara analitik. Selain akar persamaan tak-linear, contoh lain permasalahan matematika yang sulit untuk dicari solusi analitiknya adalah persamaan integral-tentu. Misalkan, terdapat integral-tentu

$$\int_{-\pi/2}^{\pi/2} \sqrt{45.2^x \frac{\sin x \tan 2x}{\ln |x|}} dx$$

Persamaan integral tersebut sangat sulit untuk dicari solusi analitiknya karena tidak ada teknik integrasi yang dapat digunakan untuk menyelesaikan integral serumit itu.

Berdasarkan beberapa contoh di atas, metode analitik hanya dapat digunakan untuk beberapa permasalahan matematika. Bahkan, sebagian besar masalah matematika tidak dapat dicari solusi analitiknya. Metode analitik hanya unggul untuk beberapa masalah matematika yang sederhana dan berdimensi rendah. Pada kenyataannya, permasalahan matematika akan melibatkan bentuk dan proses yang rumit. Dengan demikian, diperlukan cara lain yang lebih sederhana untuk mencari solusi dari permasalahan tersebut.

## 1.1 Metode Numerik

Metode numerik merupakan proses untuk merancang dan menganalisa suatu algoritma yang bertujuan untuk menyelesaikan permasalahan model matematika di berbagai macam disiplin ilmu. Metode numerik atau analisis numerik juga sering disebut komputasi ilmiah (*scientific computing*), karena sebagian besar metode numerik diselesaikan menggunakan komputer digital. Sejak penemuan komputer hingga tahun 1980-an (sebelum adanya komputer pribadi), sebagian besar komputer digunakan para ilmuan untuk memproduksi kode-kode numerik dibandingkan menjadi konsumen dari aplikasi kode-kode tersebut.

Beberapa kode algoritma yang paling populer pada abad 20 dan masih digunakan hingga sekarang antara lain, pada tahun 1946, John von Neumann, Stan Ulam dan Nick Metropolis dari laboratorium sains Los Alamos menyusun algoritma Metropolis yang sekarang lebih dikenal dengan metode Monte Carlo. Pada tahun 1947, George Dantzig dari RAND Corporation menciptakan metode Simplex untuk pemrograman linear. Pada tahun 1951, Alston Householder dari laboratorium nasional Oak Ridge memformulasikan tentang pendekatan dekomposisi untuk komputasi matriks, salah satunya yaitu dekomposisi LU. Hasil dari apa yang mereka peroleh dapat kita rasakan hingga saat ini, antara lain model pergerakan angin tornado di Amerika Serikat, sistem peringatan dini terjadinya tsunami di Indonesia, daftar rekomendasi di halaman awal aplikasi belanja anda, hingga permainan-permainan elektronik pada ponsel anda.

### 1.1.1 Perbedaan Metode Analitik dan Numerik

Perbedaan utama dari metode analitik dan numerik adalah bentuk dari solusi yang dihasilkan. Metode analitik akan menghasilkan solusi dalam bentuk formula matematika. Artinya, jika fungsi dievaluasi pada suatu titik, maka akan diperoleh solusi berupa angka. Sementara itu, metode numerik akan menghasilkan solusi dalam bentuk angka dan tidak terdapat solusi berupa formula matematika. Selain itu, perbedaan antara metode analitik dan numerik adalah akurasi dari solusi yang dihasilkan. Solusi analitik bersifat eksak, sehingga tidak memiliki galat atau kesalahan. Sebaliknya, solusi numerik bersifat hampiran, sehingga solusi numerik memiliki galat atau kesalahan. Supaya lebih jelas, perhatikan contoh penggunaan metode analitik dan numerik untuk menyelesaikan integral-tentu berikut.

**Contoh 1.1 :** Misalkan terdapat integral tentu

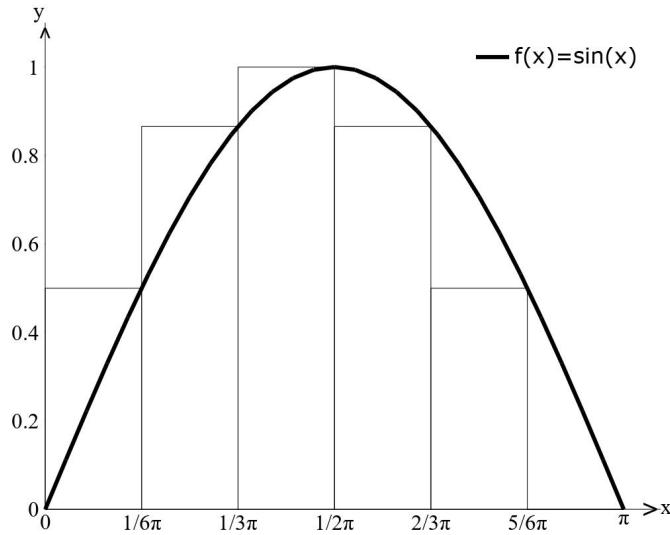
$$y = \int_0^{\pi} \sin(x) dx$$

Berdasarkan teorema dasar kalkulus kedua, masalah integral tersebut dapat diselesaikan dengan cara

$$\begin{aligned} \int_0^{\pi} \sin(x) dx &= (-\cos(x))|_0^{\pi} \\ &= -\cos(0) + \cos(\pi) \\ &= 2 \end{aligned}$$

Sementara itu, jika menggunakan pendekatan metode numerik, integral tentu tersebut dapat diselesaikan tanpa mencari fungsi anti-turunannya. Salah satu metode numerik yang dapat digunakan adalah aturan segi-empat atau dalam kalkulus biasa disebut sebagai

jumlah Riemann. Jumlah Riemann menghitung nilai integral tentu sebagai jumlah dari luas segi-empat dari sub-interval integral tentu. Perhatikan Gambar 1.1 berikut.



**Gambar 1.1:** Integral jumlah Riemann untuk  $f(x) = \sin(x)$

Gambar 1.1 menunjukkan ilustrasi integral numerik dari  $f(x) = \sin(x)$  pada interval  $[0, \pi]$  dengan 6 sub-interval. Dengan pendekatan jumlah Riemann, hampiran numerik untuk  $\int_0^\pi f(x) dx$  adalah

$$\begin{aligned} \int_0^\pi \sin(x) dx &\approx \text{Luas masing-masing persegi} \\ &\approx 0.2618 + 0.4534 + 0.5236 + 0.4534 + 0.2618 + 0. \\ &\approx 1.9541 \end{aligned}$$

Berdasarkan hasil di atas, solusi analitik menghasilkan fungsi berupa persamaan anti-turunan yaitu  $F(x) = -\cos(x)$ , dimana jika dievaluasi pada interval  $[0, \pi]$ , solusi analitik akan bernilai  $\int_0^\pi f(x) dx = 2$ . Sementara itu, proses numerik hanya menghasilkan solusi berupa angka yaitu  $y = 1.9541$  dan memiliki selisih terhadap nilai analitiknya sebesar 0.0459 yang disebut galat.

Hasil solusi numerik ini dapat diperbaiki dengan cara memperbanyak sub-interval dari fungsi  $f(x)$  atau dengan menggunakan metode yang lebih canggih, sehingga galat yang dihasilkan dapat lebih mendekati solusi analitiknya dan memenuhi toleransi yang diberikan.  $\triangle$

Konsep pada Contoh 1.1 dapat diterapkan untuk menyelesaikan masalah integral-tentu yang lebih rumit dan kompleks dengan cara yang lebih mudah mudah dan sederhana tanpa harus mencari solusi dalam bentuk persamaan matematika.

### 1.1.2 Ide Dasar dan Tahapan Metode Numerik

Ide dasar metode numerik adalah mengubah masalah yang kompleks dan rumit menjadi masalah yang lebih sederhana dan mudah, tetapi masih menghasilkan solusi yang sama atau setidaknya mendekati solusi eksaknya. Ide dasar tersebut dapat diterapkan dengan beberapa cara, antara lain

- Mengganti ruang dimensi yang tak-terbatas menjadi ruang dimensi yang terbatas,
- Mengganti deret tak-hingga menjadi hingga, termasuk diantaranya yaitu mengganti integral menjadi jumlah terbatas atau turunan menjadi beda-hingga,
- Mengganti persamaan differensial menjadi persamaan aljabar,
- Mengganti persamaan tak-linear menjadi linear,
- Mengganti sistem persamaan berordo tinggi menjadi berordo-rendah,
- Mengganti fungsi yang rumit menjadi sederhana, seperti fungsi polinomial,
- Mengganti matriks umum menjadi matriks dengan bentuk yang lebih sederhana.

Perlu diingat bahwa solusi yang dihasilkan oleh metode numerik sering kali tidak dapat memberikan jawaban yang sama dengan solusi eksaknya, karena solusi numerik hanya bersifat hampiran. Supaya ide dasar ini bekerja dengan baik dan memberikan solusi yang konvergen menuju solusi eksaknya, maka proses komputasi yang digunakan untuk menyelesaikan masalah yang diberikan harus memiliki

- Bentuk masalah alternatif dari masalah utama, dimana masalah alternatif tersebut lebih mudah untuk dicari solusinya.
- Transformasi masalah utama yang diberikan menjadi masalah alternatif masih mempertahankan solusi eksaknya dalam beberapa hal.

Dengan demikian, akan dibutuhkan usaha untuk mengidentifikasi kelas masalah yang sesuai (dengan solusi yang lebih sederhana) dan bentuk transformasi yang digunakan untuk mempertahankan solusi pada kelas tersebut. Secara umum, proses komputasi metode numerik melibatkan tahapan-tahapan berikut.

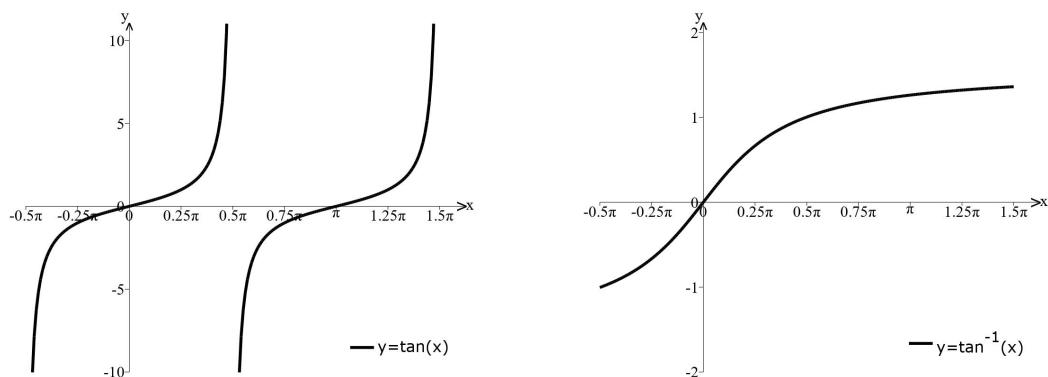
1. Pembentukan model matematika (dalam bentuk persamaan matematika) dari permasalahan nyata atau empiris.
2. Pembentukan algoritma yang dapat digunakan untuk menyelesaikan masalah model matematika secara numerik.
3. Penerjemahan algoritma yang telah disusun dalam suatu perangkat lunak (*software*) komputer.
4. Proses menjalankan program untuk menyelesaikan masalah secara numerik.
5. Proses merepresentasikan hasil dari proses komputasi secara menyeluruh, termasuk diantaranya visualisasi grafik.
6. Interpretasi dan validasi hasil komputasi, dan pengulangan sebagian (atau seluruh) tahapan jika dibutuhkan.

Tahap pertama disebut sebagai tahap pemodelan matematika. Tahap ini membutuhkan dasar ilmu yang kuat dalam matematika terapan, sehingga masalah rumit dan kompleks di dunia nyata dapat diubah menjadi bentuk permasalahan matematika yang lebih sederhana. Tahap kedua, ketiga dan keempat merupakan inti dari metode numerik, yaitu merancang, menganalisa dan menggunakan algoritma numerik (beserta perangkat lunak yang terkait) untuk menyelesaikan masalah matematika. Jika tahap ini terselesaikan dengan baik, maka tahap kelima dan keenam yaitu proses interpretasi dan validasi hasil akan terselesaikan dengan baik pula.

Tahapan-tahapan di atas saling berkaitan satu sama lain dan akan memengaruhi hasil interpretasi pada tahap akhir. Misalnya, jika terdapat kesalahan pada tahap pemodelan, maka hasil akhir solusi numerik mungkin tidak akan sesuai dengan masalah nyata sebelum proses pemodelan. Oleh karena itu, proses tinjauan ulang terhadap tahapan-tahapan komputasi metode numerik sangat diperlukan untuk meminimumkan kesalahan-kesalahan pada proses komputasi. Selain masalah pemodelan, masalah lain yang dapat memengaruhi hasil akhir solusi numerik adalah kondisi dari permasalahan matematika yang akan diselesaikan.

Secara umum, suatu permasalahan matematika dapat dikelompokkan menjadi dua kondisi, yaitu *well-posed* dan *ill-posed*. Masalah dapat dikatakan *well-posed* apabila masalah tersebut memiliki solusi tunggal dan solusi tersebut berubah secara proporsional terhadap perubahan data, artinya perubahan data tidak menyebabkan perubahan solusi secara signifikan dan tiba-tiba. Sebaliknya, jika masalah tersebut tidak memenuhi kondisi *well-posed* maka masalah tersebut memiliki kondisi *ill-posed*. Contoh berikut akan menjelaskan perbedaan masalah matematika yang memiliki kondisi *well-posed* dan *ill-posed*.

**Contoh 1.2 :** Perhatikan grafik fungsi  $f(x) = \tan(x)$  dan  $f(x) = \tan^{-1}(x)$  berikut.



(a) Grafik fungsi  $y = \tan(x)$

(b) Grafik fungsi  $y = \tan^{-1}(x)$

**Gambar 1.2:** Grafik fungsi  $y = \tan(x)$  dan  $y = \tan^{-1}(x)$

Berdasarkan Gambar 1.2a di atas, fungsi  $y = \tan(x)$  merupakan masalah matematika dengan kondisi *ill-posed* di sekitar  $x = 0.5\pi$ , karena perubahan nilai  $x$  di sekitar titik  $x = 0.5\pi$  dapat mengakibatkan terjadinya perubahan nilai  $y$  secara signifikan. Perhatikan bahwa nilai  $\tan(1.56) = 92.62$ ,  $\tan(1.57) = 1255.77$ , dan  $\tan(1.58) = -108.65$ . Dari nilai tersebut, perubahan nilai  $x$  sebesar 0.01 di sekitar  $x = 0.5\pi$  akan mengakibatkan perubahan yang sangat signifikan pada solusi  $y$ . Sebaliknya, Gambar 1.2b merupakan contoh masalah dengan kondisi *well-posed*, karena solusi yang dihasilkan tunggal dan tidak terjadi perubahan solusi secara signifikan dan tiba-tiba. △

Permasalahan matematika dengan kondisi *well-posed* maupun *ill-posed* terjadi secara natural dan tidak disebabkan oleh proses komputasi. Dengan demikian, diperlukan prosedur tertentu untuk menghindari kondisi *ill-posed*. Setelah memerhatikan kondisi dari masalah matematika, proses merancang dan menganalisa suatu algoritma dalam metode numerik akan didasarkan pada dua hal, yaitu

- waktu komputasi dan memori yang terpakai.
- penerapan algoritma, stabilitas dan akurasi solusi.

Pada umumnya, suatu algoritma yang digunakan dalam metode numerik tidak dapat memenuhi kedua kriteria tersebut dengan memuaskan secara bersamaan, sehingga sering kali terjadi pengorbanan salah satu kriteria. Pengorbanan yang dimaksud yaitu mencari akurasi yang baik namun memiliki waktu komputasi yang lama, atau memilih waktu komputasi yang singkat namun memiliki akurasi yang rendah.

## 1.2 Analisis Galat

Pada subbab sebelumnya, sudah dijelaskan bahwa solusi numerik hanya bersifat hampiran dari solusi analitiknya. Dengan demikian, solusi numerik pasti memiliki perbedaan hasil terhadap solusi analitiknya. Perbedaan antara solusi numerik dan solusi analitik tersebut sering disebut dengan galat. Semakin kecil nilai galat yang dihasilkan oleh solusi numerik, maka semakin akurat solusi numerik tersebut dengan solusi analitiknya. Oleh karena itu, analisis galat menjadi perhatian utama dalam metode numerik.

Secara etimologi, galat merupakan kata serapan bahasa arab yaitu *galat* dari kata *galiya* yang artinya 'keliru'. Dalam kamus besar bahasa Indonesia (KBBI), galat diartikan sebagai kekeliruan, kesalahan, atau kecacatan. Dalam makna komputasi, galat merupakan nilai atau kondisi yang tidak konsisten dengan nilai yang benar.

Pada bagian ini, akan dipelajari beberapa sumber dan jenis-jenis galat yang terjadi dalam metode numerik. Pemahaman tentang analisis galat sangat dibutuhkan dalam menginterpretasikan solusi dari metode numerik untuk menunjukkan bahwa solusi yang dihasilkan dapat diterima atau tidak.

### 1.2.1 Sumber-Sumber Galat

Secara umum, sumber dari galat metode numerik dapat dibedakan menjadi dua sumber, yaitu galat yang terjadi sebelum proses komputasi dan galat yang terjadi selama proses komputasi. Beberapa hal yang dapat menyebabkan galat sebelum proses komputasi antara lain:

- **Pemodelan.**

Sebagian besar masalah matematika merupakan pemodelan dari masalah rumit yang terjadi di dunia nyata. Pemodelan merupakan proses untuk membangun suatu masalah tiruan yang merepresentasikan masalah sebenarnya. Dengan demikian, terdapat kemungkinan adanya penyederhanaan dan penghilangan beberapa detail masalah yang nyata.

- **Pengukuran.**

Galat akibat pengukuran yang dimaksud adalah terbatasnya kepresisian alat ukur untuk menghitung suatu data. Dengan demikian, data mungkin memiliki akurasi yang terbatas.

- **Komputasi sebelumnya.**

Data yang akan digunakan mungkin dihasilkan oleh proses komputasi sebelumnya yang hasilnya hanya hampiran dari data sebenarnya.

Sumber galat di atas, biasanya tidak dapat dikontrol karena terjadi secara alami, namun galat tersebut memiliki peranan penting untuk menghasilkan galat akhir dari suatu proses komputasi. Sementara itu, galat yang terjadi pada saat berlangsungnya proses komputasi antara lain:

- **Galat pemotongan (*truncation error*).**

Galat ini disebabkan oleh penyederhanaan atau pemotongan suatu prosedur atau formula matematika, misalnya pemotongan deret tak-hingga, mengganti masalah turunan dengan beda-hingga, atau mengakhiri iterasi algoritma sebelum konvergen.

- **Galat pembulatan (*round-off error*).**

Galat ini disebabkan oleh keterbatasan komputer untuk merepresentasikan sistem bilangan desimal dalam sistem bilangan biner atau sebaliknya.

Galat selama proses komputasi atau yang sering disebut galat komputasi (*computational error*) merupakan jumlah antara galat pemotongan dan pembulatan.

Galat akhir dari suatu komputasi dapat terjadi karena beberapa (atau semua) sumber galat yang disebutkan di atas. Galat dari proses komputasi juga dapat diperkuat oleh kondisi masalah yang akan diselesaikan (*well-posed* atau *ill-posed*) maupun algoritma yang sedang digunakan. Ilmu yang mempelajari tentang pengaruh sumber dan jenis-jenis galat terhadap tingkat akurasi dan stabilitas suatu metode numerik sering dikenal sebagai analisis galat (*error analysis*).

**Contoh 1.3 Sumber Galat:** Luas permukaan bumi dapat dimodelkan menggunakan luas permukaan bola yaitu

$$L = 4\pi r^2 \quad (1.1)$$

Misalkan, bumi memiliki jari-jari sebesar  $r \approx 6370$  km. Maka, sumber galat penghitungan luas permukaan bumi melibatkan galat dari:

- **Pemodelan:** Bumi tidak sepenuhnya bulat, tetapi untuk menyederhanakan masalah, model luas permukaan bumi diasumsikan berbentuk bulat.
- **Pengukuran dan komputasi sebelumnya:** Tidak semua tempat di bumi memiliki jari-jari 6370 km dan angka jari-jari tersebut diperoleh dari hampiran menggunakan komputasi yang dihitung oleh peneliti sebelumnya.
- **Pemotongan (*truncation error*):** Bilangan  $\pi$  merupakan bilangan irrasional yang didapatkan dari perbandingan antara keliling lingkaran ( $K$ ) dengan diameternya ( $D$ ). Bilangan  $\pi$  memiliki angka desimal yang tak terhitung banyaknya dan tidak mungkin untuk dituliskan semua. Salah satu cara untuk menghitung nilai  $\pi$  adalah menggunakan deret tak-hingga, antara lain menggunakan deret Gregory-Leibniz pada persamaan berikut:

$$\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \dots \quad (1.2)$$

- **Pembulatan (*round-off error*):** Keluaran dari hasil penghitungan yang dilakukan oleh komputer ataupun kalkulator akan terjadi pembulatan.

Akurasi akhir dari Persamaan 1.1 adalah akumulasi dari galat-galat di atas.  $\triangle$

### 1.2.2 Galat Mutlak dan Galat Relatif

Signifikansi dari galat bergantung pada besarnya permasalahan yang diukur. Misalnya, galat sebesar 1 satuan dari penghitungan siswa di satu kelas lebih signifikan dibandingkan galat sebesar 1 satuan dari penghitungan warga di satu kota. Hal inilah yang menjadi konsep dasar galat mutlak dan galat relatif.

**Definisi 1.1 :** Misalkan  $\hat{p}$  merupakan nilai penduga atau hampiran dari  $p$ . Maka, galat mutlak didefinsikan sebagai

$$E_p = |p - \hat{p}| \quad (1.3)$$

■

**Definisi 1.2 :** Misalkan  $\hat{p}$  merupakan nilai penduga atau hampiran dari  $p$ . Maka, galat relatif didefinsikan sebagai

$$R_p = \frac{|p - \hat{p}|}{|p|} \quad (1.4)$$

dimana  $p \neq 0$ .

■

**Contoh 1.4 :** Hitunglah nilai galat mutlak dan relatif dari tiga kasus berikut.

**Kasus 1:** Diberikan  $x = 3.141592$  dan  $\hat{x} = 3.14$ , maka nilai galatnya adalah

$$E_x = |x - \hat{x}| = |3.141592 - 3.14| = 0.001592$$

dan galat relatifnya adalah

$$R_x = \frac{|x - \hat{x}|}{|x|} = \frac{0.001592}{3.141592} = 0.00507$$

**Kasus 2:** Diberikan  $y = 1000000$  dan  $\hat{y} = 999996$ , maka nilai galatnya adalah

$$E_y = |y - \hat{y}| = |1000000 - 999996| = 4$$

dan galat relatifnya adalah

$$R_y = \frac{|y - \hat{y}|}{|y|} = \frac{4}{1000000} = 0.000004$$

**Kasus 3:** Diberikan  $z = 0.000012$  dan  $\hat{z} = 0.000009$ , maka nilai galatnya adalah

$$E_z = |z - \hat{z}| = |0.000012 - 0.000009| = 0.000003$$

dan galat relatifnya adalah

$$R_z = \frac{|z - \hat{z}|}{|z|} = \frac{0.000003}{0.000012} = 0.25$$

Galat relatif dari suatu komputasi dapat diinterpretasikan sebagai suatu persentase, yaitu dengan cara mengalikan dengan angka 100. Pada kasus 1, nilai galat mutlak dan galat relatif tidak jauh berbeda, serta nilai  $\hat{x}$  dapat dikatakan baik sebagai penduga nilai  $x$  dengan galat relatif sebesar 0.5%. Sementara itu, kasus 2 memiliki nilai galat mutlak yang paling tinggi dibandingkan dua kasus lainnya, namun  $\hat{y}$  memiliki nilai galat relatif galat paling kecil. Dengan demikian,  $\hat{y}$  dapat menjadi nilai penduga atau hampiran yang baik bagi  $y$ . Sebaliknya, kasus 3 memiliki nilai galat mutlak yang terkecil, namun memiliki galat relatif terbesar yaitu 25%. Artinya,  $\hat{z}$  merupakan penduga yang buruk bagi  $z$ .  $\triangle$

Interpretasi lain dari galat relatif suatu komputasi adalah jika suatu penduga memiliki galat relatif disekitar  $10^{-d}$  maka dalam format desimal penduga memiliki  $d$  angka penting yang benar.

**Definisi 1.3 :** Nilai  $\hat{p}$  dapat dikatakan sebagai penduga dari nilai  $p$  dengan ketepatan hingga  $d$  angka penting, jika  $d$  merupakan bilangan positif terbesar yang memenuhi pertidaksamaan

$$\frac{|p - \hat{p}|}{|p|} < \frac{10^{-d}}{2} \quad (1.5)$$

■

**Contoh 1.5 :** Berdasarkan Contoh 1.4, banyaknya angka penting yang dapat dihampiri oleh penduga pada masing-masing kasus adalah

**Kasus 1.** Nilai  $R_x = 0.000507 < 10^{-2}/2$ , sehingga  $\hat{x}$  memiliki ketepatan 2 angka penting.

**Kasus 2.** Nilai  $R_y = 0.000004 < 10^{-5}/2$ , sehingga  $\hat{y}$  memiliki ketepatan 5 angka penting.

**Kasus 3.** Nilai  $R_z = 0.25 < 10^{-0}/2$ , sehingga  $\hat{z}$  memiliki ketepatan 0 angka penting.  $\triangle$

Hubungan antara galat mutlak dan relatif dapat dituliskan sebagai

$$\text{Nilai Hampiran} = \text{Nilai Sebenarnya} \times (1 + \text{Galat Relatif})$$

Karena nilai sebenarnya pada umumnya tidak diketahui, metode numerik seringkali hanya memperkirakan atau membatasi nilai galat daripada menghitung nilai galat dengan tepat. Dengan alasan yang sama, nilai galat relatif pada Definisi 1.2 seringkali menggunakan pembagi nilai hampiran daripada nilai sebenarnya.

### 1.2.3 Galat Maju dan Galat Mundur

Salah satu cara untuk menilai kualitas hasil proses komputasi adalah dengan mencoba memperkirakan besarnya galat relatif dari galat maju.

**Definisi 1.4 :** Misalkan akan dihitung nilai  $y = f(x)$ , dimana  $f : \mathbb{R} \rightarrow \mathbb{R}$ , serta  $\hat{y}$  merupakan nilai hampiran  $y$ . Maka, galat maju (*forward error*) didefinisikan sebagai

$$\Delta y = \hat{y} - y \quad (1.6) \quad \blacksquare$$

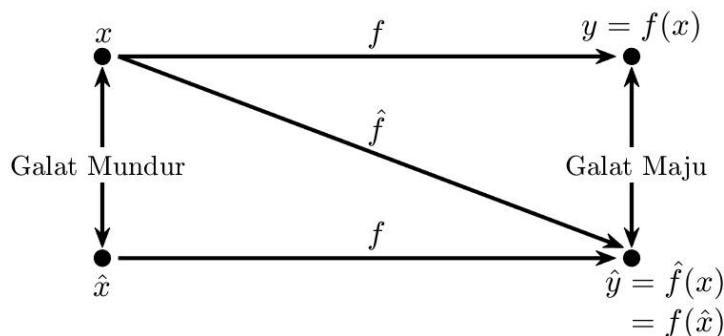
Secara umum, analisis galat maju dalam komputasi sering kali sulit untuk dihitung, karena alasan yang akan kita lihat nanti. Pendekatan lain yang dapat digunakan adalah galat mundur. Analisis galat mundur memandang solusi hampiran sebagai solusi eksak untuk masalah termodifikasi, kemudian hitung seberapa besar modifikasi untuk masalah asli yang diperlukan untuk memberikan hasil yang diharapkan. Dengan kata lain, galat mundur memandang banyaknya galat data pada input awal yang diperlukan untuk menjelaskan semua kesalahan pada hasil akhir.

**Definisi 1.5 :** Galat mundur (*backward error*) didefinisikan sebagai

$$\Delta x = \hat{x} - x \quad (1.7) \quad \blacksquare$$

dimana  $f(\hat{x}) = \hat{y}$ .

Dari sudut pandang ini, solusi hampiran untuk masalah yang asli dapat dikatakan baik, apabila solusi tersebut adalah solusi yang tepat untuk masalah termodifikasi 'terdekat', yaitu memiliki galat mundur relatif yang kecil. Hubungan antara galat maju dan galat mundur dapat dilihat pada Gambar 1.3 berikut.



**Gambar 1.3:** Hubungan antara galat maju dan mundur

**Contoh 1.6 Galat maju dan mundur:** Diketahui nilai  $y = \sqrt{2}$  dan nilai hampiran  $\hat{y} = 1.4$ . Maka nilai galat maju mutlak dari  $\hat{y}$  adalah

$$|\Delta y| = |\hat{y} - y| = |1.4 - 1.41421\ldots| \approx 0.0142$$

dan galat maju relatif

$$\frac{|\Delta y|}{|y|} = \frac{|\hat{y} - y|}{|y|} = \frac{|1.4 - 1.41421\ldots|}{|1.41421\ldots|} \approx 0.01$$

Untuk menghitung nilai galat mundur, perhatikan bahwa  $\sqrt{1.96} = 1.4$ , sehingga nilai galat mundur mutlak dari  $\hat{x}$  adalah

$$|\Delta x| = |\hat{x} - x| = |1.96 - 2| = 0.04$$

dan galat mundur relatif

$$\frac{|\Delta x|}{|x|} = \frac{|\hat{x} - x|}{|x|} = \frac{|1.96 - 2|}{|2|} = 0.02$$

△

#### 1.2.4 Sensitifitas

Solusi numerik dengan nilai akurasi yang rendah belum tentu disebabkan oleh kesalahan algoritma. Akan tetapi, mungkin disebabkan oleh kondisi awal dari masalah yang akan diselesaikan, yaitu masalah dengan kondisi *well-posed* atau *ill-posed*. Dengan demikian, masalah dengan kondisi *ill-posed* akan memiliki sensitifitas yang tinggi terhadap perubahan data masukan.

Masalah matematika disebut **tak-sensitif** (*well-conditioned*), apabila perubahan pada data masukan mengakibatkan perubahan yang sepadan pada solusi akhir. Sebaliknya, masalah matematika dapat disebut **sensitif** (*ill-conditioned*), apabila perubahan pada data masukan mengakibatkan perubahan yang signifikan pada solusi akhir. Kesensitifan suatu masalah matematika dapat ditentukan menggunakan *condition number* yaitu nilai rasio galat maju terhadap galat mundur yang didefinisikan sebagai berikut.

**Definisi 1.6 :** Jika  $\hat{x}$  adalah nilai hampiran dari  $x$ , serta  $\hat{y} = f(\hat{x})$  adalah nilai hampiran bagi  $y = f(x)$ . Maka, nilai kondisi atau *condition number* (*CN*) didefinisikan sebagai

$$CN = \frac{|(f(\hat{x}) - f(x))/f(x)|}{|(\hat{x} - x)/x|} = \frac{|(\hat{y} - y)/y|}{|(\hat{x} - x)/x|} = \frac{|\Delta y/y|}{|\Delta x/x|} \quad (1.8)$$

■

Dengan menggunakan definisi nilai kondisi pada Persamaan 1.8, masalah dapat disebut sensitif atau *ill-conditioned* jika memiliki nilai kondisi yang jauh lebih besar dari 1. Nilai kondisi merupakan rasio antara galat maju relatif dan galat mundur relatif, dengan hubungan seperti berikut.

$$|\text{Galat maju relatif}| = \text{Nilai kondisi} \times |\text{Galat mundur relatif}|$$

Seringkali, nilai kondisi pada Definisi 1.6 tidak berlaku atau tidak dapat dipakai, karena nilai  $y$  yang tidak diketahui. Dengan demikian, diperlukan suatu modifikasi pada persamaan 1.8, sehingga nilai kondisi dapat dihitung tanpa menggunakan nilai  $y$ .

Dengan formula turunan pertama pada kalkulus, nilai kondisi dapat dihitung nilai hampirannya. Misalkan, terdapat fungsi  $f : \mathbb{R} \rightarrow \mathbb{R}$ . Maka, turunan pertama dari fungsi  $f$  dapat dihampiri dengan persamaan

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Katakanlah  $h = \Delta x$ , maka persamaan di atas dapat ditulis sebagai

$$f'(x) \approx \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Jika  $\Delta x = \hat{x} - x$ , maka

$$\begin{aligned} f'(x) &\approx \frac{f(\hat{x}) - f(x)}{\Delta x} \\ \Leftrightarrow f'(x)\Delta x &\approx f(\hat{x}) - f(x) \\ \Leftrightarrow \frac{f'(x)\Delta x}{f(x)} &\approx \frac{f(\hat{x}) - f(x)}{f(x)} \end{aligned}$$

Dengan demikian, nilai kondisi ( $CN$ ) pada Persamaan 1.8 dapat dihampiri menggunakan

$$CN \approx \left| \frac{|f'(x)\Delta x/f(x)|}{|\Delta x/x|} \right| \approx \left| \frac{xf'(x)}{f(x)} \right|$$

**Contoh 1.7 Nilai Kondisi:** Diketahui suatu fungsi  $f(x) = \sqrt{x}$ . Jika  $f'(x) = \frac{1}{2\sqrt{x}}$ , maka nilai kondisi dari  $f$  adalah

$$CN \approx \left| \frac{xf'(x)}{f(x)} \right| = \frac{x/(2\sqrt{x})}{\sqrt{x}} = \frac{1}{2}$$

Nilai ini memiliki arti bahwa perubahan relatif pada keluaran adalah sekitar setengah dari perubahan relatif pada data masukan.

**Tabel 1.1:** Perbandingan nilai galat maju dan mundur relatif

$x$	$\hat{x}$	Galat Mundur Relatif	$y$	$\hat{y}$	Galat Maju Relatif	Rasio
1.0	2	1.000	1.000	1.414	0.414	0.414
1.1	2	0.818	1.049	1.414	0.348	0.426
1.2	2	0.667	1.095	1.414	0.291	0.436
1.3	2	0.538	1.140	1.414	0.240	0.446
1.4	2	0.429	1.183	1.414	0.195	0.456
1.5	2	0.333	1.225	1.414	0.155	0.464
1.6	2	0.250	1.265	1.414	0.118	0.472
1.7	2	0.176	1.304	1.414	0.085	0.480
1.8	2	0.111	1.342	1.414	0.054	0.487
1.9	2	0.053	1.378	1.414	0.026	0.494

Berdasarkan tabel 1.1, nilai galat maju relatif sekitar setengah dari nilai galat mundur relatif. Artinya, masalah  $f(x)$  merupakan masalah yang tak-sensitif atau *well-conditioned*.

△

**Contoh 1.8 Sensitifitas:** Misalkan, diketahui fungsi  $f(x) = \tan(x)$ . Dari Contoh 1.2, diketahui bahwa fungsi  $f$  memiliki kondisi ill-posed. Jika  $f'(x) = \sec^2(x) = 1 + \tan^2(x)$ , maka nilai kondisi  $f(x)$  adalah

$$CN \approx \left| \frac{xf'(x)}{f(x)} \right| = \left| \frac{x \sec^2(x)}{\tan(x)} \right| = \left| x \left( \frac{1}{\tan(x)} + \tan(x) \right) \right|$$

Nilai kondisi akan bernilai sangat besar saat  $x$  disekitar kelipatan  $\pi/2$ , hingga bernilai tak-hingga. Perbandingan antara galat maju dan mundur relatif untuk menghitung nilai  $f(x) = \tan(x)$  saat  $x$  berada disekitar  $x = 1.570$  dapat dilihat pada Tabel 1.2 berikut.

**Tabel 1.2:** Perbandingan antara galat maju dan mundur relatif fungsi  $f(x) = \tan(x)$

$x$	$\hat{x}$	Galat Mundur Relatif	$y$	$\hat{y}$	Galat Maju Relatif	Rasio
1.500	1.570	0.04667	14.10	1256	88.05	1886
1.560	1.570	0.00641	92.62	1256	12.56	1959
1.569	1.570	0.00064	556.69	1256	1.26	1970

Dari tabel 1.2, berdasarkan nilai galat maju dan mundur relatif, dapat dilihat bahwa perubahan kecil pada data masukan akan mengakibatkan perubahan yang besar pada solusi hingga hampir 2000 kali. Dengan demikian, fungsi  $f(x) = \tan(x)$  merupakan fungsi yang sensitif atau *ill-conditioned*.  $\triangle$

### 1.2.5 Stabilitas dan Akurasi

Konsep dasar stabilitas pada proses komputasi suatu algoritma mirip dengan konsep dasar sensitifitas pada masalah matematika, yaitu keduanya mengamati tentang pengaruh dari suatu perubahan. Perbedaannya adalah sensitifitas merujuk pada pengaruh perubahan input data terhadap solusi suatu permasalahan (galat data), sedangkan stabilitas merujuk pada pengaruh galat komputasi terhadap hasil komputasi dari suatu algoritma. Suatu algoritma dapat dikatakan stabil apabila solusi numerik relatif tak-sensitif terhadap galat yang terjadi selama proses komputasi. Dari sudut pandang lain, algoritma akan stabil jika solusi yang dihasilkan merupakan solusi asli dari masalah termodifikasi terdekat, artinya pengaruh dari gangguan selama proses komputasi tidak lebih buruk daripada pengaruh beberapa galat dari input data pada masalah tersebut.

Akurasi dapat diartikan sebagai kedekatan solusi numerik terhadap solusi sebenarnya. Akurasi solusi numerik tidak hanya ditentukan oleh kestabilan algoritma saja, melainkan dipengaruhi pula oleh kondisi dari masalah yang akan diselesaikan. Solusi numerik dapat menjadi tidak akurat karena penggunaan algoritma yang stabil terhadap masalah yang (atau *ill-conditioned*) ataupun penggunaan algoritma yang tak-stabil terhadap masalah yang tak-sensitif (atau *well-conditioned*). Sebaliknya, penggunaan algoritma yang stabil pada masalah yang *well-conditioned* akan menghasilkan solusi yang akurat.

## 1.3 Praktikum: Pengenalan Julia

Komputasi numerik atau yang sering disebut dengan *scientific computing* memerlukan bahasa pemrograman yang memiliki kecepatan tinggi untuk menjalankan setiap algoritma serta menyediakan kenyamanan bagi penggunanya. Bahasa pemrograman tradisional

yang memiliki performa sangat tinggi seperti C merupakan bahasa yang statis, sehingga dapat mengurangi kenyamanan saat penggunaannya. Sebaliknya, bahasa pemrograman dinamis seperti Matlab dan Octave memiliki kinerja yang lambat. Untungnya, pada saat ini, perkembangan desain bahasa modern dan teknik kompilator memungkinkan untuk menghilangkan sebagian besar *trade-off* kinerja tersebut serta menyediakan suatu lingkungan kerja yang produktif dan efisien.

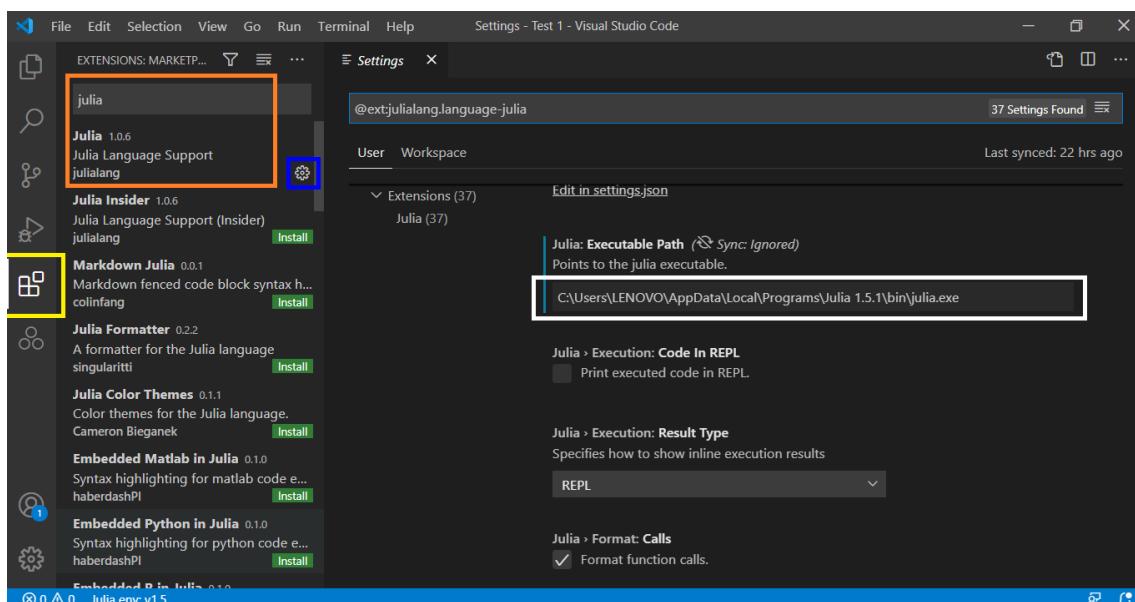
Julia merupakan bahasa pemrograman tingkat-tinggi yang dinamis, sesuai untuk komputasi ilmiah dan numerik, dengan kinerja yang sebanding dengan bahasa tradisional yang diketik secara statis. Perbedaan kompiler dengan interpreter yang ada pada Python maupun R akan menjadikan kinerja Julia menjadi tidak intuitif pada awalnya. Namun, setelah memahami cara kerja Julia, pengguna akan sangat mudah untuk menulis kode yang hampir secepat C.

### 1.3.1 Cara Memasang Julia

Selain keuntungan bahwa Julia dinamis dan cepat, Julia juga merupakan open source sehingga pengguna dapat memperoleh Julia secara gratis tanpa memerlukan lisensi. Julia dapat diunduh melalui situs resmi julialang.org. Selain itu, bahasa Julia juga telah terhubung dengan banyak editor, seperti Visual Studio Code, Atom dan Jupyter Notebook. Berikut merupakan cara untuk menghubungkan bahasa Julia pada Jupyter Notebook dan Visual Studio Code.

#### Menghubungkan Julia pada *Visual Studio Code*

*Visual Studio Code* merupakan *source-code editor* gratis yang diciptakan oleh Microsoft untuk Windows, Linux, dan macOS. Beberapa fitur pada editor ini antara lain *debugging*, penyorotan sintaks, penyelesaian kode, dan *embedded Git*. Pengguna dapat mengubah tema, *shortcut*, preferensi, dan meng-*install* ekstensi yang menambahkan fungsionalitas tambahan.



Gambar 1.4: Menghubungkan Julia pada *Visual Studio Code*

Julia dapat dihubungkan pada *Visual Studio Code* (VSCode) dengan cara meng-*install* ekstensinya. Pertama, pilih menu extensions pada VSCode (lihat kotak kuning pada Gambar 1.4). Selanjutnya, cari ekstensi Julia pada kolom search, kemudian install ekstensi tersebut (lihat kotak merah pada Gambar 1.4). Setelah itu, sesuaikan executable path pada menu *extension setting* (lihat kotak biru dan putih pada Gambar 1.4).

### Menghubungkan Julia pada *Jupyter Notebook*

*Jupyter Notebook* merupakan suatu aplikasi *open-source* yang dapat digunakan untuk membuat dan membagikan dokumen yang mengandung *live code*, persamaan, visualisasi grafik dan teks narasi. Kegunaan *Jupyter Notebook* antara lain pembersihan dan transformasi data, simulasi numerik, pemodelan statistik, visualisasi data, *machine learning*, dan banyak yang lainnya.

Julia dapat dihubungkan ke *Jupyter Notebook* dengan *command* pada Julia menggunakan paket `IJulia.jl`. IJulia adalah suatu *backend* bahasa Julia yang dikombinasikan dengan lingkungan interaktif Jupyter. Kombinasi ini memungkinkan untuk berinteraksi dengan bahasa Julia menggunakan *Jupyter/IPython notebook* yang mengkombinasikan kode, teks terformat, persamaan matematika dan multimedia dalam satu dokumen. Selain itu, juga dimungkinkan untuk bekerja dengan JupyterLab. Untuk menambahkan paket IJulia, berikut merupakan *command* pada Julia.

```
julia> using Pkg
julia> Pkg.add("IJulia")
```

Setelah paket IJulia ter-*install*, paket dapat dipanggil dengan perintah *using*. Selanjutnya, Julia dapat dihubungkan ke *Jupyter notebook* dengan perintah seperti berikut.

```
julia> using IJulia
julia> notebook() # comment
```

Selanjutnya, *Jupyter Notebook* akan secara otomatis dibuka pada *web browser default* komputer masing-masing.

### 1.3.2 Perintah Dasar

Sebagai permulaan, berikut merupakan beberapa *syntax* dasar pemrograman pada Julia, seperti mencetak sesuatu, mendefinisikan *variable* (peubah), menyisipkan komentar, dan menyelesaikan operasi dasar aritmatika. Perintah yang dapat digunakan untuk mencetak atau *print* suatu nilai maupun string adalah `println` seperti berikut.

```
Inp: println("Hello Julia, senang belajar bersama kamu!")
Out: Hello Julia, senang belajar bersama kamu!
```

Jika suatu peubah didefinisikan untuk pertama kali, Julia akan menentukan tipe peubah tersebut. Peubah dapat berupa *number*, *char*, maupun *string*. Untuk mengecek tipe dari suatu peubah, dapat digunakan perintah `typeof`. Berikut merupakan contoh penulisan peubah dan cara mengecek tipe dari peubah tersebut

```
Inp: my_answer = 42
      typeof(my_answer)
Out: Int64
Inp: my_pi = 3.14
      typeof(my_pi)
Out: Float64
```

Selain menggunakan bentuk nama peubah seperti di atas, Contohnya, peubah dengan nama  $\alpha$  dapat dituliskan dengan cara mengetikkan "\alpha" kemudian tekan tombol Tab atau Enter. Dengan demikian, peubah  $\alpha$  dapat didefinisikan seperti berikut.

```
Inp: \alpha = "Ini alpha"
      typeof(\alpha)
Out: String
```

Berdasarkan hasil di atas, peubah pertama memiliki tipe Int64 yaitu bilangan bulat dengan 64 bit, peubah kedua memiliki tipe Float64 artinya *floating number* dengan 64 bit dan terakhir memiliki tipe String.

Selanjutnya, untuk menuliskan suatu komentar pada Julia, terdapat 2 cara yaitu *single line comment* dan *multiple line comment*. Seperti pada bahasa pemrograman lain, komentar yang ditulis akan diabaikan oleh sistem. Berikut merupakan cara penulisan komentar pada Julia.

Setelah memahami cara untuk menuliskan dan mencetak peubah serta komentar, berikut merupakan tanda operasi dasar aritmatika pada Julia.

**Tabel 1.3:** Operasi Aritmatika pada Julia

Operasi	Simbol	Contoh Input	Output
tambah	+	3+7	10
Kurang	-	10-3	7
Kali	*	20*5	100
Bagi	/	100/10	10.0
Pangkat	^	10^2	100
Modulus	%	101%2	1

### 1.3.3 String

String dalam pemrograman adalah sebuah deret simbol. Dengan kata lain, tipe data string adalah tipe data yang digunakan untuk menyimpan barisan karakter. Pada Julia, string dapat didefinisikan menggunakan 2 cara yaitu menggunakan *single quote* "String" dan *triple quote* """String""". Perbedaan penulisan ini digunakan untuk menuliskan string dengan tanda kutip di dalam string tersebut. Berikut merupakan contoh penggunaan *triple quote* untuk menuliskan suatu *string*.

```
Inp: "disini terdapat "error" "
Out: ERROR: syntax: cannot juxtapose string literal
      Stacktrace:
       [1] top-level scope at REPL[16]:1
       [2] include_string(::Function, ::Module, ::String,
                         ::String) at .\ loading.jl:1088
Inp: """ disini tidak terdapat "error" """
Out: disini tidak terdapat \" error \\"
```

Berdasarkan contoh di atas, *input* pertama menghasilkan *error* karena tanda petik yang diberikan bersifat ambigu dan tidak dapat dibedakan awal dan akhirnya. Sementara itu, *input* kedua memberikan kejelasan titik awal dan akhir dari *string* menggunakan *triple quote*.

Selain menuliskan *string* secara utuh, *string* juga dapat disisipi dengan suatu peubah. Suatu peubah dapat disisipkan ke dalam *string* menggunakan tanda \$. Perhatikan contoh berikut.

```
Inp: nama = najib
      jaritangan = 10;
      jarikaki = 10;
Inp: println("Hallo, nama saya $nama")
      println("saya punya $jaritangan jari tangan")
      println("dan $jarikaki jari kaki")
      println("Jadi, total ada $(jaritangan+jarikaki) jari.")
Out: Hallo, nama saya najib
      saya punya 10 jari tangan
      dan 10 jari kaki
      Jadi, total ada 20 jari.
```

Selain disisipkan suatu peubah, suatu *string* juga dapat digabung dengan string lainnya dengan perintah *string*(*s1, s2*) atau tanda \* seperti berikut.

```
Inp: s1 = "berapa harga kucing "
      s2 = "sehingga dikatakan banyak?"
Inp: string(s1,s2)
Out: berapa banyak kucing sehingga dikatakan banyak?
Inp: s1*s2
Out: berapa banyak kucing sehingga dikatakan banyak?
```

### 1.3.4 Vektor, Matriks dan Operasinya

Pada subbab ini, akan dipelajari penggunaan Julia untuk matriks dan vektor. Matriks dibentuk menggunakan kurung siku “[ ]” dengan pemisah kolom menggunakan tanda spasi ” ” dan pemisah baris menggunakan tanda titik-koma “;”. Sementara, suatu vektor dengan jeda yang sama dibentuk menggunakan pola ”nilai awal : jeda : nilai akhir”.

**Tabel 1.4:** Operasi antar elemen dan fungsi yang berkaitan dengan matriks

+ -	: tambah dan kurang
. / . *	: bagi dan kali antar elemen
. ^ . \	: pangkat dan pembagian terbalik antar elemen
A'	: transpos A
size (A)	: dimensi matriks A
length (A)	: panjang vektor A
inv (A)	: invers dari A
det (A)	: determinan dari A
linspace (a, b, n)	: membuat vektor dari titik a sampai b sebanyak n titik
zeros (m, n)	: membuat matriks bernilai nol berukuran $m \times n$
ones (m, n)	: membuat matriks bernilai satu berukuran $m \times n$
eye (n)	: membuat matriks identitas berukuran $n \times n$
rand (m, n)	: membuat matriks acak $m \times n$ antara 0 dan 1

Tabel 1.4 menunjukkan operasi antar elemen dan fungsi yang berkaitan dengan matriks. Berikut merupakan contoh penggunaan beberapa operasi dan fungsi yang berkaitan dengan matriks pada Julia. Ketik dan jalankan beberapa perintah berikut.

```
Inp: A = [1,2,3;3,0,1;2,1,0]
Out: 3x3 Array{Int64,2}:
      1  2  3
      3  0  1
      2  1  0
```

```
Inp: B = [2 1 3]
Out: 1x3 Array{Int64,2}:
      2   1   3

Inp: C = 1:2:5
Out: 1:2:5

Inp: collect(C)
Out: 3-element Array{Int64,1}:
      1
      3
      5

Inp: D = B'
Out: 3x1 LinearAlgebra.Adjoint{Int64,Array{Int64,2}}:
      2
      1
      3

Inp: A*B
Out: DimensionMismatch("matrix A has dimensions (3,3), matrix B has dimensions (1,3)")

Inp: C.*D
Out: 3x1 Array{Int64,2}:
      2
      3
      1.5

Inp: A/B
Out: 3x1 Array{Float64,2}:
      0.9285714285714286
      0.6428571428571428
      0.35714285714285715

Inp: A\B
Out: 3x1 Array{Float64,2}:
      0.5833333333333334
      1.8333333333333335
      -0.75

Inp: inv(A)
Out: 3x3 Array{Float64,2}:
      -0.0833333   0.25   0.166667
      0.166667   -0.5    0.666667
      0.25       0.25   -0.5

Inp: length(C)
Out: 3

Inp: size(A)
Out: (3, 3)

Inp: zeros(1,4)
Out: 1x4 Array{Float64,2}:
      0.0   0.0   0.0   0.0
```

Pada metode numerik, seringkali proses komputasi menggunakan ekspresi indeks untuk mengolah data berbentuk matriks. Ekspresi indeks memungkinkan pengguna Julia untuk menunjuk satu atau beberapa elemen pada suatu vektor atau matriks. Indeks ditulis menggunakan tanda kurung siku “[ ]” setelah nama matriks menggunakan pola berikut:

”nama matriks[indeks baris, indeks kolom]”

Ekspresi indeks pertama dalam Julia selalu dimulai dari angka 1. Berikut merupakan contoh penggunaan ekspresi indeks untuk pengolahan data matriks. Ketik dan jalankan beberapa perintah berikut.

```
Inp: A = rand(-10:20, 3, 4)
Out: 3x4 Array{Int64,2}:
      17   4   -5   14
      18   5    5    2
      4    1   -9   -2

Inp: A[2,1] # Ambil baris 2 kolom 1
Out: 18

Inp: A[3, [2,4]] # Ambil baris 3 kolom 2 dan 4 (perhatikan hasil)
Out: 2-element Array{Int64,1}:
      1
      -2

Inp: A[1,1:3] # Ambil baris 1 kolom 1 sampai 3
Out: 3-element Array{Int64,1}:
      17
      4
      -5

Inp: A[:,1] # Ambil semua baris, kolom 1
Out: 3-element Array{Int64,1}:
      17
      18
      4

Inp: A[1,:]
Out: 4-element Array{Int64,1}:
      17
      4
      -5
      14
```

### 1.3.5 Fungsi

Fungsi dalam Julia merupakan suatu objek yang memetakan suatu argumen ke suatu nilai yang dikehendaki. Fungsi yang dimaksud bukan murni seperti fungsi dalam matematika, karena fungsi disini bisa diubah dan dipengaruhi oleh *global state*. Terdapat beberapa cara penulisan suatu fungsi. Berikut adalah contoh untuk mendefinisikan fungsi  $f(x) = x^2$  pada Julia.

```
Inp: function f(x)
      return x^2
    end
Inp: f(x) = x^2
Inp: f = x -> x^2
```

Ketiga bentuk fungsi di atas setara satu sama lain. Bentuk fungsi pertama yaitu dengan keyword `function` biasanya digunakan untuk fungsi yang panjang dan terdiri atas beberapa/banyak operasi. Sementara itu, bentuk fungsi kedua dan ketiga sering digunakan untuk menuliskan persamaan matematika.

Fungsi tidak harus memiliki *input* berupa angka. Asalkan *input* dapat dipahami, fungsi akan mengeluarkan *output* yang sesuai dengan *input* yang diberikan. Misalkan, fungsi di atas mendefinisikan persamaan  $f(x) = x^2$ . Namun, jika *input* merupakan suatu *string*, maka *output* akan mengeluarkan hasil pengulangan dari *string* tersebut seperti berikut.

```
Inp: f("hi")
Out: "hihi"
```

Terdapat banyak fungsi *built-in* yang tersedia pada Julia. Fungsi-fungsi tersebut dapat tersedia langsung di dalam Julia maupun terdapat pada paket-paket yang terhubung pada Julia. Beberapa contoh fungsi *built-in* tersebut adalah *rand*, *sort*, *maximum*, *minimum* dan sebagainya. Penggunaan fungsi dapat bersifat *mutating* dan tak-*mutating*. *Mutating* yang dimaksud adalah menggunakan fungsi terhadap suatu variabel akan mengubah isi variabel tersebut dan sebaliknya. Fungsi yang bersifat *mutating* biasanya diakhiri dengan tanda ! setelah nama fungsi tersebut. Contoh, `sort(x)` bersifat tak-*mutating*, tetapi `sort!(x)` bersifat *mutating* seperti berikut.

```
Inp: A = [4,1,3]
      sort(A) # sort akan mengurutkan elemen-elemen A
Out: 3-element Array{Int64,1}:
      1
      3
      4

Inp: A          # Namun bersifat tak-mutating, sehingga A tak berubah
Out: 3-element Array{Int64,1}:
      4
      1
      3

Inp: sort!(A) # bentuk mutating dari sort
      A          # Serta, A ikut mengalami perubahan
Out: 3-element Array{Int64,1}:
      1
      3
      4
```

Fungsi  $f(x)$  yang didefinisikan di atas tidak berlaku untuk input berupa struktur data seperti vektor yang disusun dalam suatu arrays. Untuk mengevaluasi fungsi satu per satu terhadap setiap elemen pada struktur data, perintah yang dapat digunakan adalah *map* dan *broadcast*. Perintah *broadcast* biasanya sering disingkat penulisannya dengan menambahkan tanda titik  $\cdot\cdot\cdot$  pada akhir nama fungsi. Berikut merupakan contoh evaluasi fungsi pada suatu array.

```
Inp: map(f,[1 2 3]) # atau
Inp: broadcast(f,[1 2 3]) # atau
Inp: f.([1 2 3])
Out: 1x3 Array{Int64,2}:
      1  4  9
```

### 1.3.6 Paket

Julia memiliki lebih dari 2000 packages yang telah terdaftar yang menjadikan packages merupakan bagian penting dalam berkembang, Julia memiliki paket untuk memanggil fungsi kelas satu dari bahasa lain. Dengan demikian, pengguna dapat dengan mudah memanggil kode dalam bahasa python atau R dengan paket PyCall atau Rcall. Paket yang telah tersedia pada Julia dapat diperiksa pada situs web <https://pkg.julialang.org/> atau <https://juliaobserver.com/>.

Saat pertama kali menggunakan paket pada instalasi Julia tertentu, pengelola paket diperlukan untuk menambahkannya secara eksplisit. Berikut adalah cara untuk menambahkan paket “Example” yang telah tersedia pada Julia.

```
Inp: using Pkg
      Pkg.add("Example")
```

Paket akan diunduh dari server yang bersangkutan, sehingga proses instalasi paket membutuhkan koneksi internet. Selain paket yang telah terdaftar pada Julia, paket yang belum terdaftar pada julia (seperti paket yang tersedia pada GitHub) juga dapat ditambahkan. Contohnya adalah paket VMLS.jl (*Vectors, Matrices, and Least Squares*) yang tersedia pada laman <https://github.com/VMLS-book/VMLS.jl>. Berikut merupakan cara menambahkan paket Julia yang tersedia pada situs web.

```
Inp: Pkg.add(url="https://github.com/VMLS-book/VMLS.jl")
```

Terdapat banyak paket yang dapat digunakan pada Julia. Beberapa paket akan digunakan dan dipelajari sesuai kebutuhan pada bab-bab selanjutnya.

Setiap kali Julia digunakan (memulai sesi baru di REPL, atau membuka notebook untuk pertama kali), pengguna harus *load* paket dengan kata kunci **using**.

```
using Example
```

Salah satu paket yang sangat berguna dalam pengolahan data vektor dan matriks adalah paket `LinearAlgebra`. Jika sintaks berikut dijalankan, maka fungsi determinan tidak tersedia pada Julia secara default.

```
Inp: A = rand(3,3)
      det(A)
Out: UndefVarError: det not defined
```

Akan tetapi, dengan paket `LinearAlgebra`, fungsi determinan dapat dijalankan.

```
Inp: using Pkg
      Pkg.add("LinearAlgebra")
      using LinearAlgebra
Inp: det(A)
Out: 0.1094330733406871
```

Pada praktikum metode numerik ini, kita hanya akan menggunakan paket-paket dasar seperti `LinearAlgebra` untuk melakukan proses komputasi metode-metode numerik yang akan dipelajari pada pertemuan-pertemuan selanjutnya.

### 1.3.7 Plot Grafik Fungsi

Plot data merupakan teknik untuk visualisasi data yang sangat penting pada komputasi numerik. Pada Julia, plot disediakan oleh paket-paket yang telah terdaftar. Contoh, paket plot standar yang digunakan pada Julia adalah paket `Plots`. Selain itu, paket lain yang dapat digunakan adalah paket `UnicodePlots`, `StatsPlots`, `PyPlot`, `VegaLite`,

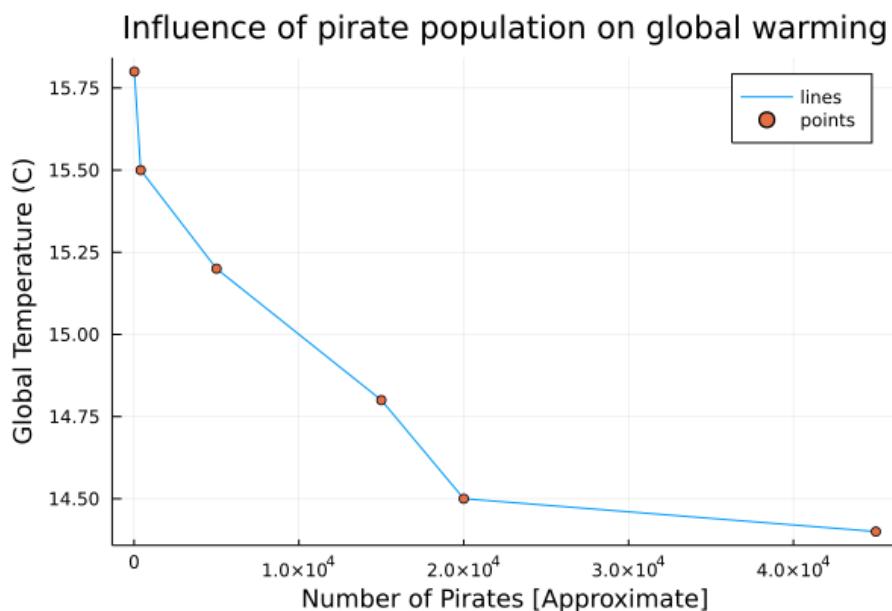
dan lain sebagainya. Berikut merupakan contoh penggunaan paket **Plots** untuk menghasilkan suatu grafik.

```
Inp: using Pkg
      Pkg.add("Plots") # Tambahkan paket jika belum tersedia

Inp: temperatures = [14.4, 14.5, 14.8, 15.2, 15.5, 15.8];
      numpirates = [45000, 20000, 15000, 5000, 400, 17];

Inp: using Plots
      gr() # untuk mendefinisikan backend

Inp: plot(numpirates,temperatures,label="lines")
      scatter!(numpirates,temperatures,label="points")
      xlabel!("Number of Pirates [Approximate]")
      ylabel!("Global Temperature (C)")
      title!("Influence of pirate population on global warming")
Out:
```



### 1.3.8 Looping dan Conditioning

Pada komputasi, *looping* dan *conditioning* merupakan hal yang sangat penting dan sering digunakan. Dua perintah *looping* yang sering digunakan adalah **for** dan **while**, sedangkan perintah *conditioning* yang digunakan adalah **if**.

*Looping* digunakan untuk melakukan suatu proses yang berulang. Terdapat banyak perintah *looping* yang dapat digunakan, namun dua perintah *looping* yang sering digunakan adalah **for** dan **while**. Bentuk umum dari perintah **for** dan **while** dapat dituliskan sebagai berikut.

#### Algoritma 1.1: Bentuk umum perintah **for**

```
for var in loop iteration
    perintah
end
```

**Algoritma 1.2:** Bentuk umum perintah **while**

```
while kondisi
    perintah
end
```

Berikut merupakan contoh penggunaan perintah for dan while untuk mencetak nilai pangkat dari angka 1 sampai 3.

```
Inp: for i in 1:3
      println(i^2)
    end
    # ATAU
Inp: i = 1
    while i<=3
        println(i^2)
        i += 1
    end
Out: 1
    4
    9
```

Untuk perintah *for* yang bertingkat, penulisan kode dapat disingkat menjadi beberapa cara seperti berikut.

```
Inp: A = fill(0, (5,5))
    for i in 1:5
        for j in 1:5
            A[i, j] = i + j
        end
    end

Inp: for i in 1:5, j in 1:5
      A[i, j] = i + j
    end

Inp: A = [i+j for i in 1:5, j in 1:5]
```

Ketiga bentuk koding di atas akan menghasilkan matriks A yang sama yaitu

```
Out: 5x5 Array{Int64,2}:
    2 3 4 5 6
    3 4 5 6 7
    4 5 6 7 8
    5 6 7 8 9
    6 7 8 9 10
```

Sementara itu, perintah *conditioning if* digunakan untuk memilih satu pilihan dari beberapa pilihan yang tersedia berdasarkan kondisi tertentu. Bentuk umum dari **if** dapat dituliskan sebagai berikut.

**Algoritma 1.3:** Bentuk umum perintah **if**

```
if kondisi 1
    perintah 1
elseif kondisi 2
    perintah 2
else
    perintah 3
end
```

Berikut merupakan implementasi untuk pengujian FizzBuzz, yaitu berdasarkan nilai  $N$  yang diberikan, cetak kata "Fizz" bila  $N$  habis dibagi 3, "Buzz" bila  $N$  habis dibagi 5, dan "FizzBuzz" bila  $N$  habis dibagi 3 dan 5. Selainnya hanya cetak nilai  $N$ .

```
Inp: N = 15
    if (N % 3 == 0) && (N % 5 == 0) # "&&" berarti "DAN"
        println("FizzBuzz")
    elseif N % 3 == 0
        println("Fizz")
    elseif N % 5 == 0
        println("Buzz")
    else
        println(N)
    end
Out: FizzBuzz
```

## 1.4 Latihan-Latihan

### 1.4.1 Ulasan Materi

1. Mengapa solusi yang dihasilkan oleh metode numerik sering kali tidak memberikan jawaban yang sama dengan solusi eksaknya? serta, mengapa solusi numerik masih tetap digunakan meskipun tidak sama dengan solusi eksaknya?
2. Dalam komputasi numerik, nilai simbol ( $\pi$ ,  $e$ , dll) dituliskan dalam bentuk ....
3. Apa saja hal-hal yang dapat menyebabkan galat sebelum proses komputasi?
4. Jika pengaruh dari galat selama proses komputasi untuk suatu algoritma tidak lebih buruk daripada pengaruh beberapa galat dari input data, maka algoritma tersebut dikatakan algoritma yang ....
5. Kondisi seperti apa yang menyebabkan suatu masalah memiliki sifat *ill-condition*?
6. Jika kapasitas memori pada komputer tidak mencukupi, maka terjadi galat ....
7. Apakah nilai galat mutlak dari galat pemotongan (*truncation*) lebih besar daripada galat pembulatan (*rounding*)?
8. Manakah pernyataan berikut yang benar dan salah?
  - (a) *Condition Number* (CN) digunakan untuk memeriksa masalah sensitifitas.
  - (b) Akurasi dapat diartikan sebagai kedekatan solusi numerik terhadap solusi sebenarnya.
  - (c) Akurasi tidak memiliki hubungan dengan galat.
  - (d) Masalah yang tak-sensitif terhadap galat berarti bersifat *ill-conditioned*.
  - (e) Galat mutlak dari suatu komputasi bernilai kurang atau sama dengan 1.
  - (f) Galat relatif dari suatu komputasi bernilai kurang atau sama dengan 1.
  - (g) Beberapa metode analitik dihasilkan dari pendekatan metode numerik.
  - (h) Metode numerik selalu dapat memberikan hasil yang pasti.
  - (i) Metode numerik dapat menyelesaikan permasalahan metode analitik.
  - (j) Semakin tinggi akurasi maka semakin lama komputer membutuhkan waktu dalam memecahkan masalah.

### 1.4.2 Soal Pemrograman

1. Tentukan nilai fungsi berikut jika diketahui  $t = 25$ ,  $x = 43$ ,  $y = 15.25$  dan  $z = 8.2$ .

(a)  $M = 4x^2 + 3y + 10$

(b)  $N = e^{2x} + x$

(c)  $O = \sqrt{\frac{1}{x+y} + \frac{1}{t+y}}$

(d)  $P = 4e^{-x/2} \sin(\pi x)$

(e)  $Q = 4 \sin(t/4) + \exp(-xy)$

2. Diketahui  $a = 3/2$ ,  $b = 1/2$  dan  $c = 2/3$

(a)  $\frac{a^2b + ab^2}{b^2 + 2ac}$

(b)  $\sqrt{abc} + \sqrt{a+b+c}$

(c)  $\sin(a) + 2 \ln(b) - 11$

(d)  $\frac{\cos(abc)}{a \sin(bc)}$

(e)  $\frac{\tan^{-1}(a) \sec(c)}{\sin^{-1}(b)}$

3. **Sistem Persamaan Linear.** Definisikan matriks  $A$  dan  $B$  seperti berikut.

$$A = \begin{pmatrix} 3 & -7 & 2 \\ 1 & 3 & 4 \\ 2 & 3 & 8 \end{pmatrix} \quad B = \begin{pmatrix} 29 \\ -3 \\ 3 \end{pmatrix}$$

(a) Hitung solusi  $X$  dari SPL  $AX = B$  menggunakan operator bagi terbalik ' $\backslash$ '.

(b) Hitung solusi  $X$  dari SPL  $AX = B$  menggunakan fungsi invers ' $\text{inv}(A)$ '.

4. **Faktorisasi QR.** Kerjakan langkah-langkah berikut.

(a) Buatlah matriks  $M$  yaitu

$$M = \begin{bmatrix} 2 & 2 \\ 0 & 1 \\ 0 & 3 \end{bmatrix}$$

(b) Dari matriks  $M$  tersebut, bentuklah matriks  $a$  dan  $b$  yaitu matriks kolom ke-1 dan 2 dari  $M$

$$a = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$$

(c) Hitung  $q_1 = \frac{a}{\|a\|}$  (gunakan fungsi  $\text{norm}(a)$  untuk menghitung  $\|a\|$ )

(d) Hitung  $p_1 = b - (b^T q_1) q_1$  dan  $q_2 = \frac{p_1}{\|p_1\|}$

(e) Bentuk matriks  $Q$  yaitu  $Q = [q_1, q_2]$ , Apakah  $Q$  matriks ortogonal?

(f) Hitung nilai  $R$  yaitu  $R = Q^T M$ , Apakah  $R$  matriks segitiga atas? Selain itu, apakah  $M = QR$ ?

**5. Pembangkitan Data.** Kerjakan langkah-langkah berikut ini.

- Tampilkan bantuan dan pelajari fungsi `rand`, `sum`, `repeat`
- Bangkitkan suatu matriks  $A_{10 \times 4}$  (10 baris, 4 kolom) yang berisi bilangan bulat antara 10-100. (Petunjuk: gunakan `rand`)
- Hitung jumlah setiap kolom kemudian simpan sebagai `jumlah`. (Petunjuk: ukuran matriks yang dihasilkan adalah  $1 \times 4$ , gunakan `sum` untuk menghitung jumlah.)
- Hitung rataan data setiap kolom dan simpan sebagai `rataan`. (Petunjuk: ukuran matriks yang dihasilkan  $1 \times 4$ , bagi `jumlah` dengan banyaknya baris.)
- matriksRataan merupakan data rataan yang disusun berulang sebanyak 10 kali. (Petunjuk: ukuran matriks yang dihasilkan  $10 \times 4$ , gunakan `repeat`.)
- $B = A - \text{matriksRataan}$ . Tunjukkan rataan kolom B mendekati 0

**6. Maksimum dan minimum.** Buatlah fungsi yang mengeluarkan nilai maksimum dan minimum dari vektor  $x$  dengan format `maks, mins=maxnmin(x)`. Contoh keluaran fungsi:

```
Inp: x = [-4, -10, 2, 1, 7, -6, 21, 32, 1, 5];
Inp: maks, mins = maxnmin(x)
      maks
Out: 32
Inp: mins
Out: -10
```

**7. Pembulatan.** Tampilkan bantuan dan pelajari fungsi `round`. Buatlah suatu fungsi dengan nama `bulat(x, n)` yaitu fungsi yang dapat membulatkan  $n$ -angka desimal dibelakang koma dari bilangan real  $x$ . Contoh keluaran fungsi:

```
Inp: x = 1.26356863;
Inp: bulat(x, 5)
Out: 1.26357
Inp: bulat(x, 6)
Out: 1.263568
```

**8. Diberikan fungsi**

$$y = f(x) = x^3 - 2$$

dengan garis singgung di  $x = 1$  yaitu  $y = L(x) = 3x - 4$ . Gambarkan grafik fungsi  $f(x)$  dan garis singgung  $L$  pada satu grafik dengan ketentuan:

- grafik  $f(x)$  berupa garis lurus berwarna hitam pada selang  $[-1, 3]$ , dan
- grafik  $L(x)$  berupa garis putus-putus berwarna biru pada selang  $[0, 2]$ .

**9. Diberikan fungsi polinom berikut.**

$$y = f(x) = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1$$

Perhatikan: gunakan *broadcast* dalam mendefinisikan fungsi polinom tersebut di Julia, yaitu gunakan `.` bukan `^` untuk operasi pangkat (kenapa?).

- Plot grafik fungsi  $f(x)$  menggunakan fungsi `plot` pada paket Plots di Julia untuk beberapa kemungkinan selang nilai  $x : 0.9 \leq x \leq 1, 0.985 \leq x \leq 1$ , dan  $0.99 \leq x \leq 1$ . Grafik seperti apa yang dihasilkan? Jelaskan!
- Plot grafik fungsi  $f(x)$  menggunakan fungsi `plot` pada paket PyPlot di Julia untuk beberapa kemungkinan selang nilai  $x : 0.9 \leq x \leq 1, 0.985 \leq x \leq 1$ , dan  $0.99 \leq x \leq 1$  dengan lebar sub-interval  $10^{-4}$  untuk membagi ketiga selang tersebut. Grafik seperti apa yang dihasilkan? Jelaskan!

10. **Segitiga bintang.** Gunakan pernyataan *if, for,* dan/atau *while* untuk membuat bentuk segitiga dari bintang seperti berikut.

(a) \*

  \*\*

  \*\*\*

  \*\*\*\*

(c) \*

  \*\*

  \*\*\*

  \*\*\*\*

(b) \*\*\*\*

  \*\*\*

  \*\*

  \*

(d) \*\*\*\*

  \*\*\*

  \*\*

  \*

11. **Matriks Pascal.** Perhatikan matriks pascal berukuran  $5 \times 5$  berikut ini.

$$P = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 6 & 10 & 15 \\ 1 & 4 & 10 & 20 & 35 \\ 1 & 5 & 15 & 35 & 70 \end{pmatrix}$$

- (a) Dengan mengamati pola antar elemen matriks  $P$ , buatlah fungsi dengan nama `pascal(n)` yang menghasilkan matriks pascal berukuran  $n \times n$ .
- (b) Keluarkan matriks  $P$  yaitu matriks pascal ukuran  $10 \times 10$
- (c) Hitung invers  $P$
- (d) Tunjukkan bahwa  $\det(PP^{-1}) = 1$

---

---

## BAB 2

---

### ***LOSS OF SIGNIFICANT DIGIT DAN REPRESENTASI BILANGAN***

#### ***2.1 Loss of Significance Digit***

Fenomena hilangnya angka penting berkaitan erat dengan tingkat presisi dan akurasi dari solusi numerik. Presisi merujuk pada banyaknya digit yang dapat dituliskan pada suatu bilangan, sedangkan akurasi merujuk pada banyaknya digit angka penting yang benar dari nilai aslinya. Misalkan, bilangan 3.143762834562 merupakan bilangan yang sangat presisi karena bilangan tersebut memiliki desimal sebanyak 12 digit, tetapi bilangan tersebut tidak akurat sebagai nilai hampiran untuk  $\pi = 3.14159265358979$  karena hanya memiliki 2 digit desimal yang benar.

Fenomena hilangnya angka penting (*loss of significance digit*) akan mengakibatkan tingkat presisi dan akurasi solusi menjadi lebih rendah daripada yang diinginkan. Hal ini disebabkan oleh pengurangan dua angka yang hampir mirip satu sama lain. Sebagai contoh, katakanlah terdapat dua bilangan yaitu  $p = 3.1415926536$  dan  $q = 3.1415957341$ . Misalkan, pada proses komputasi, terdapat operasi pengurangan keduanya yaitu  $r = q - p$ , maka akan diperoleh nilai  $r = 0.0000030805$ . Perhatikan bahwa nilai  $r$  hanya memiliki kepresisan 5 digit angka penting, sedangkan nilai  $p$  dan  $q$  masing-masing memiliki kepresisan 11 digit angka penting. Kejadian seperti ini sering disebut sebagai ***loss of significance*** atau ***subtractive cancellation***.

**Contoh 2.1 :** Bandingkan hasil dari  $f(500)$  dan  $g(500)$  dengan menggunakan 6 digit angka penting, dengan fungsi  $f$  dan  $g$  seperti berikut.

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

*Catatan :* Fungsi  $f$  ekuivalen terhadap  $g$  ( $f \equiv g$ ).

**Solusi :** Hitung  $f(500)$ :

$$\begin{aligned} f(500) &= 500 \left( \sqrt{501} - \sqrt{500} \right) \\ &= 500(22.3830 - 22.3607) \\ &= 500(0.0223) = 11.1500 \end{aligned}$$

Hitung  $g(500)$ :

$$\begin{aligned} g(500) &= \frac{500}{\sqrt{501} + \sqrt{500}} \\ &= \frac{500}{22.3830 + 22.3607} \\ &= \frac{500}{44.7437} = 11.1748 \end{aligned}$$

Jawaban kasus yang pertama yaitu  $f(500) = 11.1500$  terjadi kehilangan angka penting, dari semula memiliki 6 digit angka penting menjadi 4 digit angka penting. Sebaliknya pada kasus kedua,  $g(500) = 11.1748$  mampu mempertahankan 6 angka penting dan sama dengan solusi eksak yaitu  $11.174755300747\dots$ . Dengan demikian, salah satu cara untuk menghindari hilangnya angka penting dapat menggunakan akar sekawan untuk mencari formula baru. Pada contoh ini, fungsi  $g$  merupakan perbaikan dari fungsi  $f$ .  $\triangle$

Selain menggunakan akar sekawan, hampiran deret taylor sering kali dapat menjadi cara untuk menghindari hilangnya angka penting.

**Contoh 2.2 :** Bandingkan hasil dari  $f(0.01)$  dan  $P(0.01)$  menggunakan 6 digit angka desimal, dimana

$$\begin{aligned} f(x) &= \frac{e^x - 1 - x}{x^2} \\ P(x) &= \frac{1}{2} + \frac{x}{6} + \frac{x^2}{24} \end{aligned}$$

Fungsi  $P(x)$  merupakan deret Taylor orde-2 dari fungsi  $f$  disekitar  $x = 0$ .

**Solusi :** Hitung  $f(0.01)$ :

$$f(0.01) = \frac{e^{0.01} - 1 - 0.01}{0.01^2} = \frac{1.010050 - 1.01}{0.0001} = 0.5$$

Hitung  $P(0.01)$ :

$$P(0.01) = \frac{1}{2} + \frac{0.01}{6} + \frac{0.01^2}{24} = 0.5 + 0.001667 + 0.000004 = 0.501671$$

Hasil  $P(0.01)$  mampu mempertahankan angka penting dan memiliki akurasi yang lebih baik daripada  $f(0.01)$  dengan solusi eksak yaitu  $0.5016708416794892\dots$  hingga 6 digit desimal pembulatan.  $\triangle$

Cara lain untuk menghindari hilangnya angka penting pada persamaan polinomial adalah dengan menyusun ulang persamaan dengan aturan Horner, sehingga tidak terdapat bentuk pangkat pada persamaan polinomial. Misalkan,  $P_3(x)$  merupakan polinomial dengan pangkat 3, maka  $P_3(x)$  dapat ditulis

$$\begin{aligned} P_3(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 \\ &= a_0 + x(a_1 + x(a_2 + a_3x)) \end{aligned}$$

**Contoh 2.3 :** Misalkan terdapat fungsi  $P$  dan  $Q$  yaitu

$$P(x) = x^3 - 3x^2 + 3x - 1$$

$$Q(x) = ((x-3)x+3)x-1$$

Gunakan 3 digit angka penting untuk menghitung  $P(2.19)$  dan  $Q(2.19)$ . Bandingkan dengan nilai eksaknya yaitu  $P(2.19) = Q(2.19) = 1.685159$

**Solusi :**

$$\begin{aligned} P(2.19) &\approx (2.19)^3 - 3(2.19)^2 + 3(2.19) - 1 \\ &= 10.5 - 14.4 + 6.57 - 1 = 1.67 \end{aligned}$$

$$Q(2.19) \approx ((2.19-3)2.19+3)2.19-1 = 1.69$$

Hasil di atas menunjukkan bahwa  $Q(2.19) \approx 1.69$  lebih mendekati solusi eksak daripada  $P(2.19) \approx 1.67$  dengan galat  $P$  dan  $Q$  masing-masing sebesar 0.015159 dan  $-0.004841$ .

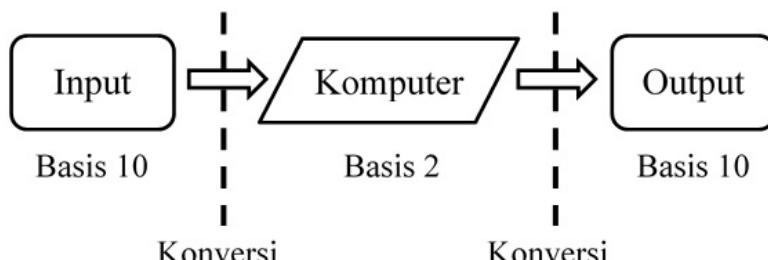
△

## 2.2 Representasi Bilangan

### 2.2.1 Bilangan Desimal dan Biner

Dalam kehidupan sehari-hari, proses hitung-menghitung akan didasarkan pada bilangan desimal (basis 10). Basis 10 yang dimaksud adalah banyaknya jenis digit yang digunakan, yaitu 0,1,2,3,4,5,6,7,8, dan 9. Namun, berbeda dengan manusia, komputer melakukan proses aritmatika menggunakan bilangan biner (basis 2), yaitu angka 0 dan 1.

Perbedaan basis yang digunakan oleh manusia dan komputer sejatinya tidak akan berpengaruh pada interaksi keduanya, karena komputer melakukan konversi bilangan desimal ke biner dan/atau sebaliknya. Dengan demikian, data masukan dari pengguna berupa bilangan desimal akan dikonversi ke biner, sehingga komputer dapat melakukan proses komputasi. Selanjutnya, sebelum ditampilkan, hasil komputasi dikonversi kembali ke bilangan desimal, sehingga dapat dipahami oleh pengguna. Proses interaksi manusia dengan komputer dapat dilihat pada Gambar 2.1.



**Gambar 2.1:** Alur proses komputasi

#### Konversi Bilangan Desimal ke Biner

Bilangan desimal merupakan bilangan yang sering digunakan pada kehidupan sehari-hari. Misalnya, bilangan bulat 1563 dalam basis 10 atau desimal dapat diekspansi menjadi bentuk

$$1563 = (1 \times 10^3) + (5 \times 10^2) + (6 \times 10^1) + (3 \times 10^0) \quad (2.1)$$

Secara umum, jika terdapat bilangan bulat positif  $N$  dan digit  $a_0, a_1, \dots, a_k$ , maka  $N$  dapat dituliskan dalam bentuk ekspansi basis 10 seperti berikut

$$N = (a_k \times 10^k) + (a_{k-1} \times 10^{k-1}) + \cdots + (a_1 \times 10^1) + (a_0 \times 10^0) \quad (2.2)$$

dimana digit  $a_k$  adalah bilangan  $[0, 1, 2, \dots, 8, 9]$ , sehingga  $N$  ditulis dalam bentuk desimal sebagai

$$N = (a_k a_{k-1} \dots a_1 a_0)_{10}$$

Dengan demikian

$$1563 = 1563_{10}$$

Basis 10 dapat diartikan sebagai penulisan bilangan dengan ekspansi pangkat dari 10. Perhatikan Persamaan 2.1, ekspansi bilangan 1563 dituliskan dalam ekspansi pangkat dari 10 dengan pangkat tertinggi benilai 3, dengan bentuk umum pada 2.2. Selain menggunakan ekspansi pangkat dari 10, suatu bilangan juga dapat dituliskan menggunakan ekspansi pangkat dari 2, yang sering dikenal sebagai **sistem bilangan biner**. Misalnya, bilangan bulat 1563 dalam basis 2 atau biner dapat diekspansi menjadi bentuk

$$\begin{aligned} 1563 &= (1 \times 2^{10}) + (1 \times 2^9) + (0 \times 2^8) + (0 \times 2^7) + (0 \times 2^6) \\ &\quad + (0 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) \\ &\quad + (1 \times 2^0) \end{aligned} \quad (2.3)$$

atau

$$1563 = 1024 + 512 + 16 + 8 + 2 + 1$$

Secara umum untuk **bilangan bulat**, misalkan  $N$  adalah bilangan bulat positif dan terdapat  $b_0, b_1, \dots, b_j$  maka  $N$  dapat diekspansi dalam persamaan 2.4 berikut

$$N = (b_j \times 2^j) + (b_{j-1} \times 2^{j-1}) + \cdots + (b_1 \times 2^1) + (b_0 \times 2^0) \quad (2.4)$$

dimana  $b_j \in \{0, 1\}$ . Dengan demikian,  $N$  dapat dituliskan dalam bilangan biner sebagai

$$N = (b_j b_{j-1} \dots b_2 b_1 b_0)_2$$

Artinya, berdasarkan Persamaan 2.3, bilangan 1563 dalam sistem bilangan biner dapat dituliskan sebagai

$$1563 = 11000011011_2$$

Penentukan nilai  $b_j$  pada Persamaan 2.4 dapat mengikuti aturan berikut. Misalkan  $Q_j$  adalah hasil bagi  $N$  terhadap 2 dan  $b_j$  adalah sisa pembagiannya. Maka, proses konversi dapat menggunakan aturan berikut

$$\begin{aligned} N &= 2 \times Q_0 + b_0 \\ Q_0 &= 2 \times Q_1 + b_1 \\ Q_1 &= 2 \times Q_2 + b_2 \\ &\vdots \\ Q_{j-1} &= 2 \times Q_j + b_j \\ Q_j &= 0 \end{aligned} \quad (2.5)$$

**Contoh 2.4 :** Misalkan terdapat bilangan bulat  $N = 50_{10}$ . Dengan aturan konversi pada Persamaan 2.5, lakukan konversi bilangan  $N$  ke sistem bilangan biner.

**Solusi :**

$$\begin{array}{rcl}
 N & = & 50 = 2 \times 25 + 0 \\
 & & 25 = 2 \times 12 + 1 \\
 & & 12 = 2 \times 6 + 0 \\
 & & 6 = 2 \times 3 + 0 \\
 & & 3 = 2 \times 1 + 1 \\
 & & 1 = 2 \times 0 + 1
 \end{array} \quad \left| \begin{array}{l}
 b_0 = 0 \\
 b_1 = 1 \\
 b_2 = 0 \\
 b_3 = 0 \\
 b_4 = 1 \\
 b_5 = 1
 \end{array} \right.$$

Dengan demikian,  $N = 50_{10} = (110010)_2$

△

### Bilangan Pecahan dalam Sistem Bilangan Biner

Dalam sistem bilangan desimal, sering kali dijumpai bilangan pecahan. Bilangan pecahan jika dituliskan dalam bentuk sistem bilangan desimal, dapat berupa bilangan desimal dengan banyak angka desimal yang berbatas atau tak-berbatas dengan pola. Misalkan,

$$\begin{aligned}
 \frac{1}{2} &= 0.5 \\
 \frac{1}{4} &= 0.25 \\
 \frac{1}{3} &= 0.\bar{3}
 \end{aligned}$$

Tanda bar di atas angka 3 menunjukkan bahwa angka tersebut berulang tak-terhingga. Dengan demikian, nilai  $0.\bar{3} = 0.33333\dots$ . Selain bilangan bulat yang dapat diekspansi dengan ekspansi pangkat, bilangan pecahan juga bisa diekspansi dengan ekspansi pangkat dari 10 ataupun 2. Misalkan, bilangan pecahan  $0.6875$  dalam basis 10 atau desimal dapat diekspansi menjadi bentuk

$$\begin{aligned}
 0.6875 &= (6 \times 10^{-1}) + (8 \times 10^{-2}) + (7 \times 10^{-3}) + (5 \times 10^{-4}) \\
 &= 0.6 + 0.08 + 0.007 + 0.0005
 \end{aligned} \tag{2.6}$$

Secara umum, jika terdapat bilangan pecahan  $R$  dan digit  $c_1, c_2, \dots$ , maka  $R$  dapat dituliskan dalam bentuk ekspansi basis 10 seperti berikut

$$R = (c_1 \times 10^{-1}) + (c_2 \times 10^{-2}) + \dots \tag{2.7}$$

dimana digit  $c_k$  adalah bilangan  $[0, 1, 2, \dots, 8, 9]$ , sehingga  $R$  ditulis dalam bentuk desimal sebagai

$$N = (0.c_1c_2\dots)_{10}$$

Dengan demikian

$$0.6875 = 0.6875_{10}$$

Sementara itu, pada sistem bilangan biner, bilangan  $R$  dapat diekspansi sebagai pangkat dari 2 dengan cara yang sama seperti Persamaan 2.6, sehingga menjadi bentuk

$$\begin{aligned} 0.6875 &= (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) + (1 \times 2^{-4}) \\ &= 0.5 + 0. + 0.125 + 0.0625 \end{aligned} \quad (2.8)$$

Secara umum, misalkan  $R$  adalah bilangan real positif yang terletak pada selang  $(0, 1)$  dan terdapat  $d_1, d_2 \dots$  maka  $R$  dapat diekspansi dalam persamaan berikut

$$R = (d_1 \times 2^{-1}) + (d_2 \times 2^{-2}) + \dots \quad (2.9)$$

dimana  $d_j \in \{0, 1\}$ . Dengan demikian,  $R$  dapat ditulis dalam bilangan biner sebagai

$$R = (0.d_1d_2\dots)_2$$

Jadi, berdasarkan persamaan 2.8, bilangan 0.6875 dapat ditulis sebagai

$$0.6875 = 0.1011$$

Untuk menentukan  $d_j$  pada persamaan 2.9 dapat mengikuti aturan berikut. Misalkan,  $d_1 = \text{int}(2R)$  dan  $F_1 = \text{frac}(2R)$ , sehingga

$$\begin{aligned} d_k &= \text{int}(2F_{k-1}) & \forall k \in \mathbb{Z}^+ \\ F_k &= \text{frac}(2F_{k-1}) \end{aligned} \quad (2.10)$$

**Contoh 2.5 :** Misalkan, terdapat bilangan pecahan  $R = 0.6875_{10}$ . Dengan menggunakan persamaan 2.10, lakukan konversi  $R$  ke biner.

**Solusi :** Diketahui  $R = 0.6875_{10}$ , sehingga

$$\begin{array}{lll} 2R = 1.375 & d_1 = \text{int}(2R) = 1 & F_1 = \text{frac}(2R) = 0.375 \\ 2F_1 = 0.750 & d_2 = \text{int}(2F_1) = 0 & F_2 = \text{frac}(2F_1) = 0.750 \\ 2F_2 = 1.500 & d_3 = \text{int}(2F_2) = 1 & F_3 = \text{frac}(2F_2) = 0.500 \\ 2F_3 = 1.000 & d_4 = \text{int}(2F_3) = 1 & F_4 = \text{frac}(2F_3) = 0.000 \end{array}$$

Dengan demikian,  $R = 0.6875_{10} = (0.1011)_2$



### Konversi Bilangan Biner ke Desimal

Bilangan biner bulat  $B = b_j \dots b_1 b_0$  dapat ditulis sebagai bilangan desimal dengan menggunakan persamaan 2.4. Sementara itu, bilangan bulat real  $D = 0.d_1d_2\dots$  dapat ditulis sebagai bilangan desimal menggunakan persamaan 2.9.

**Contoh 2.6 :** Berdasarkan persamaan 2.4 dan 2.9 bilangan biner  $(110.01101)_2$  dapat dikonversi ke desimal dengan cara

**Solusi :** Pisah bagian bulat dan desimal dari  $(110.01101)_2$ . Selanjutnya, konversi  $110_2$  ke desimal menggunakan persamaan 2.4 seperti berikut

$$\begin{aligned} (110)_2 &= (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) \\ &= 6 \end{aligned}$$

dan konversi bilangan pecahan  $(0.01101)_2$  menggunakan persamaan 2.9 seperti berikut

$$\begin{aligned}(0.01101)_2 &= (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3}) + (0 \times 2^{-4}) + (1 \times 2^{-5}) \\ &= 0.40625\end{aligned}$$

Dengan demikian, nilai

$$\begin{aligned}(110.01101)_2 &= (110)_2 + (0.01101)_2 \\ &= 6 + 0.40625 \\ &= 6.40625\end{aligned}$$

△

### 2.2.2 Floating-Point Number

Notasi ilmiah sering digunakan untuk menuliskan suatu bilangan real yang memiliki digit yang sangat besar atau sangat kecil. Sebagai contoh, bilangan 153700 dapat ditulis menjadi  $1.537 \times 10^5$  atau 0.000256 dapat ditulis menjadi  $2.56 \times 10^{-4}$ . Pada sistem penulisan ini, titik desimal bergerak sesuai dengan perubahan pangkat 10. Pada komputer digital, penulisan angka ditulis menggunakan sistem penulisan ***floating-point number***. Sistem ini memiliki dasar konsep yang mirip dengan penulisan bilangan menggunakan sistem notasi ilmiah, yaitu basis bilangan, presisi bilangan, dan eksponensial. Secara umum, bentuk sistem penulisan *floating-point* adalah

$$x = \pm \left( d_0 + \frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_{p-1}}{\beta^{p-1}} \right) \times \beta^E \quad (2.11)$$

dimana

- $\beta$  : Basis bilangan
- $p$  : Presisi bilangan
- $E$  : Bilangan eksponen,  $L \leq E \leq U$
- $d_i$  : Digit dari bilangan,  $0 \leq d_i \leq \beta - 1$ ,  $i = 0, 1, \dots, p - 1$

Bagian dalam kurung Persamaan 2.11 yang ditunjukkan dengan  $p$  bilangan berbasis  $\beta$  disebut sebagai **mantissa** dalam basis biner atau **angka penting** dalam basis desimal. Tanda (positif atau negatif), eksponen, dan mantissa dalam komputer disimpan dalam satu tempat dengan batas penyimpanan angka yang telah ditetapkan pada komputer. Meskipun banyak jenis basis bilangan yang dapat digunakan pada komputer, basis bilangan biner adalah basis yang paling sering digunakan.

Komputer dengan segala keterbatasannya menggunakan hampiran untuk menyimpan bilangan real ke dalam bilangan biner. Misalkan  $x$  bilangan real, *floating-point* untuk  $x$  pada basis bilangan biner dihampiri dengan

$$x \approx \pm q \times 2^n \quad (2.12)$$

Dengan keterangan,  $q$  disebut mantissa dimana  $(0.1)_2 \leq q < 1$  dan bilangan bulat  $n$  adalah eksponen. Dengan demikian, persamaan 2.12 dengan  $p$  digit mantissa dapat dituliskan sebagai berikut

$$x \approx 0.1d_2d_3d_4\dots d_p \times 2^n \quad (2.13)$$

Untuk keperluan penyimpanan angka yang akurat, komputer harus memiliki kemampuan menyimpan setidaknya 24-bit biner untuk mantissa. Pada masa sekarang, ada 2 jenis kapasitas komputer yang beredar di masyarakat, yaitu yang memiliki kapasitas penyimpanan angka 32-bit dan 64-bit. Hal tersebut sangat berkaitan dengan kapasitas jumlah angka biner yang dapat disimpan komputer berkaitan dengan *floating-point* number.

Komputer 32-bit mempunyai batasan penyimpanan angka yaitu 8-bit untuk eksponen dan 24-bit untuk mantissa. Artinya, komputer ini mampu merepresentasikan bilangan pada selang

$$2^{-128} \text{ sampai } 2^{127} \approx 2.93E - 39 \text{ sampai } 1.70E + 38$$

dengan sekitar 6 digit desimal kepresision angka.

Komputer 64-bit mempunyai batasan penyimpanan angka yaitu 11-bit untuk eksponen dan 53-bit untuk mantissa. Artinya, komputer ini mampu merepresentasikan bilangan pada selang

$$2^{-1024} \text{ sampai } 2^{1023} \approx 5.56E - 309 \text{ sampai } 8.98E + 307$$

dengan sekitar 16 digit desimal kepresision angka.

## 2.3 Praktikum

### 2.3.1 Kepresision Terbatas

**Contoh 2.7 :** Diketahui dua buah bilangan, yaitu  $p_1 = 1.414$  dan  $p_2 = 0.09125$ . Berikut merupakan langkah-langkah untuk menghitung nilai  $x = p_1 + p_2$  dan  $y = p_1 p_2$  dengan pembulatan 4 angka penting.

**Langkah 1:** Pendefinisian fungsi pembulatan sebanyak  $n$  angka penting.

```
function bulat(x,n)
    m = log10(abs(x));
    if m > 0
        m = ceil(m);
        y = round(x .* 10 .^ (n-m)) ./ 10 .^ (n-m);
    else
        m = ceil(-m);
        y = round(x .* 10 .^ (m+n-1)) ./ 10 .^ (m+n-1);
    end
end
```

**Langkah 2:** Penghitungan nilai  $x = p_1 + p_2$  dan  $y = p_1 p_2$ .

```
Inp: p1 = 1.414;
     p2 = 0.09125;
Inp: x = bulat(bulat(p1,4)+bulat(p2,4),4)
     y = bulat(bulat(p1,4)*bulat(p2,4),4)
     x,y
```

Out: (1.505, 0.129)

---

**Soal Latihan:** Diberikan dua bilangan, yaitu  $p_1 = -1.2412132$  dan  $p_2 = 0.0000134234$ . Gunakan pembulatan 4 angka penting untuk menghitung nilai  $x = p_1 + p_2$  dan  $y = p_1 p_2$  dengan langkah-langkah seperti pada Contoh 2.7.

---

### 2.3.2 Loss of Significant Digit

**Contoh 2.8 :** Diberikan dua fungsi  $f$  dan  $g$  seperti berikut.

$$f(x) = x(\sqrt{x+1} - \sqrt{x})$$

$$g(x) = \frac{x}{\sqrt{x+1} + \sqrt{x}}$$

Secara analitik  $f(x) = g(x)$  untuk  $x \geq 0$  (Periksa!). Akan tetapi, proses komputasi dari kedua fungsi tersebut memberikan hasil yang berbeda yang disebabkan oleh adanya *loss of significant digit*. Berikut merupakan langkah-langkah untuk mengelaborasi hal tersebut.

**Langkah 1:** Pendefinisian fungsi  $f$  dan  $g$  dengan nama  $fx$  dan  $gx$ , yaitu  $y = fx(x)$  dan  $y = gx(x)$ .

```
Inp: fx(x) = x.* (sqrt(x+1)-sqrt(x))
      gx(x) = x./ (sqrt(x+1)+sqrt(x));
```

**Langkah 2:** Pendefinisian fungsi untuk mengevaluasi  $f$  dan  $g$  dengan kepresision terbatas dengan nama  $fn$  dan  $gn$ , yaitu  $y = fn(x, n)$  dan  $y = gn(x, n)$ . Batasan kepresision proses komputasi ini tentunya menyebabkan adanya galat komputasi.

```
Inp: fn(x,n) = x .* (bulat(sqrt(x+1),n) - bulat(sqrt(x),n));
      gn(x,n) = x ./ (bulat(sqrt(x+1),n) + bulat(sqrt(x),n));
```

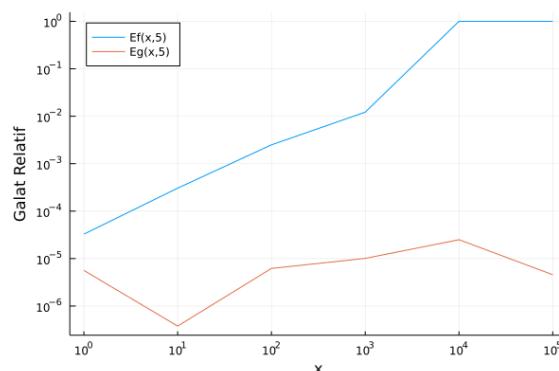
**Langkah 3:** Jika fungsi  $fx$  dan  $gx$  diasumsikan memberikan nilai eksak dari  $f$  dan  $g$ , definisikan galat relatif komputasi  $fn(x, 5)$  dan  $gn(x, 5)$  dengan  $x = 10^k, k = 0, 1, 2, \dots, 5$ , yaitu

$$E_f(x, n) = \left| \frac{fn(x, n) - fx(x)}{fx(x)} \right| \text{ dan } E_g(x, n) = \left| \frac{gn(x, n) - gx(x)}{gx(x)} \right|$$

```
Inp: Ef(x,n) = abs.((fn.(x,n)-fx.(x))./fx.(x));
      Eg(x,n) = abs.((gn.(x,n)-gx.(x))./gx.(x));
```

**Langkah 4:** Pembuatan grafik galat relatif  $E_f$  dan  $E_g$  sebagai fungsi dari  $x$  menggunakan skala logaritmik pada kedua sumbu koordinat.

```
Inp: using Plots
Inp: k = 0:5;
      x = 10 .^ k;
Inp: plot(x, Ef(x,5), xaxis=:log, yaxis=:log, label = "Ef(x,5)")
      plot!(x, Eg(x,5), xaxis=:log, yaxis=:log, label = "Eg(x,5)")
      xlabel!("x")
      ylabel!("Galat Relatif")
```



**Gambar 2.2:** Galat relatif komputasi dengan kepresision terbatas fungsi  $f$  dan  $g$ .

Berdasarkan Gambar 2.2, dapat dilihat bahwa komputasi dengan kepresisian terbatas untuk fungsi  $g$  memiliki galat relatif lebih kecil dibandingkan fungsi  $f$ . Hal ini disebabkan oleh adanya pengurangan dua angka yang saling berdekatan, yaitu  $\sqrt{x+1}$  dan  $\sqrt{x}$  untuk  $x$  bernilai besar, sehingga terjadi *loss of significant digit*. Dengan demikian, fungsi  $g$  lebih baik digunakan dalam proses komputasi dibandingkan fungsi  $f$ , meskipun secara analitik  $f$  dan  $g$  menghasilkan nilai yang sama.

**Soal Latihan:** Diberikan dua fungsi  $f$  dan  $g$  seperti berikut.

$$f(x) = \ln(x+1) - \ln(x)$$

$$g(x) = \ln\left(\frac{x+1}{x}\right)$$

Secara analitik  $f(x) = g(x)$  untuk  $-1 < x < 0$  dan  $x > 0$ . (Periksa!) Akan tetapi, proses komputasi dari kedua fungsi tersebut memberikan hasil berbeda yang disebabkan oleh adanya *loss of significant digit*. Ulangi langkah-langkah pada Contoh 2.8 untuk mengelaborasi hal tersebut.

### 2.3.3 Akurasi Komputer

**Contoh 2.9 :** Secara analitik, kalkulasi berikut akan menghasilkan nilai nol

$$\Delta = an - \sum_{i=1}^n a$$

berapa pun nilai konstanta  $a$  dan  $n$ . Berikut merupakan cara untuk menghitung secara numerik nilai  $\Delta$  dengan  $n = 10^6$  dan konstanta  $a = \frac{1}{10}$ .

```
Inp: n = 10^6;
      a = 1/10;
      jml = 0;
      for i = 1:n
          jml = jml+a;
      end
      Delta = a*n - jml;
      Delta
```

Out: -1.3328826753422618e-6

Berdasarkan hasil yang didapatkan, nilai komputasi numerik tidak menghasilkan nilai nol seperti hasil analitik. Hal ini dikarenakan ketidak-mampuan komputer untuk menyimpan angka desimal dengan tepat, sehingga terjadi galat pembulatan (*round-off error*) pada proses komputasi.

**Soal Latihan:** Secara analitik, kalkulasi berikut akan menghasilkan nilai nol

$$\Delta = an - \sum_{i=1}^n a$$

berapa pun nilai konstanta  $a$  dan  $n$ . Hitunglah secara numerik nilai  $\Delta$  untuk  $n = 10^6$  dan beberapa kemungkinan konstanta  $a$ , yaitu  $a = \frac{1}{2}$ ,  $a = \frac{1}{3}$ ,  $a = \frac{1}{10}$ ,  $a = \pi$ , dan  $a = e$ . Jelaskan hasil yang didapatkan!

## 2.4 Latihan-Latihan

### 2.4.1 Ulasan Materi

1. Jelaskan makna dari beberapa jenis galat berikut dan berikan contohnya.
  - (a) Galat mutlak dan galat relatif.
  - (b) *Truncation* dan *round-off error*.
  - (c) *Loss of significant error*
2. Jelaskan sebab munculnya fenomena *loss of significant digits* pada suatu proses komputasi numerik?
3. Jelaskan kenapa formula berikut mengalami fenomena *loss of significant digits*! Carilah formula lain untuk menghindarinya!
  - (a)  $\ln(x+1) - \ln(x)$  untuk  $x$  besar
  - (b)  $\sqrt{x^2 + 1} - x$  untuk  $x$  besar
  - (c)  $\cos^2 x - \sin^2 x$  untuk  $x \approx \pi/4$
4. Jelaskan jenis *error* yang terjadi pada beberapa persamaan berikut.
  - (a)  $\int_0^{1/4} \exp(x) dx \approx \int_0^{1/4} \left(1 + x + \frac{x^2}{2!} + \frac{x^3}{3!}\right) dx = \hat{p}$
  - (b)  $\int_0^{1/4} e dx \approx \int_0^{1/4} 2.71828 dx = \hat{p}$
  - (c)  $p = \frac{\sin(\pi + 0.00001) - \sin(\pi)}{0.00001}$
5. Dalam kegiatan sehari-hari, manusia menggunakan representasi bilangan desimal (basis 10) untuk proses menghitung, sedangkan komputer representasi menggunakan bilangan ....
6. Mengapa basis bilangan yang digunakan komputer berbeda dengan basis bilangan dalam kegiatan sehari-hari?
7. Bit merujuk pada sebuah digit dalam sistem angka biner (basis 2). Jelaskan perbedaan sistem operasi dengan 32-bit dan 64-bit? Berapakah batas maksimal bit yang mampu disimpan oleh suatu komputer pada saat ini?
8. Bagaimana suatu bilangan desimal dapat dibaca dan diproses oleh suatu komputer?
9. Manakah di antara pernyataan berikut yang benar dan salah?
  - (a) Presisi merujuk pada banyaknya digit yang dapat dituliskan oleh komputer untuk suatu bilangan
  - (b) Fenomena hilangnya angka penting tidak berkaitan dengan tingkat presisi dan akurasi dari solusi numerik
  - (c) Akurasi merujuk pada banyaknya digit angka penting yang benar dari nilai aslinya
  - (d) Fenomena hilangnya angka penting (*loss of significance digit*) sering disebut juga sebagai *subtractive cancellation*.
  - (e) Agar komputer bisa mengoperasikan bilangan, proses konversi bilangan desimal ke bilangan biner perlu dilakukan oleh komputer
  - (f) Komputer dengan kapasitas penyimpanan angka 64-bit mampu merepresentasikan bilangan pada selang yang lebih luas dan lebih presisi dibandingkan komputer yang memiliki kapasitas penyimpanan angka 32-bit.

### 2.4.2 Soal Pemrograman

1. **Galat Mutlak dan Relatif.** Hitunglah nilai error  $E_x$  dan  $R_x$  jika diketahui
  - (a)  $x = 2.71828182$ ,  $\hat{x} = 2.7182$
  - (b)  $y = 98350$ ,  $\hat{y} = 98000$
  - (c)  $z = 0.000068$ ,  $\hat{z} = 0.00006$
2. **Significant Digit.** Jika  $a = -0.00048914$ , dan  $b = 9125800$  serta  $x = 31.415$ , dan  $y = 0.027182$ .
  - (a) Hitung  $a + b$  dan  $ab$  dengan menggunakan 4 angka penting.
  - (b) Hitung  $x + y$  dan  $xy$  dengan menggunakan 5 angka penting.
3. Diketahui fungsi  $f$  dan  $g$  sebagai berikut:

$$f(x) = \sqrt{x^2 + 1} - x$$

$$g(x) = \frac{1}{\sqrt{x^2 + 1} + x}$$

- (a) Buatlah fungsi  $f$  dan  $g$  tersebut dengan nama  $y = fx(x)$  dan  $y = gx(x)$ .
- (b) Buatlah fungsi  $fn(x, n)$  dan  $gn(x, n)$  yaitu memunculkan nilai fungsi  $f$  dan  $g$  dengan kepresisan terbatas (menggunakan  $n$  angka desimal).
- (c) Bila fungsi  $fx$  dan  $gx$  diasumsikan memberikan nilai eksak dari fungsi  $f$  dan  $g$ , buatlah fungsi yang mengeluarkan nilai galat relatif dari  $fn$  dan  $gn$ , yaitu

$$E_f(x, n) = \left| \frac{fn(x, n) - fx(x)}{fx(x)} \right| \quad \text{dan} \quad E_g(x, n) = \left| \frac{gn(x, n) - gx(x)}{gx(x)} \right|$$

- (d) Gambarkan fungsi  $E_f$  dan  $E_g$  pada selang  $x : 0 \leq x \leq 1000$  dengan sub-interval sebesar 0.05 menggunakan  $n = 7$  angka penting.
- (e) Jelaskan hasil yang didapatkan dan kaitannya dengan *loss of significant digit*.
4. Konversi bilangan biner berikut ke bilangan desimal.
 

(a) 10101 <sub>2</sub>	(c) 11111110 <sub>2</sub>
(b) 111000 <sub>2</sub>	(d) 1000000111 <sub>2</sub>
5. Konversi bilangan biner berikut ke bilangan desimal.
 

(a) 0.11011 <sub>2</sub>	(c) 0.1010101 <sub>2</sub>
(b) 0.10101 <sub>2</sub>	(d) 0.110110110 <sub>2</sub>
6. Konversi bilangan biner berikut ke desimal
 

(a) 1.0110101 <sub>2</sub>	(b) 11.0010010001 <sub>2</sub>
----------------------------	--------------------------------
7. Konversi bilangan pecahan desimal berikut ke biner
 

(a) 1/10	(b) 1/3
(c) 1/7	
8. Hitung beberapa persamaan berikut, jika diketahui sebuah komputer hanya mampu menyimpan 4-bit mantissa.
 

(a) $\left(\frac{1}{3} + \frac{1}{5}\right) + \frac{1}{6}$	(c) $\left(\frac{3}{17} + \frac{1}{9}\right) + \frac{1}{7}$
(b) $\left(\frac{1}{10} + \frac{1}{3}\right) + \frac{1}{5}$	(d) $\left(\frac{7}{10} + \frac{1}{9}\right) + \frac{1}{7}$

---

---

## BAB 3

---

# SOLUSI PERSAMAAN TAK-LINEAR 1

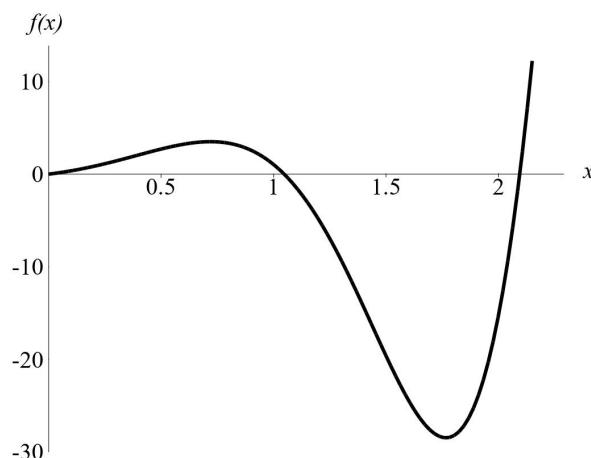
Pada berbagai disiplin ilmu, sering kali dihadapkan dengan persoalan mencari solusi akar suatu persamaan  $f(x) = 0$ . Beberapa persamaan dapat dicari akarnya dengan mudah, seperti persamaan linear dan kuadratik. Sebagai contoh, persamaan linear dengan bentuk  $ax + b = 0$  memiliki akar persamaan yaitu  $x = -b/a$ . Contoh lainnya yaitu kasus persamaan kuadratik  $ax^2 + bx + c = 0$ . Persamaan tersebut juga dengan mudah dapat diselesaikan menggunakan rumus  $abc$  yaitu

$$x_{12} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Namun, akar persamaan akan sangat sulit dicari jika persamaan  $f(x)$  merupakan persamaan yang melibatkan fungsi akar, eksponensial, trigonometri dan/atau fungsi-fungsi rumit lainnya. Contohnya, pada Bab 1, persamaan

$$f(x) = \exp(2x) \sin(3x) \quad (3.1)$$

akan sangat sulit untuk dicari formula solusi dari  $f(x) = 0$ . Dengan demikian, solusi numerik menjadi salah satu cara untuk mencari akar dari persamaan tersebut. Perhatikan grafik fungsi dari persamaan 3.1 berikut.



**Gambar 3.1:** Grafik fungsi  $f(x) = \exp(2x) \sin(3x)$

Gambar tersebut memiliki dua akar persamaan fungsi pada selang  $[0.5, 2.5]$ . Kedua akar tersebut sangat sulit untuk dicari menggunakan metode analitik. Sebaliknya, kedua akar persamaan tersebut dapat dihampiri nilainya dengan mudah menggunakan metode numerik. Salah satu caranya adalah menggunakan metode *bisection*. Hasil dari metode *bisection* menunjukkan nilai akar dengan akurasi 6 digit desimal yaitu  $c_1 = 1.047197$  dan  $c_2 = 2.094395$ .

**Definisi 3.1 Akar Persamaan:** Asumsikan bahwa fungsi  $f(x)$  merupakan suatu fungsi kontinu, maka bilangan  $r$  sembarang yang memenuhi  $f(r) = 0$  disebut akar persamaan dari  $f(x) = 0$ . ■

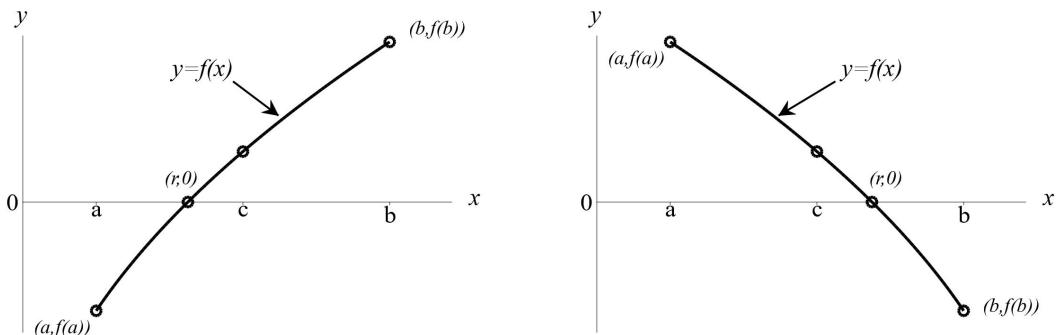
Sebagai contoh, persamaan  $2x^2 + 5x - 3 = 0$  memiliki dua akar real yang dapat dicari dengan cara

$$\begin{aligned} 2x^2 + 5x - 3 &= 0 \\ (2x - 1)(x + 3) &= 0 \end{aligned}$$

Dengan demikian, akar persamaan yang dimaksud adalah  $r_1 = 0.5$  dan  $r_2 = -3$ .

### 3.1 Metode *Bisection*

Metode *bisection* dimulai dengan interval awal  $[a, b]$  dengan  $f(a)$  dan  $f(b)$  memiliki tanda yang berlawanan (positif dan negatif). Selama grafik fungsi  $y = f(x)$  merupakan fungsi kontinu yang tidak patah, fungsi tersebut akan melewati sumbu- $x$  ketika  $x = r$  yang berada pada interval awal (perhatikan Gambar 3.2).



(a) Kondisi  $f(a)$  dan  $f(c)$  berlawanan tanda      (b) Kondisi  $f(b)$  dan  $f(c)$  berlawanan tanda  
**Gambar 3.2:** Proses pengambilan keputusan metode *bisection*.

Prinsip dasar metode *bisection* yaitu menggeser titik ujung selang awal sehingga semakin dekat satu sama lain dengan jarak sekecil mungkin yang mengapit nilai akar persamaan. Proses metode *bisection* dimulai dengan memilih nilai tengah yang terletak pada selang  $[a, b]$  yaitu  $c = (a + b)/2$ . Selanjutnya, analisa tiga kemungkinan berikut:

1. Jika  $f(a)$  dan  $f(c)$  berlawanan tanda, maka nilai akar berada pada sub-interval  $[a, c]$ .
2. Jika  $f(c)$  dan  $f(b)$  berlawanan tanda, maka nilai akar berada pada sub-interval  $[c, b]$ .
3. Jika  $f(c) = 0$ , maka  $c$  adalah nilai akar persamaan.

Jika kemungkinan 1. atau 2. yang terpenuhi (lihat Gambar 3.2a dan 3.2b), maka didapatkan interval baru yang mempunyai lebar interval setengah dari lebar interval awal, yang mengandung nilai akar persamaan. Untuk melanjutkan proses, berikan label pada interval

baru sebagai interval  $[a, b]$ , kemudian ulangi proses sehingga didapatkan interval sekecil mungkin. Selama proses *bisection*, gunakan notasi berikut untuk melacak nomor iterasi selama proses pencarian akar berlangsung.

1. Gunakan notasi  $[a_0, b_0]$  untuk interval awal dan  $c_0$  sebagai nilai tengahnya.
2. Setelah didapatkan interval baru, gunakan notasi  $[a_1, b_1]$  dimana interval baru tersebut mengandung nilai akar persamaan dan  $c_1$  sebagai nilai tengahnya.
3. Setelah diulang sebanyak  $n$  kali, maka interval yang didapatkan adalah  $[a_n, b_n]$  dengan nilai tengah  $c_n$ .

Selama proses pencarian akar, titik ujung selang sebelah kiri yaitu  $a$  semakin besar, sedangkan titik ujung selang sebelah kanan yaitu  $b$  semakin mengecil. Dalam notasi pertaksamaan dapat dituliskan sebagai

$$a_0 \leq a_1 \leq \cdots \leq a_n \leq \cdots \leq r \leq \cdots \leq b_n \leq \cdots \leq b_1 \leq b_0 \quad (3.2)$$

dimana  $c_n = \frac{a_n + b_n}{2}$  dan jika  $f(a_{n+1})$  dan  $f(b_{n+1})$  berlawanan tanda, maka

$$[a_{n+1}, b_{n+1}] = [a_n, c_n] \quad \text{atau} \quad [a_{n+1}, b_{n+1}] = [c_n, b_n] \quad \text{untuk setiap } n$$

**Teorema 3.1 :** Asumsikan bahwa  $f \in C[a, b]$  dan terdapat bilangan  $r \in [a, b]$  dimana  $f(r) = 0$ . Jika  $f(a)$  dan  $f(b)$  memiliki tanda yang berlawanan dan  $\{c_n\}_{n=0}^{\infty}$  menunjukkan nilai barisan tengah hasil dari proses *bisection* pada persamaan 3.2, maka

$$|r - c_n| < \frac{b - a}{2^{n+1}} \quad (3.3)$$

Oleh karena itu, barisan  $\{c_n\}_{n=0}^{\infty}$  konvergen menuju nilai akar persamaan  $x = r$ , yaitu

$$\lim_{n \rightarrow \infty} c_n = r \quad (3.4)$$

■

**Contoh 3.1 :** Diberikan fungsi  $h(x) = x \sin(x)$ . Carilah solusi  $h(x) = 1$  dengan  $x \in [0, 2]$  menggunakan metode *bisection*.

**Solusi :** Dengan metode *bisection*, soal di atas ekuivalen dengan mencari akar persamaan  $f(x) = 0$  dengan  $f(x) = x \sin(x) - 1$ . Metode dimulai dengan mengecek tanda dari  $f(a)$  dan  $f(b)$ , diketahui nilai awal  $a_0 = 0$  dan  $b_0 = 2$ , maka nilai

$$f(0) = -1 \quad \text{dan} \quad f(2) = 0.818595$$

Karena nilai  $f(a)$  dan  $f(b)$  berlawanan tanda, maka nilai akar dari  $f(x) = 0$  berada pada interval  $[0, 2]$ . Selanjutnya, hitung nilai tengah yaitu  $c_0 = 1$  dan  $f(c_0) = -0.158529$ . Karena  $c_0$  dan  $b_0$  berlawanan tanda, maka interval baru yang mengandung nilai akar persamaan adalah  $[c_0, b_0]$ . Selanjutnya, berikan label untuk interval baru yaitu  $a_1 = c_0$  dan  $b_1 = b_0$ . Hitung nilai tengah  $c_1 = 1.5$  dan  $f(c_1) = 0.496242$ . Pada kondisi ini, nilai  $f(a_1) = -0.158529$  dan  $f(c_1) = 0.496242$  berlawanan tanda, sehingga nilai akar persamaan berada pada selang  $[a_1, c_1] = [1, 1.5]$ . Dengan demikian, interval baru yang didapatkan adalah  $a_2 = a_1$  dan  $b_2 = c_1$ . Hasil dari proses metode *bisection* menunjukkan

bahwa  $\{c_k\}$  konvergen menuju  $r \approx 1.114157$ . Contoh penghitungan ditunjukkan pada Tabel 3.1.  $\triangle$

**Tabel 3.1:** Solusi metode *bisection* untuk  $x \sin(x) - 1 = 0$

$k$	$a_k$	$c_k$	$b_k$	$f(c_k)$
0	0.00000	1.00000	2.00000	-0.15853
1	1.00000	1.50000	2.00000	0.49624
2	1.00000	1.25000	1.50000	0.18623
3	1.00000	1.12500	1.25000	0.01505
4	1.00000	1.06250	1.12500	-0.07183
5	1.06250	1.09375	1.12500	-0.02836
6	1.09375	1.10938	1.12500	-0.00664
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Persamaan 3.3 pada Teorema 3.1 dapat digunakan untuk menghitung batas akurasi dari nilai tengah  $c_k$  terhadap nilai akar  $r$ . Pada Contoh 3.1, dapat dihitung batas akurasi nilai  $c_{10}$  yaitu

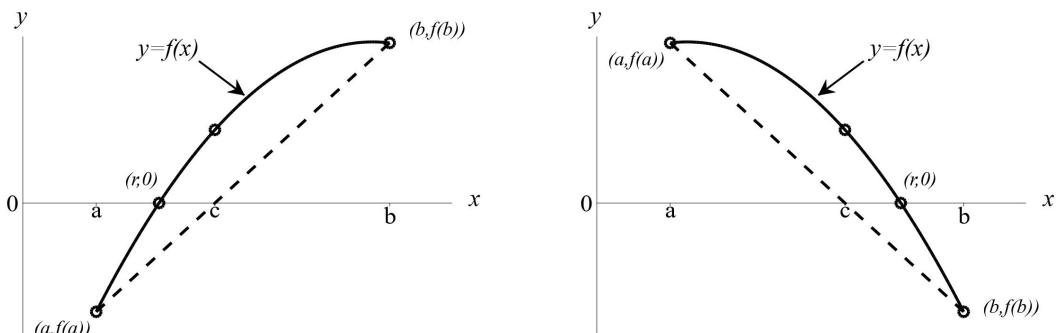
$$E_{10} \leq \frac{b_0 - a_0}{2^{10}} \approx 9.7656 \times 10^{-4}$$

Dari hasil tersebut dapat disimpulkan bahwa nilai  $c_{10}$  mempunyai akurasi terhadap nilai  $r$  hingga 3 digit desimal. Banyaknya iterasi  $N$  yang dibutuhkan untuk mencapai nilai akurasi  $\delta$  yang ditentukan dapat dihitung menggunakan rumus

$$N = \text{int} \left( \frac{\ln(b-a) - \ln(\delta)}{\ln(2)} \right)$$

## 3.2 Metode *Regula-Falsi*

Metode lain yang dapat digunakan adalah *regula falsi* atau *method of false position*. Metode ini merupakan pengembangan dari metode *bisection* yang dinilai terlalu lambat untuk memperoleh nilai akar persamaan. Sama seperti metode sebelumnya, asumsikan bahwa  $f(a)$  dan  $f(b)$  memiliki tanda yang berlawanan. Metode *bisection* menggunakan nilai tengah dari interval  $[a, b]$  sebagai peralihan interval baru. *Regula falsi* menggunakan garis yang melewati titik  $(a, f(a))$ ,  $(b, f(b))$  dan sumbu-x (perhatikan Gambar 3.3).



(a) Kondisi  $f(a)$  dan  $f(c)$  berlawanan tanda      (b) Kondisi  $f(c)$  dan  $f(b)$  berlawanan tanda  
**Gambar 3.3:** Proses pengambilan keputusan metode *regula falsi*

Untuk menghitung nilai  $c$ , dapat menggunakan persamaan kemiringan garis  $m$  dimana garis  $L$  (pada gambar 3.3 berupa garis putus-putus) dapat ditulis dalam dua versi yaitu

$$m = \frac{f(b) - f(a)}{b - a} \quad (3.5)$$

dimana menggunakan titik  $(a, f(a))$  dan  $(b, f(b))$ , serta

$$m = \frac{0 - f(b)}{c - b} \quad (3.6)$$

dimana menggunakan titik  $(c, 0)$  dan  $(b, f(b))$ . Hasil dari persamaan 3.5 dan 3.6 adalah

$$\frac{f(b) - f(a)}{b - a} = \frac{0 - f(b)}{c - b}$$

Dengan demikian, didapatkan nilai  $c$  yaitu

$$c = b - \frac{f(b)(b - a)}{f(b) - f(a)} \quad (3.7)$$

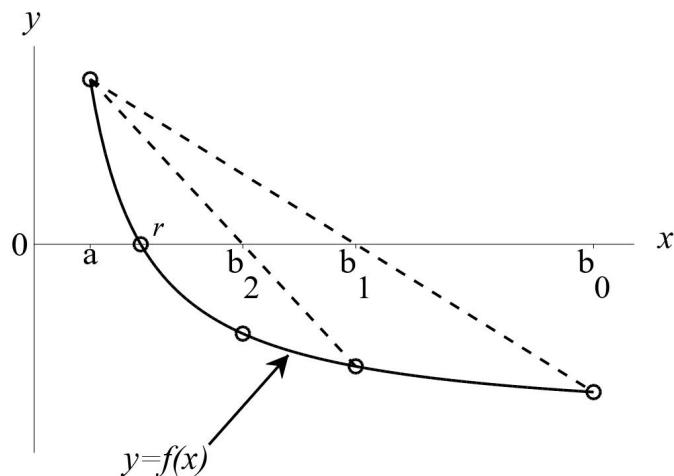
Selanjutnya, analisa tiga kemungkinan yang akan terjadi yaitu

1. Jika  $f(a)$  dan  $f(c)$  berlawanan tanda, maka nilai akar berada pada sub-interval  $[a, c]$ .
2. Jika  $f(c)$  dan  $f(b)$  berlawanan tanda, maka nilai akar berada pada sub-interval  $[c, b]$ .
3. Jika  $f(c) = 0$ , maka  $c$  adalah nilai akar persamaan.

Sama seperti metode *bisection*, proses pengambilan keputusan dari tiga kemungkinan di atas digunakan untuk membentuk barisan interval  $\{[a_n, b_n]\}$  yang mengandung nilai akar persamaan. Pada setiap langkah, akan dihitung nilai hampiran akar persamaan yaitu

$$c_n = b_n - \frac{f(b_n)(b_n - a_n)}{f(b_n) - f(a_n)} \quad (3.8)$$

dan dapat dibuktikan bahwa barisan  $\{c_n\}$  konvergen menuju  $r$ . Akan tetapi, perlu diperhatikan bahwa lebar selang  $[a, b]$  belum tentu semakin kecil mendekati nol. Jika grafik  $y = f(x)$  adalah fungsi konkaf disekitar  $(r, 0)$ , maka salah satu titik ujung akan tetap sedangkan yang lainnya akan menuju solusi (perhatikan gambar 3.4).



**Gambar 3.4:** Titik ujung stasioner pada metode *regula falsi*

Sebagai contoh, akan dicari ulang solusi dari  $x \sin(x) - 1 = 0$  dengan metode *regula falsi*. Perhatikan bahwa metode ini lebih cepat untuk mencari solusi dibandingkan metode *bisection* dan lebar interval  $\{b_n - a_n\}_{n=0}^{\infty}$  tidak menuju ke nol.

**Contoh 3.2 :** Gunakan metode *regula falsi* untuk mencari akar persamaan  $x \sin(x) - 1 = 0$  yang berada pada interval  $[0, 2]$ .

**Solusi :** Metode dimulai dengan  $a_0 = 0$  dan  $b_0 = 2$  sehingga didapatkan nilai  $f(0) = -1$  dan  $f(2) = 0.818595$ , artinya terdapat akar persamaan diantara interval  $[0, 2]$ . Dengan persamaan 3.7, didapatkan nilai

$$c_0 = 2 - \frac{0.818595(2 - 0)}{0.818595 - (-1)} = 1.099750 \quad \text{dan} \quad f(c_0) = -0.020019$$

Karena  $f(c_0)$  dan  $f(b_0)$  memiliki tanda yang berlawanan, sehingga interval baru yang mengandung akar adalah  $[c_0, b_0] = [1.099750, 2]$ . Selanjutnya, berikan label baru yaitu  $a_1 = c_0$  dan  $b_1 = b_0$ . Nilai hampiran akar selanjutnya adalah

$$c_1 = 2 - \frac{0.818595(2 - 1.099750)}{0.818595 - (-0.020019)} = 1.121240 \quad \text{dan} \quad f(c_1) = 0.009834$$

Selanjutnya,  $f(x)$  berganti tanda pada  $[a_1, c_1]$ , sehingga interval baru yang didapat adalah  $a_2 = a_1$  dan  $b_2 = c_1$ . Ringkasan penghitungan metode *regula falsi* dapat dilihat pada Tabel 3.2.  $\triangle$

**Tabel 3.2:** Solusi metode *regula falsi* untuk  $x \sin(x) - 1 = 0$

$k$	$a$	$c$	$b$	$f(c)$
0	0.00000000	1.09975017	2.00000000	-0.02001921
1	1.09975017	1.12124074	2.00000000	0.00983461
2	1.09975017	1.11416119	1.12124074	0.00000563
3	1.09975017	1.11415714	1.11416119	0.00000000

Kriteria penghentian metode *bisection* tidak terlalu berguna untuk menghentikan iterasi metode *regula falsi*, karena barisan  $\{b_n - a_n\}_{n=0}^{\infty}$  tidak selalu menuju nol. Kriteria yang dapat digunakan untuk menghentikan iterasi metode *regula falsi* adalah kedekatan hampiran yang berurutan pada setiap iterasi yaitu  $\min(c - a, b - c)$  dan nilai dari  $|f(c_n)|$ .

### 3.3 Metode Iterasi Titik Tetap

Secara bahasa, iterasi berarti perulangan. Dalam istilah komputasi, iterasi merupakan serangkaian proses operasi yang dilakukan secara berulang, sebanyak jumlah yang telah ditetapkan ataupun sampai kondisi tertentu terpenuhi. Iterasi adalah teknik dasar dalam komputasi numerik. Teknik iterasi sering digunakan untuk memecahkan masalah akar persamaan tak-linear, sistem persamaan linear, dan solusi persamaan differensial.

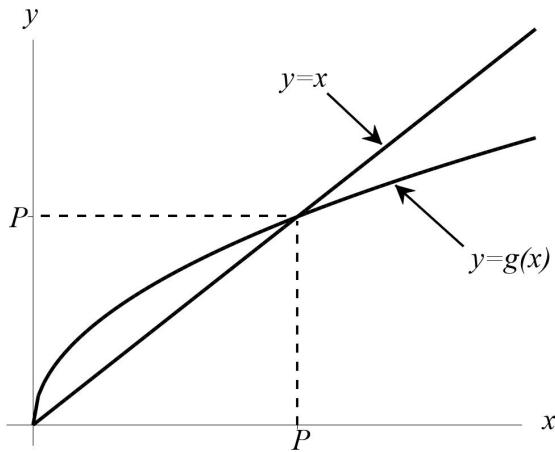
### 3.3.1 Mencari Titik Tetap

**Definisi 3.2 Titik Tetap:** Titik tetap dari fungsi  $g(x)$  adalah bilangan real  $P$  yang memenuhi persamaan

$$P = g(P) \quad (3.9)$$

■

Secara geometri, titik tetap dari fungsi  $g(x)$  adalah titik potong antara  $y = g(x)$  dan  $y = x$ . Ilustrasi geometri lokasi titik tetap dapat dilihat pada Gambar 3.5 berikut.



**Gambar 3.5:** Ilustrasi geometri lokasi titik tetap

**Definisi 3.3 Iterasi Titik Tetap:** Iterasi titik tetap didefinisikan sebagai iterasi

$$p_{n+1} = g(p_n) \quad (3.10)$$

untuk setiap  $n = 0, 1, 2, \dots$

■

**Teorema 3.2 :** Misalkan, terdapat fungsi kontinu  $g$  dan barisan  $\{p_n\}_{n=0}^{\infty}$  hasil dari iterasi titik tetap. Jika  $\lim_{n \rightarrow \infty} p_n = P$ , maka  $P$  merupakan titik tetap dari fungsi  $g(x)$ .

**Bukti** Jika  $\lim_{n \rightarrow \infty} p_n = P$ , maka  $\lim_{n \rightarrow \infty} p_{n+1} = P$ . Dari hasil tersebut, kontinuitas  $g$ , dan hubungan  $p_{n+1} = g(p_n)$ , maka

$$g(P) = g\left(\lim_{n \rightarrow \infty} p_n\right) = \lim_{n \rightarrow \infty} g(p_n) = \lim_{n \rightarrow \infty} p_{n+1} = P \quad (3.11)$$

Jadi,  $P$  merupakan titik tetap dari  $g(x)$

■

**Contoh 3.3 :** Diketahui iterasi yang konvergen yaitu

$$p_0 = 0.5 \quad \text{dan} \quad p_{k+1} = \exp(-p_k) \quad \text{untuk setiap } k = 0, 1, 2, \dots$$

Gunakan Teorema 3.2 untuk mencari nilai titik tetap dari  $\exp(-x)$

**Solusi :** Nilai 10 iterasi pertama dari persamaan tersebut adalah

$$\begin{aligned} p_1 &= \exp(-0.500000) = 0.606531 \\ p_2 &= \exp(-0.606531) = 0.545239 \\ p_3 &= \exp(-0.545239) = 0.579703 \\ &\vdots && \vdots \\ p_9 &= \exp(-0.566409) = 0.567560 \\ p_{10} &= \exp(-0.567560) = 0.566907 \end{aligned}$$

Dari hasil di atas, dapat dilihat bahwa iterasi konvergen ke suatu titik. Jika penghitungan dilanjutkan hingga  $n$  bernilai besar, maka didapatkan nilai

$$\lim_{n \rightarrow \infty} p_n = 0.567143\dots$$

Jadi, didapatkan nilai hampiran titik tetap dari fungsi  $y = e^{-x}$ , yaitu  $P = 0.567143\dots$   $\triangle$

### 3.3.2 Eksistensi dan Kekonvergenan

**Teorema 3.3 :** Asumsikan bahwa  $g \in C[a, b]$ ,

1. Jika hasil dari pemetaan  $y = g(x)$  yaitu  $y \in [a, b]$  untuk setiap  $x \in [a, b]$ , maka  $g$  memiliki titik tetap pada selang  $[a, b]$ .
2. Misalkan bahwa  $g'(x)$  terdefinisi pada selang  $(a, b)$  dan terdapat konstanta positif  $K < 1$  dimana  $|g'(x)| \leq K < 1$  untuk setiap  $x \in (a, b)$ , maka  $g$  memiliki titik tetap tunggal  $P$  pada selang  $[a, b]$ . ■

**Contoh 3.4 Eksistensi Titik Tetap:** Gunakan Teorema 3.3 untuk menunjukkan bahwa  $g(x) = \cos(x)$  memiliki titik tetap tunggal pada selang  $[0, 1]$

**Solusi :**

- Jelas bahwa  $g$  kontinu pada selang  $[0, 1]$ .
- $g(x) = \cos(x)$  merupakan fungsi turun pada selang  $[0, 1]$  dan memiliki selang solusi  $y \in [0, 1]$  untuk setiap  $x \in [0, 1]$ . Dengan demikian, kondisi Teorema 3.3.1 terpenuhi, sehingga  $g$  memiliki titik tetap pada selang  $[0, 1]$ .
- Jika  $x \in (0, 1)$  maka  $|g'(x)| = -\sin(x) \leq \sin(1) < 0.8415 < 1$ . Dengan demikian,  $K = \sin(1) < 1$  memenuhi Teorema 3.3.2, sehingga  $g$  memiliki titik tetap tunggal pada selang  $[0, 1]$ . △

**Teorema 3.4 :** Asumsikan bahwa (i)  $g, g' \in C[a, b]$ , (ii)  $K$  konstanta, (iii)  $p_0 \in (a, b)$ , dan (iv)  $g(x) \in [a, b]$  untuk setiap  $x \in [a, b]$

1. Jika  $|g'(x)| \leq K < 1$  untuk setiap  $x \in [a, b]$ , maka iterasi  $p_n = g(p_{n-1})$  akan konvergen menuju titik tetap tunggal  $P \in [a, b]$ .
2. Jika  $|g'(x)| > 1$  untuk setiap  $x \in [a, b]$ , maka iterasi  $p_n = g(p_{n-1})$  tidak akan konvergen menuju titik tetap  $P$ . ■

**Contoh 3.5 :** Diketahui iterasi  $p_{n+1} = g(p_n)$  dengan fungsi  $g(x) = 1 + x - x^2/4$ . Titik tetap dapat dicari dengan menyelesaikan persamaan  $x = g(x)$ . Terdapat dua solusi titik tetap yaitu  $x = -2$  dan  $x = 2$ . Fungsi turunan dari  $g$  adalah  $g'(x) = 1 - x/2$ , sehingga hanya ada 2 kasus yaitu

Kasus (i) : $P = -2$ Nilai awal $p_0 = -2.05$ $p_1 = -2.100625$ $p_2 = -2.203781$ $\vdots$ $\lim_{n \rightarrow \infty} p_n = -\infty$	Kasus (ii) : $P = 2$ Nilai awal $p_0 = 1.6$ $p_1 = 1.96$ $p_2 = 1.9996$ $\vdots$ $\lim_{n \rightarrow \infty} p_n = 2$
---	---

Pada kasus (i),  $|g'(x)| > \frac{3}{2}$  untuk  $x \in [-3, -1]$ , sehingga berdasarkan Teorema 3.4, barisan  $\{p_n\}_{n=0}^{\infty}$  tidak akan konvergen menuju  $P = -2$ . Sebaliknya, pada kasus (ii),  $|g'(x)| < \frac{1}{2}$  untuk  $x \in [1, 3]$ , sehingga berdasarkan Teorema 3.4, barisan  $\{p_n\}_{n=0}^{\infty}$  akan konvergen menuju  $P = 2$ .  $\triangle$

Teorema 3.4 tidak dapat diambil kesimpulan, apabila nilai  $g'(P) = 1$ . Pada contoh selanjutnya, iterasi titik tetap dengan  $g'(P) = 1$  dan  $p_0 > P$  akan konvergen, sedangkan iterasi titik tetap dengan  $g'(P) = 1$  dan  $p_0 < P$  akan divergen.

**Contoh 3.6 :** Diketahui iterasi  $p_{n+1} = g(p_n)$  dengan fungsi  $g(x) = 2(x-1)^{1/2}$  untuk  $x \geq 1$ . Hanya terdapat satu solusi titik tetap yaitu  $x = 2$  dan fungsi turunan dari  $g$  adalah  $g'(x) = 1/(x-1)^{1/2}$ . Nilai  $g'(2) = 1$ , sehingga Teorema 3.4 tidak berlaku. Ada dua kasus yang dapat terjadi yaitu ketika nilai awal lebih kecil atau lebih besar daripada  $P = 2$ .

Kasus (i) : Nilai awal $p_0 = 1.5$ $p_1 = 1.414213$ $p_2 = 1.287188$ $p_3 = 1.071799$ $p_4 = 0.535908$ $\vdots$ $p_5 = 2(-0.464092)^{1/2}$	Kasus (ii) : Nilai awal $p_0 = 2.5$ $p_1 = 2.449489$ $p_2 = 2.407895$ $p_3 = 2.373095$ $p_4 = 2.343582$ $\vdots$ $\lim_{n \rightarrow \infty} p_n = 2$
---	---

Pada kasus (i),  $p_4$  berada diluar daerah asal yaitu  $p_4 < 1$ , sehingga  $p_5$  tidak dapat dihitung. Sebaliknya, pada kasus (ii), barisan  $\{p_n\}_{n=0}^{\infty}$  akan konvergen menuju  $P = 2$  dengan sangat lambat, bahkan iterasi ke-1000 masih bernilai  $p_{1000} = 2.003987$ .  $\triangle$

### 3.3.3 Galat Mutlak dan Relatif

Pada Contoh 3.6, kasus (ii), hasil iterasi konvergen dengan sangat lambat. Berikut adalah hasil iterasi setelah iterasi ke-1000.

$$p_{1000} = 2.00398714$$

$$p_{1001} = 2.00398317$$

$$p_{1002} = 2.00397921$$

Hal ini seharusnya tidak mengganggu, iterasi dapat dilanjutkan hingga beberapa ribu iterasi untuk mendapatkan nilai hampiran yang lebih baik. Akan tetapi, apa kriteria yang dapat digunakan untuk menghentikan iterasi? Perbedaan dua iterasi yang berurutan adalah

$$|p_{1001} - p_{1002}| = 0.00000396$$

Sementara itu, diketahui bahwa galat mutlak dari hampiran  $p_{1000}$  adalah

$$|P - p_{1000}| = 0.00398714$$

Hasil ini lebih besar 1000 kali dibandingkan  $|p_{1001} - p_{1002}|$  dan menunjukkan bahwa kedekatan dari dua iterasi yang berurutan tidak menjamin akurasi yang didapatkan akan baik. Akan tetapi, kedekatan dari dua iterasi biasanya menjadi satu-satunya kriteria yang dapat digunakan untuk menghentikan iterasi.

### 3.4 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Bisection** berisi program untuk mencari hampiran akar persamaan tak-linear. Program ini secara *default* berisi 3 masukan, yaitu fungsi  $f(x)$  dan ujung selang  $[a, b]$  serta 3 luaran, yaitu solusi hampiran  $c$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $M$  berisi urutan iterasi, nilai  $a, c, b$ , dan  $f(c)$ . Berikut merupakan program metode *bisection*.

#### Algoritma 3.1: Metode *bisection*

```
function bisection(f,a,b)
    delta = 10^-7; maxi = 100; flag = 1;
    M = Array{Float64}(undef, 0, 5);
    fa = f(a); fb = f(b);
    if fa*fb>0
        c = "error: f(a) dan f(b) harus berbeda tanda";
        flag = 2;
        return;
    end
    k = 1
    while k<=maxi
        c = (b+a)/2;
        fc = f(c);
        M = [M; [k-1 a c b fc] ];
        if fc == 0
            a = c;
            b = c;
        elseif fa*fc>0
            a = c;
            fa = fc;
        else
            b = c;
            fb = fc;
        end
    end
```

```

        if b-a < delta || abs(fc) < delta
            flag = 0;
            break;
        end
        k+=1
    end
    return c,flag, M
end

```

**Metode Regula Falsi** berisi program untuk mencari hampiran akar persamaan tak-linear. Program ini secara *default* berisi 3 masukan, yaitu fungsi  $f(x)$  dan ujung selang  $[a,b]$  serta 3 luaran, yaitu solusi hampiran  $c$ , status solusi ( $flag$ ) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $M$  berisi urutan iterasi, nilai  $a$ ,  $c$ ,  $b$ , dan  $f(c)$ . Berikut merupakan program metode *regula falsi*.

### Algoritma 3.2: Metode *regula falsi*

```

function regulaFalsi(f,a,b)
    delta = 10^-7; maxi = 100; flag = 1;
    M = Array{Float64}(undef, 0, 5);
    fa = f(a); fb = f(b);
    if fa*fb > 0
        c = "error : fa fb harus beda tanda";
        flag = 2;
        return;
    end
    for k = 1:maxi
        c = b-fb*(b-a)/(fb-fa);
        fc = f(c);
        dx = min(c-a,b-c);
        M = [M ; [k-1 a c b fc] ];
        if fc == 0
            a = c;
            b = c;
        elseif fa*fc>0
            a = c;
            fa= fc;
        else
            b = c;
            fb= fc;
        end
        if abs(fc) < delta || abs(dx)< delta
            flag = 0; break;
        end
    end
    return c,flag,M
end

```

**Metode Iterasi Titik Tetap** berisi program untuk mencari hampiran akar persamaan tak-linear. Program ini secara *default* berisi 2 masukan, yaitu fungsi  $g$  dengan  $x = g(x)$  dan nilai tebakan awal  $p_0$  serta 3 luaran, yaitu solusi hampiran  $p_n$ , status solusi ( $flag$ ) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $M$  berisi urutan iterasi, nilai hampiran setiap iterasi  $p_n$ , dan galatnya. Berikut merupakan program metode iterasi titik tetap.

**Algoritma 3.3:** Metode iterasi titik tetap

```

function fixpoint(g,p0)
    # Definisikan nilai toleransi, maksimum iterasi dan tebakan awal
    delta = 10^-7;
    maxi = 100;
    flag = 1;
    pn = p0
    M = [0 pn NaN];
    for n = 2:maxi
        pn1 = pn;
        pn = g(pn1);
        err = abs(pn-pn1);
        relerr = err/(abs(pn)+eps());
        M = [M; [n-1 pn err]];
        if (err<delta) || (relerr<delta)
            flag = 0; break
        end
    end
    return pn, flag, M
end

```

### 3.4.1 Metode *Bisection*

Metode pertama yang dipelajari untuk mencari nilai hampiran akar persamaan adalah metode bagi-dua (*bisection*). Metode ini memiliki syarat, yaitu nilai evaluasi fungsi pada ujung selang  $[a, b]$  yaitu  $f(a)$  dan  $f(b)$  memiliki tanda yang berbeda.

**Contoh 3.7 :** Diberikan fungsi  $f(x) = x \sin(x) - 1$  pada interval  $[0, 2]$ . Berikut merupakan langkah-langkah untuk mencari nilai hampiran akar persamaan  $f(x) = 0$  menggunakan metode *bisection*.

**Langkah 1:** Pendefinisian fungsi  $f(x)$  dan titik ujung interval  $[0, 2]$ .

```
Inp: f(x) = x*sin(x)-1;
     a = 0; b = 2;
```

**Langkah 2:** Penghitungan nilai hampiran akar  $f(x) = 0$  menggunakan Program 3.1

```
Inp: c, flag, M = bisection(f,a,b)
```

```
@show c
```

```
@show flag
```

```
M
```

```
Out: c = 1.1141571998596191
     flag = 0
     22x5 Array{Float64,2}:
      0.0  0.0   1.0   2.0   -0.158529
      1.0  1.0   1.5   2.0   0.496242
      2.0  1.0   1.25  1.5   0.186231
      3.0  1.0   1.125 1.25  0.015051
      4.0  1.0   1.0625 1.125 -0.0718266
      5.0  1.0625 1.09375 1.125 -0.0283617
      :
     18.0  1.11415 1.11415 1.11416 -3.22926e-6
     19.0  1.11415 1.11416 1.11416 -5.80313e-7
     20.0  1.11416 1.11416 1.11416 7.44159e-7
     21.0  1.11416 1.11416 1.11416 8.19227e-8
```

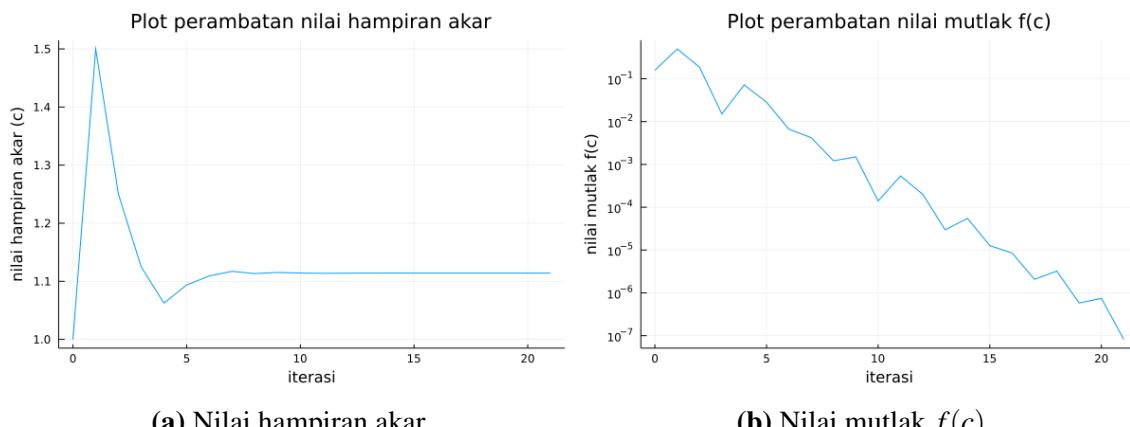
**Langkah 3:** Pembuatan plot perambatan nilai hampiran akar untuk setiap iterasi.

```
# Ambil nilai c untuk setiap n dari matriks M
iter = M[:,1];
cn   = M[:,3];
# Plot cn
p = plot(iter, cn, label = :none)
title!("Plot perambatan nilai hampiran akar")
xlabel!("iterasi")
ylabel!("nilai hampiran akar (c)")
```

**Langkah 4:** Pembuatan plot perambatan  $|f(c_n)|$  dari hampiran akar untuk setiap iterasi.

```
# Ambil nilai mutlak f(c) untuk setiap n dari matriks M
iter = M[:,1];
fc = abs.(M[:,5]);
# Plot cn
plot(iter,fc, yaxis = :log, label = :none)

title!("Plot perambatan nilai mutlak f(c)")
xlabel!("iterasi")
ylabel!("nilai mutlak f(c)")
```



(a) Nilai hampiran akar.

(b) Nilai mutlak  $f(c)$ .

**Gambar 3.6:** Plot perambatan nilai hampiran akar  $c$  dan nilai mutlak  $f(c)$  dari fungsi  $f(x) = x \sin(x) - 1$  menggunakan metode *bisection* pada interval  $[0,2]$ .

Gambar 3.6a menunjukkan perambatan nilai hampiran akar  $c$  pada setiap iterasi, yaitu iterasi ke-nol hampiran akar bernilai 1, iterasi ke-satu hampiran akar bernilai 1.5 dan seterusnya hingga pada iterasi ke-25 hampiran akar bernilai 1.11416. Gambar 3.6b menunjukkan nilai  $|f(c)|$  yang dapat diartikan sebagai akurasi dari hampiran akar, yaitu pada iterasi ke-0 hingga ke-3 akurasi hampiran lebih dari  $10^{-1}$  dan mencapai toleransi akurasi yaitu  $10^{-7}$  pada iterasi ke-24. Dengan demikian, metode *bisection* memerlukan sebanyak 24 iterasi untuk mencapai akurasi  $10^{-7}$ .

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 3.7 untuk mencari solusi akar dari persamaan

$$f(x) = \sin(x) - 2 \cos(x)$$

pada interval  $[-2, 2]$  menggunakan metode *bisection*.

### 3.4.2 Metode *Regula Falsi*

Metode *regula falsi* adalah pengembangan dari metode *bisection* dengan memperbaiki pemilihan nilai tengah  $c$  diantara selang  $[a, b]$  untuk mengurangi jumlah iterasi dan waktu komputasi. Sama seperti metode *bisection*, metode ini memiliki syarat yaitu nilai evaluasi fungsi pada ujung selang  $[a, b]$  yaitu  $f(a)$  dan  $f(b)$  memiliki tanda yang berbeda.

**Contoh 3.8 :** Diberikan fungsi  $f(x) = x \sin(x) - 1$  pada interval  $[0, 2]$ . Berikut merupakan langkah-langkah untuk mencari nilai hampiran akar  $f(x) = 0$  dengan metode *regula falsi*.

**Langkah 1:** Pendefinisian fungsi  $f(x)$  dan titik ujung interval  $[0, 2]$ .

```
Inp: f(x) = x*sin(x)-1;
      a = 0;
      b = 2;
```

**Langkah 2:** Penghitungan nilai hampiran akar  $f(x) = 0$  menggunakan Program 3.2.

```
Inp: c, flag, M = regulaFalsi(f,a,b)
      @show c
      @show flag
      M
```

```
Out: c = 1.1141571430336825
      flag = 0
      4x5 Array{Float64,2}:
      0.0 0.0 1.09975 2.0 -0.0200192
      1.0 1.09975 1.12124 2.0 0.00983461
      2.0 1.09975 1.11416 1.12124 5.63036e-6
      3.0 1.09975 1.11416 1.11416 3.00226e-9
```

**Langkah 3:** Pembuatan plot perambatan nilai hampiran akar untuk setiap iterasi.

```
# Ambil nilai c untuk setiap n dari matriks M
iter = M[:,1];
cn = M[:,3];
# Plot cn
p1 = plot(iter, cn, label = :none)

# Tambahkan title dan label
title!("Plot perambatan nilai hampiran akar")
xlabel!("iterasi")
ylabel!("nilai hampiran akar (c)")
```

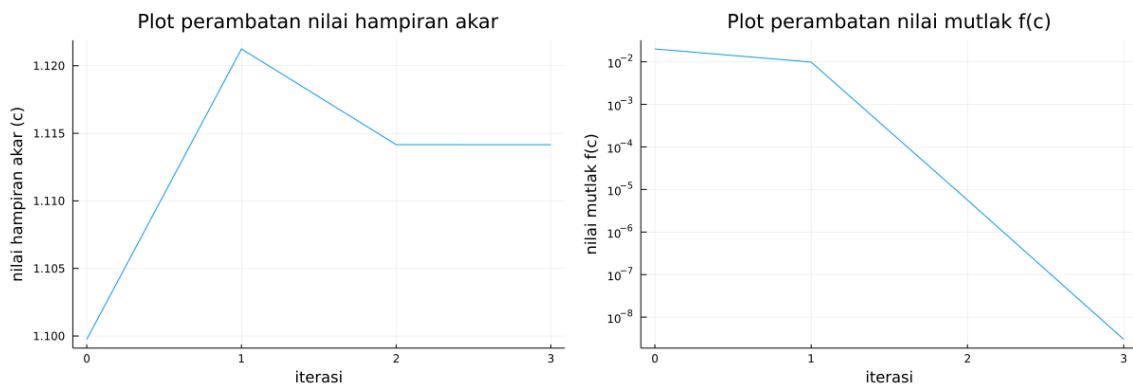
**Langkah 4:** Pembuatan plot perambatan  $|f(c_n)|$  dari hampiran akar untuk setiap iterasi.

```
# Ambil nilai mutlak f(c) untuk setiap n dari matriks M
iter = M[:,1];
fc = abs.(M[:,5]);
# Plot cn
p2 = plot(iter,fc, yaxis = :log, label = :none)

# Tambahkan title dan label
title!("Plot perambatan nilai mutlak f(c)")
xlabel!("iterasi")
ylabel!("nilai mutlak f(c)")
```

Gambar 3.7a menunjukkan perambatan nilai hampiran akar  $c$  pada setiap iterasi, yaitu iterasi ke-0 hampiran akar bernilai 1.1, iterasi ke-1 hampiran akar bernilai 1.12124 dan seterusnya hingga pada iterasi ke-4 hampiran akar bernilai 1.11416. Gambar 3.7b menunjukkan nilai  $|f(c)|$  yang dapat diartikan sebagai akurasi dari hampiran akar, yaitu pada

iterasi ke-0 akurasi hampiran lebih dari  $10^{-1}$  dan mencapai toleransi akurasi yaitu  $10^{-7}$  pada iterasi ke-3. Dengan demikian, metode *regular falsi* hanya memerlukan sebanyak 3 iterasi untuk mencapai akurasi  $10^{-7}$ , lebih sedikit dibandingkan metode *bisection*.



**Gambar 3.7:** Plot perambatan nilai hampiran akar  $c$  dan nilai mutlak  $f(c)$  dari fungsi  $f(x) = x \sin(x) - 1$  menggunakan metode *regula falsi* pada interval  $[0,2]$ .

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 3.8 untuk mencari solusi akar dari persamaan

$$f(x) = \sin(x) - 2 \cos(x)$$

pada interval  $[-2, 2]$  menggunakan metode *regula falsi*.

### 3.4.3 Iterasi Titik Tetap

Selain metode *bisection* dan *regula falsi*, terdapat metode lain yaitu metode iteratif. Pada praktikum ini, akan dipelajari salah satu metode iteratif yaitu metode iterasi titik tetap untuk mencari nilai akar persamaan.

**Contoh 3.9 :** Diberikan iterasi yang konvergen, yaitu

$$p_{k+1} = \exp(-p_k)$$

dengan  $p_0 = 0.5$ . Berikut merupakan langkah-langkah untuk menunjukkan iterasi tersebut konvergen menuju titik tetap.

**Langkah 1:** Pendefinisian persamaan iterasi dan nilai tebakan awal  $p_0$ .

```
Inp: g(x) = exp(-x);
p0 = 0.5;
```

**Langkah 2:** Penghitungan nilai iterasi titik tetap menggunakan program 3.3.

```
Inp: pn, flag, M = fixpoint(g,p0)
@show pn
@show flag
M
```

```
Out: pn = 0.5671432633594872
    flag = 0
    27x3 Array{Float64,2}:
     0.0   0.5      NaN
     1.0   0.606531  0.106531
     2.0   0.545239  0.0612914
     3.0   0.579703  0.0344639
     4.0   0.560065  0.0196385
     5.0   0.571172  0.0111075
     6.0   0.564863  0.0063092
     7.0   0.568438  0.0035751
     8.0   0.566409  0.00202859
     :
     :
     15.0  0.567157  3.82795e-5
     16.0  0.567135  2.171e-5
     17.0  0.567148  1.23127e-5
     18.0  0.567141  6.98306e-6
     19.0  0.567145  3.96039e-6
     20.0  0.567142  2.24611e-6
     21.0  0.567144  1.27387e-6
     22.0  0.567143  7.22465e-7
     23.0  0.567143  4.09741e-7
     24.0  0.567143  2.32382e-7
     25.0  0.567143  1.31794e-7
     26.0  0.567143  7.4746e-8
```

**Langkah 3:** Pembuatan plot nilai iterasi titik tetap  $p_k$  untuk setiap iterasi.

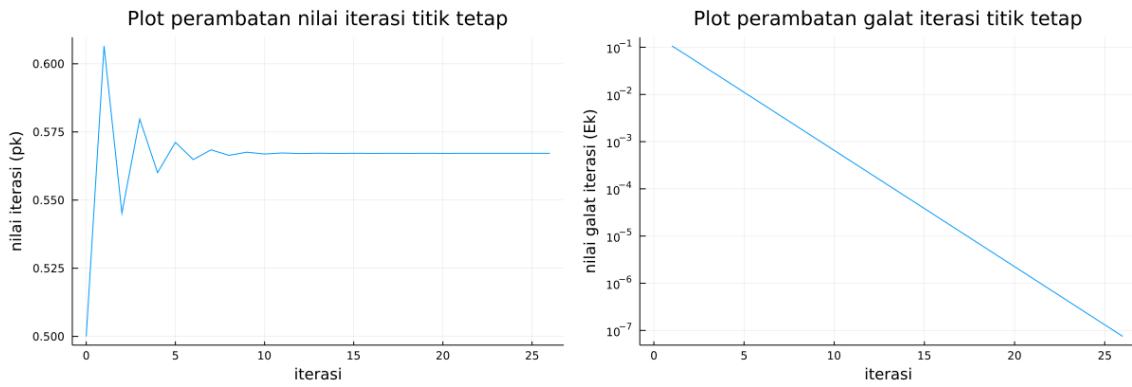
```
# Ambil nilai p_k untuk setiap k dari matriks M yaitu pada kolom ke-2.
iter = M[:,1];
pk   = M[:,2];
# Plot pk
plot(iter,pk, label = :none)
# Tambahkan grid, title dan label
title!("Plot perambatan nilai iterasi titik tetap")
xlabel!("iterasi")
ylabel!("nilai iterasi (pk)")
```

**Langkah 4:** Pembuatan plot galat iterasi titik tetap untuk setiap iterasi.

```
# Ambil nilai E_k untuk setiap k dari matriks M yaitu pada kolom ke-3.
iter = M[:,1];
Ek   = M[:,3];
# Plot Ek
plot(iter,Ek, yaxis = :log, label = :none)

title!("Plot perambatan galat iterasi titik tetap")
xlabel!("iterasi")
ylabel!("nilai galat iterasi (Ek)")
```

Gambar 3.8a menunjukkan perambatan nilai hampiran akar  $p_k$  pada setiap iterasi  $k$ , yaitu iterasi pertama ke-nol hampiran akar bernilai 1.1, iterasi ke-1 hampiran akar bernilai 1.12124 dan seterusnya hingga pada iterasi ke-4 hampiran akar bernilai 1.11416. Gambar 3.8b menunjukkan nilai  $|f(c)|$  yang dapat diartikan sebagai akurasi dari hampiran akar, yaitu pada iterasi ke-0 akurasi hampiran lebih dari  $10^{-1}$  dan mencapai toleransi akurasi yaitu  $10^{-7}$  pada iterasi ke-3. Dengan demikian, metode *regular falsi* hanya memerlukan sebanyak 3 iterasi untuk mencapai akurasi  $10^{-7}$ , lebih sedikit dibandingkan metode *bisection*.

(a) Nilai iterasi  $p_k$ .(b) Nilai galat iterasi  $E_k$ .

**Gambar 3.8:** Plot perambatan nilai iterasi  $p_k$  dan galat iterasi  $E_k$  dari persamaan iterasi  $p_{k+1} = \exp(-p_k)$ .

**Soal Latihan:** Diberikan iterasi  $p_{n+1} = g(p_n)$  menggunakan fungsi

$$g(x) = 1 + x - x^2/4$$

Secara analitik, terdapat dua titik tetap yaitu  $p = -2$  dan  $P = 2$ .

1. Tunjukkan bahwa iterasi titik tetap akan konvergen menuju  $P = 2$  apabila  $p_0 = 1.6$ .
2. Tunjukkan bahwa iterasi titik tetap akan tidak konvergen menuju  $P = -2$  apabila  $p_0 = -2.05$ .

## 3.5 Latihan-latihan

### 3.5.1 Ulasan Materi

1. Jelaskan apa yang dimaksud dengan kekonvergenan lokal dan global!
2. Jelaskan kekurangan dan kelebihan serta langkah iterasi dari metode-metode berikut
  - (a) Metode *bisection*.
  - (b) Metode *regula-falsi*.
  - (c) Metode iterasi titik tetap.
3. Jelaskan langkah iterasi dari metode *bisection* dan *regula falsi*!
4. Jelaskan syarat supaya iterasi dari metode *bisection* dan *regula falsi* menghasilkan solusi yang konvergen menuju akar persamaan!
5. Jelaskan syarat supaya iterasi titik tetap menghasilkan solusi yang konvergen menuju akar persamaan!
6. Sebutkan dan jelaskan kriteria-kriteria penghentian iterasi pada metode *bisection*, *regula falsi*, dan iterasi titik tetap!
7. Manakah di antara pernyataan-pernyataan berikut yang benar dan salah?
  - (a) Kekonvergenan lokal selalu konvergen tidak tergantung nilai tebakan awal.
  - (b) Kekonvergenan global akan konvergen hanya bila nilai tebakan awal berada dalam selang tertentu.
  - (c) Metode *bisection* dan *regula falsi* tergolong ke dalam kekonvergenan lokal.
  - (d) Metode *bisection* dan *regula falsi* dapat menemukan akar persamaan tak linear  $x^2 + 2x + 1 = 0$ .

- (e) Pada metode iterasi titik tetap, misalkan  $p_0 \neq P$ . Bila  $|g'(x)| > 1$  untuk semua  $x \in [a, b]$ , maka iterasi  $p_n = g(p_{n-1})$  akan Konvergen ke  $P$ .
- (f) Metode *bisection* lebih cepat mencari akar persamaan tak-linear dibandingkan metode regula falsi.
- (g) Metode iterasi titik tetap membutuhkan selang pencarian untuk menemukan akar persamaan tak linear, seperti metode *bisection*.

### 3.5.2 Soal Pemrograman

1. Bandingkan penggunaan metode *bisection* dan *regula falsi* untuk menghitung nilai hampiran akar dari fungsi:

- (a)  $f(x) = x^2 - e^x$  untuk  $-2 \leq x \leq 2$
- (b)  $f(x) = \sin(x) - 2 \cos(x)$  untuk  $-2 \leq x \leq 2$
- (c)  $f(x) = (x-2)^2 - \ln(x)$  untuk  $0.5 \leq x \leq 4.5$
- (d)  $f(x) = 2x - \tan(x)$  untuk  $-1.4 \leq x \leq 1.4$

2. Diketahui suatu fungsi

$$f(x) = x^4 - 5x^3 + \frac{22}{3}x^2 - \frac{116}{27}x + \frac{8}{9}$$

- (a) Cek bahwa  $f$  memiliki nilai akar persamaan  $\alpha_1$  diantara 0 dan 1 serta nilai akar persamaan lain  $\alpha_2$  diantara 1 dan 4.

(b) Cari nilai hampiran dari akar tersebut menggunakan *bisection* dan *regula-falsi*.

3. Diberikan fungsi

$$f(x) = (x-2)^2 - \ln(x)$$

pada interval  $x \in [1, 2]$ .

- (a) Buktikan bahwa fungsi  $f(x)$  pada selang tersebut memiliki tepat satu akar.
- (b) Gunakan metode *bisection* untuk menghitung nilai akar persamaan hingga memiliki 6 digit desimal akurasi.
- (c) Berapa banyak iterasi yang diperlukan untuk memenuhi akurasi tersebut.

4. Diberikan fungsi

$$f(x) = (x-2)^2 - \ln(x)$$

pada interval  $x \in [1, 2]$ .

- (a) Buktikan bahwa fungsi  $f(x)$  pada selang tersebut memiliki tepat satu akar.
- (b) Gunakan metode *regula-falsi* untuk menghitung nilai akar persamaan hingga memiliki 6 digit desimal akurasi.
- (c) Berapa banyak iterasi yang diperlukan untuk memenuhi akurasi tersebut.

5. Diketahui fungsi

$$f(x) = x + 4 \cos(x)$$

memiliki 3 akar persamaan  $f(x) = 0$ . Gunakan metode *bisection* dan *regula falsi* untuk mencari nilai hampiran ketiga akar tersebut.

6. Diberikan fungsi

$$f(x) = 0.1x^2 - x \ln(x)$$

Cari nilai akar persamaan tersebut menggunakan metode *bisection* dan *regula falsi* hingga memiliki akurasi 4 digit desimal.

7. Diberikan persamaan tak-linear  $e^x - 2 - x = 0$ . Gunakan metode bisection and regula falsi untuk menyelesaikan masalah persamaan tak-linear tersebut. Selanjutnya, tunjukkan barisan  $c_n$  (dalam bentuk tabel dan grafik) pada penggunaan kedua metode tersebut.
8. Untuk setiap fungsi berikut, tentukan interval  $[a, b]$ , sehingga solusi iterasi titik tetap  $x = g(x)$  akan konvergen.
- $g(x) = 0.2 \sin(x) + 1$
  - $g(x) = 1 - \frac{x^2}{4}$
9. Persamaan kuadrat  $f(x) = x^2 - 2x - 3$  mempunyai dua akar persamaan. Tentukan hampiran nilai akar dari  $f(x)$  menggunakan iterasi titik tetap, jika diketahui  $x_0 = 4$  dan fungsi  $g(x)$  seperti berikut.
- $g(x) = \sqrt{2x + 3}$
  - $g(x) = \frac{3}{x - 2}$
  - $g(x) = \frac{x^2 - 3}{2}$
10. Selesaikan persamaan berikut menggunakan iterasi titik tetap.
- $x = \sin(x) + x + 1$  pada selang  $x \in [3.5, 5]$
  - $x = \ln(x^2) + x - 2$  pada selang  $x \in [-4, -2]$
11. Carilah solusi beberapa fungsi berikut menggunakan iterasi titik tetap  $x = g(x)$ .
- $g(x) = x^5 - 0.25$  nilai awal  $x_0 = 0$ .
  - $g(x) = 2 \sin(x)$  nilai awal  $x_0 = 2$ .
  - $g(x) = \sqrt{3x + 1}$  nilai awal  $x_0 = 2$ .
  - $g(x) = \frac{2 - e^x + x^2}{3}$  nilai awal  $x_0 = 1$ .
12. Tunjukkan bahwa fungsi

$$f(x) = x^2 + 3x - 4$$

memiliki tepat satu nilai akar dan berada diantara  $x \in [0, 1]$ . Cari nilai akar tersebut menggunakan metode iterasi titik tetap hingga memiliki akurasi 4 angka desimal. Gunakan fungsi iterasi

$$x_k = \frac{4}{x_{k-1}^2 + 3}$$

13. Diberikan fungsi  $g(x)$  sehingga  $x = g(x)$  sebagai berikut

$$g(x) = -4 + 4x - \frac{1}{2}x^2$$

- Selesaikan  $x = g(x)$  kemudian tunjukkan  $P = 2$  dan  $P = 4$  merupakan titik tetap dari fungsi  $g$ .
- Dengan metode iterasi titik tetap, carilah titik tetap  $g(x)$  jika diketahui titik awal  $p_0 = 1$ ,  $p_0 = 1.8$ ,  $p_0 = 2.2$ ,  $p_0 = 3.6$ , dan  $p_0 = 4.4$ .
- Plot perambatan nilai iterasi  $p_k$  dan galat  $E_k$  dari masing-masing titik awal tersebut, kemudian bandingkan!
- Apakah hasil menuju titik  $P = 2$  dan  $P = 4$ . Jelaskan alasannya !

14. Carilah dua akar persamaan dari

$$f(x) = 2^x - 2x$$

15. Pada tahun 1225 Leonardo of Pisa mempelajari persamaan

$$x^3 + 2x^2 + 10x - 20 = 0$$

dan menemukan akar yaitu  $x = 1.368\ 808\ 107$ . Tidak ada satupun orang yang mengetahui metode yang digunakan untuk mencari nilai tersebut. Akan tetapi, itu merupakan hal luar biasa untuk memperoleh 9 digit desimal pada jaman itu. Carilah akar persamaan tersebut dengan metode yang telah anda pelajari.

16. Disediakan persamaan

$$\lambda x = \tan x$$

Dalam ilmu fisika, persamaan ini menggambarkan keadaan partikel mekanika kuantum dalam kotak persegi panjang dengan dinding berhingga. Persamaan ini tidak dapat diselesaikan secara analitis dan perlu menggunakan pencarian akar numerik. Analisis persamaan ini untuk tiga nilai positif  $\lambda$ , yaitu  $0 < \lambda < 1$ ,  $\lambda = 1$ , dan  $\lambda > 1$ . (pilih nilai  $\lambda$ )

17. Persamaan Peng-Robinson didefinisikan sebagai

$$P = \frac{RT}{V-b} - \frac{a}{V(V+b)+b(V-b)}$$

dimana  $P$  = tekanan,  $V$  = volume,  $T$  = suhu, dan  $R$  = konstanta gas ideal. Carilah nilai  $V$  jika diketahui

$$P = 778\text{kPa}$$

$$T = 350\text{K}$$

$$a = 365$$

$$b = 0.3$$

$$R = 1.618$$

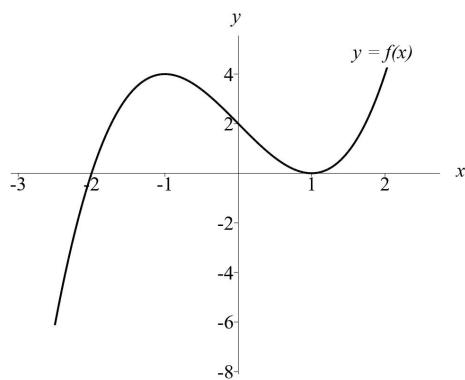
---

# BAB 4

---

## SOLUSI PERSAMAAN TAK-LINEAR 2

Pada pertemuan sebelumnya, telah dibahas beberapa metode untuk mencari hampiran solusi persamaan tak-linear. Akan tetapi, metode *bisection* dan *regula falsi* hanya dapat digunakan untuk menentukan hampiran dari akar sederhana. Perhatikan gambar berikut.



**Gambar 4.1:** Grafik fungsi  $f(x) = x^3 - 3x + 2$

Metode *bisection* dan *regula falsi* hanya mampu untuk menghitung hampiran dari nilai akar  $P = -2$  karena selang di sekitar titik tersebut memiliki tanda yang berbeda (positif dan negatif). Akan tetapi, kedua metode tadi tidak mampu untuk mengidentifikasi nilai akar  $P = 1$  karena selang di sekitar titik tersebut memiliki tanda yang sama (positif). Nilai akar  $P = 1$  disebut sebagai *double root* dan memiliki ordo  $M = 2$ .

### 4.1 Ordo Akar dan Kecepatan Kekonvergenan

**Definisi 4.1 Ordo akar:** Asumsikan bahwa  $f(x)$  dan turunannya  $f'(x), \dots, f^{(M)}(x)$  terdefinisi dan kontinu di sekitar  $x = p$ .  $f(x)$  memiliki akar dengan ordo  $M$  saat  $x = p$ , jika dan hanya jika

$$f(p), f'(p), f''(p), \dots, f^{(M-1)}(p) = 0 \quad \text{tetapi} \quad f^{(M)}(p) \neq 0 \quad (4.1)$$

Akar dengan ordo  $M = 1$  disebut dengan akar sederhana (*simple root*) dan jika  $M > 1$  disebut dengan akar berganda (*multiple root*). Akar dengan ordo  $M = 2$  seringkali disebut dengan akar ganda (*double root*) ■

**Lema 4.1 :** Jika  $f(x) = 0$  memiliki akar dengan ordo  $M$  saat  $x = p$ , maka terdapat fungsi kontinu  $h(x)$  sedemikian sehingga  $f(x)$  dapat diekspresikan dalam bentuk perkalian

$$f(x) = (x - p)^M h(x) \quad (4.2)$$

dimana  $h(p) \neq 0$  ■

**Contoh 4.1 :** Fungsi  $f(x) = x^3 - 3x + 2$  memiliki *simple root* saat  $p = -2$  dan *double root* saat  $p = 1$ . Ini dapat dibuktikan oleh  $f'(x) = 3x^2 - 3$  dan  $f''(x) = 6x$ . Saat nilai  $p = -2$ , didapatkan  $f(-2) = 0$  dan  $f'(-2) = 9$ , sehingga  $M = 1$  berdasarkan Definisi 4.1, karenanya  $p = -2$  adalah *simple root*. Sementara itu, saat  $p = 1$ , didapatkan  $f(1) = 0$ ,  $f'(1) = 0$ , dan  $f''(1) = 6$ , sehingga  $M = 2$  berdasarkan Definisi 4.1, karenanya  $p = 1$  adalah *double root*. Selain itu, dapat dilihat dari hasil faktorisasi dari fungsi  $f(x)$  yaitu  $f(x) = (x - 1)^2(x + 2)$ , sehingga  $p = 1$  berordo 2 dan  $p = -2$  berordo 1. Perbandingan grafik akar sederhana dan ganda dapat dilihat pada Gambar 4.1. △

**Definisi 4.2 Ordo kekonvergenan:** Asumsikan bahwa  $\{p_n\}_{n=0}^{\infty}$  konvergen ke  $p$  dan  $E_n = p - p_n$  untuk  $n \geq 0$ . Jika terdapat dua konstanta positif  $A \neq 0$  dan  $R > 0$ , serta

$$\lim_{n \rightarrow \infty} \frac{|p - p_{n+1}|}{|p - p_n|^R} = \lim_{n \rightarrow \infty} \frac{|E_{n+1}|}{|E_n|^R} = A \quad (4.3)$$

maka barisan  $\{p_n\}_{n=0}^{\infty}$  dapat dikatakan konvergen ke  $p$  dengan ordo kekonvergenan  $R$ . Nilai  $A$  disebut dengan konstanta galat asimtotik.

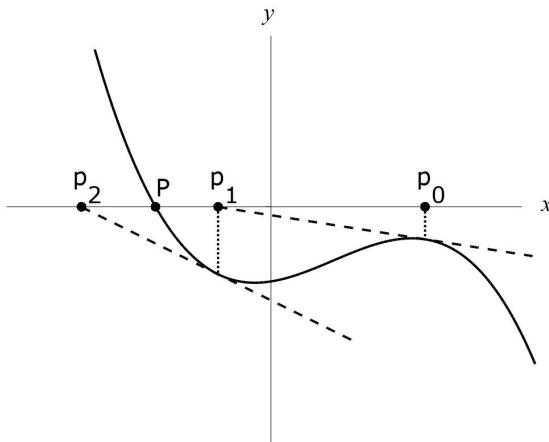
1. Jika  $R = 1$ , kekonvergenan  $\{p_n\}_{n=0}^{\infty}$  disebut linear
2. Jika  $R = 2$ , kekonvergenan  $\{p_n\}_{n=0}^{\infty}$  disebut kuadratik ■

Kelemahan dari metode tertutup adalah tidak mampu untuk mencari akar yang berordo genap. Selain itu, metode tertutup memiliki ordo kekonvergenan yang rendah, yaitu  $R = 1/2$  untuk metode *bisection* dan  $R = 1$  untuk metode *regula falsi*, sehingga membutuhkan waktu komputasi yang lama untuk memenuhi kriteria yang ditetapkan.

## 4.2 Metode Newton-Raphson

Metode Newton-Raphson (bisa disebut juga sebagai metode Newton) adalah salah satu algoritma yang sering digunakan untuk mencari solusi persamaan tak-linear. Dengan informasi mengenai kekontinuan  $f(x)$ ,  $f'(x)$ , dan  $f''(x)$ , metode Newton-Raphson dapat menghasilkan barisan  $\{p_k\}$  yang konvergen menuju nilai akar  $p$  lebih cepat dibandingkan metode *bisection* maupun *regula falsi*.

Asumsikan bahwa nilai hampiran awal  $p_0$  berada dekat dengan nilai akar  $p$ , kemudian kurva  $y = f(x)$  memotong sumbu- $x$  tepat pada titik  $(p, 0)$  dan titik  $(p_0, f(p_0))$  berada didekat titik  $(p, 0)$ . Definisikan nilai  $p_1$  sebagai titik pada sumbu- $x$  yang terpotong oleh garis singgung kurva pada titik  $(p_0, f(p_0))$ . Pada Gambar 4.2, dapat dilihat bahwa jarak  $p_1$  dengan  $p$  lebih dekat dibandingkan dengan jarak  $p_0$  dengan  $p$  pada kasus tersebut.



**Gambar 4.2:** Penentuan nilai  $p_1$  dan  $p_2$  pada metode Newton-Raphson

Persamaan yang menghubungkan antara  $p_0$  dan  $p_1$  dapat dicari melalui dua jenis persamaan kemiringan garis singgung  $L$  yaitu

$$m = \frac{0 - f(p_0)}{p_1 - p_0} \quad (4.4)$$

dimana persamaan tersebut adalah garis yang melewati garis  $(p_1, 0)$  dan  $(p_0, f(p_0))$ , serta

$$m = f'(p_0) \quad (4.5)$$

dimana persamaan tersebut adalah kemiringan pada titik  $(p_0, f(p_0))$ . Dengan menyamakan nilai dari  $m$  pada persamaan 4.4 dan 4.5, maka hasil nilai  $p_1$  adalah

$$p_1 = p_0 - \frac{f(p_0)}{f'(p_0)} \quad (4.6)$$

Proses di atas, dapat diulangi untuk mendapatkan barisan  $\{p_k\}$  yang konvergen menuju  $p$  untuk memperbaiki akurasi dan presisi solusi yang dihasilkan.

**Teorema 4.1 :** Asumsikan bahwa  $f \in C^2[a, b]$  dan terdapat  $p \in [a, b]$  dimana  $f(p) = 0$ . Jika  $f'(p) \neq 0$ , maka terdapat  $\delta > 0$  sedemikian sehingga barisan  $\{p_k\}_{k=0}^{\infty}$  yang terdefinisikan oleh iterasi

$$p_k = g(p_{k-1}) = p_{k-1} - \frac{f(p_{k-1})}{f'(p_{k-1})} \quad (4.7)$$

konvergen menuju  $p$  untuk nilai hampiran awal  $p_0 \in [p - \delta, p + \delta]$ .

*Catatan.* Fungsi  $g(x)$  yang terdefinisikan oleh persamaan

$$g(x) = x - \frac{f(x)}{f'(x)} \quad (4.8)$$

disebut sebagai fungsi iterasi Newton-Raphson. Apabila  $f(p) = 0$ , maka dapat dilihat bahwa  $g(p) = p$ . Dengan demikian, iterasi Newton-Raphson untuk mencari nilai akar persamaan  $f(x) = 0$  dapat dicapai dengan mencari titik tetap dari fungsi  $g(x)$ . ■

**Contoh 4.2 Metode Newton-Raphson untuk Simple Root:** Gunakan iterasi Newton-Raphson untuk mencari nilai akar  $p = -2$  dari fungsi  $f(x) = x^3 - 3x + 2$  dengan nilai awal  $p_0 = -2.4$ . Persamaan iterasi untuk menghitung  $\{p_k\}$  adalah

$$p_k = g(p_{k-1}) = p_{k-1} - \frac{p_{k-1}^3 - 3p_{k-1} + 2}{3p_{k-1}^2 - 3} = \frac{2p_{k-1}^3 - 2}{3p_{k-1}^2 - 3} \quad (4.9)$$

Hasil penghitungan nilai akar dapat dilihat pada Tabel 4.1. Berdasarkan tabel tersebut, kekonvergenan iterasi memiliki ordo-2 atau kuadratik dengan  $A \approx 2/3$ .  $\triangle$

**Tabel 4.1:** Metode Newton-Raphson konvergen secara kuadratik untuk *simple root*

$k$	$p_k$	$p_{k+1} - p_k$	$E_k = p - p_k$	$\frac{ E_{k+1} }{ E_k ^2}$
0	-2.400000000	0.323809524	0.400000000	0.476190476
1	-2.076190476	0.072594466	0.076190476	0.619469027
2	-2.003596011	0.003587421	0.003596011	0.664277916
3	-2.000008590	0.000008590	0.000008590	0.666642773
4	-2.000000000	0.000000000	0.000000000	

**Contoh 4.3 Metode Newton-Raphson untuk Double Root:** Gunakan iterasi Newton-Raphson untuk mencari nilai akar  $p = 1$  dari fungsi  $f(x) = x^3 - 3x + 2$  dengan nilai awal  $p_0 = 1.2$ . Dengan persamaan iterasi 4.9, hasil penghitungan nilai akar dapat dilihat pada Tabel 4.2. Berdasarkan tabel tersebut, kekonvergenan iterasi memiliki ordo-1 atau linear dengan  $A \approx 1/2$ .  $\triangle$

**Tabel 4.2:** Metode Newton-Raphson konvergen secara linear untuk *double root*

$k$	$p_k$	$p_{k+1} - p_k$	$E_k = p - p_k$	$\frac{ E_{k+1} }{ E_k ^2}$
0	1.200000000	-0.096969697	-0.200000000	0.515151515
1	1.103030303	-0.050673886	-0.103030303	0.508165226
2	1.052356417	-0.025955603	-0.052356417	0.504251732
3	1.026400814	-0.013143080	-0.026400814	0.502171404
4	1.013257734	-0.006614316	-0.013257734	0.501097536
5	1.006643418	-0.003318043	-0.006643418	0.500551785
:	:	:	:	:

### Metode Newton-Raphson untuk akar kuadrat

Asumsikan bahwa  $A > 0$  merupakan bilangan real dan diberikan  $p_0 > 0$  sebagai nilai hampiran awal untuk  $\sqrt{A}$ . Definisikan barisan  $\{p_k\}_{k=0}^{\infty}$  menggunakan aturan rekursif

$$p_k = \frac{p_{k-1} + \frac{A}{p_{k-1}}}{2} \quad (4.10)$$

untuk  $k = 1, 2, \dots$ , sedemikian sehingga barisan  $\{p_k\}_{k=0}^{\infty}$  konvergen menuju  $\sqrt{A}$ , yaitu

$$\lim_{n \rightarrow \infty} p_k = \sqrt{A}$$

**Contoh 4.4 :** Gunakan algoritma Newton untuk menghitung nilai  $\sqrt{5}$

**Solusi :** Dengan menggunakan nilai awal  $p_0 = 2$  dan formula 4.10, hitung nilai

$$\begin{aligned} p_1 &= \frac{2+5/2}{2} = 2.25 \\ p_2 &= \frac{2.25+5/2.25}{2} = 2.236111 \\ p_3 &= \frac{2.236111+5/2.236111}{2} = 2.236067978 \\ p_4 &= \frac{2.236067978+5/2.236067978}{2} = 2.236067978 \end{aligned}$$

Iterasi selanjutnya akan tetap menghasilkan nilai  $p_k = 2.236067978$  untuk  $k > 4$ , sehingga nilai akar yang didapatkan sudah memiliki nilai akurasi 9 angka desimal.  $\triangle$

### 4.3 Metode Secant

Algoritma Newton-Raphson membutuhkan evaluasi dari dua nilai untuk setiap langkahnya, yaitu  $f(p_{k-1})$  dan  $f'(p_{k-1})$ . Secara umum, evaluasi nilai turunan membutuhkan usaha yang cukup besar, meskipun dengan *software* aljabar turunan-turunan umum tidak menjadi masalah, tetapi masih banyak fungsi-fungsi rumit yang memiliki turunan yang rumit pula. Dengan demikian, dibutuhkan metode lain yang konvergen hampir secepat metode Newton, namun hanya melibatkan evaluasi fungsi  $f(x)$  dan tidak perlu evaluasi fungsi  $f'(x)$ . Metode *secant* hanya akan memerlukan evaluasi  $f(x)$  untuk setiap langkah dan untuk *simple root* memiliki ordo kekonvergenan  $R \approx 1.618033989$ , dimana hampir secepat metode Newton yang memiliki ordo 2.

Rumus yang digunakan pada metode *secant* sejatinya sama dengan yang digunakan pada metode *regula falsi*, kecuali pengambilan keputusan untuk nilai yang digunakan pada iterasi selanjutnya. Metode *regula falsi* menentukan nilai untuk iterasi selanjutnya menggunakan titik potong sumbu- $x$  dari garis yang dibentuk dari titik ujung interval  $[a, b]$ . Sebaliknya, metode *secant* tidak memerlukan interval  $[a, b]$  untuk titik awal. Metode *secant* menggunakan nilai awal  $p_0$  dan  $p_1$ , dimana  $f(p_0)$  dan  $f(p_1)$  tidak harus memiliki tanda yang berlawanan. Dua titik awal  $(p_0, f(p_0))$  dan  $(p_1, f(p_1))$  harus berada dekat dengan titik  $(p, 0)$ , seperti yang diperlihatkan oleh Gambar 4.3. Definisikan nilai  $p_2$  sebagai titik pada sumbu- $x$  yang dilewati oleh garis yang terbentuk oleh dua titik awal sebelumnya, sehingga nilai  $p_2$  berada lebih dekat dengan  $p$  dibandingkan  $p_1$  dan  $p_0$ .

Persamaan yang menghubungkan  $p_2, p_1$ , dan  $p_0$  dapat dicari menggunakan rumus kemiringan antara dua titik yaitu

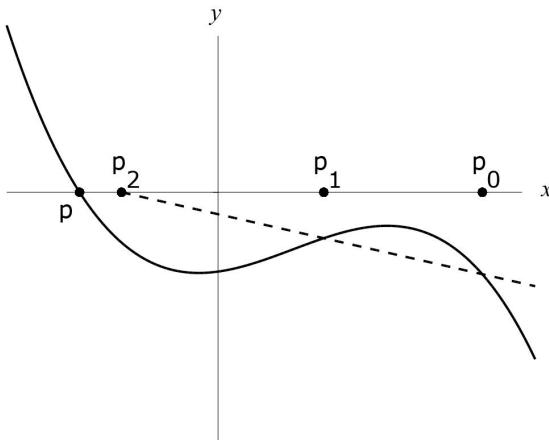
$$m = \frac{f(p_1) - f(p_0)}{p_1 - p_0} \quad \text{dan} \quad m = \frac{0 - f(p_1)}{p_2 - p_1} \quad (4.11)$$

Tentukan persamaan untuk  $p_2$  dengan cara menyamakan kedua rumus  $m$  tersebut, yaitu

$$p_2 = g(p_1, p_0) = p_1 - \frac{f(p_1)(p_1 - p_0)}{f(p_1) - f(p_0)} \quad (4.12)$$

Dengan demikian, didapatkan bentuk umum persamaan iterasi *secant* yaitu

$$p_{k+1} = g(p_k, p_{k-1}) = p_k - \frac{f(p_k)(p_k - p_{k-1})}{f(p_k) - f(p_{k-1})} \quad (4.13)$$



**Gambar 4.3:** Gambaran geometri pembentukan nilai  $p_2$  pada metode *secant*.

**Contoh 4.5 Metode *secant* untuk *simple root*:** Gunakan metode *secant* untuk mencari nilai akar  $p = -2$  dari fungsi  $f(x) = x^3 - 3x + 2$ . Iterasi dimulai dengan  $p_0 = -2.6$  dan  $p_1 = -2.4$ . Pada kasus ini, iterasi *secant* pada persamaan 4.13 berubah menjadi

$$p_{k+1} = p_k - \frac{(p_k^3 - 3p_k + 2)(p_k - p_{k-1})}{p_k^3 - p_{k-1}^3 - 3p_k + 3p_{k-1}} \quad (4.14)$$

Hasil penghitungan metode *secant* dapat dilihat pada Tabel 4.3. Berdasarkan tabel tersebut, kekonvergenan iterasi memiliki ordo  $R \approx 1.618$  dengan  $A \approx (2/3)^{1.618}$ .  $\triangle$

**Tabel 4.3:** Kekonvergenan metode *secant* untuk *simple root*

$k$	$p_k$	$p_{k+1} - p_k$	$E_k = p - p_k$	$\frac{ E_{k+1} }{ E_k ^{1.618}}$
0	-2.600000000000	0.200000000000	0.600000000000	0.91413696
1	-2.400000000000	0.293401015228	0.400000000000	0.46948314
2	-2.106598984772	0.083957572465	0.106598984772	0.84722556
3	-2.022641412307	0.021130314977	0.022641412307	0.69351930
4	-2.001511097330	0.001488560847	0.001511097330	0.82564051
5	-2.000022536484	0.000022513798	0.000022536484	0.74952218
6	-2.000000022686	0.000000022685	0.000000022686	0.79449020
7	-2.000000000000	0.000000000000	0.000000000000	

## 4.4 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Newton-Raphson** berisi program untuk mencari hampiran akar persamaan tak-linear. Program ini secara *default* berisi 3 masukan, yaitu fungsi  $f$ , turunannya  $f'$ ,

dan nilai tebakan awal  $p_0$ , serta 3 luaran, yaitu solusi hampiran  $p_k$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $M$  berisi urutan iterasi, nilai hampiran setiap iterasi  $p_k$ , dan galatnya. Berikut merupakan program metode Newton-Raphson.

#### Algoritma 4.1: Metode Newton-Raphson

```
function newtonRaphson(f, df, p0)
    # Definisikan nilai toleransi, maksimum iterasi dan tebakan awal
    delta = 10^-7;
    maxi = 100;
    pk = p0
    M = [0 pk NaN];
    # Mulai langkah iterasi
    flag = 1;
    for k = 2:maxi
        # Rumus metode Newton-Raphson
        pk1 = pk
        pk = pk1 - f(pk1)/df(pk1);
        # Hitung nilai galat mutlak dan relatif
        err = abs(pk-pk1);
        rel = 2*err/(abs(pk)+eps());
        M = [M; [k-1 pk err]];
        # Kriteria penghentian iterasi jika galat memenuhi toleransi.
        if err<delta || rel<delta
            flag = 0;
            break
        end
    end
    return pk, flag, M
end
```

**Metode Secant** berisi program untuk mencari nilai hampiran akar persamaan tak-linear. Program ini secara *default* berisi 3 masukan, yaitu fungsi  $f$ , dua tebakan awal  $p_0$ , dan  $p_1$ , serta 3 luaran, yaitu solusi hampiran  $p_k$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $M$  berisi urutan iterasi, nilai hampiran setiap iterasi  $p_k$ , dan galatnya. Berikut merupakan program metode *secant*.

#### Algoritma 4.2: Metode secant

```
function secant(f, p0, p1)
    # Definisikan nilai toleransi, maksimum iterasi dan tebakan awal.
    delta = 10^-12;
    maxi = 100;
    M = [0 p0 NaN
          1 p1 NaN];
    p = [p0 p1];
    pk = p1;
    flag = 1;
    # Mulai langkah iterasi
    for k = 2:maxi
        # rumus metode secant
        pk=p[k]-f(p[k])*(p[k]-p[k-1])/(f(p[k])-f(p[k-1]));
        p = [p pk];
        # Hitung nilai galat mutlak dan relatif
        err=abs(p[k+1]-p[k]);
        rel=2*err/(abs(p[k]) + eps());
```

```

M = [M; [k pk err]]
# Kriteria penghentian iterasi jika galat memenuhi toleransi.
if err<delta || rel<delta
    flag = 0;
    break
end
end
return pk, flag, M
end

```

#### 4.4.1 Metode Newton-Raphson

Metode *bisection* dan *regula falsi* yang telah dipelajari sebelumnya masih memiliki jumlah iterasi dan waktu komputasi yang cukup lama. Dengan tambahan informasi mengenai turunan dari fungsi  $f(x)$ , metode Newton-Raphson dapat digunakan untuk mencari nilai akar dengan lebih cepat dibandingkan metode sebelumnya.

##### Solusi *Simple Root*

Persamaan dengan solusi *simple root* adalah suatu persamaan yang memiliki akar tunggal. Contohnya, persamaan  $x^2 - 1 = 0$  memiliki 2 akar tunggal, yaitu  $x = -1$  dan  $x = 1$ .

**Contoh 4.6 :** Diberikan fungsi  $f(x) = x^3 - 3x + 2$  dan  $p_0 = -2.4$ . Berikut merupakan langkah-langkah untuk mencari hampiran akar  $f(x) = 0$  yaitu  $P = -2$  menggunakan metode Newton-Raphson serta menunjukkan bahwa metode Newton-Raphson memiliki ordo kekonvergenan  $R = 2$  untuk *simple root*.

**Langkah 1:** Penghitungan fungsi turunan dari  $f(x)$ .

Diketahui bahwa  $f(x) = x^3 - 3x + 2$ , sehingga turunan fungsi  $f$  adalah

$$f'(x) = 3x^2 - 3$$

**Langkah 2:** Pendefinisian fungsi  $f$  dan turunannya  $f'$  serta nilai awal  $p_0$ .

```
Inp: f(x) = x^3-3*x+2;
df(x) = 3*x^2-3;
p0 = -2.4;
```

**Langkah 3:** Penghitungan nilai hampiran akar menggunakan metode Newton-Raphson pada Program 4.1.

```
Inp: pk,flag,M = newtonRaphson(f,df,p0)
@show pk
@show flag
M
```

```
Out: pk = -2.0
flag = 0
6x3 Array{Float64,2}:
 0.0  -2.4      NaN
 1.0  -2.07619   0.32381
 2.0  -2.0036    0.0725945
 3.0  -2.00001   0.00358742
 4.0  -2.0       8.58992e-6
 5.0  -2.0       4.91913e-11
```

**Langkah 4:** Penghitungan nilai galat  $E_k = P - p_k$ .

```
Inp: P = -2;
     Ek = P.-M[:, 2]
```

```
Out: 6-element Array{Float64,1}:
      0.3999999999999999
      0.07619047619047592
      0.0035960106756567356
      8.589972221084707e-6
      4.9191317685881586e-11
      0.0
```

**Langkah 5:** Penghitungan nilai rasio galat  $|E_{k+1}|/|E_k|^2$ .

```
Inp: R = 2;
      RasioGalat = abs.(Ek[2:end]) ./ abs.(Ek[1:end-1].^R)
```

```
Out: 5-element Array{Float64,1}:
      0.4761904761904747
      0.6194690265486843
      0.6642779161836095
      0.6666608280164035
      0.0
```

Berdasarkan hasil yang didapatkan, nilai  $|E_{k+1}|/|E_k|^2$  konvergen menuju suatu bilangan, yaitu  $\frac{2}{3}$ , sehingga metode Newton-Raphson memiliki ordo kekonvergenan  $R = 2$ .

---

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 4.6 untuk mencari nilai hampiran akar persamaan dari  $f(x) = x^3 + 3x^2 - 4$  dengan nilai awal  $p_0 = 1.2$  menggunakan metode Newton-Raphson.

---

### Solusi Double Root

Persamaan dengan solusi *double root* adalah suatu persamaan yang memiliki akar ganda, artinya satu persamaan memiliki 2 akar yang bernilai sama. Contohnya, persamaan tak-linear  $x^2 + 2x + 1 = 0$  memiliki akar ganda, yaitu  $x_1 = 1$  dan  $x_2 = 1$ .

**Contoh 4.7 :** Diberikan fungsi  $f(x) = x^3 - 3x + 2$  dengan  $p_0 = 1.2$ . Berikut merupakan langkah-langkah untuk mencari nilai hampiran akar  $f(x) = 0$  yaitu  $P = 1$  menggunakan metode Newton-Raphson serta menunjukkan bahwa metode Newton-Raphson memiliki ordo kekonvergenan  $R = 1$  untuk *double root*.

**Langkah 1:** Penghitungan fungsi turunan dari  $f(x)$ .

Diketahui bahwa  $f(x) = x^3 - 3x + 2$ , sehingga turunan fungsi tersebut adalah

$$f'(x) = 3x^2 - 3$$

**Langkah 2:** Pendefinisian fungsi  $f$  dan turunannya  $f'$  serta nilai awal  $p_0$ .

```
Inp: f(x) = x^3-3*x+2;
      df(x) = 3*x^2-3;
      p0 = 1.2;
```

**Langkah 3:** Penghitungan nilai hampiran akar menggunakan metode Newton-Raphson pada Program 4.1.

```
Inp: pk, flag, M = newtonRaphson(f,df,p0)
@show pk
@show flag
M

Out: pk = 1.000000050926111
flag = 0
23x3 Array{Float64,2}:
 0.0 1.2      NaN
 1.0 1.10303  0.0969697
 2.0 1.05236  0.0506739
 3.0 1.0264   0.0259556
 4.0 1.01326  0.0131431
  :
18.0 1.0      8.12787e-7
19.0 1.0      4.06442e-7
20.0 1.0      2.03214e-7
21.0 1.0      1.01505e-7
22.0 1.0      5.06448e-8
```

**Langkah 4:** Penghitungan nilai galat  $E_k = P - p_k$  dan nilai  $|E_{k+1}|/|E_k|$ .

```
Inp: P = 1;
Ek = P .- M[:,2]
R = 1;
RatioGalat = abs.(Ek[2:end]) ./ abs.(Ek[1:end-1].^R)

Inp: 22-element Array{Float64,1}:
 0.5151515151515141
 0.5081652257444805
 0.5042517320382931
 0.5021714044158508
 0.5010975357376475
  :
0.499982757558765
0.4999060464130258
0.49983004507077766
0.5001628260176164
0.5013849762527124
```

Berdasarkan hasil yang didapatkan, nilai  $|E_{k+1}|/|E_k|$  konvergen menuju suatu bilangan, yaitu  $\frac{1}{2}$ , sehingga metode Newton-Raphson memiliki ordo kekonvergenan  $R = 1$ .

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 4.7 untuk mencari nilai hampiran akar persamaan dari  $f(x) = x^3 + 3x^2 - 4$  dengan nilai awal  $p_0 = -2.4$  menggunakan metode Newton-Raphson.

#### 4.4.2 Metode Secant

Metode *secant* merupakan metode alternatif yang digunakan, apabila metode Newton-Raphson tidak mungkin untuk dicari fungsi turunannya. Metode *secant* memiliki ordo kekonvergenan yaitu  $R = (1 + \sqrt{5})/2 \approx 1.618$  yang hampir mendekati metode Newton-Raphson yaitu  $R = 2$ .

**Contoh 4.8 :** Diberikan fungsi  $f(x) = x^3 - 3x + 2$  dengan  $p_0 = -2.6$  dan  $p_1 = -2.4$ . Berikut merupakan langkah-langkah untuk mencari nilai hampiran akar  $f(x) = 0$  yaitu  $P = -2$  menggunakan metode *secant* serta menunjukkan bahwa metode *secant* memiliki ordo kekonvergenan  $R = (1 + \sqrt{5})/2$  untuk *simple root*.

**Langkah 1:** Pendefinisian fungsi  $f$  serta nilai awal  $p_0$  dan  $p_1$ .

```
Inp: f(x) = x^3-3*x+2;
     p0 = -2.6;
     p1 = -2.4;
```

**Langkah 2:** Penghitungan nilai hampiran akar  $f(x) = 0$  menggunakan metode *secant* pada Program 4.2.

```
Inp: pk, flag, M = secant(f,p0,p1)
      @show pk
      @show flag
      M
```

```
Out: pk = -2.0
     flag = 0
     9x3 Array{Float64,2}:
      0.0  -2.6      NaN
      1.0  -2.4      NaN
      2.0  -2.1066   0.293401
      3.0  -2.02264  0.0839576
      4.0  -2.00151  0.0211303
      5.0  -2.00002  0.00148856
      6.0  -2.0       2.25138e-5
      7.0  -2.0       2.26855e-8
      8.0  -2.0       3.40616e-13
```

**Langkah 3:** Penghitungan nilai galat  $E_k = P - p_k$  dan rasio galat  $|E_{k+1}|/|E_k|^{(1+\sqrt{5})/2}$

```
Inp: P = -2;
     Ek = P .- M[:,2]
Inp: R = (1+sqrt(5))/2;
     RasioGalat = abs.(Ek[2:end]) ./ abs.(Ek[1:end-1].^R)
```

```
Out: 8-element Array{Float64,1}:
      0.9141528310457633
      0.46949776380882363
      0.8472900257102459
      0.6936085986041765
      0.8258227893506624
      0.7497950349957269
      0.7960031113186452
      0.0
```

Berdasarkan hasil yang didapatkan, nilai  $|E_{k+1}|/|E_k|^R$  konvergen menuju suatu bilangan, yaitu 0.8. Jadi, metode Newton-Raphson memiliki ordo kekonvergenan  $R = (1 + \sqrt{5})/2$  untuk *single root*.

**Soal Latihan:** Diberikan  $f(x) = x^3 + 3x^2 - 4$  dengan nilai  $p_0 = 1.6$  dan  $p_1 = 1.4$ . Carilah nilai hampiran akar persamaan  $f(x) = 0$  menggunakan metode *secant* serta tunjukkan bahwa metode *secant* memiliki ordo kekonvergenan  $R = 1$  untuk *double root*.

## 4.5 Latihan-latihan

### 4.5.1 Ulasan Materi

1. Jelaskan yang dimaksud ordo kekonvergenan!
2. Jelaskan apa yang dimaksud laju kekonvergenan linear, kuadratik, dan superlinear!
3. Jelaskan kekurangan dan kelebihan metode Newton-Raphson dan *secant* untuk mencari akar persamaan tak-linear!
4. Jelaskan langkah iterasi metode Newton-Raphson dan *secant* untuk mencari akar persamaan tak-linear!
5. Jelaskan syarat supaya iterasi metode Newton-Raphson dan *secant* menghasilkan solusi yang konvergen menuju akar persamaan!
6. Sebutkan dan jelaskan kriteria-kriteria penghentian iterasi pada metode Newton-Raphson dan *secant*!
7. Manakah di antara pernyataan-pernyataan berikut yang benar?
  - (a) Laju kekonvergenan dari metode regula-falsi adalah linear.
  - (b) Laju kekonvergenan linear lebih cepat konvergen menuju solusi jika dibandingkan dengan laju kekonvergenan superlinear.
  - (c) Metode *secant* memiliki laju kekonvergenan kuadratik.
  - (d) Suatu metode dengan ordo kekonvergenan  $R = 2$  memiliki laju kekonvergenan superlinear.
  - (e) Salah satu kelebihan metode Newton-Raphson adalah bisa terjadinya fenomena *cyclic* atau osilasi.
  - (f) Dalam metode Newton-Raphson, turunan fungsi pada suatu titik digunakan untuk membuat garis singgung, sedangkan dalam metode *secant*, aproksimasi numerik dari turunan berdasarkan dua titik digunakan untuk membuat garis potong.
  - (g) Untuk kasus *simple root*, metode *secant* memiliki  $R \approx 1.618033989$ , sedangkan metode Newton-Raphson memiliki  $R = 2$ .
  - (h) Metode *secant* menggunakan nilai awal  $p_0$  dan  $p_1$ , dimana  $f(p_0)$  dan  $f(p_1)$  tidak harus memiliki tanda yang berlawanan.
  - (i) Metode *secant* memerlukan selang pencarian awal  $[a, b]$  untuk memulai iterasi pencarian akar persamaan tak-linear.

### 4.5.2 Soal Pemrograman

1. Dengan metode **Newton-Raphson** dan ***secant***, carilah akar fungsi dari

$$f(x) = x^4 - 6x^3 + 3x^2 - 2x + \cos(3x)$$

Gunakan  $x_0 = 1$  dan  $x_1 = 0.25$  untuk *secant*.

2. Dengan metode **Newton-Raphson** dan ***secant***, carilah tiga akar fungsi dari

$$f(x) = x^5 - 21x^2 - 8x^3 - 4x^4 - 28x + 60$$

3. Diketahui fungsi

$$f(x) = x + 4 \cos(x)$$

memiliki 3 akar persamaan. Gunakan metode Newton-Raphson dan *secant* untuk mencari dua dari tiga akar tersebut.

4. Diberikan fungsi

$$f(x) = 0.1x^2 - x \ln(x)$$

Cari nilai akar persamaan tersebut menggunakan metode Newton-Raphson dan *secant* hingga memiliki akurasi 4 digit desimal.

5. Diberikan fungsi

$$f(x) = (x - 2)^2 - \ln(x)$$

pada interval  $x \in [1, 2]$

- (a) Gunakan metode Newton-Raphson dan *secant* untuk menghitung nilai akar persamaan hingga memiliki 6 digit desimal akurasi.
- (b) Berapa banyak iterasi yang diperlukan untuk memenuhi akurasi tersebut.

6. Diberikan persamaan tak-linear

$$e^x - 2 - x = 0$$

Persamaan tersebut memiliki dua akar persamaan pada interval  $x \in [-3, 2]$ .

- (a) Gunakan metode Newton-Raphson dan *secant* untuk mencari kedua nilai akar tersebut hingga memiliki akurasi 8 angka desimal.
  - (b) Tunjukkan kemajuan per iterasi (dalam bentuk tabel dan gambar) dari kedua metode untuk menyelesaikan persamaan tak-linear tersebut.
  - (c) Tunjukkan suatu contoh (jika ada) pemberian nilai awal yang membuat metode tersebut konvergen dan yang membuat metode tersebut divergen.
7. Bandingkan penggunaan metode **Newton-Raphson** dan *secant* dalam mencari akar fungsi:
- (a)  $f(x) = x^2 - 2x - 1$ . Gunakan tebakan awal  $x_0 = 2.6$  dan untuk metode *secant* gunakan  $x_1 = 2.5$ .
  - (b)  $f(x) = x^3 - x + 2$ . Gunakan tebakan awal  $x_0 = -1.5$  dan untuk metode *secant* gunakan pula  $x_1 = -1.52$ .

Untuk setiap fungsi, plot grafik (pada satu gambar) fungsi  $f$  dan barisan koordinat  $(x_i, f(x_i))$  untuk  $i = 1, 2, 3, \dots$

8. Diberikan fungsi

$$f(x) = x^5 + 2.5x^4 - 10x^3 - 17.5x^2 + 30x + 18$$

Gunakan metode Newton Raphson untuk mencari hampiran akar  $f(x) = 0$ , yaitu  $P = -3$  dengan langkah-langkah berikut.

- (a) Hitung fungsi turunan dari  $f(x)$ .
  - (b) Definisikan fungsi  $f$  dan turunannya  $f'$  serta nilai awal  $p_0$ .
  - (c) Gunakan Program 4.1 untuk mengeluarkan hasil metode Newton-Raphson.
  - (d) Tunjukkan bahwa metode Newton-Raphson mempunyai ordo kekonvergenan  $R = 1$  pada kasus ini.
9. Diberikan fungsi

$$f(x) = x^5 + 2.5x^4 - 10x^3 - 17.5x^2 + 30x + 18$$

Gunakan metode Newton Raphson dan *secant* untuk mencari hampiran salah satu akar persamaan  $f(x) = 0$ , yaitu  $P = 0.5$  dengan langkah-langkah berikut.

- (a) Hitung fungsi turunan dari  $f(x)$ .
- (b) Definisikan fungsi  $f$  dan turunannya  $f'$  serta nilai awal  $p_0$ .
- (c) Gunakan Program 4.1 untuk mengeluarkan hasil metode Newton-Raphson.

- (d) Tunjukkan bahwa metode Newton-Raphson mempunyai ordo kekonvergenan  $R = 2$  pada kasus ini.
- (e) Definisikan fungsi  $f$  serta nilai awal  $p_0$  dan  $p_1$ .
- (f) Gunakan Program 4.2 untuk mengeluarkan hasil metode *secant*.
- (g) Tunjukkan bahwa metode *secant* mempunyai ordo kekonvergenan  $R \approx 1.618$  pada kasus ini.
10. Carilah dua akar persamaan dari

$$f(x) = 2^x - 2x$$

11. Pada tahun 1225 Leonardo of Pisa mempelajari persamaan

$$x^3 + 2x^2 + 10x - 20 = 0$$

dan menemukan akar yaitu  $x = 1.368\ 808\ 107$ . Tidak ada satupun orang yang mengetahui metode yang digunakan untuk mencari nilai tersebut, tapi itu merupakan hal luar biasa untuk memperoleh 9 digit desimal pada jaman itu. Carilah akar persamaan tersebut !

12. Disediakan persamaan

$$\lambda x = \tan x$$

Dalam ilmu fisika, persamaan di atas menggambarkan keadaan partikel mekanika kuantum dalam kotak persegi panjang dengan dinding berhingga. Persamaan di atas tidak dapat diselesaikan secara analitis dan perlu menggunakan pencarian akar numerik. Carilah hampiran akar persamaan tersebut untuk tiga nilai positif  $\lambda$ , yaitu  $0 < \lambda < 1$ ,  $\lambda = 1$ , dan  $\lambda > 1$ .

13. Persamaan Peng-Robinson didefinisikan sebagai

$$P = \frac{RT}{V-b} - \frac{a}{V(V+b)+b(V-b)}$$

dimana  $P$  = tekanan,  $V$  = volume,  $T$  = suhu, dan  $R$  = konstanta gas ideal. Carilah nilai  $V$  jika diketahui

$$P = 778\ kPa$$

$$T = 350\ K$$

$$a = 365$$

$$b = 0.3$$

$$R = 1.618$$

14. Carilah solusi beberapa nilai akan berikut menggunakan metode pencarian akan numerik

- (a)  $\sqrt{8}$   
 (b)  $-\sqrt{8}$   
 (c)  $\sqrt[3]{7}$   
 (d)  $-\sqrt[3]{7}$

---

---

## BAB 5

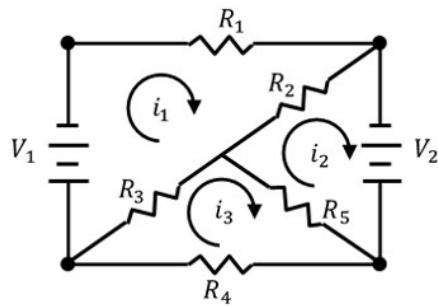
---

### SOLUSI SISTEM PERSAMAAN LINEAR: METODE LANGSUNG

Sistem persamaan linear merupakan bagian dari ilmu aljabar yang sering muncul dalam kehidupan sehari-hari, dari kasus-kasus yang sederhana maupun kasus yang kompleks. Contoh sederhananya, jika massa dua bola merah dan tiga bola hijau adalah 13 kg, serta massa tiga bola merah dan dua bola hijau adalah 12 kg, maka berapakah massa masing-masing bola merah dan hijau. Dalam bentuk matematika, pertanyaan tersebut dapat ditulis sebagai berikut.

$$\begin{aligned}2 \text{ bola merah} + 3 \text{ bola hijau} &= 13\text{kg} \\3 \text{ bola merah} + 2 \text{ bola hijau} &= 12\text{kg}\end{aligned}$$

Masalah lain yang melibatkan sistem persamaan linear adalah hukum Ohm dan Kirchoff dalam bidang fisika. Misalkan, diketahui suatu rangkaian listrik dengan tegangan ( $V$ ) dan hambatan ( $R$ ) seperti gambar berikut.



**Gambar 5.1:** Rangkaian listrik dengan sumber tegangan dan resistor

Dengan menerapkan hukum Ohm dan Kirchoff untuk setiap *loop*, rangkaian listrik pada Gambar 5.1 dapat ditulis dalam sistem persamaan linear berikut.

$$\begin{aligned}(R_1 + R_2 + R_3)i_1 - R_2i_2 - R_3i_3 &= -V_1 \\-R_2i_1 + (R_2 + R_5)i_2 - R_5i_3 &= V_2 \\-R_3i_1 - R_5i_2 + (R_3 + R_4 + R_5)i_3 &= 0\end{aligned}$$

Dalam bentuk perkalian matriks, persamaan di atas dapat ditulis sebagai berikut.

$$\begin{bmatrix} R_1 + R_2 + R_3 & -R_2 & -R_3 \\ -R_2 & R_2 + R_5 & -R_5 \\ -R_3 & -R_5 & R_3 + R_4 + R_5 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix} = \begin{bmatrix} -V_1 \\ V_2 \\ 0 \end{bmatrix}$$

Selain contoh tersebut masih banyak penerapan sistem persamaan linear dalam bidang sains dan teknik. Secara umum, bentuk sistem persamaan linear dapat dituliskan sebagai berikut.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

Dalam notasi matriks, persamaan tersebut sering dituliskan sebagai sistem persamaan differensial dalam bentuk perkalian matriks  $Ax = B$ , yaitu

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

Secara garis besar, metode untuk menyelesaikan sistem persamaan linear dibagi menjadi dua macam, yaitu metode langsung dan metode iteratif. Metode langsung yang dapat digunakan dalam mencari solusi sistem persamaan linear antara lain metode substitusi mundur, eliminasi Gauss dan faktorisasi  $LU$ .

## 5.1 Substitusi Mundur

Algoritma substitusi mundur digunakan untuk mencari solusi dari sistem persamaan linear yang memiliki koefisien berupa matriks segitiga atas. Algoritma ini merupakan modal untuk metode eliminasi Gauss maupun metode faktorisasi  $LU$ .

**Definisi 5.1 :** Suatu matriks  $A_{N \times N} = a_{ij}$  disebut sebagai **matriks segitiga atas** jika elemen  $a_{ij} = 0$  ketika  $i > j$ . Sementara itu, matriks  $A_{N \times N} = a_{ij}$  disebut sebagai **matriks segitiga bawah** jika elemen  $a_{ij} = 0$  ketika  $i < j$ . ■

Jika  $A$  merupakan matriks segitiga atas, maka sistem  $AX = B$  disebut sistem segitiga atas. Dalam notasi matriks, dapat dituliskan sebagai berikut.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22} & a_{23} & \dots & a_{2n} \\ 0 & 0 & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix} \quad (5.1)$$

**Teorema 5.1 Substitusi Mundur:** Misalkan bahwa  $AX = B$  merupakan sistem segitiga atas dengan bentuk seperti persamaan 5.1. Jika

$$a_{kk} \neq 0 \text{ untuk } k = 1, 2, 3, \dots, n. \quad (5.2)$$

maka sistem tersebut memiliki solusi tunggal. ■

Solusi dari sistem segitiga atas dapat dengan mudah untuk dicari. Persamaan terakhir hanya melibatkan  $x_n$ , sehingga langkah pertama substitusi mundur adalah mencari solusi nilai  $x_n$ , yaitu

$$x_n = \frac{b_n}{a_{nn}} \quad (5.3)$$

Sekarang, nilai  $x_n$  telah diketahui dan dapat digunakan untuk menyelesaikan nilai  $x_{n-1}$ , yaitu

$$x_{n-1} = \frac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}} \quad (5.4)$$

Setelah itu, nilai  $x_n$  dan  $x_{n-1}$  telah diketahui dan dapat digunakan untuk menyelesaikan nilai  $x_{n-2}$ , yaitu

$$x_{n-2} = \frac{b_{n-2} - a_{n-2,n-1}x_{n-1} - a_{n-2,n}x_n}{a_{n-2,n-2}} \quad (5.5)$$

Selanjutnya, dari persamaan 5.4 dan 5.5, jika nilai  $x_n, x_{n-1}, \dots, x_{k+1}$  telah diketahui maka nilai  $x_k$  dapat dicari dengan langkah

$$x_k = \frac{b_k - \sum_{j=k+1}^n a_{kj}x_j}{a_{kk}} \quad (5.6)$$

untuk  $k = n-1, n-2, \dots, 1$ .

**Contoh 5.1 :** Gunakan substitusi mundur untuk menyelesaikan sistem berikut.

$$\begin{aligned} 4x_1 - x_2 + 2x_3 + 3x_4 &= 20 \\ -2x_2 + 7x_3 - 4x_4 &= -7 \\ 6x_3 + 5x_4 &= 4 \\ 3x_4 &= 6 \end{aligned}$$

**Solusi :** Selesaikan  $x_4 = 2$  dari persamaan terakhir, yaitu

$$x_4 = \frac{6}{3} = 2$$

Gunakan  $x_4$  untuk persamaan ketiga, sehingga didapatkan

$$x_3 = \frac{4 - 5(2)}{6} = -1$$

Sekarang, gunakan  $x_4 = 2$  dan  $x_3 = -1$  untuk mencari nilai  $x_2$ , yaitu

$$x_2 = \frac{-7 - 7(-1) + 4(2)}{-2} = -4$$

Terakhir,  $x_1$  didapatkan dari persamaan pertama, yaitu

$$x_1 = \frac{20 + 1(-4) - 2(-1) - 3(2)}{4} = 3$$

Jadi, solusi yang didapatkan merupakan solusi tunggal yaitu  $x = (3 \quad -4 \quad -1 \quad 2)^T$ , karena nilai  $a_{kk} \neq 0$  untuk  $k = 1, 2, 3, 4$ . △

## 5.2 Eliminasi Gauss

Eliminasi Gauss merupakan metode untuk mencari solusi dari sistem persamaan linear umum yaitu  $AX = B$ . Tujuan metode ini adalah untuk menyusun sistem segitiga atas yang ekuivalen dengan sistem persamaan awal. Dua sistem persamaan linear dapat dikatakan ekuivalen jika memiliki solusi yang sama. Dalam aljabar linear, terdapat transformasi yang dapat digunakan, namun tidak mengubah solusi sistem awal. Untuk mengefisiensikan penulisan, akan lebih baik jika koefisien dari sistem  $AX = B$  disimpan dalam suatu matriks gandeng berdimensi  $N \times (N + 1)$ . Matriks gandeng dinotasikan dengan  $[A|B]$  dan sistem linear direpresentasikan sebagai berikut.

$$[A|B] = \left[ \begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} & b_n \end{array} \right] \quad (5.7)$$

Sistem  $AX = B$  yang telah direpresentasikan kedalam matriks gandeng dapat diselesaikan dengan melakukan operasi baris, sehingga dihasilkan sistem segitiga atas yang ekuivalen.

**Teorema 5.2 Operasi baris dasar:** Operasi-operasi berikut dapat memnghasilkan sistem persamaan linear yang ekuivalen.

- Operasi  $E_{ij}$  yaitu tukar baris ke- $i$  dengan baris ke- $j$ .
- Operasi  $E_{i(k)}$  yaitu baris ke- $i$  dikali sebesar  $k$ .
- Operasi  $E_{ij(k)}$  yaitu baris ke- $i$  ditambah  $k$  kali baris ke- $j$ .

■

### 5.2.1 Eliminasi Gauss tanpa Pivoting

**Teorema 5.3 Eliminasi Gauss dengan Substitusi Mundur:** Jika  $A_{N \times N}$  adalah matriks non-singular, dan terdapat sistem  $UX = Y$  yang ekuivalen terhadap  $AX = B$  dengan  $U$  adalah matriks segitiga atas dan  $u_{kk} \neq 0$ . Setelah  $U$  dan  $Y$  disusun, substitusi mundur dapat digunakan untuk menyelesaikan sistem  $UX = Y$ . ■

Untuk membentuk sistem segitiga atas yang ekuivalen dengan sistem awal, dapat menggunakan lankah-langkah berikut.

**Langkah 1:** Bentuk matriks gandeng  $[A|B]$  yang berisi koefisien dari sistem persamaan linear awal.

$$\left[ \begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \dots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} & b_n^{(1)} \end{array} \right]$$

Superskrip pada  $a_{ij}^{(1)}$  menujukkan langkah, dalam hal ini menunjukkan bahwa matriks gandeng merupakan hasil dari langkah pertama.

**Langkah 2:** Gunakan operasi  $E_{ij(k)}$  untuk mengubah elemen  $a_{21}, a_{31}, \dots, a_{n1}$  menjadi bernilai nol. Proses diawali dengan mencari nilai  $k$  yaitu  $k_{i1} = -a_{i1}/a_{11}$ . Selanjutnya, gunakan operasi  $E_{i1(k_{i1})}$  untuk mengubah baris ke- $i$ . Dengan demikian, didapatkan hasil langkah ke-2 seperti berikut.

$$\left[ \begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n2}^{(2)} & \dots & a_{nn}^{(2)} & b_n^{(2)} \end{array} \right]$$

**Langkah p+1:** Bentuk umum dari langkah 2 yaitu menggunakan operasi  $E_{ij(k)}$  untuk mengubah elemen  $a_{p+1p}, a_{p+2p}, \dots, a_{np}$  menjadi bernilai nol, dimana  $p = 1, 2, \dots, n-1$ . Proses diawali dengan mencari nilai  $k$  yaitu  $k_{ip} = -a_{ip}/a_{pp}$ , dimana  $i = p+1, \dots, n$ . Selanjutnya, gunakan operasi  $E_{ip(k_{ip})}$  untuk mengubah baris ke- $i$ . Dengan demikian, setelah menghitung hingga langkah ke- $n$  didapatkan hasil langkah ke- $n$  seperti berikut.

$$\left[ \begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & a_{nn}^{(n)} & b_n^{(n)} \end{array} \right]$$

Proses penyusunan sistem segitiga atas telah selesai. Selama  $A$  tak-singular, maka matriks yang dihasilkan juga tak-singular. Hal ini menjadi jaminan bahwa  $a_{kk}^{(k)} \neq 0$  untuk setiap  $k$ , sehingga substitusi mundur dapat digunakan untuk mencari solusi  $X$  dari  $UX = Y$ .

**Contoh 5.2 :** Lakukan eliminasi Gauss untuk membentuk sistem segitiga atas dari sistem persamaan linear berikut, kemudian cari solusi menggunakan substitusi mundur.

$$\begin{bmatrix} -1 & 2 & 4 \\ 1 & -1 & 3 \\ 2 & -2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 9 \\ 3 \end{bmatrix}$$

**Solusi :**

**Langkah 1:** Buatlah matriks gandeng  $[A|B]$  seperti berikut.

$$\left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 1 & -1 & 3 & 9 \\ 2 & -2 & 3 & 3 \end{array} \right]$$

**Langkah 2:** Ubah elemen  $a_{21}$  dan  $a_{31}$  menjadi bernilai nol.

$$\begin{aligned} pivot &\rightarrow \left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 1 & -1 & 3 & 9 \\ 2 & -2 & 3 & 3 \end{array} \right] \xrightarrow{E_{21}(1)} \left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 0 & 1 & 7 & 17 \\ 2 & -2 & 3 & 3 \end{array} \right] \\ k_{21} &= 1 \\ k_{31} &= 2 \end{aligned}$$

**Langkah 3:** Ubah elemen  $a_{32}$  menjadi bernilai nol.

$$pivot \rightarrow k_{32} = -1 \quad \left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 0 & 1 & 7 & 17 \\ 0 & 2 & 11 & 19 \end{array} \right] \xrightarrow{E_{32}(-1)} \left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 0 & 1 & 7 & 17 \\ 0 & 0 & -3 & -15 \end{array} \right]$$

Setelah langkah ke-3, didapatkan sistem segitiga atas yang ekuivalen terhadap sistem awal. Dengan substitusi mundur, solusi sistem persamaan linear yaitu

$$\begin{array}{l|l|l} -3x_3 = -15 & x_2 + 7x_3 = 17 & -x_1 + 2x_2 + 4x_3 = 8 \\ x_3 = 5 & x_2 = -18 & x_1 = -24 \end{array}$$

Jadi, solusi yang didapatkan adalah  $X = (-24 \quad -18 \quad 5)^T$ .  $\triangle$

### 5.2.2 Eliminasi Gauss dengan *Pivoting*

**Definisi 5.2 Pivot:** Bilangan  $a_{rr}$  pada koefisien matriks  $A$  yang digunakan untuk mengeliminasi  $a_{kr}$  dimana  $k = r+1, r+2, \dots, n$  disebut sebagai elemen pivot ke- $r$  dan baris ke- $r$  disebut baris pivot. ■

Pada Contoh 5.2, jika pivot bernilai nol yaitu  $a_{rr} = 0$ , maka baris ke- $r$  tidak dapat digunakan untuk mengeliminasi  $a_{kr}$ . Dengan demikian, baris ke- $r$  harus ditukar posisi dengan baris lain, sehingga pivot bernilai tak-nol. Jika pivot tidak memungkinkan bernilai tak-nol, maka koefisien matriks dari sistem persamaan linear merupakan matriks singular dan sistem tidak memiliki solusi tunggal. Eliminasi Gauss dengan menggunakan operasi tukar baris untuk menentukan pivot sering disebut dengan *pivoting*. Terdapat banyak strategi yang dapat digunakan untuk *pivoting*, salah satu diantaranya adalah strategi *partial pivoting*. Strategi *partial pivoting* sering digunakan untuk menghindari pivot bernilai nol  $a_{pp}^{(p)} = 0$  dan memperkecil galat dari solusi. Untuk mengurangi galat, strategi ini memeriksa nilai dari setiap elemen matriks pada kolom ke- $p$  yang berada pada diagonal utama atau dibawahnya. Pemilihan pivot pada kolom  $k$  didasarkan pada elemen yang memiliki nilai absolut terbesar, yaitu

$$|a_{kp}| = \max \{|a_{pp}|, |a_{p+1p}|, \dots, |a_{np}| \}$$

Selanjutnya, tukar baris ke- $p$  dan baris ke- $k$ .

**Contoh 5.3 :** Lakukan eliminasi Gauss dengan *partial pivoting* untuk membentuk sistem segitiga atas dari sistem persamaan linear berikut, kemudian cari solusi menggunakan substitusi mundur.

$$\left[ \begin{array}{ccc} -1 & 2 & 4 \\ 1 & -1 & 3 \\ 2 & -2 & 3 \end{array} \right] \left[ \begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array} \right] = \left[ \begin{array}{c} 8 \\ 9 \\ 3 \end{array} \right]$$

**Solusi :**

**Langkah 1:** Buatlah matriks gandeng  $[A|B]$  seperti berikut.

$$\left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 1 & -1 & 3 & 9 \\ 2 & -2 & 3 & 3 \end{array} \right]$$

**Langkah 2:** Ubah elemen  $a_{21}$  dan  $a_{31}$  menjadi bernilai nol. Perhatikah bahwa pivot berada pada elemen  $a_{11}$ . Pivot dipilih dengan membandingkan nilai mutlak dari elemen  $a_{11}, a_{21}$ , dan  $a_{31}$ . Dengan demikian, pivot pada kolom ke-1 adalah

$$\begin{aligned}|a_{k1}| &= \max \{|a_{11}|, |a_{21}|, |a_{31}|\} \\&= \max \{|-1|, |1|, |2|\} \\&= 2 = a_{31}\end{aligned}$$

Dengan demikian, pivot terpilih berada pada elemen  $a_{31}$ , sehingga operasi yang digunakan adalah operasi tukar baris  $a_{11}$  dan  $a_{31}$ .

$$\text{pivot} \rightarrow \left[ \begin{array}{ccc|c} -1 & 2 & 4 & 8 \\ 1 & -1 & 3 & 9 \\ 2 & -2 & 3 & 3 \end{array} \right] \xrightarrow{E_{13}} \left[ \begin{array}{ccc|c} 2 & -2 & 3 & 3 \\ 1 & -1 & 3 & 9 \\ -1 & 2 & 4 & 8 \end{array} \right] \xrightarrow[E_{21}(-0.5)]{} \left[ \begin{array}{ccc|c} 2 & -2 & 3 & 3 \\ 0 & 0 & 1.5 & 7.5 \\ 0 & 1 & 5.5 & 9.5 \end{array} \right]$$

**Langkah 3:** Ubah elemen  $a_{32}$  menjadi bernilai nol. Perhatikah bahwa pivot berada pada elemen  $a_{22}$ . Pivot dipilih dengan membandingkan nilai mutlak dari elemen  $a_{22}$  dan  $a_{32}$ . Dengan demikian, pivot pada kolom ke-1 adalah

$$\begin{aligned}|a_{k2}| &= \max \{|a_{22}|, |a_{32}|\} \\&= \max \{|0|, |1|\} \\&= 1 = a_{32}\end{aligned}$$

Dengan demikian, pivot terpilih berada pada elemen  $a_{32}$ , sehingga operasi yang digunakan adalah operasi tukar baris  $a_{22}$  dan  $a_{32}$ .

$$\text{pivot} \rightarrow \left[ \begin{array}{ccc|c} 2 & -2 & 3 & 3 \\ 0 & 0 & 1.5 & 7.5 \\ 0 & 1 & 5.5 & 9.5 \end{array} \right] \xrightarrow{E_{23}} \left[ \begin{array}{ccc|c} 2 & -2 & 3 & 3 \\ 0 & 1 & 5.5 & 9.5 \\ 0 & 0 & 1.5 & 7.5 \end{array} \right]$$

**Langkah terakhir:** Setelah langkah ke-3, didapatkan sistem segitiga atas yang ekuivalen terhadap sistem awal. Dengan substitusi mundur, solusi sistem persamaan linear yaitu

$$\begin{array}{l} 1.5x_3 = 7.5 \\ x_3 = 5 \end{array} \quad \left| \begin{array}{l} x_2 + 5.5x_3 = 9.5 \\ x_2 = -18 \end{array} \right. \quad \left| \begin{array}{l} -x_1 + 2x_2 + 4x_3 = 8 \\ x_1 = -24 \end{array} \right.$$

Jadi, solusi yang didapatkan adalah  $X = (-24 \quad -18 \quad 5)^T$ . △

### 5.3 Faktorisasi LU

Telah diketahui bahwa sistem persamaan linear dengan matriks koefisien segitiga atas ataupun bawah dapat dengan mudah dicari solusinya, yaitu dengan substitusi. Faktorisasi LU akan mengubah matriks  $A$  pada sistem persamaan linear  $AX = B$  menjadi perkalian matriks segitiga bawah  $L$  dan matriks segitiga atas  $U$ . Untuk mempermudah penjelasan, konsep akan diilustrasikan dengan matriks berukuran  $4 \times 4$ , tetapi secara umum dapat diterapkan untuk matriks berukuran  $N \times N$ .

### 5.3.1 Faktorisasi $LU$ tanpa pivoting

**Definisi 5.3 Faktorisasi  $LU$ :** Suatu matriks tak-singular  $A$  memiliki faktor segitiga, jika matriks  $A$  dapat diekspresikan sebagai perkalian dari matriks segitiga bawah  $L$  dan matriks segitiga atas  $U$ .

$$A = LU \quad (5.8)$$

Dalam bentuk matriks, persamaan 5.8 dapat ditulis sebagai

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix} \quad (5.9) \blacksquare$$

Kondisi  $A$  yang merupakan matriks tak-singular akan mengakibatkan  $u_{kk} \neq 0$  untuk setiap  $k$ . Notasi untuk matriks  $L$  merupakan kebalikan dari  $k_{ij}$ , yaitu  $m_{ij} = -k_{ij}$ . Matriks  $L$  dan  $U$  dari persamaan  $A = LU$  dapat dicari dengan langkah-langkah sebagai berikut.

**Langkah 1:** Buatlah matriks gandeng matriks  $A$  dan matriks 0 yaitu  $[A|0]$ . Dalam bentuk matriks dapat ditulis sebagai

$$\left[ \begin{array}{cccc|cccc} a_{11} & a_{12} & a_{13} & a_{14} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 0 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 0 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 & 0 & 0 & 0 \end{array} \right] \quad (5.10)$$

**Langkah 2-N:** Lakukan operasi baris dasar  $E_{ij(k)}$  terhadap Persamaan 5.10 kemudian simpan pengalinya pada matriks 0. Lakukan operasi baris dasar hingga matriks  $A$  menjadi matriks segitiga atas seperti berikut.

$$\left[ \begin{array}{cccc|cccc} u_{11} & u_{12} & u_{13} & u_{14} & 0 & 0 & 0 & 0 \\ 0 & u_{22} & u_{23} & u_{24} & m_{21} & 0 & 0 & 0 \\ 0 & 0 & u_{33} & u_{34} & m_{31} & m_{32} & 0 & 0 \\ 0 & 0 & 0 & u_{44} & m_{41} & m_{42} & m_{43} & 0 \end{array} \right] \quad (5.11)$$

Dengan demikian, matriks  $U$  adalah matriks segitiga atas hasil operasi baris dasar (OBD) dari matriks  $A$ , sedangkan matriks  $L$  adalah catatan pengali ditambah dengan matriks identitas. Dengan demikian,

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 1 \end{bmatrix} \quad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix} \quad (5.12)$$

#### Solusi dari Sistem Persamaan Linear

Misalkan bahwa matriks koefisien  $A$  untuk sistem linear  $AX = B$  memiliki faktor segitiga, maka solusi menjadi

$$LUX = B \quad (5.13)$$

dapat diselesaikan dengan mendefinisikan  $Y = UX$ . Dengan demikian, solusi  $AX = B$  dapat dicari dengan menyelesaikan dua sistem yaitu

1. Solusi  $Y$  dari  $LY = B$ .
2. Solusi  $X$  dari  $UX = Y$ .

Dalam bentuk matriks, pertama akan dicari solusi dari sistem segitiga bawah

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

untuk mendapatkan nilai  $y_1, y_2, y_3, y_4$  yang akan digunakan untuk menyelesaikan sistem segitiga atas

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

Dengan demikian, didapatkan solusi  $x_1, x_2, x_3, x_4$ .

**Contoh 5.4 :** Diberikan sistem persamaan linear seperti berikut

$$\begin{bmatrix} 4 & 2 & 3 & -1 \\ 8 & 3 & 8 & -1 \\ -4 & -4 & 3 & 6 \\ 4 & -3 & 5 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -11 \\ -20 \\ 11 \\ 7 \end{bmatrix}$$

Lakukan faktorisasi  $LU$  terhadap matriks koefisien  $A$ , kemudian carilah solusi sistem tersebut.

**Solusi :**

**Langkah 1:** Buatlah matriks  $[A|0]$ , yaitu

$$\left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 \\ -4 & -4 & 3 & 6 & 0 & 0 & 0 & 0 \\ 4 & -3 & 5 & -4 & 0 & 0 & 0 & 0 \end{array} \right]$$

**Langkah 1:** Gunakan operasi  $E_{ij(k)}$  untuk membuat elemen  $a_{21}, a_{31}$ , dan  $a_{41}$  bernilai nol, kemudian simpan pengali  $m_{ij} = -k_{ij}$  pada matriks 0. Perhatikan bahwa elemen pivot berada di  $a_{11}$ .

$$\begin{aligned} \text{pivot} &\rightarrow \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 \\ -4 & -4 & 3 & 6 & 0 & 0 & 0 & 0 \\ 4 & -3 & 5 & -4 & 0 & 0 & 0 & 0 \end{array} \right] \\ k_{21} &= -2 \\ k_{31} &= 1 \\ k_{41} &= -1 \end{aligned}$$

Perhatikan baris 2. Baris 2 memiliki nilai  $k = -2$ , sehingga operasi yang digunakan adalah  $E_{21(-2)}$  dan simpan nilai pengali yaitu  $m = -k = 2$  pada matriks pengali. Hal serupa juga diterapkan pada baris ke-3 dan ke-4. Dengan demikian, hasil operasi baris dasar pada langkah 2 adalah

$$\begin{array}{c} E_{21(-2)} \\ \hline \xrightarrow{\hspace{1cm}} \\ E_{31(1)} \\ E_{41(-1)} \end{array} \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 1 & 2 & 0 & 0 & 0 \\ 0 & -2 & 6 & 5 & -1 & 0 & 0 & 0 \\ 0 & -5 & 2 & -3 & 1 & 0 & 0 & 0 \end{array} \right]$$

**Langkah 3:** Gunakan operasi  $E_{ij(k)}$  untuk membuat elemen  $a_{32}$  dan  $a_{42}$  bernilai nol, kemudian simpan pengali  $m_{ij} = -k_{ij}$  pada matriks pengali. Perhatikan bahwa elemen pivot berada di  $a_{22}$ .

$$\begin{array}{l} \text{pivot} \rightarrow \\ k_{32} = -2 \\ k_{42} = -5 \end{array} \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 1 & 2 & 0 & 0 & 0 \\ 0 & -2 & 6 & 5 & -1 & 0 & 0 & 0 \\ 0 & -5 & 2 & -3 & 1 & 0 & 0 & 0 \end{array} \right]$$

Perhatikan baris 3. Baris 3 memiliki nilai  $k = -2$ , sehingga operasi yang digunakan adalah  $E_{32(-2)}$  dan simpan nilai pengali yaitu  $m = -k = 2$  pada matriks pengali. Hal serupa juga diterapkan pada baris ke-4. Dengan demikian, hasil operasi baris dasar pada langkah 3 adalah

$$\xrightarrow{\begin{array}{l} E_{32(-2)} \\ E_{42(-5)} \end{array}} \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & -1 & 2 & 0 & 0 \\ 0 & 0 & -8 & -8 & 1 & 5 & 0 & 0 \end{array} \right]$$

**Langkah 4:** Gunakan operasi  $E_{ij(k)}$  untuk membuat elemen  $a_{43}$  bernilai nol, kemudian simpan pengali  $m_{ij} = -k_{ij}$  pada matriks pengali. Perhatikan bahwa elemen pivot berada di  $a_{33}$ .

$$\begin{array}{l} \text{pivot} \rightarrow \\ k_{43} = 4 \end{array} \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & \frac{2}{4} & 3 & -1 & 2 & 0 & 0 \\ 0 & 0 & -8 & -8 & 1 & 5 & 0 & 0 \end{array} \right]$$

Perhatikan baris 4. Baris 4 memiliki nilai  $k = 4$ , sehingga operasi yang digunakan adalah  $E_{43(4)}$  dan simpan nilai pengali yaitu  $m = -k = -4$  pada matriks pengali. Dengan demikian, hasil operasi baris dasar pada langkah 4 adalah

$$\xrightarrow{E_{32(4)}} \left[ \begin{array}{cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & -1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 4 & 1 & 5 & -4 & 0 \end{array} \right]$$

Dengan demikian, didapatkan faktorisasi segitiga dari matriks koefisien  $A$ , yaitu matriks segitiga bawah  $L$  dan matriks segitiga atas  $U$  sebagai berikut.

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 1 & 5 & -4 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 4 & 2 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

**Solusi Sistem Persamaan Linear** Hitung solusi  $Y$  dari sistem  $LY = B$  menggunakan substitusi maju, yaitu

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 1 & 5 & -4 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} -11 \\ -20 \\ 11 \\ 7 \end{bmatrix}$$

Solusi  $Y$  dari sistem tersebut adalah  $Y = [-11 \ 2 \ -4 \ -8]^T$ . Selanjutnya, gunakan nilai  $Y$  tersebut untuk menghitung solusi  $X$  dari sistem  $UX = Y$  seperti berikut.

$$\begin{bmatrix} 4 & 2 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -11 \\ 2 \\ -4 \\ -8 \end{bmatrix}$$

Dengan demikian, didapatkan solusi persamaan linear  $X = [-3 \ -2 \ 1 \ -2]^T$ .  $\triangle$

### 5.3.2 Faktorisasi LU dengan pivoting

Sama halnya pada eliminasi Gauss, jika nilai pivot bernilai nol, maka pivot tersebut tidak dapat mengeliminasi elemen lain menjadi bernilai nol. Dengan demikian, diperlukan strategi pivoting untuk membuat nilai pivot bernilai tak-nol.

**Teorema 5.4 Matriks Permutasi:** Matriks  $P_{N \times N}$  disebut matriks permutasi, apabila matriks  $P$  memiliki tepat satu entri bernilai 1 pada setiap baris dan kolom, serta selainnya bernilai 0.  $\blacksquare$

Sebagai contoh, berikut merupakan contoh matriks permutasi berukuran  $N \times N$ .

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

**Teorema 5.5 Faktorisasi LUP:** Jika  $A$  merupakan matriks tak-singular, maka terdapat matriks permutasi  $P$  sedemikian sehingga  $PA$  memiliki faktor segitiga

$$PA = LU \quad (5.14) \quad \blacksquare$$

Matriks  $P, L$ , dan  $U$  dari persamaan 5.14 dapat dicari menggunakan langkah-langkah seperti berikut.

**Langkah 1:** Buatlah matriks gandeng matriks  $A$ , matriks 0, dan matriks identitas  $I$  yaitu  $[A|0|I]$ . Dalam bentuk matriks dapat ditulis sebagai

$$\left[ \begin{array}{cccc|cccc|cccc} a_{11} & a_{12} & a_{13} & a_{14} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \quad (5.15)$$

**Langkah 2-N:** Lakukan operasi baris dasar pindah baris yaitu  $E_{ij}$  untuk memilih pivot (gunakan strategi *partial pivoting*) diikuti dengan operasi  $E_{ij(k)}$  terhadap Persamaan 5.15 kemudian simpan pengalinya pada matriks 0. Lakukan operasi baris dasar hingga matriks  $A$  menjadi matriks segitiga atas seperti berikut.

$$\left[ \begin{array}{cccc|cccc|c} u_{11} & u_{12} & u_{13} & u_{14} & 0 & 0 & 0 & 0 & P \\ 0 & u_{22} & u_{23} & u_{24} & m_{21} & 0 & 0 & 0 & \\ 0 & 0 & u_{33} & u_{34} & m_{31} & m_{32} & 0 & 0 & \\ 0 & 0 & 0 & u_{44} & m_{41} & m_{42} & m_{43} & 0 & \end{array} \right] \quad (5.16)$$

Dengan demikian, matriks  $U$  adalah matriks segitiga atas hasil operasi baris dasar (OBD) matriks  $A$ . Sementara itu, Matriks  $L$  adalah catatan pengali ditambah dengan matriks identitas sehingga

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 0 \end{bmatrix} \quad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix} \quad (5.17)$$

### Solusi dari Sistem Persamaan Linear

Misalkan  $P$  merupakan matriks permutasi dan  $PA$  memiliki faktor segitiga sedemikian sehingga  $PA = LU$ , maka solusi sistem persamaan linear  $AX = B$  dapat dicari dengan empat langkah berikut.

1. Susun matriks  $L, U$ , dan  $P$ .
2. Hitung vektor kolom  $PB$ .
3. Cari solusi  $Y$  dari  $LY = PB$  menggunakan substitusi maju.
4. Cari solusi  $X$  dari  $UX = Y$  menggunakan substitusi mundur.

**Contoh 5.5 :** Diberikan sistem persamaan linear seperti berikut

$$\begin{bmatrix} 4 & 2 & 3 & -1 \\ 8 & 3 & 8 & -1 \\ -4 & -4 & 3 & 6 \\ 4 & -3 & 5 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -11 \\ -20 \\ 11 \\ 7 \end{bmatrix}$$

Lakukan faktorisasi  $LU$  dengan *partial pivoting* untuk menyusun matriks  $L, U$ , dan  $P$  dari matriks koefisien  $A$ , kemudian carilah solusi sistem tersebut.

**Solusi :**

**Langkah 1:** Buatlah matriks  $[A|0|I]$ , yaitu

$$\left[ \begin{array}{cccc|cccc|cccc} 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -4 & -4 & 3 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 4 & -3 & 5 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right]$$

**Langkah 2:** Perhatikan bahwa pivot kolom ke-1 berada pada elemen  $a_{11}$ . Pivot dipilih dengan membandingkan nilai mutlak dari elemen  $a_{11}, a_{21}, a_{31}$ , dan  $a_{41}$ . Dengan demikian, pivot pada kolom ke-1 adalah

$$\begin{aligned} |a_{k1}| &= \max \{|a_{11}|, |a_{21}|, |a_{31}|, |a_{41}|\} \\ &= \max \{|4|, |8|, |-4|, |4|\} \\ &= 8 = a_{21} \end{aligned}$$

Jadi, pivot terpilih berada pada elemen  $a_{21}$ , akibatnya operasi yang digunakan adalah

operasi tukar baris  $a_{11}$  dan  $a_{21}$ .

$$\begin{array}{l} \text{pivot} \rightarrow \\ \left[ \begin{array}{cccc|ccccc} \frac{4}{4} & 2 & 3 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -4 & -4 & 3 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 4 & -3 & 5 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \\ \xrightarrow{E_{12}} \left[ \begin{array}{cccc|ccccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 4 & 2 & 3 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -4 & -4 & 3 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 4 & -3 & 5 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \end{array}$$

Selanjutnya, eliminasi elemen  $a_{21}, a_{31}$ , dan  $a_{41}$  seperti berikut.

$$\begin{array}{l} E_{21}(-0.5) \\ \xrightarrow{} \left[ \begin{array}{cccc|ccccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0.5 & -1 & -0.5 & 0.5 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right] \\ E_{31}(0.5) \\ E_{41}(-0.5) \end{array} \left[ \begin{array}{cccc|ccccc} 0 & -2.5 & 7 & 5.5 & -0.5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right]$$

**Langkah 3:** Perhatikan bahwa pivot kolom ke-2 berada pada elemen  $a_{22}$ . Pivot dipilih dengan membandingkan nilai mutlak dari elemen  $a_{22}, a_{32}$ , dan  $a_{42}$ . Dengan demikian, pivot pada kolom ke-2 adalah

$$\begin{aligned} |a_{k1}| &= \max \{|a_{22}|, |a_{32}|, |a_{42}|\} \\ &= \max \{|0.5|, |-2.5|, |-4.5|\} \\ &= 4.5 = a_{42} \end{aligned}$$

Jadi, pivot terpilih berada pada elemen  $a_{42}$ , akibatnya operasi yang digunakan adalah operasi tukar baris  $a_{22}$  dan  $a_{42}$ .

$$\begin{array}{l} \text{pivot} \rightarrow \\ \left[ \begin{array}{cccc|ccccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \underline{0.5} & -1 & -0.5 & 0.5 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -2.5 & 7 & 5.5 & -0.5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \\ \xrightarrow{E_{24}} \left[ \begin{array}{cccc|ccccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -2.5 & 7 & 5.5 & -0.5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0.5 & -1 & -0.5 & 0.5 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right] \end{array}$$

Selanjutnya, eliminasi elemen  $a_{32}$ , dan  $a_{42}$  seperti berikut.

$$\begin{array}{l} E_{32}(-0.56) \\ \xrightarrow{} \left[ \begin{array}{cccc|ccccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 6.44 & 7.44 & -0.5 & 0.56 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -0.89 & -0.89 & 0.5 & -0.11 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right] \\ E_{42}(0.11) \end{array}$$

**Langkah 4:** Perhatikan bahwa pivot kolom ke-3 berada pada elemen  $a_{33}$ . Pivot dipilih dengan membandingkan nilai mutlak dari elemen  $a_{33}, a_{43}$ . Dengan demikian, pivot pada kolom ke-3 adalah

$$\begin{aligned} |a_{k3}| &= \max \{|a_{33}|, |a_{43}|\} \\ &= \max \{|6.44|, |-0.89|\} \\ &= 6.44 = a_{33} \end{aligned}$$

Jadi, pivot terpilih berada pada elemen  $a_{33}$ , akibatnya tidak memerlukan operasi pindah baris.

$$pivot \rightarrow \left[ \begin{array}{cccc|cccc|cccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 6.44 & 7.44 & -0.5 & 0.56 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -0.89 & -0.89 & 0.5 & -0.11 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right]$$

Selanjutnya, eliminasi elemen  $a_{43}$  seperti berikut.

$$\xrightarrow{E_{43}(0.14)} \left[ \begin{array}{cccc|cccc|cccc} 8 & 3 & 8 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -4.5 & 1 & -3.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 6.44 & 7.44 & -0.5 & 0.56 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.14 & 0.5 & -0.11 & -0.14 & 0 & 1 & 0 & 0 & 0 \end{array} \right]$$

Dengan demikian, diperoleh matriks  $L, U$ , dan  $P$  yaitu

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ -0.5 & 0.56 & 1 & 0 \\ 0.5 & -0.11 & -0.14 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 8 & 3 & 8 & -1 \\ 0 & -4.5 & 1 & -3.5 \\ 0 & 0 & 6.44 & 7.44 \\ 0 & 0 & 0 & 0.14 \end{bmatrix} \quad P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

### Solusi dari Sistem Persamaan Linear

Setelah didapatkan matriks  $L, U$ , dan  $P$ , hitung vektor kolom  $PB$  yaitu

$$PB = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -11 \\ -20 \\ 11 \\ 7 \end{bmatrix} = \begin{bmatrix} -20 \\ 7 \\ 11 \\ -11 \end{bmatrix}$$

Selanjutnya, Hitung solusi  $Y$  dari sistem  $LY = PB$  seperti berikut.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ -0.5 & 0.56 & 1 & 0 \\ 0.5 & -0.11 & -0.14 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} -20 \\ 7 \\ 11 \\ -11 \end{bmatrix}$$

Dengan substitusi maju, solusi sistem  $LY = PB$  adalah  $Y = [-20 \ 17 \ -8.44 \ -0.28]^T$ . Terakhir, gunakan solusi  $Y$  untuk menghitung solusi  $X$  dari sistem  $UX = Y$  dengan substitusi mundur seperti berikut.

$$\begin{bmatrix} 8 & 3 & 8 & -1 \\ 0 & -4.5 & 1 & -3.5 \\ 0 & 0 & 6.44 & 7.44 \\ 0 & 0 & 0 & 0.14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -20 \\ 17 \\ -8.44 \\ -0.28 \end{bmatrix}$$

Jadi, solusi dari sistem persamaan linear adalah  $X = [-3 \ -2 \ 1 \ -2]^T$ .  $\triangle$

## 5.4 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Substitusi Mundur** berisi program untuk mencari solusi dari sistem persamaan linear dengan secara langsung. Program ini secara *default* berisi 2 masukan, yaitu matriks SPL A, dan B, serta 1 luaran, yaitu solusi sistem persamaan linear. Berikut merupakan program metode substitusi mundur.

**Algoritma 5.1:** Metode substitusi mundur

```
function backsub(A,B)
    # Hitung ukuran matriks dan inisiasi solusi X
    n = length(B);
    X = zeros(n,1);
    # Hitung nilai solusi X ke-n
    X[n] = B[n]/A[n,n];
    # Hitung nilai solusi X ke- n-1 sampai X ke-1
    for i = n-1:-1:1
        X[i] = (B[i] - A[i,i+1:n]'X[i+1:n])/A[i,i];
    end
    return X
end
```

**Metode Eliminasi Gauss** berisi program untuk mencari solusi sistem persamaan linear dengan secara langsung. Metode eliminasi Gauss terbagi menjadi 2 jenis yaitu tanpa *pivoting* dan dengan *pivoting*. Program ini secara *default* berisi 2 masukan, yaitu matriks SPL A, dan B, serta 1 luaran, yaitu solusi sistem persamaan linear. Berikut merupakan program metode eliminasi Gauss tanpa dan dengan *pivoting*.

**Algoritma 5.2:** Metode eliminasi Gauss tanpa *pivoting*

```
# Metode Eliminasi Gauss Tanpa Pivoting
function elimGaussNonPivoting(A,b)
    # Hitung ukuran matriks dan inisiasi solusi X
    n = length(b);
    X = zeros(n,1);
    # Buatlah matriks Gandeng [A|b]
    Aug = [A b];
    # Lakukan operasi baris dasar terhadap matriks gandeng
    for p = 1:n-1 # p menunjukkan kolom
        # Jika pivot bernilai nol, maka gagal.
        if Aug[p,p]==0
            error("Harus Pakai Pivoting")
        end
        # Lakukan eliminasi menggunakan operasi Eij(k)
        for i = p+1:n # i menunjukkan baris
            k = Aug[i,p]/Aug[p,p];
            Aug[i,:] = Aug[i,:] - k*Aug[p,:];
        end
    end
    # Pisahkan matriks gandeng, kemudian substitusi mundur.
```

```

A = Aug[:,1:n];
b = Aug[:,1+n];
X = backsub(A,b);
end

```

**Algoritma 5.3:** Metode eliminasi Gauss dengan *pivoting*

```

# Metode Eliminasi Gauss Dengan Pivoting
function elimGaussWithPivoting(A,b)
    # Hitung ukuran matriks dan inisiasi solusi X
    n = length(b);
    X = zeros(n,1);
    # Buatlah matriks Gandeng [A|b]
    Aug = [A b];
    # Lakukan operasi baris dasar terhadap matriks gandeng
    for p = 1:n-1
        # Lakukan operasi pindah baris untuk menentukan nilai pivot
        val,j = findmax(abs.(Aug[p:n,p]));
        C = Aug[p,:];
        Aug[p,:] = Aug[j+p-1,:];
        Aug[j+p-1,:] = C;
        # Jika pivot bernilai nol, maka gagal.
        if Aug[p,p] == 0;
            error("Matriks tidak punya solusi tunggal")
        end
        # Lakukan eliminasi menggunakan operasi Eij(k)
        for i = p+1:n
            k = Aug[i,p]/Aug[p,p];
            Aug[i,:] = Aug[i,:] - k*Aug[p,:];
        end
    end
    # Pisahkan matriks gandeng, kemudian substitusi mundur.
    A = Aug[:,1:n];
    b = Aug[:,1+n];
    X = backsub(A,b);
end

```

**Metode Faktorisasi LU** berisi program untuk mencari solusi sistem persamaan linear dengan secara langsung. Metode faktorisasi LU terbagi menjadi 2 jenis, yaitu faktorisasi LU tanpa *pivoting* dan faktorisasi LU dengan *pivoting*. Program faktorisasi LU tanpa *pivoting* secara *default* berisi 2 masukan, yaitu matriks SPL *A*, dan *B*, serta 2 luaran, yaitu matriks *L* dan *U*. Sementara itu, program faktorisasi LU dengan *pivoting* secara *default* berisi 2 masukan, yaitu matriks SPL *A*, dan *B*, serta 3 luaran, yaitu matriks *L*, *U*, dan *P*. Berikut merupakan program metode faktorisasi LU tanpa dan faktorisasi LU dengan *pivoting*.

**Algoritma 5.4:** Metode faktorisasi LU tanpa *pivoting*

```

# Metode Faktorisasi LU Tanpa Pivoting
function LUtanpaP(A);
    # Definisikan matriks L sebagai pencatat pengali.
    n,n = size(A);
    L = zeros(n,n);
    # Lakukan operasi baris dasar terhadap matriks
    Aug = copy(A);
    for p = 1:n-1

```

```

# Jika pivot bernilai nol, maka gagal.
if Aug[p,p]==0
    error("Pivot bernilai nol");
end
# Lakukan eliminasi lalu simpan pengali pada matriks L.
for i = p+1:n
    k = Aug[i,p]/Aug[p,p];
    Aug[i,1:n] = Aug[i,1:n] - k*Aug[p,1:n];
    L[i,p] = k;
end
U = Aug;
L = L .+ I(n);
return L, U
end

```

### Algoritma 5.5: Metode faktorisasi LU dengan pivoting

```

# Metode Faktorisasi LU Dengan Pivoting
function LUdenganP(A)
    # Definisikan matriks L dan P.
    n,n = size(A);
    L = zeros(n,n);
    P = Array(I(n));
    # Lakukan operasi baris dasar terhadap matriks gandeng
    Aug = copy(A)
    for p = 1:n-1
        # Lakukan operasi pindah baris untuk menentukan nilai pivot
        val,j = findmax(abs.(Aug[p:n,p]));
        # Pivoting U
        C = Aug[p,:];
        Aug[p,:] = Aug[j+p-1,:];
        Aug[j+p-1,:] = C;
        # Pivoting L
        C = L[p,:];
        L[p,:] = L[j+p-1,:];
        L[j+p-1,:] = C;
        # Pivoting P
        C = P[p,:];
        P[p,:] = P[j+p-1,:];
        P[j+p-1,:] = C;
        # Jika pivot bernilai nol, maka gagal.
        if Aug[p,p]==0
            error("Matriks Singular")
        end
        # Lakukan eliminasi dan simpan pengali.
        for i = p+1:n
            k = Aug[i,p]/Aug[p,p]
            Aug[i,1:n] = Aug[i,1:n] - k*Aug[p,1:n];
            L[i,p] = k;
        end
    end
    U = Aug;
    L = L .+ I(n);
    return L,U,P
end

```

### 5.4.1 Metode Substitusi Mundur

Metode dasar yang digunakan untuk mencari solusi dari sistem persamaan linear adalah metode substitusi. Metode substitusi ini membutuhkan persyaratan yaitu matriks  $A$  pada sistem  $AX = B$  merupakan matriks segitiga. Jika  $A$  merupakan matriks segitiga atas, maka metode substitusi yang digunakan adalah substitusi mundur. Contoh berikut ini menunjukkan penyelesaian dari sistem segitiga atas dengan metode substitusi mundur. Adapun untuk substitusi maju, dapat dipelajari sendiri dengan analogi yang sama dengan substitusi mundur.

**Contoh 5.6 :** Diberikan sistem segitiga atas seperti berikut.

$$\begin{aligned} 4x_1 - x_2 + 2x_3 + 3x_4 &= 20 \\ -2x_2 + 7x_3 - 4x_4 &= -7 \\ 6x_3 + 5x_4 &= 4 \\ 3x_4 &= 6 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menyelesaikan sistem persamaan linear tersebut menggunakan metode substitusi mundur.

**Langkah 1:** Pendefinisian matriks  $A$  dan  $B$ .

```
Inp: A = [ 4 -1 2 3
           0 -2 7 -4
           0 0 6 5
           0 0 0 3];
      B = [20; -7; 4; 6];
```

**Langkah 2:** Penghitungan solusi sistem persamaan linear menggunakan metode substitusi mundur pada Program 5.1.

```
Inp: X = backsud(A, B)
```

```
Out: 4x1 Array{Float64,2}:
      3.0
      -4.0
      -1.0
      2.0
```

Dengan demikian, solusi dari SPL di atas adalah  $X = (3 \ -4 \ -1 \ 2)^T$ . △

### 5.4.2 Metode Eliminasi Gauss

Metode eliminasi Gauss berangkat dari metode substitusi mundur. Maksudnya adalah sistem persamaan linear dengan matriks  $A$  bukan matriks segitiga atas diubah terlebih dahulu menjadi matriks segitiga atas dengan eliminasi. Selanjutnya, matriks hasil eliminasi tersebut diselesaikan menggunakan substitusi mundur.

**Contoh 5.7 :** Diberikan sistem persamaan linear dengan 4 persamaan seperti berikut.

$$\begin{pmatrix} 2 & 5 & 4 & 2 & 3 \\ 4 & 1 & 0.4 & 6 & 5 \\ 0.9 & 4 & 8 & 7 & 10 \\ 5 & 7 & 0.5 & 2 & 7 \\ 2 & 6 & 8 & 9 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 26 \\ 24.4 \\ 32.8 \\ 30.5 \\ 53 \end{pmatrix}$$

Berikut merupakan langkah-langkah untuk menghitung solusi penyelesaikan sistem persamaan tersebut menggunakan metode eliminasi Gauss tanpa *pivoting* dan eliminasi Gauss dengan *pivoting* serta menunjukkan bahwa eliminasi Gauss dengan *pivoting* memiliki galat yang lebih kecil daripada eliminasi Gauss tanpa *pivoting*, jika diketahui solusi eksak persamaan, yaitu  $X^* = (2 \ 3 \ 1 \ 3 \ -1)^T$ .

**Langkah 1:** Pendefinisian matriks A dan B.

```
Inp: A = [ 2      5      4      2      3
          4      1      0.4     6      5
          0.9    4      8      7      10
          5      7      0.5     2      7
          2      6      8      9      4];
B = [26; 24.4; 32.8; 30.5; 53];
```

**Langkah 2:** Penghitungan solusi sistem persamaan linear menggunakan eliminasi Gauss tanpa *pivoting* pada Program 5.2 dan eliminasi Gauss dengan *pivoting* pada Program 5.3.

```
Inp: Xn = elimGaussNonPivoting(A,B)
```

```
Out: 5x1 Array{Float64,2}:
      2.0000000000000018
      2.999999999999999
      1.0000000000000013
      2.999999999999999
      -1.0000000000000009
```

```
Inp: Xn = elimGaussNonPivoting(A,B)
```

```
Out: 5x1 Array{Float64,2}:
      1.999999999999993
      3.000000000000001
      1.0
      3.0
      -1.000000000000007
```

**Langkah 2:** Penghitungan panjang vektor galat dari masing-masing solusi.

```
Inp: using LinearAlgebra
Inp: Xeksak = [2;3;1;3;-1];
      En = norm(Xn-Xeksak)
```

```
Out: 2.7012892057857038e-15
```

```
Inp: Ep = norm(Xp-Xeksak)
```

```
Out: 1.2947314098277875e-15
```

Berdasarkan hasil yang didapatkan, nilai galat metode eliminasi Gauss dengan *pivoting* lebih kecil daripada metode eliminasi Gauss tanpa *pivoting*. Hal ini ditunjukkan dari nilai  $E_p < E_n$ .

**Soal Latihan:** Diberikan SPL seperti berikut.

$$\begin{pmatrix} 4.0 & 2.0 & 1.0 & 3.0 & 9.0 & 1.0 \\ 5.0 & 3.0 & 9.0 & 3.0 & 1.0 & 2.0 \\ 10.0 & 7.0 & 2.0 & 1.0 & 8.0 & 3.0 \\ 9.0 & 0.3 & 8.0 & 4.0 & 8.0 & 5.0 \\ 8.0 & 0.7 & 0.7 & 2.0 & 7.0 & 1.0 \\ 2.0 & 6.0 & 10.0 & 2.0 & 6.0 & 5.0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} 17.0 \\ 42.0 \\ 30.0 \\ 58.4 \\ 33.7 \\ 23.0 \end{pmatrix}$$

Gunakan metode eliminasi Gauss tanpa *pivoting* dan eliminasi Gauss dengan *pivoting* untuk mencari penyelesaikan sistem persamaan tersebut, kemudian tunjukkan bahwa metode eliminasi Gauss dengan *pivoting* memiliki galat lebih kecil dibandingkan metode eliminasi Gauss tanpa *pivoting*.

### 5.4.3 Metode Faktorisasi LU

Metode faktorisasi *LU* tanpa *pivoting* mendefinisikan matriks  $A$  pada sistem persamaan linear sebagai  $A = LU$ , sedangkan metode faktorisasi *LU* dengan *pivoting* mendefinisikan matriks  $A$  pada sistem persamaan linear sebagai  $PA = LU$ . Solusi sistem persamaan linear dapat dicari dengan substitusi maju dan mundur dari matriks  $P$ ,  $L$ , dan  $U$ .

**Contoh 5.8 :** Diberikan sistem persamaan linear dengan 4 persamaan seperti berikut.

$$\begin{pmatrix} 2 & 5 & 4 & 2 & 3 \\ 4 & 1 & 0.4 & 6 & 5 \\ 0.9 & 4 & 8 & 7 & 10 \\ 5 & 7 & 0.5 & 2 & 7 \\ 2 & 6 & 8 & 9 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 26 \\ 24.4 \\ 32.8 \\ 30.5 \\ 53 \end{pmatrix}$$

Berikut merupakan langkah-langkah untuk menghitung solusi penyelesaikan sistem persamaan tersebut menggunakan metode faktorisasi *LU* tanpa *pivoting* dan faktorisasi *LU* dengan *pivoting* serta menunjukkan bahwa faktorisasi *LU* dengan *pivoting* memiliki galat yang lebih kecil daripada faktorisasi *LU* tanpa *pivoting*, jika diketahui solusi eksak persamaan, yaitu  $X^* = (2 \ 3 \ 1 \ 3 \ -1)^T$ .

**Langkah 1:** Pendefinisian matriks  $A$  dan  $B$ .

```
A = [ 2      5      4      2      3
      4      1      0.4    6      5
      0.9    4      8      7     10
      5      7      0.5    2      7
      2      6      8      9      4];
B = [26; 24.4; 32.8; 30.5; 53];
```

**Langkah 2:** Penghitungan solusi sistem persamaan linear menggunakan faktorisasi *LU* tanpa *pivoting* pada Program 5.4 dan faktorisasi *LU* dengan *pivoting* pada Program 5.5.

Untuk mencari solusi dari SPL  $Ax = B$  menggunakan faktorisasi *LU* tanpa *pivoting*. Perhatikan bahwa  $A = LU$ , sehingga  $Ax = B \Leftrightarrow LUx = B$ . Misalkan bahwa  $Ux = Y$ , maka  $LY = B$ . Selesaikan  $LY = B$  dengan substitusi maju, maka solusi  $x$  dapat dicari dari  $Ux = Y$  dengan substitusi mundur.

```
Inp: L,U = LUtanpaP(A)
Inp: Y = forwardsub(L,B)
Xn = backsub(U,Y)
```

```
Inp: 5x1 Array{Float64,2}:
2.000000000000053
2.99999999999997
1.000000000000036
2.999999999999998
-1.000000000000009
```

Untuk mencari solusi dari SPL  $Ax = B$  menggunakan faktorisasi LU dengan *pivoting*. Perhatikan bahwa  $PA = LU$ , sehingga  $Ax = B \Leftrightarrow PAx = PB \Leftrightarrow LUx = PB$ . Misalkan bahwa  $Ux = Y$ , maka  $LY = PB$ . Selesaikan  $LY = PB$  dengan substitusi maju, maka solusi  $x$  dapat dicari dari  $Ux = Y$  dengan substitusi mundur.

```
Inp: L,U,P = LUDenganP(A)
Inp: Y = forwardsub(L,P*B);
      Xp = backsub(U,Y)
```

```
Inp: 5x1 Array{Float64,2}:
      1.9999999999999993
      3.0000000000000001
      1.0000000000000004
      3.0
      -1.0000000000000007
```

**Langkah 2:** Penghitungan panjang vektor galat dari masing-masing solusi.

```
Inp: using LinearAlgebra
Inp: Xeksak = [2;3;1;3;-1];
      norm(Xn-Xeksak)
```

```
Out: 2.7012892057857038e-15
```

```
Inp: norm(Xp-Xeksak)
```

```
Out: 1.368774871883577e-15
```

Berdasarkan hasil yang didapatkan, nilai galat metode faktorisasi LU dengan *pivoting* lebih kecil daripada metode faktorisasi LU tanpa *pivoting*. Hal ini ditunjukkan dari nilai  $E_p < E_n$ .

**Soal Latihan:** Diberikan sistem persamaan linear seperti berikut.

$$\begin{pmatrix} 4.0 & 2.0 & 1.0 & 3.0 & 9.0 & 1.0 \\ 5.0 & 3.0 & 9.0 & 3.0 & 1.0 & 2.0 \\ 10.0 & 7.0 & 2.0 & 1.0 & 8.0 & 3.0 \\ 9.0 & 0.3 & 8.0 & 4.0 & 8.0 & 5.0 \\ 8.0 & 0.7 & 0.7 & 2.0 & 7.0 & 1.0 \\ 2.0 & 6.0 & 10.0 & 2.0 & 6.0 & 5.0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} 17.0 \\ 42.0 \\ 30.0 \\ 58.4 \\ 33.7 \\ 23.0 \end{pmatrix}$$

Gunakan metode faktorisasi LU tanpa *pivoting* dan faktorisasi LU dengan *pivoting* untuk mencari penyelesaikan sistem persamaan tersebut, kemudian tunjukkan bahwa metode faktorisasi LU dengan *pivoting* memiliki galat lebih kecil dibandingkan metode faktorisasi LU tanpa *pivoting*.

## 5.5 Latihan-latihan

### 5.5.1 Ulasan Materi

1. Jelaskan bagaimana algoritma substitusi mundur dan maju menyelesaikan suatu sistem persamaan linear! Apakah ada kriteria sistem persamaan linear yang diperlukan oleh substitusi mundur dan maju? Sebutkan!

2. Jelaskan fungsi dan tujuan penggunaan teknik *pivoting* pada eliminasi gauss dan faktorisasi LU!
3. Apakah sistem persamaan linear  $Ax = b$  dengan  $A$  merupakan matriks singular dapat diselesaikan (terdapat solusi  $x$ )? Jelaskan!
4. Apakah algoritma penyelesaian SPL dengan metode langsung memberikan hasil yang tepat (tanpa *error*) jika dikerjakan menggunakan pendekatan numerik? Mengapa demikian?
5. Apakah algoritma penyelesaian SPL dengan metode langsung memberikan hasil yang tepat (tanpa *error*) jika dikerjakan menggunakan cara manual (tulis tangan)? Mengapa demikian?
6. Jika terdapat *error* pada algoritma penyelesaian SPL dengan metode langsung, apakah *error* tersebut dapat dikendalikan dan diatur toleransinya?

### 5.5.2 Soal Pemrograman

1. Diberikan sistem persamaan linear (SPL) berikut:

$$\begin{aligned} 4x_1 + 8x_2 + 4x_3 &= 8 \\ x_1 + 5x_2 + 4x_3 - 3x_4 &= -4 \\ x_1 + 4x_2 + 7x_3 + 2x_4 &= 10 \\ x_1 + 3x_2 - 2x_4 &= -4 \end{aligned}$$

- (a) Selesaikan SPL  $AX = B$  menggunakan eliminasi Gauss tanpa *pivoting*.  
 (b) Selesaikan SPL  $AX = B$  menggunakan eliminasi Gauss dengan *pivoting*.
2. Pembentukan matriks acak
  - Bangkitkan matriks bilangan acak  $A = (a_{ij})_{(n \times n)} \in Z$  pada selang  $[-n, n]$ .
  - Tetapkan matriks  $B = A \bullet (1)_{(n \times 1)}$  sehingga solusi dari SPL  $AX = B$  adalah  $X^* = (1)_{(n \times 1)}$
  - Cari solusi numerik  $X_1$  menggunakan eliminasi Gauss tanpa *pivoting*.
  - Cari solusi numerik  $X_2$  menggunakan eliminasi Gauss dengan *pivoting*.
  - Hitung panjang vektor galat, yaitu solusi numerik  $X_1$  dan  $X_2$  terhadap solusi eksak  $X^*$
 Lakukan beberapa tahapan tersebut untuk:
  - (a)  $n = 20$
  - (b)  $n = 50$
  - (c)  $n = 100$
  - (d)  $n = 500$
3. Diberikan matriks tridiagonal sebagai berikut:

$$\begin{bmatrix} -4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

- (a) Gunakan metode eliminasi Gauss untuk mencari solusi SPL di atas.
- (b) Apa yang terjadi terhadap nilai solusi jika dimensi SPL  $AX = B$  diubah menjadi  $A_{10 \times 10}$ ,  $A_{25 \times 25}$ , dan  $A_{50 \times 50}$ .

4. Diberikan suatu sistem persamaan linear (SPL) sebagai berikut:

$$\begin{bmatrix} 12 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ -2 & 12 & -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 12 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 12 & -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & -2 & 12 & -2 & 1 & 0 \\ 0 & 0 & 0 & 1 & -2 & 12 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -2 & 12 & -2 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 12 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{pmatrix} = \begin{pmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \end{pmatrix}$$

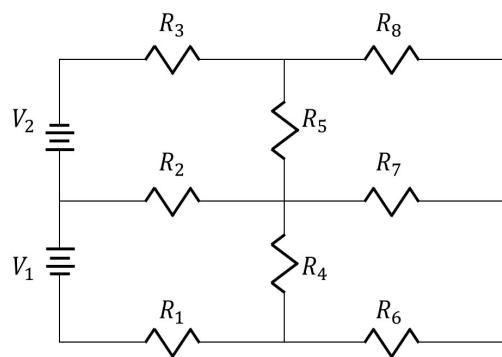
- (a) Gunakan metode eliminasi Gauss untuk mencari solusi SPL di atas.
- (b) Apa yang terjadi terhadap nilai solusi jika dimensi SPL  $AX = B$  diubah menjadi  $A_{10 \times 10}$ ,  $A_{25 \times 25}$ , dan  $A_{50 \times 50}$ .

5. Diberikan sistem persamaan linear seperti berikut.

$$\begin{bmatrix} 8 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 8 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 8 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 8 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 8 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 8 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 8 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 8 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 8 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \end{pmatrix}$$

- (a) Gunakan faktorisasi  $LU$  tanpa pivoting Program 5.4 untuk mengeluarkan matriks  $L$  dan  $U$ , kemudian hitung nilai solusi sistem persamaan linear.
- (b) Gunakan faktorisasi  $LU$  dengan pivoting Program 5.5 untuk mengeluarkan matriks  $L$ ,  $U$  dan  $P$ , kemudian hitung nilai solusi sistem persamaan linear.

6. Sebuah rangkaian resistor dengan 2 sumber tegangan diberikan seperti berikut.



Dengan menerapkan hukum Ohm dan Kirchoff didapatkanlah SPL seperti berikut

$$\begin{bmatrix} R_1 + R_2 + R_4 & -R_2 & 0 & -R_4 \\ -R_2 & R_2 + R_3 + R_5 & -R_5 & 0 \\ 0 & -R_5 & R_5 + R_7 + R_8 & -R_7 \\ -R_4 & 0 & -R_7 & R_4 + R_6 + R_7 \end{bmatrix} \begin{pmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{pmatrix} = \begin{pmatrix} -V_1 \\ V_2 \\ 0 \\ 0 \end{pmatrix}$$

Hitung solusi  $i$  dengan metode eliminasi Gauss, jika diketahui  $R_1 = R_2 = R_3 = 100$ ,  $R_4 = R_5 = 150$ ,  $R_6 = R_7 = R_8 = 200$ ,  $V_1 = 10$ , dan  $V_2 = 12$

7. Diberikan suatu sistem persamaan linear yaitu

$$x + \alpha y = 1$$

$$x + \beta y = 1$$

dengan terdapat empat kasus untuk nilai  $\alpha$  dan  $\beta$ , yaitu

- $\alpha = 101/100$  dan  $\beta = 99/100$
- $\alpha = 11/10$  dan  $\beta = 9/10$
- $\alpha = 15$  dan  $\beta = 5$
- $\alpha = 1 + \delta$  dan  $\beta = 1 - \delta$  dengan  $\delta = 10^{-8}$

Untuk setiap kasus tersebut,

- (a) Cari solusi numerik dari SPL tersebut menggunakan metode eliminasi Gauss (dengan ataupun tanpa *pivoting*)
- (b) Periksa galat absolut masing-masing solusi numerik, jika diketahui solusi eksak dari keempat kasus adalah  $x = 1$  dan  $y = 0$ .
- (c) Jika kondisi dari suatu matriks  $A$  dari SPL  $Ax = b$  dapat dihitung pada Julia dengan perintah `cond(A, 2)` yang terdapat pada paket `LinearAlgebra.jl`, jelaskan kaitan antara kondisi matriks tersebut dengan besarnya galat absolut yang diperoleh sebelumnya!

---

---

## BAB 6

---

### SOLUSI SISTEM PERSAMAAN LINEAR: METODE ITERATIF

Metode eliminasi Gauss dan faktorisasi *LU* melibatkan banyak galat pembulatan. Galat pembulatan yang terjadi pada eliminasi Gauss dapat menyebabkan solusi yang diperoleh “jauh” dari solusi sebenarnya. Gagasan tentang teknik iterasi pada pencarian akar persamaan tak-linear dapat juga diterapkan untuk menyelesaikan sistem persamaan linear. Dengan teknik iterasi, galat pembulatan dapat diperkecil, karena kita dapat meneruskan iterasi sampai solusinya seteliti mungkin, sesuai dengan batas galat yang kita tetapkan. Dengan kata lain, besar galat dapat dikendalikan sampai batas yang bisa diterima.

Jika metode eliminasi Gauss dan faktorisasi *LU* dinamakan metode langsung (*direct*) karena solusi SPL diperoleh tanpa iterasi maka metode dengan iterasi dinamakan metode tidak langsung (*indirect*) atau metode iteratif. Metode iteratif yang akan dipelajari adalah metode Jacobi dan Gauss-Seidel. Seperti pada iterasi pencarian akar persamaan, metode iteratif memiliki kemungkinan untuk konvergen maupun divergen. Syarat cukup bagi metode iteratif agar konvergen adalah matriks koefisien dominan mutlak secara diagonal (*strictly diagonally dominant*).

**Definisi 6.1 :** Matriks  $A$  yang berukuran  $N \times N$  dapat dikatakan *strictly diagonally dominant* apabila memenuhi

$$|a_{kk}| > \sum_{\substack{j=1 \\ j \neq k}}^N |a_{kj}| \quad (6.1)$$

untuk  $k = 1, 2, \dots, N$ . ■

Katakanlah terdapat sistem dengan tiga persamaan linear yaitu

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Matriks koefisien dari SPL tersebut dapat dikatakan dominan mutlak secara diagonal apa-

bila memenuhi

$$\begin{aligned}|a_{11}| &> |a_{12}| + |a_{13}| \\|a_{22}| &> |a_{21}| + |a_{23}| \\|a_{33}| &> |a_{31}| + |a_{32}|\end{aligned}$$

## 6.1 Metode Jacobi

Misalkan, akan dicari solusi dari sistem persamaan linear berukuran  $n \times n$  yaitu  $AX = B$  dengan bentuk matriks seperti berikut.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Pisahkan matriks  $A$  menjadi 3 bagian yaitu  $A = D + L + U$ , dimana matriks  $D$  adalah diagonal utama dari matriks  $A$ , matriks  $L$  adalah elemen matriks  $A$  yang berada di bawah diagonal utama, dan matriks  $U$  adalah elemen matriks  $A$  yang berada di atas diagonal utama. Dalam notasi matriks, matriks  $A$  dipisah menjadi seperti berikut.

$$A = \underbrace{\begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix}}_D + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 \\ a_{31} & a_{32} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & 0 \end{bmatrix}}_L + \underbrace{\begin{bmatrix} 0 & a_{12} & a_{13} & a_{14} \\ 0 & 0 & a_{23} & a_{24} \\ 0 & 0 & 0 & a_{34} \\ 0 & 0 & 0 & 0 \end{bmatrix}}_U$$

Katakanlah  $M = L + U$ , maka  $A = D + M$ . Dengan demikian, sistem persamaan linear

$$\begin{aligned}AX &= B \\(D + M)X &= B\end{aligned}$$

Dengan sifat distributif matriks, didapatkan

$$\begin{aligned}DX + MX &= B \\DX &= B - MX\end{aligned}$$

Kalikan masing-masing ruas dengan  $D^{-1}$ , sehingga diperoleh

$$\begin{aligned}D^{-1}DX &= D^{-1}(B - MX) \\X &= D^{-1}(B - MX)\end{aligned}$$

Dengan demikian, didapatkan formula iterasi metode Jacobi, yaitu

$$X^{(k)} = D^{-1}(B - MX^{(k-1)}) \quad (6.2)$$

untuk  $k = 1, 2, 3, 4, \dots$  dan  $X^{(0)}$  merupakan tebakan awal solusi  $X$ . Untuk menyelesaikan sistem persamaan linear menggunakan metode Jacobi, dapat mengikuti langkah-langkah berikut ini.

1. Ubah sistem berbentuk persamaan kedalam notasi matriks.
2. Cek syarat cukup kekonvergenan metode Jacobi.
3. Lakukan dekomposisi terhadap matriks koefisien yaitu  $A = D + M$ .
4. Rumuskan langkah iterasi metode Jacobi dengan persamaan 6.2.
5. Lakukan iterasi diawali dengan substitusi nilai tebakan awal  $X^{(0)}$ .

Untuk menghentikan iterasi, kriteria yang dapat digunakan adalah menghitung kedekatan dari vektor solusi numerik dan vektor solusi menggunakan panjang vektor galat.

**Definisi 6.2 Panjang vektor:** Misalkan  $X = (x_1, x_2, \dots, x_n) \in \mathbb{R}$ , panjang vektor Euclid atau norm Euclid dari  $X$  dinotasikan dengan  $\|X\|$  dan didefinisikan sebagai

$$\|X\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2} \quad (6.3)$$

■

**Definisi 6.3 Panjang vektor galat:** Misalkan  $X^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)}) \in \mathbb{R}$  merupakan solusi numerik pada iterasi ke- $k$  dan  $X^* = (x_1, x_2, \dots, x_n) \in \mathbb{R}$  merupakan solusi eksak dari SPL  $AX = B$ , maka panjang vektor galat pada iterasi ke- $k$  didefinisikan sebagai

$$\|X^* - X^{(k)}\| = \sqrt{(x_1 - x_1^{(k)})^2 + (x_2 - x_2^{(k)})^2 + \dots + (x_n - x_n^{(k)})^2} \quad (6.4)$$

■

Pada kenyataannya, definisi panjang vektor galat pada persamaan 6.4 sangatlah sulit diterapkan, karena solusi eksak pada umumnya tidak diketahui. Oleh karena itu, kriteria penghentian sering menggunakan panjang vektor galat dari dua solusi numerik dengan iterasi yang berdekatan, yaitu

$$\|X^{(k)} - X^{(k-1)}\| < \delta \quad (6.5)$$

dimana  $\delta$  merupakan nilai toleransi yang telah ditetapkan.

**Contoh 6.1 :** Diberikan sistem persamaan linear

$$\begin{aligned} 3x_1 + x_2 - x_3 &= 6 \\ -x_1 - 5x_2 + x_3 &= -11 \\ 2x_2 + 4x_3 &= 0 \end{aligned}$$

Gunakan iterasi Jacobi untuk menghitung solusi  $X$  dari SPL tersebut hingga 3 iterasi, jika diberikan nilai tebakan awal  $X^{(0)} = (1 \ 1 \ 1)^T$ .

**Solusi :**

**Langkah 1:** Ubah sistem kedalam notasi matriks, yaitu

$$\begin{bmatrix} 3 & 1 & -1 \\ -1 & -5 & 1 \\ 0 & 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -11 \\ 0 \end{bmatrix}$$

**Langkah 2:** Cek syarat cukup metode iteratif yaitu matriks koefisien  $A$  memiliki sifat dominan mutlak secara diagonal.

$$\begin{aligned}|3| &> |1| + |-1| \\|-5| &> |-1| + |1| \\|4| &> |0| + |2|\end{aligned}$$

Jadi, matriks koefisien  $A$  memiliki sifat dominan mutlak secara diagonal, sehingga metode Jacobi akan menghasilkan solusi yang konvergen menuju solusi eksak.

**Langkah 3:** Lakukan dekomposisi terhadap matriks koefisien yaitu  $A = D + M$ .

$$A = \begin{bmatrix} 3 & 0 & 0 \\ 0 & -5 & 0 \\ 0 & 0 & 4 \end{bmatrix} + \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 0 & 2 & 0 \end{bmatrix}$$

**Langkah 4:** Berdasarkan persamaan 6.2, rumus langkah iterasi metode Jacobi adalah

$$\begin{aligned}X^{(k)} &= D^{-1}(B - MX^{(k-1)}) \\&= \begin{bmatrix} 1/3 & 0 & 0 \\ 0 & -1/5 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \left( \begin{bmatrix} 6 \\ -11 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 0 & 2 & 0 \end{bmatrix} X^{(k-1)} \right) \\&= \begin{bmatrix} 2 \\ 2.2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 1/5 & 0 & -1/5 \\ 0 & 1/2 & 0 \end{bmatrix} X^{(k-1)}\end{aligned}$$

**Langkah 5:** Substitusi nilai tebakan awal yaitu  $X^{(0)} = (1 \ 1 \ 1)^T$  hingga didapatkan solusi pada iterasi ke-3.

$$\begin{aligned}X^{(1)} &= \begin{bmatrix} 2 \\ 2.2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 1/5 & 0 & -1/5 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2.2 \\ -0.5 \end{bmatrix} \\X^{(2)} &= \begin{bmatrix} 2 \\ 2.2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 1/5 & 0 & -1/5 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2.2 \\ -0.5 \end{bmatrix} = \begin{bmatrix} 1.1 \\ 1.7 \\ -1.1 \end{bmatrix} \\X^{(3)} &= \begin{bmatrix} 2 \\ 2.2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 1/5 & 0 & -1/5 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1.1 \\ 1.7 \\ -1.1 \end{bmatrix} = \begin{bmatrix} 1.07 \\ 1.76 \\ -0.85 \end{bmatrix}\end{aligned}$$

Jadi, solusi yang dihasilkan setelah iterasi ke-3 adalah  $X = [1.07 \ 1.76 \ -0.85]^T$ .  $\triangle$

## 6.2 Metode Gauss-Seidel

Sama halnya pada metode Jacobi, misalkan akan dicari solusi sistem persamaan linear berukuran  $n \times n$  yaitu  $AX = B$  dengan bentuk matriks seperti berikut.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Pisahkan matriks  $A$  menjadi 3 bagian yaitu  $A = D + L + U$ , dimana matriks  $D$  adalah diagonal utama dari matriks  $A$ , matriks  $L$  adalah elemen matriks  $A$  yang berada di bawah diagonal utama, dan matriks  $U$  adalah elemen matriks  $A$  yang berada di atas diagonal utama. Dalam notasi matriks, matriks  $A$  dipisah menjadi seperti berikut.

$$A = \underbrace{\begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix}}_D + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 \\ a_{31} & a_{32} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & 0 \end{bmatrix}}_L + \underbrace{\begin{bmatrix} 0 & a_{12} & a_{13} & a_{14} \\ 0 & 0 & a_{23} & a_{24} \\ 0 & 0 & 0 & a_{34} \\ 0 & 0 & 0 & 0 \end{bmatrix}}_U$$

Katakanlah  $R = L + D$ , maka  $A = R + U$ . Dengan demikian, sistem persamaan linear

$$\begin{aligned} AX &= B \\ (R + U)X &= B \end{aligned}$$

Dengan sifat distributif matriks, didapatkan

$$\begin{aligned} RX + UX &= B \\ RX &= B - UX \end{aligned}$$

Kalikan masing-masing ruas dengan  $R^{-1}$ , sehingga diperoleh

$$\begin{aligned} R^{-1}DX &= R^{-1}(B - UX) \\ X &= R^{-1}(B - UX) \end{aligned}$$

Dengan demikian, didapatkan formula iterasi metode Gauss-Seidel, yaitu

$$X^{(k)} = R^{-1}(B - UX^{(k-1)}) \quad (6.6)$$

untuk  $k = 1, 2, 3, 4, \dots$  dan  $X^{(0)}$  merupakan tebakan awal solusi  $X$ . Untuk menyelesaikan sistem persamaan linear menggunakan metode Gauss-Seidel, dapat mengikuti langkah-langkah berikut ini.

1. Ubah sistem berbentuk persamaan kedalam notasi matriks.
2. Cek syarat cukup kekonvergenan metode iterasi Gauss-Seidel.
3. Lakukan dekomposisi terhadap matriks koefisien yaitu  $A = R + U$ .
4. Rumuskan langkah iterasi metode Gauss-Seidel dengan persamaan 6.6.
5. Lakukan iterasi diawali dengan substitusi nilai tebakan awal  $X^{(0)}$ .

Untuk menghentikan iterasi Gauss-Seidel, kriteria yang dapat digunakan adalah dengan menghitung kedekatan dari vektor solusi numerik dan vektor solusi eksak menggunakan panjang vektor galat. Sama seperti metode Jacobi, solusi eksak pada umumnya tidak diketahui. Dengan demikian, panjang vektor galat yang memungkinkan untuk diterapkan pada metode Gauss-Seidel adalah kedekatan solusi numerik dengan iterasi yang berdekatan pada persamaan 6.5.

**Contoh 6.2 :** Diberikan sistem persamaan linear

$$\begin{aligned} 3x_1 + x_2 - x_3 &= 6 \\ -x_1 - 5x_2 + x_3 &= -11 \\ 2x_2 + 4x_3 &= 0 \end{aligned}$$

Gunakan iterasi Gauss-Seidel untuk menghitung solusi  $X$  dari SPL tersebut hingga tiga iterasi, jika diberikan nilai tebakan awal  $X^{(0)} = (1 \ 1 \ 1)^T$ .

**Solusi :**

**Langkah 1:** Ubah sistem kedalam notasi matriks, yaitu

$$\begin{bmatrix} 3 & 1 & -1 \\ -1 & -5 & 1 \\ 0 & 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -11 \\ 0 \end{bmatrix}$$

**Langkah 2:** Cek syarat cukup metode iteratif yaitu matriks koefisien  $A$  memiliki sifat dominan mutlak secara diagonal.

$$\begin{aligned} |3| &> |1| + |-1| \\ |-5| &> |-1| + |1| \\ |4| &> |0| + |2| \end{aligned}$$

Jadi, matriks koefisien  $A$  memiliki sifat dominan mutlak secara diagonal, sehingga metode Gauss-Seidel akan menghasilkan solusi yang konvergen menuju solusi eksak.

**Langkah 3:** Lakukan dekomposisi terhadap matriks koefisien yaitu  $A = R + U$ .

$$A = \begin{bmatrix} 3 & 0 & 0 \\ -1 & -5 & 0 \\ 0 & 2 & 4 \end{bmatrix} + \begin{bmatrix} 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

**Langkah 4:** Sebelum merumuskan langkah iterasi Gauss-Seidel, akan dihitung terlebih dahulu nilai  $R^{-1}$ . Dari matriks  $R$ , didapatkan nilai  $\det(R) = -60$  dan matriks adjoint sebagai berikut.

$$C = \begin{bmatrix} -20 & 4 & -2 \\ 0 & 12 & -6 \\ 0 & 0 & -15 \end{bmatrix}$$

Dengan demikian, didapatkan matriks  $R^{-1}$  yaitu

$$R^{-1} = \frac{1}{\det(R)} C^T = \frac{1}{-60} \begin{bmatrix} -20 & 0 & 0 \\ 4 & 12 & 0 \\ -2 & -6 & -15 \end{bmatrix} = \begin{bmatrix} 1/3 & 0 & 0 \\ -1/15 & -1/5 & 0 \\ 1/30 & 1/10 & 1/4 \end{bmatrix}$$

Berdasarkan persamaan 6.6, rumus langkah iterasi metode Gauss-Seidel adalah

$$\begin{aligned} X^{(k)} &= R^{-1} (B - UX^{(k-1)}) \\ &= \begin{bmatrix} 1/3 & 0 & 0 \\ -1/15 & -1/5 & 0 \\ 1/30 & 1/10 & 1/4 \end{bmatrix} \left( \begin{bmatrix} 6 \\ -11 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} X^{(k-1)} \right) \\ &= \begin{bmatrix} 2 \\ 1.8 \\ -0.9 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 0 & -1/15 & -4/30 \\ 0 & 1/30 & 2/30 \end{bmatrix} X^{(k-1)} \end{aligned}$$

**Langkah 5:** Substitusi nilai tebakan awal yaitu  $X^{(0)} = (1 \ 1 \ 1)^T$  hingga didapatkan

solusi pada iterasi ke-3.

$$\begin{aligned} X^{(1)} &= \begin{bmatrix} 2 \\ 1.8 \\ -0.9 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 0 & -1/15 & -4/30 \\ 0 & 1/30 & 2/30 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ -1 \end{bmatrix} \\ X^{(2)} &= \begin{bmatrix} 2 \\ 1.8 \\ -0.9 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 0 & -1/15 & -4/30 \\ 0 & 1/30 & 2/30 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1.8 \\ -0.9 \end{bmatrix} \\ X^{(3)} &= \begin{bmatrix} 2 \\ 1.8 \\ -0.9 \end{bmatrix} - \begin{bmatrix} 0 & 1/3 & -1/3 \\ 0 & -1/15 & -4/30 \\ 0 & 1/30 & 2/30 \end{bmatrix} \begin{bmatrix} 1 \\ 1.8 \\ -0.9 \end{bmatrix} = \begin{bmatrix} 1.1 \\ 1.8 \\ -0.9 \end{bmatrix} \end{aligned}$$

Jadi, solusi yang dihasilkan setelah iterasi ke-3 adalah  $X = [1.1 \ 1.8 \ -0.9]^T$ .  $\triangle$

### 6.3 Metode Rekonstruksi Aljabar

Metode atau teknik rekonstruksi aljabar (*algebraic reconstruction technique*) merupakan salah satu metode pertama yang diimplementasikan dalam komputer untuk menyelesaikan sistem persamaan linear. Metode dikembangkan oleh S. Kaczmarz pada tahun 1937. Misalkan SPL yang terdiri dari  $M$  persamaan dengan  $N$  faktor yang tidak diketahui seperti berikut.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= b_2 \\ &\vdots \\ a_{M1}x_1 + a_{M2}x_2 + \cdots + a_{MN}x_N &= b_M \end{aligned}$$

Persamaan linear simultan tersebut dapat dituliskan sebagai

$$\mathbf{a}_i^T \mathbf{x} = b_i \quad \text{dengan } i = 1, 2, \dots, M \quad (6.7)$$

dengan vektor kolom  $\mathbf{a}$  dan  $\mathbf{x}$  yaitu

$$\mathbf{a}_i = \begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{iN} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \quad \text{dengan } i = 1, 2, \dots, M$$

Perhatikan bahwa sebanyak  $M$  persamaan pada Persamaan 6.7 merupakan **hiperbidang** (*hyperplane*) pada ruang Euclidean berdimensi  $N$ ,  $\mathbb{R}^N$ . Setiap  $M$  persamaan disebut garis dan bidang untuk kasus di  $\mathbb{R}^2$  dan  $\mathbb{R}^3$ .

Sistem persamaan linear tidak akan mempunyai sebuah penyelesaian eksak, apabila  $M$  hiperbidangnya tidak memiliki perpotongan yang sama. Oleh karena itu, perlu ditentukan suatu titik pada  $\mathbb{R}^N$  yang relatif "dekat" dengan seluruh hiperbidang tersebut. Titik tersebut akan menjadi hampiran dari solusi sistem persamaan linear. Metode rekonstruksi aljabar akan menghasilkan iterasi berupa siklus proyeksi ortogonal yang berurutan pada  $M$  hiperbidang.

**Teorema 6.1 Proyeksi Ortogonal:** Misalkan  $L$  merupakan sebuah hiperbidang pada  $\mathbb{R}^n$  dengan persamaan  $\mathbf{a}^T \mathbf{x} = b$  dan misalkan  $\mathbf{x}^*$  adalah sebarang titik pada  $\mathbb{R}^n$ , maka proyeksi ortogonal dari  $\mathbf{x}^*$  terhadap  $L$  yaitu  $\mathbf{x}_p$  dapat dinyatakan sebagai berikut.

$$\mathbf{x}_p = \mathbf{x}^* + \frac{(b - \mathbf{a}^T \mathbf{x}^*)}{\mathbf{a}^T \mathbf{a}} \mathbf{a} \quad (6.8)$$

■

Misalkan  $\mathbf{x}_k^{(p)}$  adalah titik yang terletak pada hiperbidang ke- $k$  yang dihasilkan saat siklus iterasi ke- $p$ . Solusi numerik sistem persamaan linear dapat dicari menggunakan metode rekonstruksi aljabar dengan langkah-langkah seperti berikut.

1. Pilih titik sebarang pada  $\mathbb{R}^N$  dan tandai dengan  $\mathbf{x}_0$ .
2. Untuk siklus iterasi pertama, tetapkan  $p = 1$ .
3. Untuk  $k = 1, 2, \dots, M$ , hitung proyeksi ortogonal hiperbidang ke- $k$  menggunakan persamaan berikut.

$$\mathbf{x}_k^{(p)} = \mathbf{x}_{k-1}^{(p)} + \frac{(b_k - \mathbf{a}_k^T \mathbf{x}_{k-1}^{(p)})}{\mathbf{a}_k^T \mathbf{a}_k} \mathbf{a}_k \quad (6.9)$$

4. Tetapkan  $\mathbf{x}_0^{(p+1)} = \mathbf{x}_M^{(p)}$ .
5. Naikkan jumlah siklus  $p$  sebanyak satu dan ulangi Langkah 3.

**Contoh 6.3 :** Diketahui suatu sistem persamaan linear sebagai berikut.

$$\begin{aligned} -x_2 + 2x_3 &= -2 \\ 2x_1 + 3x_2 - x_3 &= 8 \\ -x_1 + x_2 + 2x_3 &= 1 \end{aligned}$$

Gunakan metode rekonstruksi aljabar untuk menyelesaikan sistem tersebut dengan nilai tebakan awal  $\mathbf{x}_0 = [1, 1, 1]^T$  hingga siklus iterasi  $p = 2$ , serta gunakan pembulatan 6 angka desimal pada setiap proses penghitungan. Bandingkan solusi numerik pada setiap langkah dengan solusi eksaknya yaitu  $\mathbf{x} = [1, 2, 0]^T$ .

**Solusi :** Berdasarkan sistem persamaan di atas, dapat diperoleh vektor kolom  $\mathbf{a}_k$  untuk  $k = 1, 2, 3$ , yaitu

$$\mathbf{a}_1 = \begin{bmatrix} 0 \\ -1 \\ 2 \end{bmatrix} \quad \mathbf{a}_2 = \begin{bmatrix} 2 \\ 3 \\ -1 \end{bmatrix} \quad \mathbf{a}_3 = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}$$

Selain itu, diketahui pula nilai tebakan awal yaitu  $\mathbf{x}_0^{(1)} = [1, 1, 1]^T$ .

**Iterasi 1.** Akan dihitung nilai proyeksi ortogonal untuk iterasi ke-1 dan hiperbidang ke-1, 2, 3 menggunakan Persamaan 6.9 seperti berikut.

$$\begin{aligned} \mathbf{x}_1^{(1)} &= \mathbf{x}_0^{(1)} + \frac{(b_1 - \mathbf{a}_1^T \mathbf{x}_0^{(1)})}{\mathbf{a}_1^T \mathbf{a}_1} \mathbf{a}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + (-0.6) \begin{bmatrix} 0 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1.6 \\ -0.2 \end{bmatrix} \\ \mathbf{x}_2^{(1)} &= \mathbf{x}_1^{(1)} + \frac{(b_2 - \mathbf{a}_2^T \mathbf{x}_1^{(1)})}{\mathbf{a}_2^T \mathbf{a}_2} \mathbf{a}_2 = \begin{bmatrix} 1 \\ 1.6 \\ -0.2 \end{bmatrix} + (0.071429) \begin{bmatrix} 2 \\ 3 \\ -1 \end{bmatrix} = \begin{bmatrix} 1.14286 \\ 1.81429 \\ -0.27143 \end{bmatrix} \\ \mathbf{x}_3^{(1)} &= \mathbf{x}_2^{(1)} + \frac{(b_3 - \mathbf{a}_3^T \mathbf{x}_2^{(1)})}{\mathbf{a}_3^T \mathbf{a}_3} \mathbf{a}_3 = \begin{bmatrix} 1.14286 \\ 1.81429 \\ -0.27143 \end{bmatrix} + (0.14524) \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0.997619 \\ 1.959524 \\ 0.019047 \end{bmatrix} \end{aligned}$$

Berdasarkan hasil di atas, galat dari setiap proses dapat dihitung menggunakan panjang vektor, yaitu  $\varepsilon_k^{(p)} = \|\mathbf{x}_k^{(p)} - \mathbf{x}\|$  dengan hasil  $\varepsilon_1^{(1)} = 0.4472136$ ,  $\varepsilon_2^{(1)} = 0.3585687$ , dan  $\varepsilon_3^{(1)} = 0.0447969$ .

*Iterasi 2.* Dengan cara yang sama seperti iterasi 1, diperoleh nilai proyeksi ortogonal untuk iterasi ke-2 dan hiperbidang ke-1, 2, 3 yaitu

$$\mathbf{x}_1^{(2)} = \begin{bmatrix} 0.997619 \\ 1.975238 \\ -0.012381 \end{bmatrix} \quad \mathbf{x}_2^{(2)} = \begin{bmatrix} 1.007143 \\ 1.989524 \\ -0.017143 \end{bmatrix} \quad \mathbf{x}_3^{(2)} = \begin{bmatrix} 0.998492 \\ 1.998175 \\ 0.000159 \end{bmatrix}$$

dan galat pada setiap langkah, yaitu  $\varepsilon_1^{(2)} = 0.027787$ ,  $\varepsilon_2^{(2)} = 0.021323$ , dan  $\varepsilon_3^{(2)} = 0.002373$ .

Berdasarkan hasil yang diperoleh pada iterasi 1 dan 2, metode rekonstruksi aljabar memiliki galat yang terus mengecil dan solusi yang akan dihasilkan konvergen menuju solusi eksaknya yaitu  $\mathbf{x} = [1, 2, 0]^T$ .  $\triangle$

Solusi numerik sistem persamaan linear menggunakan metode rekonstruksi aljabar memiliki tingkat kesalahan yang kecil. Namun, pada beberapa sistem, solusi numerik membutuhkan iterasi yang banyak dan waktu yang lama. Proses mendapatkan solusi numerik juga dipengaruhi oleh penentuan nilai tebakan awal  $\mathbf{x}_0$  solusi numerik.

## 6.4 Metode *Conjugate Gradient*

Metode *conjugate gradient* (CG) merupakan suatu metode iteratif untuk menyelesaikan sistem persamaan linear  $A\mathbf{x} = \mathbf{b}$ , dengan matriks koefisien  $A$  bersifat **simetrik definit positif**. Metode *conjugate gradient* banyak digunakan untuk menyelesaikan SPL yang berukuran besar. Ide metode ini adalah mencari titik minimum suatu fungsi kuadrat dari vektor yang berbentuk

$$f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x} + \mathbf{c} \quad (6.10)$$

Kemiringan atau *gradient* dari fungsi tersebut adalah

$$f'(\mathbf{x}) = \frac{1}{2} A^T \mathbf{x} + \frac{1}{2} A \mathbf{x} - \mathbf{b} \quad (6.11)$$

Jika  $A$  merupakan matriks simetrik, maka  $A^T \mathbf{x} = A\mathbf{x}$ . Dengan demikian, fungsi *gradient* dari  $f$  menjadi  $f'(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$ .

Jika  $A$  definit positif, maka fungsi  $f$  akan minimum apabila  $f'(\mathbf{x}) = 0$  atau dengan kata lain  $A\mathbf{x} = \mathbf{b}$ . Dengan demikian,  $\mathbf{x}$  merupakan titik minimum global dari  $f$ , sehingga  $\mathbf{x}$  merupakan solusi tunggal dari sistem persamaan linear.

Misalkan  $\mathbf{x}^{(p)}$  merupakan niali perkiraan solusi vektor  $\mathbf{x}$  pada iterasi ke- $p$ . Iterasi solusi numerik metode *conjugate gradient* dapat dihitung menggunakan langkah-langkah seperti berikut.

1. Pilih nilai tebakan awal  $\mathbf{x}^{(0)}$  serta nilai vektor residual awal  $r_{(0)}$  dan vektor arah awal  $d_{(0)}$  yaitu

$$r_{(0)} = \mathbf{b} - A\mathbf{x}_{(0)} \quad \text{dan} \quad d_{(0)} = r_{(0)} \quad (6.12)$$

2. Mulai iterasi dengan  $i = 0$ , kemudian hitung nilai hampiran  $x_{(i+1)}$  menggunakan persamaan

$$x_{(i+1)} = x_{(i)} + \frac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} d_{(i)} \quad (6.13)$$

3. Hitung nilai vektor residual  $r_{(i+1)}$  dan vektor arah  $d_{(i+1)}$  menggunakan persamaan

$$r_{(i+1)} = r_{(i)} - \frac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} A d_{(i)} \quad \text{dan} \quad d_{(i+1)} = r_{(i+1)} - \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} d_{(i)} \quad (6.14)$$

4. Naikkan siklus iterasi  $i$  sebanyak 1 kemudian ulangi langkah ke-2.

**Contoh 6.4 :** Diketahui suatu sistem persamaan linear sebagai berikut.

$$\begin{bmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 2 & 1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix}$$

Gunakan metode *conjugate gradient* untuk menyelesaikan sistem tersebut dengan nilai tebakan awal  $x_{(0)} = [0, 0, 0]^T$  hingga siklus mendapatkan hampiran  $x_{(2)}$ . Hitung panjang vektor residu sebagai galat hampiran setiap iterasi.

**Solusi :** Diketahui bahwa nilai solusi tebakan awal SPL adalah  $[0, 0, 0]^T$ . Metode *conjugate gradient* dimulai dengan menghitung vektor residual awal dan vektor arah awal, yaitu

$$r_{(0)} = d_{(0)} = \mathbf{b} - A\mathbf{x}_{(0)} = \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix} - A \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix}$$

*Iterasi 1.* Akan dihitung nilai hampiran solusi SPL  $\mathbf{x}_{(1)}$  menggunakan Persamaan 6.13, yaitu

$$x_{(1)} = x_{(0)} + \frac{r_{(0)}^T r_{(0)}}{d_{(0)}^T A d_{(0)}} d_{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + (0.16248) \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix} = \begin{bmatrix} 0.97489 \\ 0.81241 \\ 1.13737 \end{bmatrix}$$

Selanjutnya, hitung vektor residu  $r_{(1)}$  dan vektor arah  $d_{(1)}$  menggunakan Persamaan 6.14, yaitu

$$r_{(1)} = r_{(0)} - \frac{r_{(0)}^T r_{(0)}}{d_{(0)}^T A d_{(0)}} A d_{(0)} = \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix} - (0.16248) \begin{bmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 2 & 1 & 4 \end{bmatrix} \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix} = \begin{bmatrix} -0.011817 \\ 0.450517 \\ -0.311669 \end{bmatrix}$$

$$d_{(1)} = r_{(1)} - \frac{r_{(1)}^T r_{(1)}}{r_{(0)}^T r_{(0)}} d_{(0)} = \begin{bmatrix} -0.011817 \\ 0.450517 \\ -0.311669 \end{bmatrix} - (0.0027295) \begin{bmatrix} 6 \\ 5 \\ 7 \end{bmatrix} = \begin{bmatrix} -0.028194 \\ 0.436870 \\ -0.330775 \end{bmatrix}$$

*Iterasi 2.* Dengan mengulangi langkah-langkah pada iterasi 1, akan diperoleh nilai hampiran  $x_{(2)}$  dan nilai vektor residu  $r_{(2)}$  serta vektor arah  $d_{(2)}$ , yaitu

$$x_{(2)} = \begin{bmatrix} 0.96339 \\ 0.99056 \\ 1.00248 \end{bmatrix} \quad r_{(2)} = \begin{bmatrix} 0.114299 \\ 0.062442 \\ 0.072728 \end{bmatrix} \quad d_{(2)} = \begin{bmatrix} 0.116389 \\ 0.030063 \\ 0.097243 \end{bmatrix}$$

Panjang vektor residu dari masing-masing hampiran adalah

$$\begin{aligned} \|r_{(0)}\| &= 10.48806 \\ \|r_{(1)}\| &= 0.54794 \\ \|r_{(2)}\| &= 0.14917 \end{aligned}$$

Berdasarkan hasil yang diperoleh, metode *conjugate gradient* menghasilkan solusi yang konvergen, karena nilai panjang galat residu akan semakin kecil jika iterasi bertambah. Dengan mengulangi proses komputasi hingga iterasi ke-15, akan diperoleh solusi hampiran SPL yaitu  $x_{(15)} = [1, 1, 1]^T$  dengan panjang vektor residu  $\|r_{(15)}\| = 0.000067$ .  $\triangle$

## 6.5 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Jacobi** berisi program untuk mencari solusi sistem persamaan linear secara iteratif. Program ini secara *default* berisi 3 masukan, yaitu matriks SPL  $A$ ,  $B$ , dan nilai tebakan awal  $X_0$ , serta 4 luaran, yaitu solusi sistem persamaan linear  $X$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, matriks  $M$  berisi nilai hampiran solusi SPL pada setiap iterasi, dan matriks *err* berisi panjang vektor galat pada setiap iterasi. Berikut merupakan program untuk metode Jacobi.

**Algoritma 6.1:** Metode Jacobi

```
using LinearAlgebra
function jacobi(A, B, Xawal::Array{Int64,1})
    delta = 10^-7;
    maxi = 100;
    flag = 1;
    D = Diagonal(diag(A))
    R = A - D;
    X = Xawal;
    M = [0 X' NaN];
    for k = 1:maxi
        Xlama = X;
        X = inv(D) * (B - R*Xlama);
        err = norm(X-Xlama);
        M = [M; [k X' err] ];
        if err<delta || norm(B-A*X)<delta
            flag = 0;
            break
        end
    end
    err = M[end,end]
    return X, flag, err, M
end
```

**Metode Gauss-Seidel** berisi program untuk mencari solusi sistem persamaan linear secara iteratif. Program ini secara *default* berisi 3 masukan, yaitu matriks SPL  $A$ ,  $B$ , dan nilai tebakan awal  $X_0$ , serta 4 luaran, yaitu solusi sistem persamaan linear  $X$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, matriks  $M$  berisi nilai hampiran solusi SPL pada setiap iterasi, dan matriks *err* berisi panjang vektor galat pada setiap iterasi. Berikut merupakan program untuk metode Gauss-Seidel.

### Algoritma 6.2: Metode Gauss-Seidel

```
using LinearAlgebra
function gaussSeidel(A,B,Xawal::Array{Int64,1})
    delta = 10^-7;
    maxi = 100;
    flag = 1;
    D = Diagonal(diag(A))
    R = tril(A);
    U = triu(A,1);
    X = Xawal;
    M = [0 X' NaN];
    for k = 1:maxi
        Xlama = X;
        X = inv(R)*(B - U*Xlama);
        err = norm(X-Xlama);
        M = [M; [k X' err] ];
        if err<delta
            flag = 0;
            break
        end
    end
    err = M[end,end]
    return X, flag, err, M
end
```

**Metode Rekonstruksi Aljabar** berisi program untuk mencari solusi sistem persamaan linear secara iteratif. Program ini secara *default* berisi 3 masukan, yaitu matriks SPL  $A$ ,  $B$ , dan nilai tebakan awal  $X_0$ , serta 4 luaran, yaitu solusi sistem persamaan linear  $X$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, matriks  $M$  berisi nilai hampiran solusi SPL pada setiap iterasi, dan matriks *err* berisi panjang vektor galat pada setiap iterasi. Berikut merupakan program untuk metode Rekonstruksi Aljabar.

### Algoritma 6.3: Metode Rekonstruksi Aljabar

```
using LinearAlgebra
function rekons(A,B,Xawal::Array{Int64,1})
    delta = 10^-7;
    maxi = 100;
    flag = 1;
    X = Xawal;
    Xlama = X;
    M = [0 X' NaN];
    for k = 1:maxi
        for i = 1:length(B)
            Xlama = X;
            X = Xlama + A[i,:]*((B[i]-A[i,:]'Xlama)/(A[i,:]'A[i,:]))
        end
        err = norm(X-Xlama)
    end
end
```

```

M = [M; [k X' err] ];
if err<delta
    flag = 0;
    break
end
end
err = M[end,end]
return X, flag, err, M
end

```

**Metode Conjugate Gradient** berisi program untuk mencari solusi sistem persamaan linear secara iteratif khusus untuk matriks koefisien dengan sifat simetris definit positif. Program ini secara *default* berisi 3 masukan, yaitu matriks SPL  $A$ ,  $B$ , dan nilai tebakan awal  $X_0$ , serta 4 luaran, yaitu solusi sistem persamaan linear  $X$ , status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, matriks  $M$  berisi nilai hampiran solusi SPL pada setiap iterasi, dan matriks *err* berisi panjang vektor galat pada setiap iterasi. Berikut merupakan program untuk metode *Conjugate Gradient*.

#### Algoritma 6.4: Metode Conjugate Gradient

```

using LinearAlgebra
function conGrad(A,B,Xawal::Array{Int64,1})
    if ~(isposdef(A) && issymmetric(A))
        error("matriks A harus simetrik definit positif")
    end
    delta = 10^-7;
    maxi = 100;
    flag = 1;
    X = Xawal;
    r = B-A*X
    d = r
    M = [0 X' NaN];
    for k = 1:maxi
        Xlama = X
        rlama = r
        dlama = d
        X = Xlama + ((rlama'*rlama)/(dlama'*A*dlama))*dlama
        r = rlama - ((rlama'*rlama)/(dlama'*A*dlama))*A*dlama
        d = r - ((r'*r)/(rlama'*rlama))*dlama
        err = norm(X-Xlama)
        M = [M; [k X' err] ];
        if err<delta
            flag = 0;
            break
        end
    end
    err = M[end,end]
    return X, flag, err, M
end

```

### 6.5.1 Metode Jacobi

Metode Jacobi merupakan salah satu contoh dari metode iteratif. Solusi yang dihasilkan oleh metode iteratif merupakan hampiran dari solusi eksak SPL. Tujuan utama dari metode

Jacobi adalah melakukan serangkaian iterasi, sehingga hampiran solusi SPL memenuhi toleransi galat yang ditentukan terhadap solusi eksak SPL.

**Contoh 6.5 :** Diberikan sistem persamaan linear tiga peubah seperti berikut.

$$\begin{aligned} 4x - y + z &= 7 \\ 4x - 8y + z &= -21 \\ -2x + y + 5z &= 15 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menunjukkan bahwa metode Jacobi menghasilkan solusi yang konvergen menuju solusi eksak SPL tersebut dengan nilai tebakan awal yaitu  $X_0 = (1 \ 2 \ 2)^T$ .

**Langkah 1:** Pendefinisian matriks A, B, dan tebakan solusi awal.

```
Inp: A = [4 -1 1
          4 -8 1
          -2 1 5];
      B = [7;-21;15];
      Xa = [1,2,2];
```

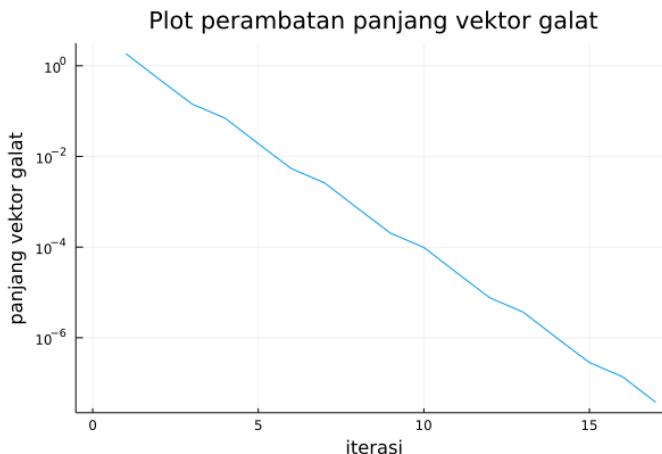
**Langkah 2:** Penghitungan iterasi Jacobi menggunakan Program 6.1.

```
Inp: X,flag,err,M = jacobi(A,B,Xa)
      @show X
      @show flag
      @show err
      M

Out: X = [1.999999884128572, 3.999999907302857, 3.0000000018539428]
      flag = 0
      err = 3.777053140308286e-8
      18x5 Array{Float64,2}:
      0.0 1.0 2.0 2.0 NaN
      1.0 1.75 3.375 3.0 1.85826
      2.0 1.84375 3.875 3.025 0.509327
      3.0 1.9625 3.925 2.9625 0.143205
      4.0 1.99062 3.97656 3.0 0.0696847
      5.0 1.99414 3.99531 3.00094 0.0190998
      :
      13.0 2.0 4.0 3.0 3.67478e-6
      14.0 2.0 4.0 3.0 1.00721e-6
      15.0 2.0 4.0 3.0 2.83194e-7
      16.0 2.0 4.0 3.0 1.37804e-7
      17.0 2.0 4.0 3.0 3.77705e-8
```

**Langkah 3:** Pembuatan plot perambatan nilai panjang vektor galat yang diperoleh.

```
Inp: using Plots
      # Plot panjang vektor galat.
      iter = M[:,1]
      err = M[:,end]
      plot(iter, err, yaxis = :log, label = :none)
      # Tambahkan title dan label
      title!("Plot perambatan panjang vektor galat")
      xlabel!("iterasi")
      ylabel!("panjang vektor galat")
```



**Gambar 6.1:** Plot perambatan vektor galat metode Jacobi.

Gambar 6.1 menunjukkan perambatan panjang vektor galat dari metode Jacobi pada setiap iterasi, yaitu iterasi ke-1 hampiran solusi SPL memiliki galat lebih dari 1, iterasi ke-2 dan ke-3 hampiran solusi SPL memiliki galat lebih dari  $10^{-1}$ , dan seterusnya hingga pada iterasi ke-17 galat hampiran solusi SPL memenuhi toleransi akurasi yang diberikan, yaitu  $10^{-7}$ . Dengan demikian, metode Jacobi memerlukan sebanyak 17 iterasi untuk mencapai akurasi  $10^{-7}$ .

---

**Soal Latihan:** Diketahui sistem persamaan linear 4 peubah seperti berikut.

$$\begin{pmatrix} 8 & 2 & -3 & 2 \\ -2 & -7 & -1 & 3 \\ 1 & -2 & 6 & -2 \\ 2 & -1 & -4 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ -11 \\ 15 \\ 11 \end{pmatrix}$$

Gunakan metode iterasi Jacobi untuk menyelesaikan SPL tersebut, kemudian gambarkan plot perambatan panjang vektor galatnya.

---

### 6.5.2 Metode Gauss-Seidel

Kelemahan dari metode Jacobi adalah proses komputasi membutuhkan jumlah iterasi yang cukup banyak. Dengan demikian, diperlukan modifikasi terhadap metode ini agar jumlah iterasi berkurang. Salah satu metode tersebut adalah metode Gauss-Seidel.

**Contoh 6.6 :** Diberikan sistem persamaan linear tiga peubah seperti berikut.

$$\begin{aligned} 4x - y + z &= 7 \\ 4x - 8y + z &= -21 \\ -2x + y + 5z &= 15 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menunjukkan bahwa metode Gauss-Seidel menghasilkan solusi yang konvergen menuju solusi eksak SPL dengan nilai tebakan awal yaitu  $X_0 = (1 \ 2 \ 2)^T$ .

**Langkah 1:** Pendefinisian matriks A, B, dan tebakan solusi awal.

```
Inp: A = [4 -1 1
          4 -8 1
          -2 1 5];
B = [7;-21;15];
Xa = [1,2,2];
```

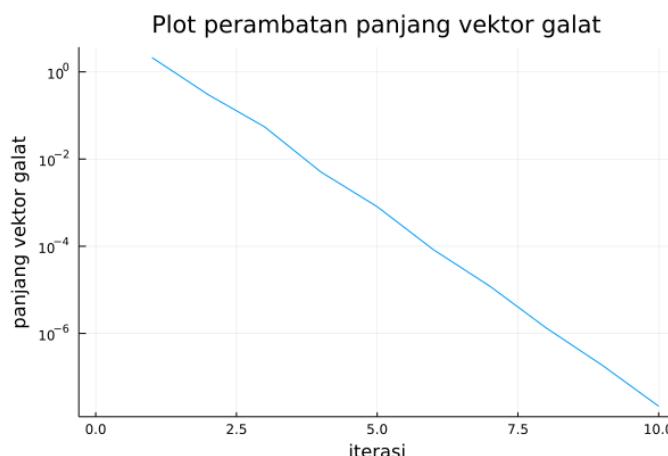
**Langkah 2:** Penghitungan iterasi Gauss-Seidel menggunakan Program 6.2.

```
Inp: X,flag,err,M = gaussSeidel(A,B,Xa)
@show X
@show flag
@show err
M

Out: X = [1.99999999743314, 3.999999998137355, 2.999999999345785]
flag = 0
err = 2.1378379298357413e-8
11x5 Array{Float64,2}:
 0.0 1.0 2.0 2.0      NaN
 1.0 1.75 3.75 2.95 2.12779
 2.0 1.95 3.96875 2.98625 0.298606
 3.0 1.99562 3.99609 2.99903 0.0547054
 4.0 1.99927 3.99951 2.9998 0.00505308
 5.0 1.99993 3.99994 2.99998 0.000807443
 6.0 1.99999 3.99999 3.0 8.30727e-5
 7.0 2.0 4.0 3.0 1.21519e-5
 8.0 2.0 4.0 3.0 1.34052e-6
 9.0 2.0 4.0 3.0 1.85294e-7
10.0 2.0 4.0 3.0 2.13784e-8
```

**Langkah 3:** Pembuatan plot perambatan nilai panjang vektor galat yang diperoleh.

```
Inp: # Plot panjang vektor galat.
iter = M[:,1]
err = M[:,end]
plot(iter, err, yaxis = :log, label = :none)
# Tambahkan title dan label
title!("Plot perambatan panjang vektor galat")
xlabel!("iterasi")
ylabel!("panjang vektor galat")
```



**Gambar 6.2:** Plot perambatan vektor galat metode Gauss-Seidel.

Gambar 6.2 menunjukkan perambatan panjang vektor galat dari metode Gauss-Seidel pada setiap iterasi, yaitu iterasi ke-1 hampiran solusi SPL memiliki galat lebih dari 1, iterasi ke-2 hampiran solusi SPL memiliki galat lebih dari  $10^{-1}$ , dan seterusnya hingga pada iterasi ke-10, galat hampiran solusi SPL memenuhi toleransi akurasi yang diberikan, yaitu  $10^{-7}$ . Dengan demikian, metode Gauss-Seidel memerlukan sebanyak 10 iterasi untuk mencapai akurasi  $10^{-7}$ . Jadi, metode Gauss-Seidel lebih cepat mencari solusi SPL untuk mencapai toleransi yang diberikan daripada metode Jacobi.

**Soal Latihan:** Diketahui sistem persamaan linear 4 peubah seperti berikut.

$$\begin{pmatrix} 8 & 2 & -3 & 2 \\ -2 & -7 & -1 & 3 \\ 1 & -2 & 6 & -2 \\ 2 & -1 & -4 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ -11 \\ 15 \\ 11 \end{pmatrix}$$

Gunakan metode iterasi Gauss-Seidel untuk menyelesaikan SPL tersebut, kemudian gambarkan plot perambatan panjang vektor galatnya.

### 6.5.3 Metode Rekonstruksi Aljabar

**Contoh 6.7 :** Diberikan sistem persamaan linear tiga peubah seperti berikut.

$$\begin{aligned} 4x - y + z &= 7 \\ 4x - 8y + z &= -21 \\ -2x + y + 5z &= 15 \end{aligned}$$

Berikut merupakan langkah-langkah menunjukkan bahwa metode rekonstruksi aljabar menghasilkan solusi yang konvergen menuju solusi eksak SPL dan bandingkan dengan metode Jacobi serta Gauss-Seidel dengan nilai tebakan awal yaitu  $X_0 = (1 \ 2 \ 2)^T$ .

**Langkah 1:** Pendefinisian matriks A, B, dan tebakan solusi awal.

```
Inp: A = [4 -1 1
          4 -8 1
          -2 1 5];
B = [7;-21;15];
Xa = [1,2,2];
```

**Langkah 2:** Penghitungan iterasi rekonstruksi aljabar menggunakan Program 6.3.

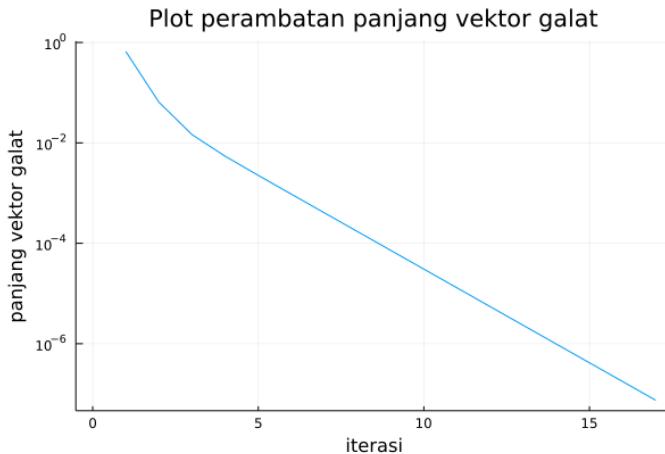
```
Inp: X,flag,err,M = rekons(A,B,Xa)
@show X
@show flag
@show err
M

Out: X = [1.9999987931831584, 3.9999993707528194, 2.9999996431226994]
flag = 0
err = 7.477270454885512e-8
18x5 Array{Float64,2}:
 0.0 1.0 2.0 2.0      NaN
 1.0 0.677229 3.45151 2.58059 0.658544
 2.0 1.51207 3.75397 2.85403 0.0644652
 3.0 1.79762 3.89498 2.94005 0.0145717
```

4.0	1.91453	3.95547	2.97472	0.0054161
5.0	1.96381	3.98113	2.9893	0.00224922
:				
13.0	1.99996	3.99998	2.99999	2.32292e-6
14.0	1.99998	3.99999	3.0	9.83924e-7
15.0	1.99999	4.0	3.0	4.16762e-7
16.0	2.0	4.0	3.0	1.76529e-7
17.0	2.0	4.0	3.0	7.47727e-8

**Langkah 3:** Pembuatan plot perambatan nilai panjang vektor galat yang diperoleh.

```
Inp: # Plot panjang vektor galat.
iter = M[:,1]
err = M[:,end]
plot(iter, err, yaxis = :log, label = :none)
# Tambahkan title dan label
title!("Plot perambatan panjang vektor galat")
xlabel!("iterasi")
ylabel!("panjang vektor galat")
```



**Gambar 6.3:** Plot perambatan vektor galat metode rekonstruksi aljabar.

Gambar 6.3 menunjukkan perambatan panjang vektor galat dari metode rekonstruksi aljabar pada setiap iterasi. Metode ini memerlukan sebanyak 17 iterasi untuk mencapai akurasi  $10^{-7}$ , hampir setara dengan metode Jacobi. Namun, pada iterasi-iterasi awal, metode ini memberikan error yang lebih kecil dibandingkan metode Jacobi maupun Gauss-Seidel.

**Soal Latihan:** Diketahui sistem persamaan linear 4 peubah seperti berikut.

$$\begin{pmatrix} 8 & 2 & -3 & 2 \\ -2 & -7 & -1 & 3 \\ 1 & -2 & 6 & -2 \\ 2 & -1 & -4 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ -11 \\ 15 \\ 11 \end{pmatrix}$$

Gunakan metode iterasi rekonstruksi aljabar untuk menyelesaikan SPL tersebut, kemudian gambarkan plot perambatan panjang vektor galatnya.

### 6.5.4 Conjugate Gradient

**Contoh 6.8 :** Diberikan sistem persamaan linear tiga peubah seperti berikut.

$$\begin{aligned} 4x - y + z &= 7 \\ 4x - 8y + z &= -21 \\ -2x + y + 5z &= 15 \end{aligned}$$

Apakah metode *conjugate gradient* menghasilkan solusi yang konvergen menuju solusi eksak SPL tersebut? Jika iya, gunakan metode *conjugate gradient* untuk menyelesaikan SPL dengan nilai tebakan awal yaitu  $X_0 = (1 \ 2 \ 2)^T$ . Jika tidak, modifikasi terlebih dulu SPL sehingga dapat diselesaikan menggunakan metode *conjugate gradient*, kemudian selesaikan dengan nilai tebakan awal yaitu  $X_0 = (1 \ 2 \ 2)^T$ .

**Langkah 1:** Pendefinisian matriks A, B, dan tebakan solusi awal.

```
Inp: A = [4 -1 1
          4 -8 1
          -2 1 5];
      B = [7;-21;15];
      Xa = [1,2,2];
```

**Langkah 2:** Penghitungan iterasi rekonstruksi aljabar menggunakan Program 6.4.

```
Inp: X,flag,err,M = conGrad(A,B,Xa)
```

```
Out: matriks A harus simetrik definit positif
```

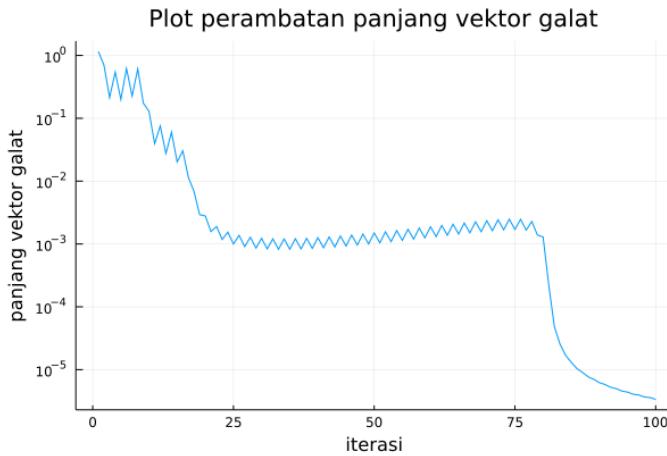
**Langkah 3:** Karena, matriks A tidak simetrik definit positif, maka diperlukan transfor-masi terlebih dulu untuk menyelesaikan SPL tersebut dengan mengalikan ruas kanan dan kiri menjadi  $A^T A x = A^T B$ , kemudian selesaikan masalah.

```
Inp: Amod = A'A
      Bmod = A'B
      Xa = [1,2,2];
Inp: X,flag,err,M = conGrad(Amod,Bmod,Xa)
      @show X
      @show flag
      @show err
      M

Out: X = [1.9999823763728397, 3.9999905656839267, 3.0000343341958913]
      flag = 1
      err = 3.3614379666212563e-6
      101x5 Array{Float64,2}:
      0.0 1.0 2.0 2.0      NaN
      1.0 0.51751 3.03391 2.19529 1.15754
      2.0 1.05617 3.03635 2.64322 0.700569
      3.0 0.910543 3.16552 2.55507 0.213692
      4.0 1.33662 3.26162 2.87937 0.544011
      5.0 1.1788 3.314 2.76781 0.200238
      :
      96.0 1.99998 3.99999 3.00004 4.05917e-6
      97.0 1.99998 3.99999 3.00004 3.94649e-6
      98.0 1.99998 3.99999 3.00004 3.67159e-6
      99.0 1.99998 3.99999 3.00003 3.59295e-6
      100.0 1.99998 3.99999 3.00003 3.36144e-6
```

**Langkah 4:** Pembuatan plot perambatan nilai panjang vektor galat yang diperoleh.

```
Inp: iter = M[:,1]
err = M[:,end]
plot(iter, err, yaxis = :log, label = :none)
title!("Plot perambatan panjang vektor galat")
xlabel!("iterasi")
ylabel!("panjang vektor galat")
```



**Gambar 6.4:** Plot perambatan vektor galat metode *conjugate gradient*.

Gambar 6.4 menunjukkan perambatan panjang vektor galat dari metode *conjugate gradient* pada setiap iterasi. Metode ini sangat lambat untuk mencapai toleransi yang ditetapkan. Hingga 100 iterasi, galat yang diperoleh belum mencapai toleransi yaitu  $10^{-7}$ . Jika iterasi dilanjutkan, solusi numerik mencapai toleransi pada iterasi ke-7906.

**Soal Latihan:** Diketahui sistem persamaan linear 4 peubah seperti berikut.

$$\begin{pmatrix} 8 & 2 & -3 & 2 \\ -2 & -7 & -1 & 3 \\ 1 & -2 & 6 & -2 \\ 2 & -1 & -4 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ -11 \\ 15 \\ 11 \end{pmatrix}$$

Gunakan metode *conjugate gradient* untuk menyelesaikan SPL tersebut, kemudian gambarkan plot perambatan panjang vektor galatnya.

## 6.6 Latihan-latihan

### 6.6.1 Ulasan Materi

1. Sebutkan dan jelaskan keunggulan dan kelemahan metode iteratif dalam menyelesaikan sistem persamaan linear dibandingkan metode langsung!
2. Apakah metode iterasi Jacobi dan Gauss-Seidel dapat selalu menyelesaikan masalah sistem persamaan linear? Jika ada, jelaskan syarat agar metode iterasi Jacobi dan Gauss-Seidel konvergen menuju solusi SPL!
3. Apakah metode rekonstruksi aljabar dapat selalu menyelesaikan masalah sistem persamaan linear? Jika ada, jelaskan syarat agar metode rekonstruksi aljabar konvergen menuju solusi SPL!

4. Apakah metode *conjugate gradient* dapat selalu menyelesaikan masalah sistem persamaan linear? Jika ada, jelaskan syarat agar metode *conjugate gradient* konvergen menuju solusi SPL!
5. Jelaskan penurunan rumus dari suatu SPL sembarang  $Ax = B$  sehingga diperoleh persamaan iterasi untuk metode
  - (a) Jacobi
  - (b) Gauss-Seidel
  - (c) rekonstruksi aljabar
  - (d) *conjugate gradient*
6. Sebutkan dan jelaskan keunggulan serta kelemahan beberapa metode iterasi berikut dibandingkan metode iterasi yang lain.
  - (a) Metode Jacobi
  - (b) Metode Gauss-Seidel
  - (c) Metode rekonstruksi aljabar
  - (d) Metode *conjugate gradient*
7. Sebutkan lima bidang ilmu pengetahuan yang menerapkan atau mengaplikasikan sistem persamaan linear! Berikan contoh kasus hingga diperoleh suatu SPL dari kelima bidang yang disebutkan tersebut!

### 6.6.2 Soal Pemrograman

1. Dengan menggunakan metode **Jacobi** dan **Gauss-Seidel**, carilah solusi SPL dibawah ini ! Apakah kedua metode konvergen ke solusi ? Jelaskan !
  - (a) Matriks 1

$$\begin{pmatrix} 4 & 8 & 4 & 0 \\ 1 & 5 & 4 & -3 \\ 1 & 4 & 7 & 2 \\ 1 & 3 & 0 & -2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 16 \\ 7 \\ 14 \\ 2 \end{pmatrix}$$

- (b) Matriks 2

$$\begin{pmatrix} 8 & 1 & 4 & 0 \\ 1 & -9 & 1 & -2 \\ 1 & 1 & 7 & 2 \\ 1 & 3 & 0 & -5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 13 \\ -9 \\ 11 \\ -1 \end{pmatrix}$$

2. Pembentukan matriks acak untuk metode iteratif
  - Bangkitkan matriks bilangan acak  $A = (a_{ij})_{(n \times n)} \in Z$  pada selang  $[-n, n]$  yang memenuhi syarat kekonvergenan metode iteratif.
  - Tetapkan matriks  $B = A \cdot (1)_{(n \times 1)}$ .
  - Cari solusi numerik  $X_1$  menggunakan metode Jacobi.
  - Cari solusi numerik  $X_2$  menggunakan metode Gauss-Seidel.
  - Cari solusi numerik  $X_3$  menggunakan metode rekonstruksi aljabar.
  - Cari solusi numerik  $X_4$  menggunakan metode *conjugate gradient*.
  - Buatlah plot perambatan panjang vektor galat dari keempat metode tersebut.

Lakukan beberapa tahapan tersebut untuk:

- |              |               |
|--------------|---------------|
| (a) $n = 20$ | (c) $n = 100$ |
| (b) $n = 50$ | (d) $n = 500$ |

3. Diberikan suatu sistem persamaan linear (SPL) sebagai berikut:

$$\begin{bmatrix} 12 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ -2 & 12 & -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 12 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 12 & -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & -2 & 12 & -2 & 1 & 0 \\ 0 & 0 & 0 & 1 & -2 & 12 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -2 & 12 & -2 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \\ 5 \end{bmatrix}$$

- (a) Gunakan metode iterasi Jacobi dan Gauss-Seidel untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
  - (b) Gunakan metode iterasi rekonstruksi aljabar dan *conjugate gradient* untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
  - (c) Apa yang terjadi terhadap nilai solusi jika dimensi SPL  $AX = B$  diubah menjadi  $A_{10 \times 10}$ ,  $A_{25 \times 25}$ , dan  $A_{50 \times 50}$ .
4. Diberikan matriks tridiagonal sebagai berikut:

$$\begin{bmatrix} -4 & 1 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

- (a) Gunakan metode iterasi Jacobi dan Gauss-Seidel untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
  - (b) Gunakan metode iterasi rekonstruksi aljabar dan *conjugate gradient* untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
  - (c) Apa yang terjadi terhadap nilai solusi jika dimensi SPL  $AX = B$  diubah menjadi  $A_{10 \times 10}$ ,  $A_{25 \times 25}$ , dan  $A_{50 \times 50}$ .
5. Diberikan sistem persamaan linear seperti berikut.

$$\begin{pmatrix} 8 & 1 & 0 & 0 & 0 \\ 1 & 8 & 1 & 0 & 0 \\ 0 & 1 & 8 & 1 & 0 \\ 0 & 0 & 1 & 8 & 1 \\ 0 & 0 & 0 & 1 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 4 \\ 4 \\ 4 \end{pmatrix}$$

- (a) Gunakan metode iterasi Jacobi dan Gauss-Seidel untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
- (b) Gunakan metode iterasi rekonstruksi aljabar dan *conjugate gradient* untuk mencari solusi SPL di atas, lalu gambarkan plot perambatan galat masing-masing metode.
- (c) Apa yang terjadi terhadap nilai solusi jika dimensi SPL  $AX = B$  diubah menjadi  $A_{10 \times 10}$ ,  $A_{25 \times 25}$ , dan  $A_{50 \times 50}$ .

6. Diberikan dua sistem persamaan linear (SPL) berikut:

$$M_p \cdot X_p = B_p$$

dengan

$$M_p = \begin{bmatrix} 232 & 3 & 4 & 9 & 19 & 20 \\ 14 & 272 & 18 & 18 & 9 & 4 \\ 17 & 1 & 156 & 1 & 4 & 14 \\ 1 & 5 & 2 & 112 & 10 & 7 \\ 18 & 19 & 11 & 14 & 280 & 4 \\ 30 & 7 & 17 & 14 & 7 & 208 \end{bmatrix}, \quad B_p = M_p \cdot \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

dan

$$M_q \cdot X_q = B_q$$

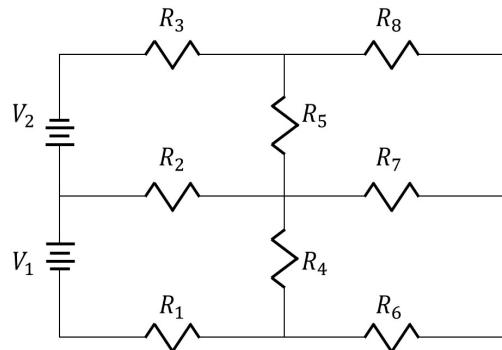
dengan

$$M_q = \begin{bmatrix} 232 & 3 & 4 & 9 & 19 & 240 \\ 14 & 272 & 18 & 18 & 9 & 4 \\ 17 & 1 & 156 & 1 & 4 & 14 \\ 1 & 5 & 2 & 112 & 10 & 7 \\ 18 & 19 & 11 & 14 & 280 & 4 \\ 200 & 7 & 17 & 14 & 7 & 208 \end{bmatrix}, \quad B_q = M_q \cdot \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

Dengan pengaturan seperti itu, solusi eksak kedua sistem persamaan linear di atas adalah  $X_p = X_q = [1, 2, 3, 4, 5, 6]^T$ .

- (a) Dengan metode Jacobi, tentukan solusi numerik kedua SPL tersebut menggunakan toleransi galat sebesar  $10^{-7}$ . Sebut hampiran solusi numerik kedua SPL tersebut masing-masing adalah  $\hat{X}_p$  dan  $\hat{X}_q$ .
  - i. Apakah iterasi metode Jacobi konvergen ke solusi pada saat menyelesaikan kedua SPL tersebut?
  - ii. Berapakah nilai norm galat absolut  $\hat{X}_p$  dan  $\hat{X}_q$  ?
  - iii. Berapakah nilai residual  $r_p = \|B_p - M_p \cdot \hat{X}_p\|_2$  dan  $r_q = \|B_q - M_q \cdot \hat{X}_q\|_2$  ?
  - iv. Bila jumlah iterasi maksimum dinaikkan, akankah metode Jacobi konvergen ke solusi?
- (b) Dengan metode Gauss-Seidel, tentukan solusi numerik kedua SPL tersebut menggunakan toleransi galat sebesar  $10^{-7}$ . Sebut hampiran solusi numerik kedua SPL tersebut masing-masing adalah  $\hat{X}_p$  dan  $\hat{X}_q$ .
  - i. Apakah iterasi metode Gauss-Seidel konvergen ke solusi pada saat menyelesaikan kedua SPL tersebut?
  - ii. Berapakah nilai norm galat absolut  $\hat{X}_p$  dan  $\hat{X}_q$  ?
  - iii. Berapakah nilai residual  $r_p = \|B_p - M_p \cdot \hat{X}_p\|_2$  dan  $r_q = \|B_q - M_q \cdot \hat{X}_q\|_2$  ?
  - iv. Bila jumlah iterasi maksimum dinaikkan, akankah metode Gauss-Seidel konvergen ke solusi?
- (c) Bila metode Jacobi dan Gauss-Seidel keduanya konvergen ke solusi, metode manakah yang lebih baik?
- (d) Matriks pada kedua SPL sebenarnya sangat mirip, namun keduanya memberikan solusi hampiran yang berbeda. Jelaskan!

7. Sebuah rangkaian resistor dengan 2 sumber tegangan diberikan seperti berikut



Dengan menerapkan hukum Ohm dan Kirchoff, didapatkanlah SPL seperti berikut.

$$\begin{bmatrix} R_1 + R_2 + R_4 & -R_2 & 0 & -R_4 \\ -R_2 & R_2 + R_3 + R_5 & -R_5 & 0 \\ 0 & -R_5 & R_5 + R_7 + R_8 & -R_7 \\ -R_4 & 0 & -R_7 & R_4 + R_6 + R_7 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix} = \begin{bmatrix} -V_1 \\ V_2 \\ 0 \\ 0 \end{bmatrix}$$

Hitung nilai solusi menggunakan metode iteratif Jacobi dan Gauss-Seidel, jika diketahui  $R_1 = R_2 = R_3 = 100$ ,  $R_4 = R_5 = 150$ ,  $R_6 = R_7 = R_8 = 200$ ,  $V_1 = 10$ , dan  $V_2 = 12$

---

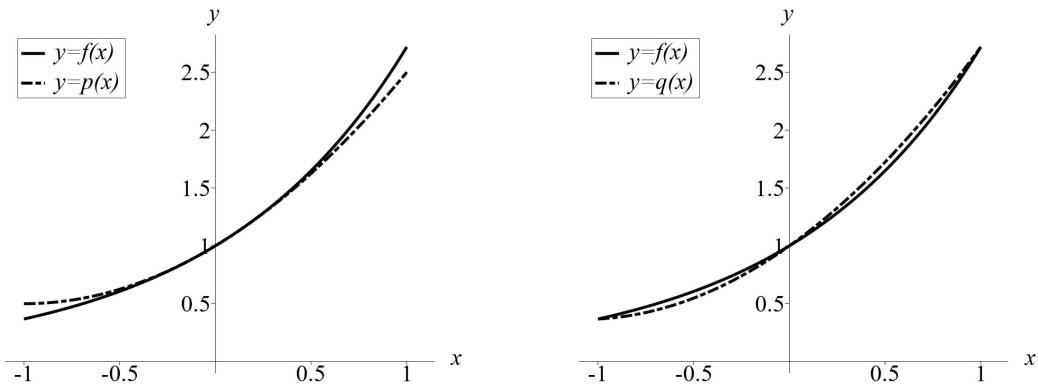
---

## BAB 7

---

### HAMPIRAN POLINOMIAL: DERET TAYLOR DAN INTERPOLASI

Pada perangkat lunak komputer, proses komputasi untuk mengevaluasi fungsi *built-in*, seperti  $\sin(x)$  atau  $\exp(x)$ , akan melibatkan hampiran polinomial. Hampiran polinomial dari suatu fungsi dapat dihitung menggunakan beberapa metode, yaitu deret Taylor dan interpolasi. Tidak seperti deret Taylor yang membutuhkan fungsi eksak dan turunannya, interpolasi hanya membutuhkan beberapa pasangan titik untuk membentuk hampiran polinomial.



(a) Hampiran Taylor  $p(x)$  untuk  $f(x) = e^x$       (b) Garis interpolasi  $q(x)$  untuk  $f(x) = e^x$   
**Gambar 7.1:** Hampiran polinomial untuk  $f(x) = e^x$  pada interval  $[-1, 1]$

Misalkan, fungsi  $f(x) = \exp(x)$  akan digambarkan dengan hampiran deret Taylor hingga derajat  $n = 2$  pada interval  $[-1, 1]$ . Hasil hampiran deret Taylor dapat dilihat pada Gambar 7.1a. Bandingkan gambar tersebut dengan hasil interpolasi titik pada Gambar 7.1b. Interpolasi titik akan menghubungkan tiga titik yaitu  $\{(-1, e^{-1}), (0, e^0), (1, e^1)\}$ . Kedua gambar menghasilkan hampiran polinom dari fungsi  $f(x)$  dengan derajat 2, yaitu hampiran Taylor  $p(x) = 1 + x + 0.5x^2$  dan interpolasi  $q(x) = 1 + 1.17520x + 0.54308x^2$ . Namun, kedua grafik fungsi memiliki nilai maksimum galat yang berbeda, yaitu galat maksimum hampiran Taylor 0.21828 dan galat maksimum interpolasi 0.078491. Pada Bab ini, akan dibahas pembentukan hampiran polinomial menggunakan deret Taylor dan interpolasi titik.

## 7.1 Deret Taylor

Deret Taylor merupakan salah satu jenis dari proses limit. Proses limit merupakan teknik dasar untuk kalkulus. Pada kasus ini, deret Taylor menghampiri suatu fungsi dengan menambahkan secara tak-hingga suku-suku yang dihitung dari turunan fungsi tersebut. Beberapa fungsi dan ekspansi deret Taylor fungsi tersebut dapat dilihat pada Tabel 7.1. Penjumlahan suku-suku untuk hampiran dari fungsi dilakukan hingga mendapatkan nilai akurasi yang dibutuhkan.

**Tabel 7.1:** Ekspansi deret Taylor dari beberapa fungsi umum

$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$	untuk setiap $x$
$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$	untuk setiap $x$
$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$	untuk setiap $x$
$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots$	untuk setiap $-1 \leq x \leq 1$

Sebagai ilustrasi untuk menunjukkan seberapa bagus penjumlahan berhingga untuk menghampiri penjumlahan tak-hingga, akan digunakan ekspansi Taylor fungsi eksponen untuk menghitung bilangan  $e = \exp(1)$ . Dengan memilih  $x = 1$ , deret eksponen pada Tabel 7.1 menjadi seperti berikut.

$$\exp(1) = 1 + \frac{1}{1} + \frac{1^2}{2!} + \frac{1^3}{3!} + \frac{1^4}{4!} + \dots + \frac{1^k}{k!} + \dots$$

Hasil dari penjumlahan deret di atas dapat dilihat pada Tabel 7.2 berikut.

**Tabel 7.2:** Penjumlahan deret  $S_n$  untuk menghitung bilangan  $e$

$n$	$S_n = 1 + \frac{1}{1} + \frac{1^2}{2!} + \frac{1^3}{3!} + \frac{1^4}{4!} + \dots + \frac{1^n}{n!}$
0	1.
1	2.
2	2.5
3	2.6666666...
4	2.7083333...
5	2.7166666...
6	2.7180555...
7	2.7182539...
8	2.7182787...
9	2.7182815...
10	2.7182818...

Jika deret yang ditambahkan cukup banyak, maka nilai hampiran yang akurat dapat diperoleh. Dengan demikian, hampiran yang diperoleh seharusnya presisi. Dari Tabel 7.2, kepresisionan nilai hampiran  $e$  setelah penjumlahan hingga derajat ke-10 adalah 6 angka desimal.

Derajat dan koefisien dari fungsi hampiran polinomial deret Taylor dapat ditentukan menggunakan teorema berikut.

**Teorema 7.1 Hampiran Polinomial Taylor:** Asumsikan bahwa  $f \in C^{N+1}[a, b]$  dan diketahui  $x_0 \in [a, b]$ . Jika  $x \in [a, b]$ , maka

$$f(x) = P_N(x) + E_N(x) \quad (7.1)$$

dimana  $P_N(x)$  adalah fungsi polinomial yang dapat digunakan untuk menghampiri  $f(x)$ , yaitu

$$f(x) \approx P_N(x) = \sum_{k=0}^N \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k \quad (7.2)$$

dan galat  $E_N(x)$  memiliki bentuk

$$E_N(x) = \frac{f^{(N+1)}(c)}{(N+1)!} (x - x_0)^{N+1} \quad (7.3)$$

untuk suatu nilai  $c = c(x)$  yang terletak diantara  $x$  dan  $x_0$ . ■

**Contoh 7.1 :** Diberikan fungsi  $f(x) = e^x$ .

1. Gunakan  $x_0 = 0$  dan carilah  $P_5(x)$ ,  $P_7(x)$ , dan  $P_9(x)$ .
2. Hitung nilai batas galat hampiran  $f(x) \approx P_9(x)$  jika  $|x| \leq 1$ .

**Solusi :** Pertama-tama, akan dicari nilai  $f^{(k)}(x_0)$ .  $f^{(k)}(x_0)$  dapat dicari dengan cara menu-runkan fungsi  $f(x)$  hingga beberapa ordo turunan, kemudian cari pola dari turunan tersebut untuk menentukan turunan pada ordo ke- $k$ . Diketahui bahwa nilai  $f(x) = e^x$ , sehingga dengan kalkulus didapatkan fungsi turunan

$$\begin{aligned} f'(x) &= e^x \\ f''(x) &= e^x \\ f'''(x) &= e^x \\ f^{(4)}(x) &= e^x \end{aligned}$$

Gunakan  $x_0 = 0$ , sehingga didapatkan

$$\begin{aligned} f'(x_0) &= e^0 = 1 \\ f''(x_0) &= e^0 = 1 \\ f'''(x_0) &= e^0 = 1 \\ f^{(4)}(x_0) &= e^0 = 1 \end{aligned}$$

Dengan memperhatikan pola tersebut, didapatkan nilai  $f^{(k)}(x_0) = 1$ . Dengan demikian, berdasarkan Teorema 7.1, diperoleh formula hampiran polinomial Taylor yaitu

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

Gunakan  $N = 5, 7, 9$  untuk menghitung  $P_5(x)$ ,  $P_7(x)$ , dan  $P_9(x)$  seperti berikut.

$$\begin{aligned} P_5(x) &= \sum_{k=0}^5 \frac{1}{k!} x^k = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} \\ P_7(x) &= \sum_{k=0}^7 \frac{1}{k!} x^k = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} \\ P_9(x) &= \sum_{k=0}^9 \frac{1}{k!} x^k = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} + \frac{x^8}{8!} + \frac{x^9}{9!} \end{aligned}$$

Galat untuk hampiran  $P_9(x)$  adalah

$$E_9(x) = \frac{f^{(10)}(c)}{10!} x^{10}$$

Karena  $|x| \leq 1$  sehingga nilai  $0 \leq c \leq 1$ . Serta, karena  $f^{(10)}(c) = e^c$  maka  $e^c$  merupakan fungsi naik. Dengan demikian, nilai  $e^c < e^1 < 3$ , sehingga nilai batas galat untuk hampiran  $P_9(x)$  adalah

$$|E_9(x)| = \frac{|f^{(10)}(c)|}{10!} x^{10} \leq \frac{e^c}{10!} < \frac{3}{10!} < 8.26719 \times 10^{-7}$$

Jadi, hampiran  $E_9(x)$  memiliki batas akurasi hingga 7 angka desimal. Dengan demikian, nilai hampiran  $e$  ketika  $n = 9$  pada Tabel 7.2 yaitu  $e = 2.718281$  benar hingga 6 angka desimal.  $\triangle$

Teori dasar dalam hampiran fungsi adalah untuk mencari hampiran polinomial yang akurat terhadap fungsi analitik  $f$  pada interval  $[a, b]$ . Fungsi  $f$  dapat dikatakan analitik di  $x_0$  apabila memiliki turunan untuk semua ordo dan dapat direpresentasikan sebagai deret Taylor pada interval sekitar  $x_0$ . Teknik ini sering digunakan dalam pengembangan *software* komputer. Nilai akurasi dari polinomial Taylor akan meningkat ketika  $N$  yang dipilih bernilai besar. Selain itu, nilai akurasi dari hampiran polinomial tertentu secara umum akan menurun ketika nilai  $x$  bergerak menjauhi nilai tengah  $x_0$ . Oleh karena itu, nilai akurasi yang baik dapat diperoleh dengan cara memilih nilai  $N$  yang cukup besar dan membatasi nilai maksimum  $|x - x_0|$ , sehingga kesalahan tidak melebihi batas yang ditentukan. Jika lebar interval yang dipilih adalah  $2R$  dan  $x_0$  berada ditengahnya, maka nilai absolut galat dari hampiran polinomial akan memenuhi formula berikut.

$$|E_N(x)| \leq \frac{MR^{N+1}}{(N+1)!} \quad (7.4)$$

dimana  $M \leq \max \left\{ \left| f^{(N+1)}(z) \right| : x_0 - R \leq z \leq x_0 + R \right\}$ .

**Contoh 7.2 :** Gunakan Persamaan 7.4 untuk menghitung nilai batas galat dari hampiran  $e^x \approx P_8(x)$  pada interval  $|x| \leq 1$  dan  $|x| \leq 0.5$ .

**Solusi :** Jika  $|x| \leq 1$ , maka nilai  $R = 1$  dan  $|f^{(9)}(c)| = |e^c| \leq e^1 = M$ . Dengan demikian, didapatkan nilai batas galat yaitu

$$|\text{galat}| = |E_8(x)| \leq \frac{e^1(1)^9}{9!} \approx 0.00000749$$

Jika  $|x| \leq 0.5$ , maka nilai  $R = 0.5$  dan  $|f^{(9)}(c)| = |e^c| \leq e^{0.5} = M$ . Dengan demikian, didapatkan nilai batas galat yaitu

$$|\text{galat}| = |E_8(x)| \leq \frac{e^{0.5}(0.5)^9}{9!} \approx 0.00000001$$

△

Tabel 7.3 berikut menunjukkan pemilihan dua parameter (yaitu nilai  $R$  dan  $N$ ) akan memengaruhi nilai akurasi dari hampiran  $e^x \approx P_N(x)$  pada interval  $|x| \leq R$ . Galat terkecil terjadi ketika nilai  $N$  terbesar dan  $R$  terkecil.

**Tabel 7.3:** Nilai batas galat untuk hampiran  $e^x \approx P_N(x)$  pada interval  $|x| \leq R$

	$R = 2$ $ x  \leq 2$	$R = 1.5$ $ x  \leq 1.5$	$R = 1$ $ x  \leq 1$	$R = 0.5$ $ x  \leq 0.5$
$e^x \approx P_5(x)$	0.65680499	0.07090172	0.00377539	0.00003578
$e^x \approx P_6(x)$	0.18765857	0.01519323	0.00053934	0.00000256
$e^x \approx P_7(x)$	0.04691464	0.00284873	0.00006742	0.00000016
$e^x \approx P_8(x)$	0.01042548	0.00047479	0.00000749	0.00000001

### Metode Evaluasi Fungsi Polinomial

Terdapat beberapa cara untuk mengevaluasi suatu fungsi polinomial. Misalkan, terdapat fungsi

$$f(x) = (x - 1)^8$$

Evaluasi fungsi  $f$  akan membutuhkan fungsi eksponensial. Fungsi  $f$  juga dapat diekspansi menjadi pangkat dari  $x$  seperti berikut.

$$\begin{aligned} f(x) &= (x - 1)^8 \\ &= x^8 - 8x^7 + 28x^6 - 56x^5 + 70x^4 - 56x^3 + 28x^2 - 8x + 1 \end{aligned} \tag{7.5}$$

Untuk mengevaluasi fungsi polinomial dengan menghindari bentuk pangkat, cara lain yang dapat digunakan adalah metode Horner atau *nested multiplication*. Dengan metode Horner, Persamaan 7.5 dapat ditulis seperti berikut.

$$f(x) = (((((x - 8)x + 28)x - 56)x + 70)x - 56)x + 28)x - 8)x + 1 \tag{7.6}$$

Jadi, evaluasi fungsi polinomial dengan metode Horner membutuhkan 7 operasi perkalian dan 8 operasi penjumlahan atau pengurangan. Dengan demikian, kebutuhan evaluasi fungsi polinomial menggunakan operasi eksponensial dapat dihindari.

## 7.2 Interpolasi

Pada subbab sebelumnya, telah dipelajari bahwa deret Taylor dapat digunakan untuk menghampiri fungsi  $f(x)$ . Informasi yang dibutuhkan untuk menyusun fungsi hampiran polinomial Taylor adalah nilai dari  $f$  dan turunannya pada  $x_0$ . Kekurangan metode ini adalah turunan ordo-tinggi harus diketahui. Namun, sering kali turunan tersebut tak terdefinisi ataupun sulit untuk dihitung. Dengan demikian, diperlukan metode lain untuk menghampiri fungsi  $f(x)$  dengan fungsi polinomial tanpa perlu menggunakan turunan.

Misalkan, diketahui fungsi  $y = f(x)$  dengan  $N + 1$  titik, yaitu  $(x_0, y_0), \dots, (x_N, y_N)$ , dimana nilai  $x_k$  menyebar pada interval  $[a, b]$  dan memenuhi

$$a \leq x_0 < x_1 < \dots < x_N \leq b \quad \text{dan} \quad y_k = f(x_k)$$

Fungsi polinomial  $P(x)$  dengan derajat  $N$  dapat disusun melewati  $N + 1$  titik tersebut. Metode ini hanya memerlukan nilai numerik dari  $x_k$  dan  $y_k$ . Oleh karena itu, turunan fungsi ordo-tinggi tidak lagi diperlukan. Fungsi polinomial  $P(x)$  dapat digunakan untuk menghampiri fungsi  $f(x)$  pada interval  $[a, b]$ .

Pada sudut pandang lain, jika fungsi  $f(x)$  hanya diketahui berupa pasangan titik terurut  $(x_k, y_k)$ , maka diperlukan metode selain deret Taylor untuk menghampiri fungsi  $f(x)$  berupa persamaan garis. Jika terdapat galat yang signifikan pada data titik terurut, maka metode yang digunakan untuk memperoleh fungsi hampiran adalah percocokan kurva yang akan dibahas pada Bab 8. Sebaliknya, jika diketahui titik  $(x_k, y_k)$  memiliki akurasi yang tinggi, maka fungsi polinomial  $y = P(x)$  harus melewati semua titik yang diketahui. Jika nilai  $x_0 \leq x \leq x_N$  maka hampiran  $P(x)$  disebut sebagai nilai **interpolasi**. Sementara itu, jika nilai  $x \leq x_0$  atau  $x > x_N$  maka hampiran  $P(x)$  disebut sebagai nilai **ekstrapolasi**. Hampiran polinomial digunakan untuk merancang algoritma *software* sebagai hampiran suatu fungsi, untuk turunan dan integral numerik, dan membuat kurva yang melewati beberapa titik. Metode paling dasar yang dapat digunakan untuk membentuk persamaan garis interpolasi adalah menggunakan sistem persamaan linear.

**Contoh 7.3 :** Carilah koefisien polinomial  $P(x) = A + Bx + Cx^2 + Dx^3$  yang melewati empat titik, yaitu  $(1, 1.06)$ ,  $(2, 1.12)$ ,  $(3, 1.34)$ , dan  $(5, 1.78)$ . Gunakan hampiran polinomial yang diperoleh untuk menghitung  $P(4)$  dan  $P(5.5)$ .

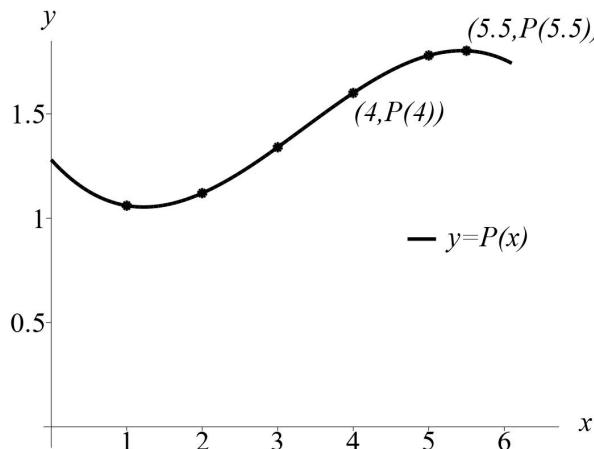
**Solusi :** Berdasarkan titik-titik yang diketahui, didapatkan persamaan linear untuk setiap  $x = 1, 2, 3, 5$  seperti berikut.

$$\begin{aligned} \text{Ketika } x = 1 : \quad A + 1B + 1C + 1D &= 1.06 \\ \text{Ketika } x = 2 : \quad A + 2B + 4C + 8D &= 1.12 \\ \text{Ketika } x = 3 : \quad A + 3B + 9C + 27D &= 1.34 \\ \text{Ketika } x = 5 : \quad A + 5B + 25C + 125D &= 1.78 \end{aligned}$$

Dengan metode penyelesaian sistem persamaan linear pada Bab sebelumnya, didapatkan nilai koefisien  $A = 1.28$ ,  $B = -0.4$ ,  $C = 0.2$ , dan  $D = 0 - 0.2$ . Jadi, fungsi polinomial yang melewati keempat titik yang diberikan adalah  $P(x) = 1.28 - 0.4x + 0.2x^2 - 0.02x^3$ . Dengan metode Horner, hasil interpolasi  $P(4)$  dan ekstrapolasi  $P(5.5)$  adalah

$$\begin{aligned} P(x) &= 1.28 - 0.4x + 0.2x^2 - 0.02x^3 \\ &= 1.28 - x(0.4 + x(0.2 - 0.02x)) \\ P(4) &= 1.28 - 4(0.4 + 4(0.2 - 0.02(4))) \\ &= 1.6 \\ P(5.5) &= 1.28 - 5.5(0.4 + 5.5(0.2 - 0.02(5.5))) \\ &= 1.8025 \end{aligned}$$

Untuk lebih jelasnya, Gambar 7.2 menunjukkan grafik fungsi  $P(x)$  terhadap titik-titik yang diketahui serta hasil interpolasi dan ekstrapolasi.  $\triangle$



**Gambar 7.2:** Hampiran polinomial  $P(x) = 1.28 - 0.4x + 0.2x^2 - 0.02x^3$ .

Contoh selanjutnya menunjukkan hampiran polinomial  $P(x)$  dapat digunakan untuk menghampiri grafik fungsi  $f(x)$ .

**Contoh 7.4 :** Diberikan fungsi  $f(x) = \ln(1+x)$  dan polinomial

$$P(x) = 0.02957x^5 - 0.12895x^4 + 0.28250x^3 - 0.48908x^2 + 0.99911x$$

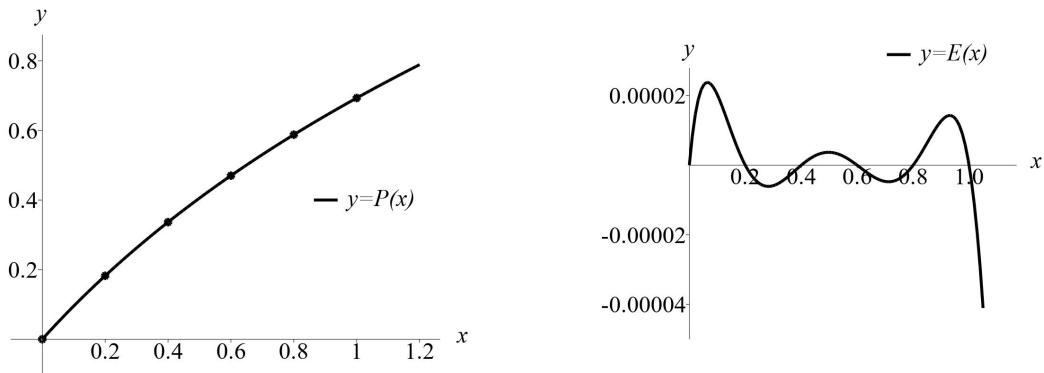
merupakan hampiran dari  $f(x)$  berdasarkan titik  $x_k = 0, 0.2, 0.4, 0.6, 0.8, 1$  yang dihitung dengan cara yang sama seperti pada Contoh 7.3. Hampiran  $P(x) \approx \ln(1+x)$  tersebut memiliki deskripsi seperti berikut.

1.  $P(x_k) = f(x_k)$  pada setiap titik yang diberikan. (Lihat Tabel 7.4)
2. Grafik fungsi  $y = P(x)$  tampak identik dengan  $f(x) = \ln(1+x)$ , dimana grafik  $P(x)$  berada 'di atas' grafik fungsi  $f(x)$  (Lihat Gambar 7.3a).
3. Pada interval  $[0, 1]$ , galat maksimum hampiran  $P(x) \approx \ln(1+x)$  berada pada titik  $x = 0.06472456$ , yaitu  $|\text{galat}| \leq 0.00002385$ . (Lihat Gambar 7.3b)

Perhatikan bahwa pada titik  $x_k$  nilai  $f(x_k) = P(x_k)$ , sehingga  $E(x_k) = 0$  pada titik tersebut. Oleh karena itu, pada gambar 7.3b, grafik  $E(x)$  memotong sumbu- $x$  pada titik  $x_k$ .  $\triangle$

**Tabel 7.4:** Nilai  $f(x) = \ln(1+x)$  dan hampirannya  $P(x) \approx \ln(1+x)$  beserta galatnya pada interval  $[0, 1]$ .

$x$	$f(x) = \ln(1+x)$	$P(x)$	galat
0.0	0.000000	0.000000	0.00000000
0.1	0.095310	0.095290	0.00002030
0.2	0.182322	0.182322	-0.00000000
0.3	0.262364	0.262370	-0.00000588
0.4	0.336472	0.336472	-0.00000000
0.5	0.405465	0.405461	0.00000372
0.6	0.470004	0.470004	-0.00000000
0.7	0.530628	0.530633	-0.00000467
0.8	0.587787	0.587787	-0.00000000
0.9	0.641854	0.641841	0.00001271
1.0	0.693147	0.693147	-0.00000000

(a) Grafik  $y = P(x)$ .(b) Grafik galat  $y = E(x) = \ln(1+x) - P(x)$ .**Gambar 7.3:** Grafik hampiran polinomial  $P(x)$  dan galatnya  $E(x)$ .

### 7.2.1 Interpolasi Lagrange

Interpolasi berarti memperkirakan nilai fungsi  $f(x)$  yang tidak diketahui dengan mengambil rata-rata terboboti dari nilai fungsi yang diketahui di titik sekitarnya. Suatu interpolasi linear akan mengambil garis lurus yang melewati dua titik. Misalkan terdapat dua titik  $(x_0, y_0)$  dan  $(x_1, y_1)$ , maka dua titik tersebut memiliki kemiringan

$$m = \frac{y_1 - y_0}{x_1 - x_0}$$

Dengan demikian, persamaan garis yang ditarik dari kedua titik yaitu  $y = m(x - x_0) + y_0$  dapat ditulis menjadi

$$y = P(x) = y_0 + (y_1 - y_0) \frac{x - x_0}{x_1 - x_0} \quad (7.7)$$

Jika  $P(x)$  dievaluasi pada  $x_0$  dan  $x_1$ , maka akan menghasilkan nilai  $y_0$  dan  $y_1$ .

$$\begin{aligned} P(x_0) &= y_0 + (y_1 - y_0)(0) = y_0 \\ P(x_1) &= y_0 + (y_1 - y_0)(1) = y_1 \end{aligned}$$

Seorang matematikawan bernama Joseph Louis Lagrange menggunakan metode yang sedikit berbeda untuk mencari fungsi polinomial. Dia menyatakan bahwa Persamaan 7.7 dapat ditulis dalam bentuk

$$y = P_1(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0} \quad (7.8)$$

Pembagian pada Persamaan 7.8 dapat dinotasikan dengan

$$\begin{aligned} L_{1,0}(x) &= \frac{x - x_1}{x_0 - x_1} \\ L_{1,1}(x) &= \frac{x - x_0}{x_1 - x_0} \end{aligned}$$

Bentuk  $L_{1,0}(x)$  dan  $L_{1,1}(x)$  disebut sebagai koefisien polinomial Lagrange berdasarkan titik  $x_0$  dan  $x_1$ . Dalam notasi sigma, Persamaan 7.8 dapat ditulis menjadi

$$P_1(x) = \sum_{k=0}^1 y_k L_{1,k}(x) \quad (7.9)$$

Seperti Persamaan 7.7, jika  $P(x)$  pada Persamaan 7.8 dievaluasi di  $x_0$  dan  $x_1$ , maka akan menghasilkan nilai  $y_0$  dan  $y_1$  seperti berikut.

$$\begin{aligned}P_1(x_0) &= y_0(1) + y_1(0) = y_0 \\P_1(x_1) &= y_0(0) + y_1(1) = y_1\end{aligned}$$

Dapat dilihat bahwa nilai  $P_1(0)$  dan  $P_1(1)$  didapatkan dari nilai  $y_0$  dan  $y_1$  yang diboboti. Jika  $P_1(x)$  digunakan untuk menghampiri  $f(x)$  dengan  $x \in [x_0, x_1]$ , maka hampiran disebut sebagai **interpolasi**. Sebaliknya, jika  $x < x_0$  atau  $x > x_1$  maka  $P_1(x)$  disebut **ekstrapolasi**.

**Contoh 7.5 :** Diberikan fungsi  $f(x) = \cos(x)$  pada interval  $[0, 1.2]$ .

1. Gunakan titik  $x_0 = 0$  dan  $x_1 = 1.2$  untuk menyusun interpolasi polinomial  $P_1(x)$ .
2. Gunakan titik  $x_0 = 0.2$  dan  $x_1 = 1$  untuk menyusun interpolasi polinomial  $Q_1(x)$ .

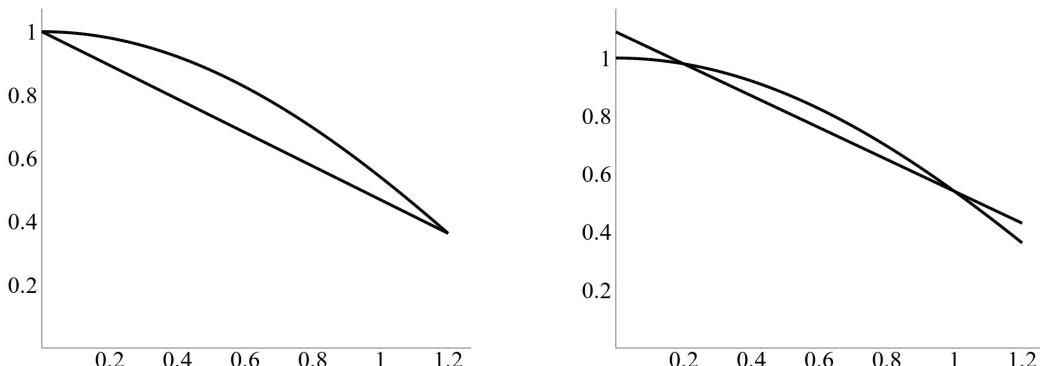
**Solusi :** Dengan Persamaan 7.8, gunakan titik  $x_0 = 0$  dan  $x_1 = 1.2$  serta nilai  $f(x_0) = 1$  dan  $f(x_1) = 0.36236$ , sehingga didapatkan fungsi  $P_1(x)$  sebagai berikut.

$$\begin{aligned}P_1(x) &= 1.00000 \frac{x - 1.2}{0 - 1.2} + 0.36236 \frac{x - 0}{1.2 - 0} \\&= -0.83333(x - 1.2) + 0.30197(x - 0) \\&= -0.53136x + 1\end{aligned}$$

Gunakan titik  $x_0 = 0.2$  dan  $x_1 = 1$  serta nilai  $f(x_0) = 0.98007$  dan  $f(x_1) = 0.54030$ , sehingga didapatkan fungsi  $Q_1(x)$  sebagai berikut.

$$\begin{aligned}Q_1(x) &= 0.98007 \frac{x - 1}{0.2 - 1} + 0.54030 \frac{x - 0.2}{1 - 0.2} \\&= -1.22508(x - 1) + 0.67538(x - 0.2) \\&= -0.54970x + 1.09\end{aligned}$$

Gambar 7.4 menunjukkan perbandingan antara grafik  $f(x)$  terhadap hampiran polinomial  $P_1(x)$  dan  $Q_1(x)$ . Perhitungan numerik pada Tabel 7 menunjukkan bahwa  $Q_1(x)$  memiliki nilai galat yang lebih sedikit pada titik  $x_k \in [0.1, 1.1]$ . Galat terbesar untuk kedua hampiran terjadi saat  $x_k = 0.6$ , yaitu  $f(0.6) - P_1(0.6) = 0.144157$  dan berkurang menjadi  $f(0.6) - Q_1(0.6) = 0.065151$  menggunakan  $Q_1(x)$ .  $\triangle$



(a) Perbandingan grafik  $f(x)$  terhadap  $P_1(x)$       (b) Perbandingan grafik  $f(x)$  terhadap  $Q_1(x)$   
**Gambar 7.4:** Perbandingan grafik  $f(x)$  terhadap hampiran polinomial  $P_1(x)$  dan  $Q_1(x)$  pada interval  $[0, 1.2]$ .

**Tabel 7.5:** Perbandingan fungsi  $f(x) = \cos(x)$  terhadap hampiran linear  $P_1(x)$  dan  $Q_1(x)$ 

$x_k$	$f(x_k)$	$P_1(x)$	$f(x_k) - P_1(x_k)$	$Q_1(x)$	$f(x_k) - Q_1(x_k)$
0.0	1.000000	1.000000	0.000000	1.090008	-0.090008
0.1	0.995004	0.946863	0.048141	1.035037	-0.040033
0.2	0.980067	0.893726	0.086340	0.980067	0.000000
0.3	0.955336	0.840589	0.114747	0.925096	0.030240
0.4	0.921061	0.787453	0.133608	0.870126	0.050935
0.5	0.877583	0.734316	0.143267	0.815155	0.062428
0.6	0.825336	0.681179	0.144157	0.760184	0.065151
0.7	0.764842	0.628042	0.136800	0.705214	0.059628
0.8	0.696707	0.574905	0.121802	0.650243	0.046463
0.9	0.621610	0.521768	0.099842	0.595273	0.026337
1.0	0.540302	0.468631	0.071671	0.540302	0.000000
1.1	0.453596	0.415495	0.038102	0.485332	-0.031736
1.2	0.362358	0.362358	-0.000000	0.430361	-0.068003

Dengan memperumum Persamaan 7.9, hampiran polinomial  $P_N(x)$  dengan derajat paling tinggi  $N$  dapat disusun menggunakan  $N + 1$  titik yaitu  $(x_0, y_0), (x_1, y_1), \dots, (x_N, y_N)$  dan memiliki bentuk umum

$$P_N(x) = \sum_{k=0}^N y_k L_{N,k}(x) \quad (7.10)$$

dimana  $L_{N,k}(x)$  adalah koefisien polinomial Lagrange, yaitu

$$L_{N,k}(x) = \frac{(x - x_0) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_N)}{(x_k - x_0) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_N)}$$

Untuk mempermudah mengingat notasi  $L_{N,k}$ , bentuk  $(x - x_k)$  tidak muncul pada pembilang dan  $(x_k - x_k)$  tidak muncul pada penyebut. Dalam notasi perkalian, dapat ditulis sebagai berikut.

$$L_{N,k}(x) = \frac{\prod_{\substack{j=0 \\ j \neq k}}^N (x - x_j)}{\prod_{\substack{j=0 \\ j \neq k}}^N (x_k - x_j)}$$

Hampiran polinomial Lagrange kuadratik  $P_2(x)$  dapat disusun dengan mengekspansi Persamaan 7.10 menggunakan 3 titik  $(x_0, y_0), (x_1, y_1)$ , dan  $(x_2, y_2)$ , yaitu

$$P_2(x) = y_0 \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + y_1 \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + y_2 \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \quad (7.11)$$

Dengan cara yang sama, hampiran polinomial Lagrange kubik  $P_3(x)$  dapat disusun menggunakan 4 titik  $(x_0, y_0), (x_1, y_1), (x_2, y_2)$ , dan  $(x_3, y_3)$ , yaitu

$$\begin{aligned} P_3(x) &= y_0 \frac{(x - x_1)(x - x_2)(x - x_3)}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} + y_1 \frac{(x - x_0)(x - x_2)(x - x_3)}{(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)} \\ &\quad + y_2 \frac{(x - x_0)(x - x_1)(x - x_3)}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} + y_3 \frac{(x - x_0)(x - x_1)(x - x_2)}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)} \end{aligned} \quad (7.12)$$

**Contoh 7.6 :** Diberikan fungsi  $f(x) = \cos(x)$  pada interval  $[0, 1.2]$ .

1. Gunakan titik  $x_0 = 0, x_1 = 0.6$ , dan  $x_2 = 1.2$  untuk menyusun interpolasi polinomial kuadratik  $P_2(x)$ .
2. Gunakan titik  $x_0 = 0, x_1 = 0.4, x_2 = 0.8$ , dan  $x_3 = 1.2$  untuk menyusun interpolasi polinomial kubik  $P_3(x)$ .

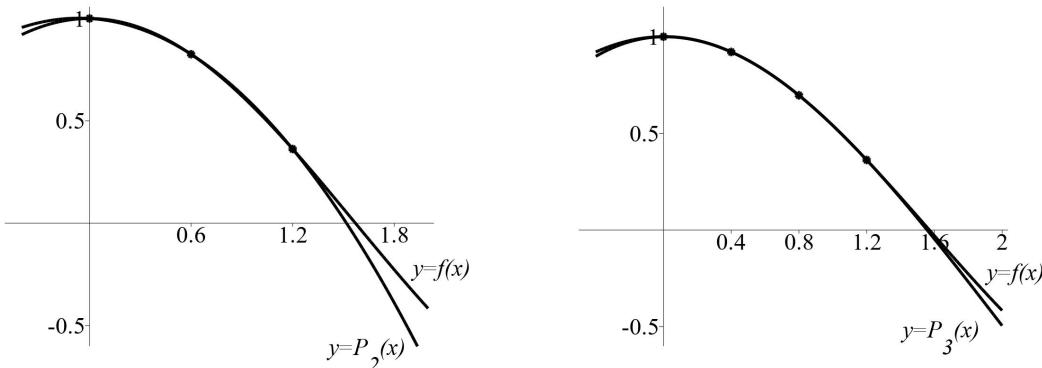
**Solusi :** Dengan Persamaan 7.11, gunakan titik  $x_0 = 0, x_1 = 0.6$ , dan  $x_2 = 1.2$  serta nilai  $y_0 = f(x_0) = 1.00000, y_1 = f(x_1) = 0.82534$ , dan  $y_2 = f(x_2) = 0.36236$  sehingga didapatkan fungsi  $P_2(x)$  sebagai berikut.

$$\begin{aligned} P_2(x) &= 1.00000 \frac{(x-0.6)(x-1.2)}{(0-0.6)(0-1.2)} + 0.82534 \frac{(x-0)(x-1.2)}{(0.6-0)(0.6-1.2)} \\ &\quad + 0.36236 \frac{(x-0)(x-0.7)}{(1.2-0)(1.2-0.6)} \\ &= -0.400435x^2 - 0.050846x + 1 \end{aligned}$$

Dengan Persamaan 7.12, gunakan titik  $x_0 = 0, x_1 = 0.4, x_2 = 0.8$ , dan  $x_3 = 1.2$  serta nilai  $y_0 = 1.00000, y_1 = 0.92106, y_2 = 0.69671$ , dan  $y_3 = 0.36236$  sehingga didapatkan fungsi  $P_3(x)$  sebagai berikut.

$$\begin{aligned} P_3(x) &= 1.00000 \frac{(x-0.4)(x-0.8)(x-1.2)}{(0-0.4)(0-0.8)(0-1.2)} + 0.92106 \frac{(x-0)(x-0.8)(x-1.2)}{(0.4-0)(0.4-0.8)(0.4-1.2)} \\ &\quad + 0.69671 \frac{(x-0)(x-0.4)(x-1.2)}{(0.8-0)(0.8-0.4)(0.8-1.2)} + 0.3624 \frac{(x-0)(x-0.4)(x-0.8)}{(1.2-0)(1.2-0.4)(1.2-0.8)} \\ &= 0.092241x^3 - 0.565112x^2 + 0.013939x + 1 \end{aligned}$$

Grafik perbandingan fungsi  $y = \cos(x)$  terhadap hampiran polinomial  $P_2(x)$  dan  $P_3(x)$  masing-masing dapat dilihat pada Gambar 7.5.  $\triangle$



(a) Perbandingan grafik  $f(x)$  terhadap  $P_2(x)$       (b) Perbandingan grafik  $f(x)$  terhadap  $P_3(x)$   
**Gambar 7.5:** Perbandingan antara grafik  $f(x)$  terhadap hampiran polinomial  $P_2(x)$  dan  $P_3(x)$ .

## 7.2.2 Interpolasi Newton

Dalam mencari hampiran polinomial yang sesuai, sering kali dibutuhkan beberapa hampiran polinomial seperti  $P_1(x), P_2(x), \dots, P_N(x)$  kemudian dari beberapa polinomial

tersebut dipilih satu yang paling cocok. Jika interpolasi Lagrange digunakan, maka diperlukan waktu komputasi yang cukup lama karena tidak terdapat hubungan antara  $P_{N-1}(x)$  dan  $P_N(x)$ . Setiap polinomial harus dibangun satu per satu, sehingga untuk menghitung polinomial berderajat tinggi, dibutuhkan banyak proses komputasi.

Pendekatan lain yang dapat digunakan untuk membentuk hampiran polinomial adalah menggunakan interpolasi Newton. Interpolasi Newton memiliki pola rekursif yaitu

$$\begin{aligned}
 P_1(x) &= a_0 + a_1(x - x_0) \\
 P_2(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \\
 P_3(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2) \\
 &\vdots \\
 P_N(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2) \\
 &\quad + \cdots + a_N(x - x_0) \dots (x - x_{N-1})
 \end{aligned} \tag{7.13}$$

Berdasarkan persamaan 7.13, diperoleh hubungan antara polinomial  $P_{N-1}(x)$  dan  $P_N(x)$ , yaitu

$$P_N(x) = P_{N-1}(x) + a_N(x - x_0)(x - x_1) \dots (x - x_{N-1}) \tag{7.14}$$

$P_N(x)$  disebut sebagai polinomial Newton dengan  $N$  titik tengah, yaitu  $a_0, a_1, a_2, \dots, a_{N-1}$ . Persamaan tersebut diperoleh dari penjumlahan dari faktor linear hingga

$$a_N(x - x_0)(x - x_1)(x - x_2) \dots (x - x_{N-1})$$

sehingga penyederhanaan bentuk penjumlahan pada persamaan  $P_N(x)$  akan menghasilkan polinom berderajat  $\leq N$ .

**Definisi 7.1 Selisih Terbagi:** Selisih terbagi dari suatu fungsi  $f(x)$  didefinisikan sebagai berikut.

$$\begin{aligned}
 f[x_k] &= f(x_k) \\
 f[x_{k-1}, x_k] &= \frac{f[x_k] - f[x_{k-1}]}{x_k - x_{k-1}} \\
 f[x_{k-2}, x_{k-1}, x_k] &= \frac{f[x_{k-1}, x_k] - f[x_{k-2}, x_{k-1}]}{x_k - x_{k-2}} \\
 f[x_{k-3}, x_{k-2}, x_{k-1}, x_k] &= \frac{f[x_{k-2}, x_{k-1}, x_k] - f[x_{k-3}, x_{k-2}, x_{k-1}]}{x_k - x_{k-3}}
 \end{aligned} \tag{7.15}$$

Aturan rekursif yang diperoleh berdasarkan Persamaan 7.15 untuk selisih terbagi dengan ordo yang tinggi adalah

$$f[x_{k-j}, x_{k-j+1}, \dots, x_k] = \frac{f[x_{k-j+1}, \dots, x_k] - f[x_{k-j}, \dots, x_{k-1}]}{x_k - x_{k-j}} \tag{7.16}$$

Hasil dari Persamaan rekursif 7.16 dapat disusun menjadi suatu matriks seperti yang dapat dilihat pada Tabel 7.6 berikut. ■

**Tabel 7.6:** Tabel selisih terbagi untuk  $y = f(x)$ .

$x_k$	$f[x_k]$	$ST_1 = f[ , ]$	$ST_2 = f[ , , ]$	$ST_3 = f[ , , , ]$
$x_0$	$f[x_0]$			
$x_1$	$f[x_1]$	$f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$		
$x_2$	$f[x_2]$	$f[x_1, x_2] = \frac{f[x_2] - f[x_1]}{x_2 - x_1}$	$f[x_0, x_1, x_2] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0}$	
$x_3$	$f[x_3]$	$f[x_2, x_3] = \frac{f[x_3] - f[x_2]}{x_3 - x_2}$	$f[x_1, x_2, x_3] = \frac{f[x_3, x_2] - f[x_2, x_1]}{x_3 - x_1}$	$f[x_0, x_1, x_2, x_3]$

Koefisien  $a_k$  dari  $P_N(x)$  bergantung pada nilai  $f(x_j)$  untuk  $j = 0, 1, \dots, k$ . Teorema selanjutnya akan menunjukkan bahwa  $a_k$  dapat dihitung menggunakan selisih terbagi, yaitu

$$a_k = f[x_0, x_1, \dots, x_k] \quad (7.17)$$

**Teorema 7.2 :** Misalkan bahwa  $x_0, x_1, \dots, x_N$  merupakan  $N + 1$  bilangan yang berbeda pada selang  $[a, b]$  sedemikian sehingga terdapat tepat satu polinomial  $P_N(x)$  dengan derajat  $\leq N$  yaitu

$$f(x_j) = P_N(x_j) \quad \text{untuk } j = 0, 1, \dots, N$$

dengan bentuk polinomial Newton yaitu

$$P_N(x) = a_0 + a_1(x - x_0) + \dots + a_N(x - x_0)(x - x_1) \dots (x - x_{N-1}) \quad (7.18)$$

dengan  $a_k = f[x_0, x_1, \dots, x_k]$  untuk  $k = 0, 1, \dots, N$ . ■

**Contoh 7.7 :** Diketahui  $f(x) = x^3 - 4x$ . Susunlah tabel selisih terbagi berdasarkan nilai  $x_0 = 1, x_1 = 2, \dots, x_5 = 6$  dan carilah polinomial Newton  $P_3(x)$  berdasarkan  $x_0, x_1, x_2$ , dan  $x_3$ .

**Solusi :** Berdasarkan Definisi 7.1, tabel selisih terbagi untuk  $f(x^3 - 4x)$  berdasarkan node  $x_0 = 1, x_1 = 2, \dots, x_5 = 6$  dapat dilihat pada Tabel 7.7 berikut.

**Tabel 7.7:** Selisih terbagi Contoh 7.7

$x_t$	$f[x_t]$	$ST_1$	$ST_2$	$ST_3$	$ST_4$	$ST_5$
$x_0 = 1$	<u>-3</u>					
$x_1 = 2$	0	<u>3</u>				
$x_2 = 3$	15	15	<u>6</u>			
$x_3 = 4$	48	33	9	<u>1</u>		
$x_4 = 5$	105	57	12	1	<u>0</u>	
$x_5 = 6$	192	87	15	1	0	<u>0</u>

Berdasarkan Tabel 7.7, didapatkan nilai koefisien  $a_0 = -3, a_1 = 3, a_2 = 6$  dan  $a_3 = 1$  untuk  $P_3(x)$  pada diagonal tabel selisih terbagi. Berdasarkan Persamaan 7.18,  $P_3(x)$  dapat ditulis dalam bentuk

$$P_3(x) = -3 + 3(x - 1) + 6(x - 1)(x - 2) + (x - 1)(x - 2)(x - 3)$$

△

**Contoh 7.8 :** Susun suatu tabel selisih terbagi untuk  $f(x) = \cos(x)$  berdasarkan 5 titik  $(k, \cos(k))$  untuk  $k = 0, 1, 2, 3, 4$ . Gunakan tabel tersebut untuk mencari koefisien  $a_k$  dan 4 polinomial  $P_k(x)$  untuk  $k = 1, 2, 3, 4$ .

**Solusi :** Berdasarkan Definisi 7.1, tabel selisih terbagi untuk  $(k, \cos(k))$  untuk  $k = 0, 1, 2, 3, 4$  dengan pembulatan 7 angka desimal dapat dilihat pada Tabel 7.8 berikut.

**Tabel 7.8:** Selisih terbagi Contoh 7.8

$x_t$	$f[x_t]$	$ST_1$	$ST_2$	$ST_3$	$ST_4$
$x_0 = 0$	1.0000000				
$x_1 = 1$	0.5403023	-0.4596977			
$x_2 = 2$	-0.4161468	-0.9564491	-0.2483757		
$x_3 = 3$	-0.9899925	-0.5738457	0.1913017	0.1465592	
$x_4 = 4$	-0.6536436	0.3363489	0.4550973	0.0879318	-0.0146568

Berdasarkan Tabel 7.8 dan Persamaan 7.18, didapatkan polinomial Newton  $P_k(x)$  untuk  $k = 1, 2, 3, 4$  seperti berikut.

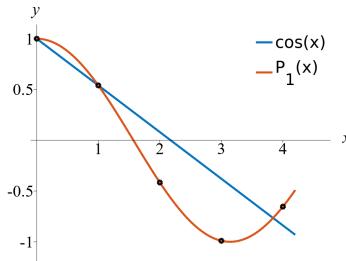
$$P_1(x) = 1.0000000 + (-0.4596977)(x - 0)$$

$$P_2(x) = 1.0000000 + (-0.4596977)(x - 0) + (-0.2483757)(x - 0)(x - 1)$$

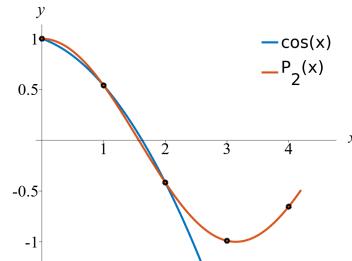
$$P_3(x) = 1.0000000 + (-0.4596977)(x - 0) + (-0.2483757)(x - 0)(x - 1) \\ + (0.1465592)(x - 0)(x - 1)(x - 2)$$

$$P_4(x) = 1.0000000 + (-0.4596977)(x - 0) + (-0.2483757)(x - 0)(x - 1) \\ + (0.1465592)(x - 0)(x - 1)(x - 2) + (-0.0146568)(x - 0)(x - 1)(x - 2)(x - 3)$$

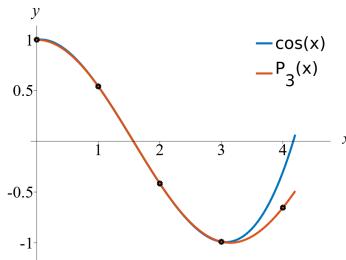
Grafik  $y = \cos(x)$  dan hampiran polinomial  $P_k(x)$  untuk  $k = 1, 2, 3, 4$  ditunjukkan pada Gambar 7.6 berikut.  $\triangle$



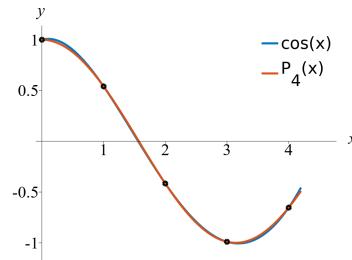
(a)  $y = \cos(x)$  dan  $P_1(x)$



(b)  $y = \cos(x)$  dan  $P_2(x)$



(c)  $y = \cos(x)$  dan  $P_3(x)$



(d)  $y = \cos(x)$  dan  $P_4(x)$

**Gambar 7.6:** Grafik  $y = \cos(x)$  dan hampiran polinomial  $P_k(x)$  untuk  $k = 1, 2, 3, 4$

## 7.3 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Interpolasi Lagrange** berisi program untuk mencari interpolasi pada suatu titik, jika diketahui dataset pada titik lain. Program ini secara *default* berisi 3 masukan, yaitu titik yang ingin dicari hasil interpolasinya  $x$  dan data terurut yang diketahui  $(x_d, y_d)$  serta 1 luaran, yaitu hasil interpolasi dari titik  $x$  yaitu  $y$ . Berikut merupakan program untuk interpolasi Lagrange.

**Algoritma 7.1:** Metode interpolasi Lagrange

```
function lagrange(x, xd, yd)
    % sedikit trik agar bisa menghitung nilai interpolasi pada banyak
    % titik sekaligus. Intinya, tetap diproses satu titik demi titik.
    m=length(x);
    y=zeros(m);
    if m > 1
        for i=1:m
            # proses satu titik demi satu titik
            y[i]=lagrange(x[i], xd, yd);
        end
        return y
    end
    % periksa jumlah titik dan tentukan derajat polinom
    ntitik = length(xd);
    n = ntitik-1; # derajat maksimum, sesuai jumlah titik yang ada
    # hitung L_{n,k}(x) untuk k=1:(n+1)
    Ln=ones(1,ntitik);
    for i=1:ntitik
        for j=1:ntitik
            if i!=j
                Ln[i] = Ln[i] * (x-xd[j]) / (xd[i]-xd[j]);
            end
        end
    end
    % hitung P_N(x)
    y = 0;
    for i=1:ntitik
        y = y + yd[i]*Ln[i];
    end
    return y
end
```

**Metode Interpolasi Newton** berisi program untuk mencari interpolasi pada suatu titik, jika diketahui dataset pada titik lain. Program ini secara *default* berisi 3 masukan, yaitu titik yang ingin dicari hasil interpolasinya yaitu  $x$  dan data terurut yang diketahui  $(x_d, y_d)$  serta 2 luaran, yaitu hasil interpolasi dari titik  $x$  yaitu  $y$  dan tabel beda terbagi  $D$ . Berikut merupakan program untuk interpolasi Newton.

**Algoritma 7.2:** Metode interpolasi Newton

```

function newton(x, xd, yd)
    m=length(x);
    y=zeros(m)
    if m > 1
        for i=1:m
            # proses satu titik demi satu titik
            y[i],D =newton(x[i], xd, yd);
        end
        return y,D
    end
    #% periksa jumlah titik dan tentukan derajat polinom
    ntitik = length(xd);
    #% hitung tabel beda-terbagi (divided-difference)
    D = zeros(ntitik,ntitik)
    D[:,1]=yd;           #% kolom pertama
    for j=2:ntitik      #% kolom ke-2 dan seterusnya
        for k=j:ntitik
            D[k,j] = (D[k,j-1]-D[k-1,j-1]) / (xd[k]-xd[k-j+1]);
        end
    end
    #% hitung interpolasi Newton
    y = D[1,1]; s = 1;
    for i=2:ntitik
        s = s * (x-xd[i-1]);
        y = y + D[i,i]*s;
    end
    return y, D
end

```

**7.3.1 Hampiran Deret Taylor**

Metode pertama yang akan dipelajari untuk membentuk hampiran polinomial dari suatu fungsi adalah menggunakan deret Taylor.

**Contoh 7.9 :** Diberikan fungsi  $f(x) = \exp(x)$ . Berikut merupakan proses pembentukan hampiran polinomial dari fungsi  $f$  dengan derajat  $n = 1, 2, 3, 4$  pada interval  $[-3, 3]$  menggunakan teorema deret Taylor dan galat dari masing-masing hampiran terhadap fungsi  $f$ .

**Langkah 1:** Perumusan hampiran deret taylor dari fungsi  $f(x) = \exp(x)$ . Berdasarkan teorema Taylor, hampiran polinomial dari  $f(x) = \exp(x)$  adalah

$$P_N(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^N}{N!} + \dots$$

**Langkah 2:** Pendefinisian fungsi  $P_N(x)$  dengan masukan yaitu nilai  $x$  dan derajat  $n$ . (petunjuk: gunakan *header*  $y = pn\_exp(x, n)$ ).

```

function pn_exp(x,n)
    y = 0;
    for k = 0:n
        y = y + x.^k./factorial(k);
    end
    return y
end

```

**Langkah 3:** Pendefinisian interval  $x$  yaitu  $[-3, 3]$  dan nilai eksak  $y = \exp(x)$ .

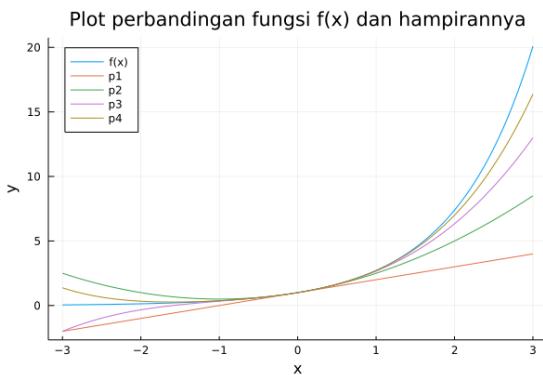
```
Inp: x = -3:0.01:3;
y = exp.(x);
```

**Langkah 4:** Pembuatan plot perbandingan grafik fungsi  $P_1(x)$ ,  $P_2(x)$ ,  $P_3(x)$ , dan  $P_4(x)$  terhadap  $f(x)$ .

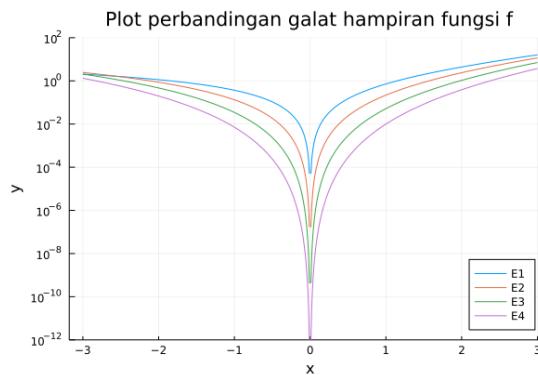
```
Inp: using Plots
Inp: p1 = pn_exp.(x, 1);
      p2 = pn_exp.(x, 2);
      p3 = pn_exp.(x, 3);
      p4 = pn_exp.(x, 4);
Inp: plot(x, y, label = "f(x)")
      plot!(x, p1, label = "p1")
      plot!(x, p2, label = "p2")
      plot!(x, p3, label = "p3")
      plot!(x, p4, label = "p4")
Inp: title!("Plot perbandingan fungsi f(x) dan hampirannya")
      xlabel!("x")
      ylabel!("y")
```

**Langkah 5:** Gambarkan perbandingan galat hampiran  $P_1(x)$ ,  $P_2(x)$ ,  $P_3(x)$ , dan  $P_4(x)$ .

```
Inp: e1 = abs.(y-p1);
      e2 = abs.(y-p2);
      e3 = abs.(y-p3);
      e4 = abs.(y-p4);
Inp: plot(x, e1, label = "E1", yaxis=:log, ylim=(1e-12, 100))
      plot!(x, e2, label = "E2", legend=:bottomright)
      plot!(x, e3, label = "E3")
      plot!(x, e4, label = "E4")
Inp: title!("Plot perbandingan galat hampiran fungsi f")
      xlabel!("x")
      ylabel!("y")
```



(a) Perbandingan grafik fungsi  $f(x)$  terhadap  $P_1(x)$ ,  $P_2(x)$ ,  $P_3(x)$ , dan  $P_4(x)$ .



(b) Perbandingan grafik galat hampiran  $P_1(x)$ ,  $P_2(x)$ ,  $P_3(x)$ , dan  $P_4(x)$ .

**Gambar 7.7:** Grafik fungsi hampiran polinomial dari fungsi  $f(x) = \exp(x)$  dengan nilai derajat  $n = 1, 2, 3, 4$  pada interval  $[-3, 3]$  beserta nilai galatnya.

Gambar 7.7a menunjukkan perbandingan dari masing-masing hampiran terhadap fungsi eksaknya yaitu  $f(x) = \exp(x)$ . Semakin tinggi derajat dari hampiran deret Taylor, maka semakin dekat hampiran polinomial mendekati fungsi eksaknya. Untuk lebih jelasnya, Gambar 7.7b menunjukkan perbandingan galat dari masing-masing hampiran terhadap

fungsi eksaknya. Galat dari hampiran polinomial dengan derajat 4 berada paling bawah, yang artinya memiliki galat paling kecil dibandingkan hampiran yang lainnya.

**Soal Latihan:** Diberikan fungsi

$$f(x) = \cos(x)$$

Ulangi langkah-langkah pada Contoh 7.9 untuk membentuk hampiran polinomial dengan derajat  $n = 1, 2, 3, 4$  pada interval  $[-4, 4]$ .

### 7.3.2 Interpolasi Linear

Hampiran polinomial digunakan untuk merancang algoritma *software* sebagai hampiran suatu fungsi, untuk turunan dan integral numerik, dan membuat kurva yang melewati beberapa titik. Metode paling dasar yang dapat digunakan untuk membentuk persamaan garis interpolasi adalah menggunakan sistem persamaan linear.

**Contoh 7.10 :** Carilah koefisien polinomial  $P(x) = A + Bx + Cx^2 + Dx^3$  yang melewati empat titik, yaitu  $(1, 1.06)$ ,  $(2, 1.12)$ ,  $(3, 1.34)$ , dan  $(5, 1.78)$ . Gunakan hampiran polinomial yang diperoleh untuk menghitung  $P(4)$  dan  $P(5.5)$ .

**Langkah 1:** Definisikan SPL yang berkorespondensi dengan titik titik data dan persamaan yang diinginkan. Akan dicari nilai A, B, C, dan D. polinomial  $P(x) = A + Bx + Cx^2 + Dx^3$

Berdasarkan titik-titik yang diketahui, didapatkan persamaan linear untuk setiap  $x = 1, 2, 3, 5$  seperti berikut.

$$\begin{aligned} \text{Ketika } x = 1 : & A + 1B + 1C + 1D = 1.06 \\ \text{Ketika } x = 2 : & A + 2B + 4C + 8D = 1.12 \\ \text{Ketika } x = 3 : & A + 3B + 9C + 27D = 1.34 \\ \text{Ketika } x = 5 : & A + 5B + 25C + 125D = 1.78 \end{aligned}$$

**Langkah 2:** Selesaikan SPL yang diperoleh.

```
Inp: A = [1 1 1 1
          1 2 4 8
          1 3 9 27
          1 5 25 125];
      B = [1.06; 1.12; 1.34; 1.78];
      C = A\B

Out: 4-element Array{Float64,1}:
      1.2799999999999998
      -0.3999999999999997
      0.1999999999999999
      -0.01999999999999999
```

Jadi, fungsi polinomial yang melewati keempat titik yang diberikan adalah

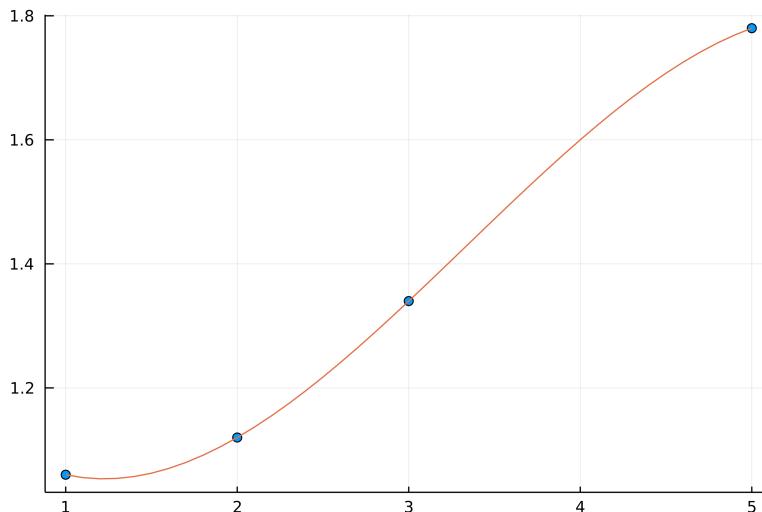
$$P(x) = 1.28 - 0.4x + 0.2x^2 - 0.02x^3$$

**Langkah 3:** Hitung  $P(4)$  dan  $P(5.5)$ .

```
Inp: f(4), f(5.5)

Out: (1.6, 1.8025000000000007)
```





**Gambar 7.8:** Garis interpolasi linear dari Contoh 7.10.

### 7.3.3 Interpolasi Lagrange

Setelah menerapkan deret Taylor untuk fungsi hampiran polinomial, dapat dilihat kelemahan dari hampiran deret Taylor yaitu untuk memperoleh akurasi yang baik, diperlukan derajat deret Taylor yang tinggi. Untuk mengatasi hal tersebut, metode lain yang dapat digunakan untuk menghampiri suatu fungsi adalah menggunakan metode interpolasi Lagrange.

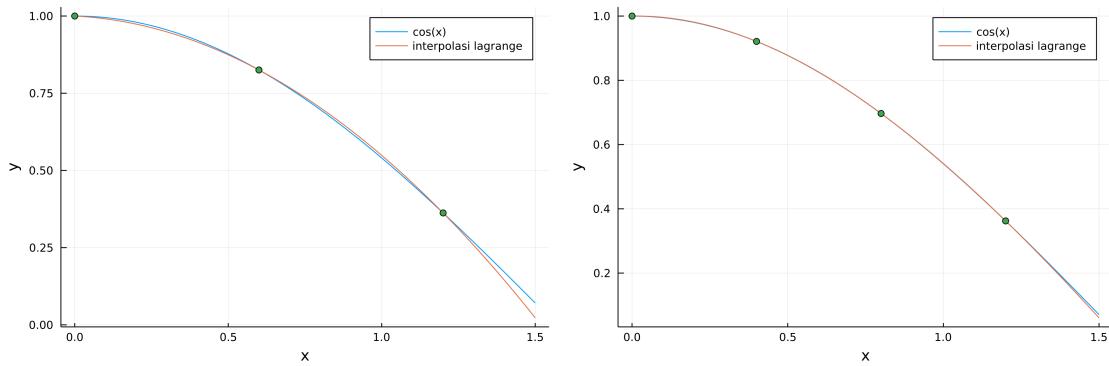
**Contoh 7.11 :** Diberikan fungsi  $f(x) = \cos(x)$  pada interval  $[0, 1.2]$ . Berikut merupakan proses pembentukan hampiran polinomial  $P_2(x)$  dan  $P_3(x)$  dari fungsi  $f$  menggunakan interpolasi Lagrange beserta galat dari masing-masing hampiran.

**Langkah 1:** Pembentukan hampiran  $P_2(x)$  dengan titik  $x_0 = 0$ ,  $x_1 = 0.6$  dan  $x_2 = 1.2$ , serta pembentukan grafik fungsi  $P_2(x)$  pada interval  $[0, 1.5]$ .

```
Inp: f(x) = cos(x);
    xd = [0, 0.6, 1.2];
    yd = f.(xd);
    x = 0:0.01:1.5
    p2 = lagrange(x, xd, yd)
Inp: plot(x, f.(x), label = "cos(x)")
    plot!(x, p2, label = "interpolasi lagrange")
    scatter!(xd, yd, label = :none)
    xlabel!("x")
    ylabel!("y")
```

**Langkah 2:** Pembentukan hampiran  $P_3(x)$  dengan titik  $x_0 = 0$ ,  $x_1 = 0.4$ ,  $x_2 = 0.8$  dan  $x_3 = 1.2$  serta pembentukan grafik fungsi  $P_3(x)$  pada interval  $[0, 1.5]$ .

```
Inp: f(x) = cos(x);
    xd = [0, 0.4, 0.8, 1.2];
    yd = f.(xd);
    x = 0:0.01:1.5
    p3 = lagrange(x, xd, yd)
Inp: plot(x, f.(x), label = "cos(x)")
    plot!(x, p3, label = "interpolasi lagrange")
    scatter!(xd, yd, label = :none)
    xlabel!("x")
    ylabel!("y")
```



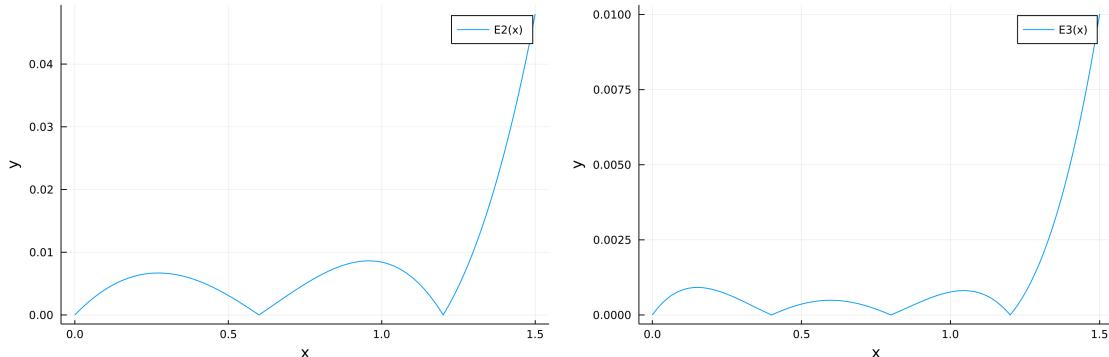
(a) Garis interpolasi  $P_2(x)$  berdasarkan titik  $x_0 = 0, x_1 = 0.6$  dan  $x_2 = 1.2$ .

(b) Garis interpolasi  $P_3(x)$  berdasarkan titik  $x_0 = 0, x_1 = 0.4, x_2 = 0.8$  dan  $x_3 = 1.2$ .

**Gambar 7.9:** Garis interpolasi kuadratik  $P_2(x)$  dan kubik  $P_3(x)$  dari  $f(x) = \cos(x)$ .

**Langkah 3:** Penghitungan galat dari kedua garis interpolasi yang diperoleh.

```
Inp: e2 = abs.(f.(x)-p2);
      plot(x,e2,label = "E2(x)")
      xlabel!("x")
      ylabel!("y")
Inp: e3 = abs.(f.(x)-p3);
      plot(x,e3,label = "E3(x)")
      xlabel!("x")
      ylabel!("y")
```



(a) Gambar fungsi galat  $E_2(x)$

(b) Gambar fungsi galat  $E_3(x)$

**Gambar 7.10:** Gambar fungsi galat  $E_2(x)$  dan  $E_3(x)$

Gambar 7.9 dan Gambar 7.10 menunjukkan bahwa garis interpolasi kubik  $P_3(x)$  lebih mendekati fungsi asli dengan galat kurang dari 0.002 dibandingkan interpolasi kuadratik  $P_2(x)$  dengan galat kurang dari 0.01 pada interval  $[0, 1.2]$ .

**Soal Latihan:** Diberikan fungsi

$$f(x) = \sin(x)$$

Ulangi langkah-langkah pada Contoh 7.11 untuk membentuk hampiran  $P_2(x)$  dan  $P_3(x)$  pada interval  $[0, 1.2]$  kemudian hitung galat masing-masing hampiran tersebut.

### 7.3.4 Interpolasi Newton

Selain interpolasi Lagrange, teknik lain yang dapat digunakan untuk membentuk hampiran polinomial dari suatu fungsi adalah menggunakan interpolasi Newton.

**Contoh 7.12 :** Diberikan fungsi  $f(x) = x^3 - 4x$ . Jika diketahui titik  $x_0 = 1, x_1 = 2, \dots, x_5 = 6$ , maka tabel beda-terbagi untuk interpolasi Newton dapat dihitung menggunakan Program 7.2 dengan cara sebagai berikut.

```
Inp: f(x) = x.^3-4*x;
xd = 1:6;
yd = f(xd);
Inp: # masukkan nilai x sembarang karena yang dibutuhkan
# hanyalah matriks beda-terbagi
ynew,D = newton(1,xd,yd)
D
```

```
Out: 6x6 Array{Float64,2}:
-3.0 0.0 0.0 0.0 0.0 0.0
0.0 3.0 0.0 0.0 0.0 0.0
15.0 15.0 6.0 0.0 0.0 0.0
48.0 33.0 9.0 1.0 0.0 0.0
105.0 57.0 12.0 1.0 0.0 0.0
192.0 87.0 15.0 1.0 0.0 0.0
```

**Soal Latihan:** Tampilkan matriks beda-terbagi, jika diberikan fungsi  $f(x) = \cos(x)$  dan empat titik  $(k, f(k))$  dengan  $k = 0, 1, 2, 3$

**Contoh 7.13 :** Diberikan fungsi

$$f(x) = \cos(x)$$

dan empat titik  $(k, f(k))$  dengan  $k = 0, 1, 2, 3$ . Berikut merupakan langkah-langkah untuk membentuk fungsi interpolasi  $P_1(x)$ ,  $P_2(x)$ , dan  $P_3(x)$  menggunakan metode Newton, serta pembentukan grafik perbandingan fungsi  $f(x)$  dan garis interpolasi pada interval  $[0, 3.1]$ .

**Langkah 1:** Pendefinisian fungsi  $f(x)$  dan urutan titik data  $(x_d, y_d)$  serta interval grafik, yaitu  $x_{\text{new}}$ .

```
Inp: f(x) = cos(x);
xd = 0:3;
yd = f.(xd);
x = 0:0.1:3.1;
```

**Langkah 2:** Penghitungan nilai  $y_{\text{new}}$  dengan interpolasi Newton, yaitu  $P_1(x)$  dengan titik  $x_0$  dan  $x_1$  serta grafik perbandingan fungsi  $f(x)$  dan  $P_1(x)$ .

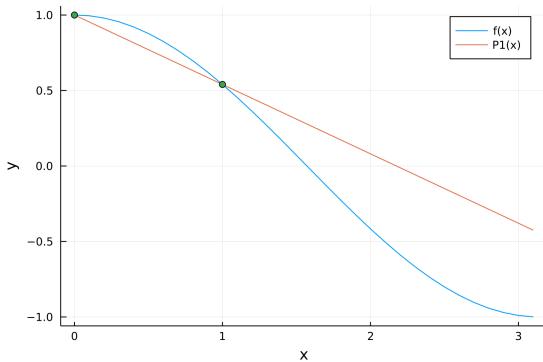
```
Inp: ynew,D = newton(x,xd[1:2],yd[1:2]);
Inp: plot(x,f.(x),label = "f(x)")
plot!(x,ynew,label = "P1(x)")
scatter!(xd[1:2],yd[1:2],label = :none)
xlabel!("x")
ylabel!("y")
```

**Langkah 3:** Penghitungan nilai  $y_{\text{new}}$  dengan interpolasi Newton, yaitu  $P_2(x)$  dengan titik  $x_0, x_1$ , dan  $x_2$  serta grafik perbandingan fungsi  $f(x)$  dan  $P_2(x)$ .

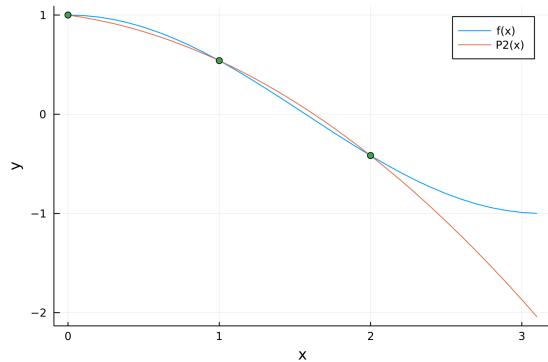
```
Inp: ynew,D = newton(x,xd[1:3],yd[1:3]);
Inp: plot(x,f.(x),label = "f(x)")
      plot!(x,ynew,label = "P2(x)")
      scatter!(xd[1:3],yd[1:3],label = :none)
      xlabel!("x")
      ylabel!("y")
```

**Langkah 4:** Penghitungan nilai  $y_{\text{new}}$  dengan interpolasi Newton, yaitu  $P_3(x)$  dengan titik  $x_0, x_1, x_2$ , dan  $x_3$  serta grafik perbandingan fungsi  $f(x)$  dan  $P_3(x)$ .

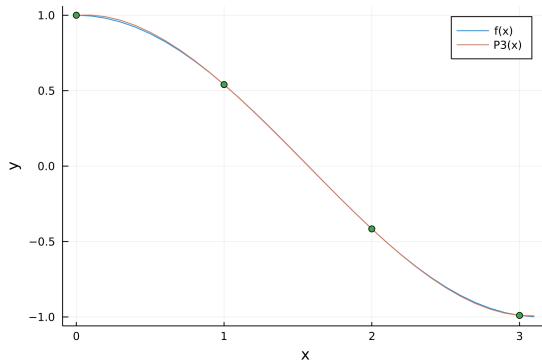
```
Inp: ynew,D = newton(x,xd[1:4],yd[1:4]);
Inp: plot(x,f.(x),label = "f(x)")
      plot!(x,ynew,label = "P3(x)")
      scatter!(xd[1:4],yd[1:4],label = :none)
      xlabel!("x")
      ylabel!("y")
```



(a) Perbandingan fungsi  $f(x)$  dan  $P_1(x)$ .



(b) Perbandingan fungsi  $f(x)$  dan  $P_2(x)$ .



(c) Perbandingan fungsi  $f(x)$  dan  $P_3(x)$ .

**Gambar 7.11:** Plot perbandingan fungsi  $f(x) = \cos(x)$  dan hampiran interpolasi Newton.

Gambar 7.11 menunjukkan plot perbandingan hampiran polinomial yang dihasilkan oleh metode interpolasi Newton, yaitu  $P_1(x)$ ,  $P_2(x)$ , dan  $P_3(x)$  terhadap fungsi eksaknya yaitu  $f(x) = \cos(x)$ .

**Soal Latihan:** Diberikan fungsi  $f(x) = \sin(x)$ . Ulangi Contoh 7.13 untuk membentuk hampiran  $P_1(x)$ ,  $P_2(x)$  dan  $P_3(x)$  dengan titik  $(k, f(k))$  dimana  $k = 0, 1, 2, 3$  lalu gambarkan masing-masing grafik hampiran Newton dan fungsi  $f(x)$  pada interval  $[0, 3.1]$ .

**Contoh 7.14 Soal Kuliah (*Polynomial Wiggle*):** Misalkan diberikan suatu fungsi sebagai berikut.

$$f(x) = \frac{1}{1+12x^2}$$

Dengan mengambil titik  $(x_d, y_d)$  pada  $x_d = -1, -0.8, -0.6, \dots, 0.6, 0.8, 1$  dan  $y_d = f(x_d)$ . Hitunglah

1. Hampiran interpolasi dari  $f(x)$  menggunakan data tersebut.
2. Hampiran interpolasi dari  $f(x)$  dengan membagi data menjadi 2 bagian yaitu  $[-1, 0]$  dan  $[0, 1]$

**Langkah 1:** Hampiran interpolasi dari  $f(x)$  menggunakan data pada selang  $[-1, 1]$  secara langsung dengan metode Newton pada 7.2.

```
Inp: f(x) = 1/(1+12*x^2);
xd = -1:0.2:1;
yd = f.(xd);
x = -1:.01:0;
ynew = newton(x, xd, yd);

plot(x,f.(x),label = "f(x)")
plot!(x,ynew,label = "P10(x)")
scatter!(xd,yd,label = :none)

xlabel!("x")
ylabel!("y")
```

**Langkah 2:** Hampiran interpolasi dari  $f(x)$  dengan membagi data menjadi 2 bagian yaitu  $[-1, 0]$  dan  $[0, 1]$  dengan metode Newton pada 7.2.

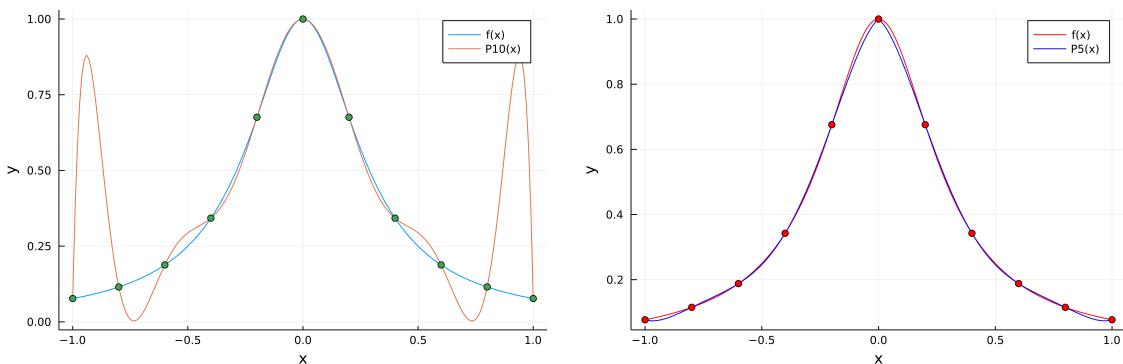
```
Inp: f(x) = 1/(1+12*x^2);
xd = -1:0.2:0; # bagian [-1, 0]
yd = f.(xd);
x = -1:.01:0;
ynew = newton(x, xd, yd);

plot(x,f.(x),color = :red,label = "f(x)")
plot!(x,ynew,color = :blue,label = "P5(x)")
scatter!(xd,yd,color = :red,label = :none)

Inp: xd = 0:0.2:1; # bagian [0,1]
yd = f.(xd);
x = 0:.01:1;
ynew = newton(x, xd, yd);

plot!(x,f.(x),color = :red,label = :none)
plot!(x,ynew,color = :blue,label = :none)
scatter!(xd,yd,color = :red,label = :none)
```

Terjadi ketidaksesuaian yang amat tinggi pada titik mendekati  $x = -1$  dan  $x = 1$ . Ketidaksesuaian itu sering disebut sebagai goncangan polinomial (*polynomial wiggle*). Hal ini terjadi karena proses interpolasi melibatkan fungsi polinomial berderajat terlalu tinggi. Sebagai alternatifnya, derajat fungsi polinomial dapat dipangkas dengan cara mempartisi selang awal menjadi dua atau lebih selang bagian (sub-interval). Berdasarkan Gambar 7.12, hasil interpolasi menggunakan cara kedua jauh lebih akurat daripada menggunakan cara yang pertama.



(a) Tanpa pembagian selang.

(b) Dengan pembagian selang.

**Gambar 7.12:** Perbandingan metode Newton untuk interpolasi kasus pada Contoh 7.14 tanpa dan dengan pembagian selang data.

**Soal Latihan:** Misalkan diberikan suatu fungsi sebagai berikut.

$$f(x) = e^{-\sin(x)}$$

Dengan mengambil titik  $(x_d, y_d)$  pada  $x_d = -2, -1, 0, 1, 2, 3, 4$  dan  $y_d = f(x_d)$ . Hitunglah

1. Hampiran interpolasi dari  $f(x)$  menggunakan data tersebut.
2. Hampiran interpolasi dari  $f(x)$  dengan membagi data menjadi 2 bagian yaitu  $[-2, 1]$  dan  $[1, 4]$

## 7.4 Latihan-latihan

### 7.4.1 Ulasan Materi

1. Bagaimana cara deret Taylor untuk menghampiri nilai dari suatu fungsi yang rumit?
2. Bagaimana cara mendapatkan nilai hampiran polinomial yang akurat berdasarkan deret Taylor?
3. Sebutkan dan jelaskan beberapa cara untuk mengevaluasi suatu fungsi polinomial!
4. Sebutkan dan jelaskan kelebihan serta kekurangan dari hampiran polinomial dari suatu fungsi menggunakan deret Taylor!
5. Sebutkan dan jelaskan kelebihan serta kekurangan dari hampiran polinomial dari suatu fungsi menggunakan metode interpolasi!
6. Apa perbedaan antara interpolasi dan ekstrapolasi?
7. Metode paling dasar untuk membentuk garis interpolasi adalah menggunakan SPL, jelaskan dan berikan contoh bagaimana cara membentuk garis interpolasi adalah menggunakan SPL!
8. Jelaskan tahapan dalam membentuk garis interpolasi Lagrange dan Newton!
9. Manakah di antara pernyataan berikut yang benar/salah?
  - (a) Metode hampiran digunakan untuk mengevaluasi fungsi-fungsi yang tidak bisa dievaluasi secara eksak seperti fungsi  $e^x$ ,  $\ln(x)$ ,  $\sin(x)$ , dan sebagainya.
  - (b) Jika deret yang ditambahkan cukup banyak, maka semakin akurat nilai hampiran yang akan diperoleh.
  - (c) Hampiran polinomial menggunakan metode interpolasi lebih akurat dibandingkan metode deret Taylor.

- (d) Hasil garis interpolasi Lagrange lebih akurat daripada hasil garis interpolasi Newton.
- (e) Metode interpolasi membutuhkan turunan ordo-tinggi untuk memperoleh garis hampiran polinomial.

### 7.4.2 Soal Pemrograman

1. Dengan menggunakan Teorema Taylor,  $f(x) = \cos(x)$  dapat dirumuskan dengan hampiran polinomial, yaitu  $f(x) \approx P_n(x)$ 
  - (a) Rumuskan  $P_n(x)$  dengan menggunakan deret Taylor
  - (b) Hitung nilai  $P_2(0.25)$  dan  $P_6(0.25)$
  - (c) Hitung galat absolut dan relatif jika diketahui  $\cos(0.25) = 0.968912$
  - (d) Jelaskan dan hitung galat pemotongan (*truncation error*) dari hampiran polinomial tersebut.
2. Diberikan beberapa fungsi sebagai berikut:
  - i.  $f(x) = \frac{1}{1-x}$        $P_n(x) = \sum_{k=0}^n x^n$        $x \in (-0.9, 0.9)$
  - ii.  $f(x) = \ln(x+1)$      $P_n(x) = \sum_{k=1}^n (-1)^{k+1} \frac{x^k}{k}$        $x \in (-1, 1)$
  - iii.  $f(x) = \sin(x)$        $P_n(x) = \sum_{k=0}^n \frac{(-1)^k}{(2k+1)!} x^{2k+1}$        $x \in (-10, 10)$

Dari fungsi tersebut, kerjakan untuk masing-masing fungsi:

- (a) Buatlah fungsi hampiran  $P_n(x)$  dengan format
  - i.  $y=pn\_har(x)$
  - ii.  $y=pn\_ln(x)$
  - iii.  $y=pn\_sin(x)$
- (b) Bangkitkan vektor  $x$  pada selang yang ditentukan menggunakan `linspace`.
- (c) Hitung  $f(x)$  pada selang  $x$ .
- (d) Hitung  $P_n(x)$  dengan  $n = 1, 2, \dots, 7$ .
- (e) Hitung nilai error yaitu  $E_n(x) = |f(x) - P_n(x)|$ .
- (f) Plot  $f(x)$  dan  $P_n(x)$  untuk setiap  $n$  dalam satu grafik.
- (g) Plot  $E_n(x)$  untuk setiap  $n$  dalam satu grafik.

3. Hampiran suatu fungsi dapat dilakukan antara lain menggunakan deret Taylor berupa suatu polinomial

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

untuk  $x$  di sekitar  $a$ . Komputasi numerik untuk memperoleh hampiran tersebut dilakukan hanya dengan banyaknya suku berhingga, yaitu

$$f(x) \approx P_N(x) = \sum_{n=0}^N \frac{f^{(n)}(a)}{n!} (x-a)^n$$

Tunjukkan bagaimana banyaknya suku  $N$  mempengaruhi lebar selang hampiran efektif  $P_N(x)$  terhadap fungsi  $f(x)$  berikut:

- (a)  $f(x) = \sin(x)$
- (b)  $f(x) = \exp(x)$

Gunakan  $N = 2, 5$  dan  $10$  untuk masing-masing fungsi.

**Catatan:** Lebar selang hampiran efektif adalah suatu selang yang memiliki selisih (antara  $P_N$  dan  $f$ ) yang *acceptable*, misalkan tak lebih dari suatu toleransi tertentu  $\text{tol} = 10^{-5}$ .

4. Pada praktikum biologi, pertumbuhan suatu bakteri dicatat setiap 3 menit. Data dari praktikum tersebut yaitu

$t$	0	3	6	9
$p(t)$	2	3	4	6

dimana  $t$  menunjukkan waktu (dalam menit) dan  $p(t)$  adalah populasi bakteri pada waktu  $t$  (dalam ratusan). Berapa populasi bakteri pada  $t = 5$  ?

5. Suatu masalah nilai awal  $dy/dt = f(t, y)$  mempunyai nilai solusi di beberapa titik sebagai berikut:

$t$	0	0.1	0.2	0.3	0.4	0.5
$y$	-1	-0.9	-0.79	-0.67	-0.54	-0.41

Buatlah hampiran polinomial dengan interpolasi yang menghubungkan titik-titik tersebut, kemudian gunakan untuk menebak nilai  $y(0.26), y(0.35)$ , dan  $y(0.43)$ .

6. Diberikan data  $x = 1, 2, 3$  dan  $y = f(x)$  dengan  $f(x) = x^3$
- (a) Tuliskan formula interpolasi Lagrange dan Newton untuk  $P_2(x)$ .
  - (b) Bandingkan fungsi  $P_2(x)$  dan  $f(x)$  kemudian hitunglah error  $x \in [1, 3]$ .
  - (c) Hitung nilai galat maksimum dari poin b.
  - (d) Hitung nilai galat maksimum jika  $x \in [1.9, 2.1]$ . Bandingkan dengan poin c !
7. Diberikan fungsi

$$f(x) = 2x^2 e^x + 1$$

Buatlah hampiran polinom berderajat dua atau kurang dengan  $x = 0, 0.5$ , dan 1. Gunakan fungsi hampiran tersebut untuk menebak  $f(0.8)$ .

8. Diberikan suatu fungsi sebagai berikut:

$$f(x) = \frac{1}{1 + 12x^2}$$

Misalkan terdapat  $x$  sebanyak 11 titik diantara selang  $[-1, 1]$  dengan jeda yang sama dan  $y = f(x)$ .

- (a) Buatlah hampiran polinomial  $f_{10}(x)$  menggunakan interpolasi Lagrange atau Newton dari titik-titik tersebut.
  - (b) Bandingkan fungsi yang anda dapat dengan fungsi asli  $f$ .
  - (c) Apakah interpolasi polinom dengan orde tinggi baik digunakan? Jika tidak, bagaimana cara mengatasinya?
9. Suatu masalah nilai awal  $dy/dt = f(t, y)$  mempunyai nilai solusi di beberapa titik sebagai berikut:

$t$	0	0.1	0.2	0.3	0.4	0.5
$y$	-1	-0.9	-0.79	-0.67	-0.54	-0.41

- (a) Buatlah garis interpolasi yang menghubungkan titik-titik tersebut.
- (b) Plot fungsi hasil interpolasi yang anda dapat beserta titik-titik di atas.
- (c) Gunakan fungsi interpolasi yang diperoleh untuk menebak nilai  $y(0.26), y(0.35)$ , dan  $y(0.43)$ .

10. Berikut merupakan data yang diperoleh SHP Chen dan SC Saxena dalam percobaan mereka.

$T$	$e$	$T$	$e$
300	0.024	1200	0.140
400	0.035	1300	0.155
500	0.046	1400	0.170
600	0.056	1500	0.186
700	0.067	1600	0.202
800	0.083	1700	0.219
900	0.097	1800	0.235
1000	0.111	1900	0.252
1100	0.125	2000	0.269

Mereka menemukan bahwa persamaan

$$e(T) = 0.02424 \left( \frac{T}{303.16} \right)^{1.27591}$$

berkorelasi terhadap data dengan akurasi hingga 3 digit desimal.

- (a) Buatlah hampiran polinomial  $p(T)$  yang menghubungkan data di atas dengan interpolasi.  
 (b) Plot  $p(T)$  dan  $e(T)$  dalam satu grafik pada interval [300,2000].  
 (c) Jelaskan gambar yang anda peroleh.
11. Diberikan data nilai  $C$  dari titanium sebagai fungsi dari suhu  $T$  sebagai berikut.

$T$	$C$
605	0.622
685	0.655
725	0.668
765	0.679
825	0.730
855	0.907
875	1.336

- (a) Carilah hampiran polinomial dari data di atas.  
 (b) Perkirakan nilai  $C(645), C(795)$ , dan  $C(845)$ .  
 (c) Bandingkan dengan nilai aslinya yaitu  $C(645) = 0.639, C(795) = 0.694$ , dan  $C(845) = 0.812$ .
12. Tentukan polinom interpolasi  $P_6(x)$  yang melalui semua titik berikut

$X_k$	0	1	2	3	4	5	6
$Y_k$	5	5	3	5	17	45	95

- (a) Gunakan metode interpolasi Lagrange  
 (b) Gunakan metode interpolasi Newton  
 (c) Bandingkan kedua polinom interpolasi tersebut, yaitu kedekatan solusi dan waktu komputasi yang diperlukan  
 (d) Manakah yang lebih baik? Jelaskan!

13. Diberikan fungsi  $f(x) = \frac{1}{\sin(x^2 + x - 1)}$ ,  $-2 \leq x \leq 2$ .
- Bentuk fungsi interpolasi polinom  $P(x)$  untuk menghampiri fungsi  $f(x)$ .
  - Tentukan metode interpolasi polinom dan derajat polinom yang paling tepat! Jelaskan!
14. Diberikan fungsi  $f(x) = 2 \sin(\pi x/6)$ ,  $-4 \leq x \leq 4$ .
- Bentuk fungsi interpolasi polinom  $P(x)$  untuk menghampiri fungsi  $f(x)$ .
  - Tentukan metode interpolasi polinom dan derajat polinom yang paling tepat! Jelaskan!
15. Diberikan fungsi  $f(x) = x^{1/2} \sin(x^{-1})$ ,  $0 \leq x \leq 6$ .
- Bentuk fungsi interpolasi polinom  $P(x)$  untuk menghampiri fungsi  $f(x)$ .
  - Tentukan metode interpolasi polinom dan derajat polinom yang paling tepat! Jelaskan!

---

---

# BAB 8

---

## PENCOCOKAN KURVA (REGRESI)

Suatu penelitian pada bidang ilmu pengetahuan maupun teknik sering kali menghasilkan suatu barisan data pasangan titik  $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$  dengan nilai  $x_k$  yang berbeda. Misalkan, pada suatu praktikum biologi, akan diamati pertumbuhan suatu bakteri. Jumlah bakteri tersebut akan dihitung setiap tiga menit. Setelah 10 menit pengamatan, didapatkan data seperti berikut.

waktu $t$	jumlah bakteri $p(t)$
0	2
3	3
6	4
9	6

dimana  $t$  menunjukkan waktu (dalam menit) dan  $p(t)$  menunjukkan banyaknya populasi bakteri pada waktu  $t$  (dalam ratusan). Pertanyaan yang sering muncul adalah berapakah nilai diantara titik-titik yang diketahui tersebut tanpa harus melakukan pengukuran lagi. Contohnya, pada Tabel di atas, berapakah jumlah populasi bakteri pada waktu 8 menit?

Salah satu tujuan dari komputasi numerik adalah menentukan suatu persamaan garis  $y = f(x)$  yang dapat menghubungkan data-data tersebut. Jika diketahui bahwa data yang diperoleh memiliki akurasi yang tinggi bahkan tidak memiliki galat data dan *noise*, persamaan garis  $y = f(x)$  dapat dibentuk menggunakan metode interpolasi seperti yang telah dibahas pada Bab 7. Sementara itu, jika diketahui bahwa data yang diperoleh memiliki galat dan *noise*, persamaan garis  $y = f(x)$  dapat dibentuk menggunakan metode pencocokan kurva atau sering disebut juga metode regresi. Beberapa penelitian menggunakan alat ukur khusus sedemikian sehingga data yang dihasilkan memiliki akurasi setidaknya lima angka desimal. Namun, banyak penelitian yang terselesaikan hanya memiliki akurasi tiga angka desimal atau kurang. Berdasarkan hal tersebut, terdapat galat pengukuran dan tertekam pada data  $x_k$  dan  $y_k$ , sehingga nilai data sebenarnya  $f(x_k)$  memenuhi persamaan

$$f(x_k) = y_k + e_k \quad (8.1)$$

dengan  $e_k$  merupakan galat pengukuran. Dengan demikian, dibutuhkan diskusi tentang galat pengukuran yaitu

$$e_k = f(x_k) - y_k \quad (8.2)$$

untuk menentukan seberapa dekat garis regresi  $y = f(x)$  dapat mewakili data. Beberapa bentuk *norm* yang dapat digunakan untuk menentukan kedekatan garis regresi terhadap data sebenarnya antara lain.

1. *Maximum Error*

$$E_{\infty}(f) = \max_{1 \leq k \leq n} \{|f(x_k) - y_k|\} \quad (8.3)$$

2. *Average Error*

$$E_1(f) = \frac{1}{N} \sum_{k=1}^N |f(x_k) - y_k| \quad (8.4)$$

3. *Root Mean Squared Error*

$$E_2(f) = \sqrt{\frac{1}{N} \sum_{k=1}^N (f(x_k) - y_k)^2} \quad (8.5)$$

Berikut merupakan contoh penghitungan nilai galat dari suatu garis regresi terhadap data sebenarnya.

**Contoh 8.1 :** Bandingkan nilai *maximum error*, *average error* dan *rms error* dari garis regresi linear  $y = f(x) = 8.6 - 1.6x$  terhadap data berikut.

$x_k$	-1	0	1	2	3	4	5	6
$y_k$	10	9	7	5	4	3	0	-1

**Solusi :** Penghitungan galat  $e_k$  pada setiap titik dapat dilihat pada Tabel berikut.

**Tabel 8.1:** Penghitungan galat  $e_k$  pada setiap titik

$x_k$	$y_k$	$f(x_k)$	$ e_k $	$e_k^2$
-1	10	10.2	0.2	0.04
0	9	8.6	0.4	0.16
1	7	7.0	0.0	0.00
2	5	5.4	0.4	0.16
3	4	3.8	0.2	0.04
4	3	2.2	0.8	0.64
5	0	0.6	0.6	0.36
6	-1	-1.0	0.0	0.00
			2.6	1.40

Dengan demikian, nilai *maximum error*, *average error* dan *rms error* adalah

$$E_{\infty}(f) = \max\{0.2, 0.4, 0.0, 0.4, 0.2, 0.8, 0.6, 0.0\} = 0.8 \quad (8.6)$$

$$E_1(f) = \frac{1}{8}(2.6) = 0.325 \quad (8.7)$$

$$E_2(f) = \left(\frac{1.4}{8}\right)^{1/2} \approx 0.41833 \quad (8.8)$$

Dapat dilihat bahwa nilai *maximum error* memiliki nilai yang paling besar yaitu menunjukkan galat terbesar dari fungsi  $f(x)$  terhadap data  $y_k$ , *average error* secara sederhana menghitung rataan dari nilai galat absolut dan *rms error* sering digunakan apabila sifat statistik dari galat ikut dipertimbangkan.  $\triangle$

## 8.1 Regresi Linear

Misalkan, diberikan data  $\{(x_k, y_k)\}_{k=1}^N$  dengan nilai  $x_k$  berbeda. Fungsi regresi  $y = Ax + B$  adalah suatu garis yang meminimumkan nilai *root mean squared error*  $E_2(f)$ . Nilai  $E_2(f)$  akan minimum, jika dan hanya jika nilai  $N(E_2(f))^2 = \sum_{k=1}^N (Ax_k + B - y_k)^2$  minimum.

**Teorema 8.1 Garis Regresi Linear:** Misalkan bahwa  $\{(x_k, y_k)\}_{k=1}^N$  sebanyak  $N$  titik dengan absis  $\{x_k\}_{k=1}^N$  berbeda. Koefisien garis regresi linear

$$y = Ax + B \quad (8.9)$$

merupakan solusi dari sistem persamaan linear berikut, yang disebut *normal equations*.

$$\begin{aligned} \left( \sum_{k=1}^N x_k^2 \right) A + \left( \sum_{k=1}^N x_k \right) B &= \sum_{k=1}^N x_k y_k \\ \left( \sum_{k=1}^N x_k \right) A + NB &= \sum_{k=1}^N y_k \end{aligned} \quad (8.10)$$

**Bukti** Misalkan, nilai titik data adalah  $(x_k, y_k)$  dan fungsi regresi linear  $y(x_k) = Ax_k + B$  memberikan nilai dugaan  $(x_k, Ax_k + B)$ , maka jarak antara penduga dan nilai sebenarnya adalah  $d_k = |Ax_k + B - y_k|$ . Koefisien  $A$  dan  $B$  dari fungsi regresi dapat dicari dengan meminimumkan jumlah kuadrat dari jarak  $d_k$  seperti berikut.

$$E(A, B) = \sum_{k=1}^N d_k^2 = \sum_{k=1}^N (Ax_k + B - y_k)^2 \quad (8.11)$$

Nilai minimum dari Persamaan 8.11 dapat diperoleh dengan menghitung nilai turunan parsial  $\partial E / \partial A$  dan  $\partial E / \partial B$  sama dengan 0, sehingga didapatkan nilai koefisien  $A$  dan  $B$ . Perhatikan bahwa pada Persamaan 8.11 nilai  $x_k$  dan  $y_k$  merupakan konstanta serta  $A$  dan  $B$  merupakan suatu peubah. Untuk menghitung koefisien  $A$ , anggap  $B$  diketahui, sehingga  $\partial E / \partial A$  dapat didefinisikan sebagai berikut.

$$\frac{\partial E(A, B)}{\partial A} = \sum_{k=1}^N 2(Ax_k + B - y_k)(x_k) = 2 \sum_{k=1}^N (Ax_k^2 + Bx_k - x_k y_k) \quad (8.12)$$

Selanjutnya, anggap koefisien  $A$  diketahui, sehingga  $\partial E / \partial B$  dapat didefinisikan sebagai berikut.

$$\frac{\partial E(A, B)}{\partial B} = \sum_{k=1}^N 2(Ax_k + B - y_k) = 2 \sum_{k=1}^N (Ax_k + B - y_k) \quad (8.13)$$

Dengan mengatur Persamaan 8.12 dan 8.13 sama dengan 0, didapatkan persamaan masing-masing seperti berikut.

$$2 \sum_{k=1}^N (Ax_k^2 + Bx_k - x_k y_k) = 0 \Leftrightarrow \left( \sum_{k=1}^N x_k^2 \right) A + \left( \sum_{k=1}^N x_k \right) B = \sum_{k=1}^N x_k y_k \quad (8.14)$$

$$2 \sum_{k=1}^N (Ax_k + B - y_k) = 0 \Leftrightarrow \left( \sum_{k=1}^N x_k \right) A + NB = \sum_{k=1}^N y_k \quad (8.15)$$

Bukti selesai. ■

**Contoh 8.2 :** Carilah persamaan garis regresi linear dari barisan data berikut.

$x_k$	-1	0	1	2	3	4	5	6
$y_k$	10	9	7	5	4	3	0	-1

**Solusi :** Penjumlahan yang dibutuhkan untuk menghitung *normal equations* seperti pada Persamaan 8.10 dapat dilihat pada Tabel berikut.

**Tabel 8.2:** Penjumlahan yang dibutuhkan untuk menghitung *normal equations*

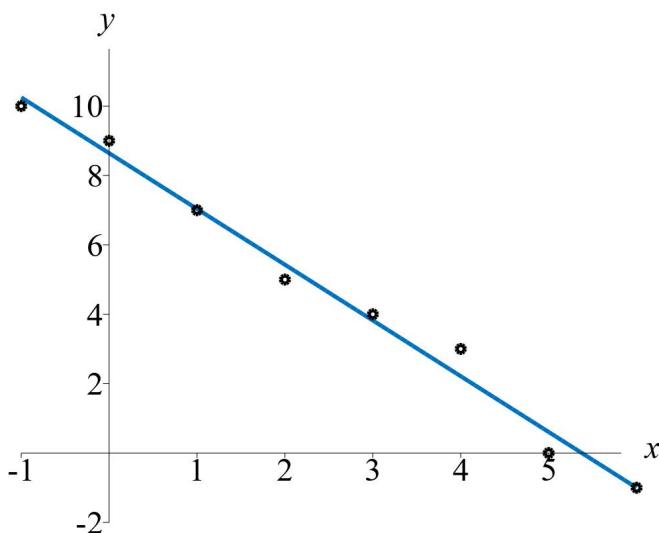
$x_k$	$y_k$	$x_k^2$	$x_k y_k$
-1	10	1	-10
0	9	0	0
1	7	1	7
2	5	4	10
3	4	9	12
4	3	16	12
5	0	25	0
6	-1	36	-6
20	37	92	25

Berdasarkan Persamaan 8.10, *normal equations* untuk menghitung koefisien regresi adalah

$$\begin{aligned} 92A + 20B &= 25 \\ 20A + 8B &= 37 \end{aligned}$$

Solusi dari sistem persamaan linear di atas adalah  $A = -1.6071429$  dan  $B = 8.6428571$ . Dengan demikian, persamaan regresi linear yang didapat adalah (lihat Gambar)

$$y = -1.6071429x + 8.6428571$$



**Gambar 8.1:** Kurva regresi linear  $y = -1.6071429x + 8.6428571$ .



## 8.2 Regresi Pangkat

Pada beberapa situasi, hubungan suatu barisan data dapat dibentuk dalam suatu fungsi pangkat  $f(x) = Ax^M$  dengan  $M$  merupakan konstanta yang telah ditetapkan.

**Teorema 8.2 Regresi Pangkat:** Misalkan bahwa  $(x_k, y_k)_{k=1}^N$  adalah barisan data sebanyak  $N$  titik dengan nilai  $x_k$  yang berbeda. Koefisien  $A$  dari fungsi pangkat  $y = Ax^M$  adalah

$$A = \left( \sum_{k=1}^N x_k^M y_k \right) / \left( \sum_{k=1}^N x_k^{2M} \right) \quad (8.16)$$

**Bukti** Dengan cara yang sama seperti pada pembuktian Teorema 8.1, koefisien  $A$  dari garis regresi  $y = Ax^M$  dapat dicari dengan cara meminimumkan persamaan berikut.

$$E(A) = \sum_{k=1}^N (Ax_k^M - y_k)^2 \quad (8.17)$$

Dengan demikian, turunan dari  $E(A)$  adalah

$$E'(A) = \sum_{k=1}^N (Ax_k^M - y_k)(x_k^M) = 2 \sum_{k=1}^N (Ax_k^{2M} - x_k^M y_k) \quad (8.18)$$

Dalam kasus ini,  $E(A)$  bernilai minimum apabila  $E'(A) = 0$ . Hasil reduksi dari  $E'(A) = 0$  akan menghasilkan Persamaan 8.16. **Bukti selesai.** ■

**Contoh 8.3 :** Pada suatu praktikum fisika, seorang mahasiswa mendapatkan kumpulan data percobaan seperti berikut.

**Tabel 8.3:** Data percobaan pada suatu praktikum fisika.

Waktu, $t_k$	Jarak, $d_k$
0.2	0.1960
0.4	0.7850
0.6	1.7665
0.8	3.1405
1.0	4.9075

Hubungan dari data tersebut adalah  $d = \frac{1}{2}gt^2$  dengan  $d$  merupakan jarak dalam meter dan  $t$  merupakan waktu dalam detik. Carilah koefisien gravitasi  $g$ .

**Solusi :** Tabel 8.4 berikut menunjukkan proses penjumlahan yang diperlukan untuk Persamaan 8.16 dengan pangkat  $M = 2$ . Dengan regresi pangkat, persamaan regresi menjadi  $d = At^2$  dengan  $A = \frac{1}{2}g$ .

**Tabel 8.4:** Penjumlahan yang diperlukan untuk regresi pangkat.

Waktu, $t_k$	Jarak, $d_k$	$d_k t_k^2$	$t_k^4$
0.2	0.1960	0.00784	0.0016
0.4	0.7850	0.12560	0.0256
0.6	1.7665	0.63594	0.1296
0.8	3.1405	2.00992	0.4096
1.0	4.9075	4.90750	1.0000
Total:		7.68680	1.5664

Berdasarkan Persamaan 8.16, koefisien regresi pangkat  $A = 7.6868/1.5664 = 4.9073$ . Dengan demikian, diperoleh persamaan regresi pangkat  $d = 4.9073t^2$ , sehingga koefisien gravitasi  $g = 2A = 9.7146 \text{ m/s}^2$ .  $\triangle$

### 8.3 Linearisasi Data

Sering kali barisan data dari suatu percobaan atau penelitian memiliki nilai galat yang buruk apabila didekati menggunakan regresi linear maupun pangkat. Dengan demikian, diperlukan garis regresi lain seperti persamaan tak linear untuk mendekati data tersebut. Jika hubungan data berupa persamaan tak-linear seperti eksponensial, maka garis persamaan regresi tak-linear dapat dibentuk menggunakan metode linearisasi data.

#### Linearisasi Data untuk $y = Ce^{Ax}$

Misalkan bahwa diberikan titik  $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$  dan ingin dicocokan dengan grafik eksponensial dengan bentuk

$$y = Ce^{Ax} \quad (8.19)$$

Langkah pertama adalah me-logaritma-kan kedua sisi, sehingga Persamaan 8.19 menjadi seperti berikut.

$$\begin{aligned} \ln(y) &= \ln(Ce^{Ax}) \\ \Leftrightarrow \ln(y) &= \ln(e^{Ax}) + \ln(C) \\ \Leftrightarrow \ln(y) &= Ax + \ln(C) \end{aligned} \quad (8.20)$$

Selanjutnya, perubahan peubah didefinisikan sebagai berikut.

$$Y = \ln(y), \quad X = x, \quad \text{dan} \quad B = \ln(C) \quad (8.21)$$

Hasil ini menunjukkan hubungan linear antara peubah baru  $X$  dan  $Y$ , yaitu  $Y = AX + B$ . Titik asli  $(x_k, y_k)$  pada bidang-xy ditransformasikan menjadi titik  $(X_k, Y_k) = (x_k, \ln(y_k))$  pada bidang-XY. Proses ini dinamakan **linearisasi data**. Langkah selanjutnya adalah menghitung nilai koefisien  $A$  dan  $B$  menggunakan *normal equation* pada Persamaan 8.10 yang cocok pada titik  $(X_k, Y_k)$ . Setelah nilai  $A$  dan  $B$  didapatkan, nilai parameter  $C$  pada Persamaan 8.19 adalah  $C = e^B$ .

**Contoh 8.4 :** Gunakan metode linearisasi data untuk mencari kurva regresi eksponensial  $y = Ce^{Ax}$  untuk 5 titik data, yaitu  $(0, 1.5), (1, 2.5), (2, 3.5), (3, 5.0)$ , dan  $(4, 7.5)$ .

**Solusi :** Dengan transformasi pada Persamaan 8.21, titik data awal ditransformasikan menjadi titik  $(X_k, Y_k)$  sebagai berikut.

**Tabel 8.5:** Proses linearisasi data untuk membentuk kurva regresi  $y = Ce^{Ax}$ .

$x_k$	$y_k$	$X_k = x_k$	$Y_k = \ln(y_k)$	$X_k^2$	$X_k Y_k$
0	1.5	0	0.405465	0	0.000000
1	2.5	1	0.916291	1	0.916291
2	3.5	2	1.252763	4	2.505526
3	5.0	3	1.609438	9	4.828314
4	7.5	4	2.014903	16	8.059612
Total:		10	6.198860	30	16.309742

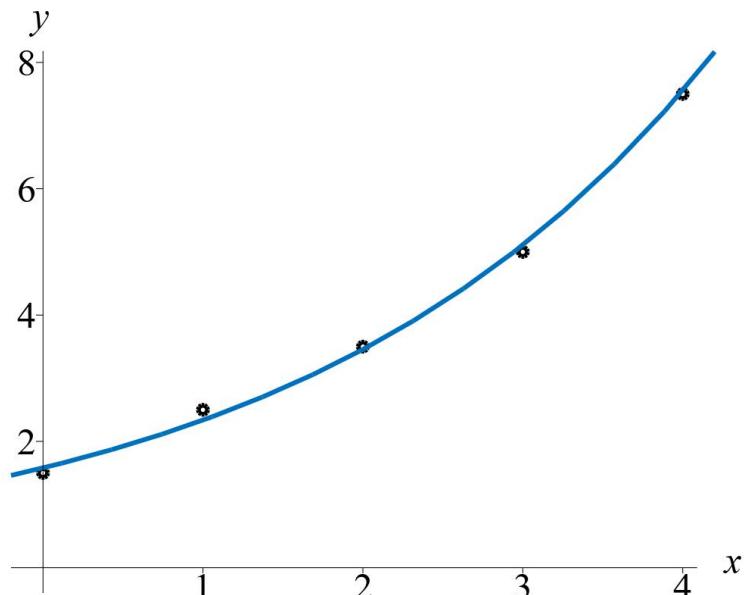
Berdasarkan Tabel 8.5, *normal equation* dari data  $X$  dan  $Y$  adalah

$$\begin{aligned} 30A + 10B &= 16.309742 \\ 10A + 5B &= 6.198860 \end{aligned} \quad (8.22)$$

Berdasarkan sistem linear di atas, didapatkan nilai  $A = 0.3912023$  dan  $B = 0.457367$ . Dengan demikian, nilai  $C$  dapat diperoleh yaitu  $C = e^{0.457367} = 1.579910$ . Dengan men-substitusikan nilai  $A$  dan  $C$  pada Persamaan 8.19, akan diperoleh kurva regresi eksponensial, yaitu

$$y = 1.579910e^{0.3912023x} \quad (8.23)$$

Persamaan eksponensial tersebut *fit* terhadap data  $x$  dan  $y$  berdasarkan metode linearisasi data dan membentuk grafik seperti pada Gambar 8.2 berikut.



**Gambar 8.2:** Kurva regresi eksponensial  $y = 1.579910e^{0.3912023x}$  yang didapatkan menggunakan metode linearisasi data.



## Transformasi untuk Linearisasi Data

Metode linearisasi data telah banyak digunakan para ilmuwan untuk menentukan mencocokan kurva dari beberapa persamaan tak-linear seperti  $y = Ce^{Ax}$ ,  $y = \ln(x) + B$  dan  $y = A/x + B$ . Setelah menentukan kurva yang akan dicocokan terhadap data, transformasi yang tepat terhadap peubah  $x$  dan  $y$  harus ditentukan sedemikian sehingga didapatkan hubungan linear. Sebagai contoh, persamaan  $y = D/(x + C)$  dapat ditransformasikan menjadi masalah linear  $Y = AX + B$  dengan cara mengganti peubah dan konstanta menjadi  $X = xy$ ,  $Y = y$ ,  $A = -1/C$ , dan  $B = D/C$ . Beberapa bentuk persamaan tak-linear lain beserta transformasinya kedalam bentuk linear dapat dilihat pada Tabel 8.6 berikut.

**Tabel 8.6:** Transformasi peubah untuk linearisasi data.

Fungsi, $y = f(x)$	Bentuk linear, $Y = AX + B$	Transformasi peubah dan konstanta
$y = \frac{A}{x} + B$	$y = A\frac{1}{x} + B$	$X = \frac{1}{x}, Y = y$
$y = \frac{D}{x+C}$	$y = \frac{-1}{C}xy + \frac{D}{C}$	$X = xy, Y = y$ $A = \frac{-1}{C}, B = \frac{D}{C}$
$y = \frac{1}{Ax+B}$	$\frac{1}{y} = Ax + B$	$X = x, Y = \frac{1}{y}$
$y = \frac{x}{Ax+B}$	$\frac{1}{y} = A\frac{1}{x} + B$	$X = \frac{1}{x}, Y = \frac{1}{y}$
$y = A \ln(x) + B$	$y = A \ln(x) + B$	$X = \ln(x), Y = y$
$y = Ce^{Ax}$	$\ln(y) = Ax + \ln(C)$	$X = x, Y = \ln(y)$ $C = e^B$
$y = Cx^A$	$\ln(y) = A \ln(x) + \ln(C)$	$X = \ln(x), Y = \ln(y)$ $C = e^B$
$y = (Ax+B)^{-2}$	$y^{-1/2} = Ax + B$	$X = x, Y = y^{-1/2}$
$y = Cxe^{-Dx}$	$\ln\left(\frac{y}{x}\right) = -Dx + \ln(C)$	$X = x, Y = \ln\left(\frac{y}{x}\right)$ $C = e^B, D = -A$
$y = \frac{L}{1+Ce^{Ax}}$	$\ln\left(\frac{L}{y} - 1\right) = Ax + \ln(C)$	$X = x, Y = \ln\left(\frac{L}{y} - 1\right)$ $C = e^B, L$ diketahui

## 8.4 Regresi Polinomial

Bentuk fungsi lain yang sering digunakan untuk mencari garis regresi dari kumpulan data adalah fungsi polinomial. Fungsi polinomial  $f(x)$  dengan derajat  $M$  didefinisikan sebagai berikut.

$$f(x) = c_1 + c_2x + c_3x^2 + \cdots + c_{M+1}x^M \quad (8.24)$$

Berikut ini adalah cara untuk mencari fungsi regresi polinomial berderajat dua (parabola) dan dapat dikembangkan untuk polinomial dengan derajat yang lebih tinggi.

**Teorema 8.3 Regresi Parabola:** Misalkan bahwa diberikan data  $\{(x_k, y_k)\}_{k=1}^N$  sebanyak  $N$  titik dengan nilai  $x_k$  yang berbeda. Koefisien dari garis regresi parabola

$$y = Ax^2 + Bx + C \quad (8.25)$$

merupakan nilai solusi  $A$ ,  $B$ , dan  $C$  dari sistem linear

$$\begin{aligned} \left( \sum_{k=1}^N x_k^4 \right) A + \left( \sum_{k=1}^N x_k^3 \right) B + \left( \sum_{k=1}^N x_k^2 \right) C &= \sum_{k=1}^N y_k x_k^2 \\ \left( \sum_{k=1}^N x_k^3 \right) A + \left( \sum_{k=1}^N x_k^2 \right) B + \left( \sum_{k=1}^N x_k \right) C &= \sum_{k=1}^N y_k x_k \\ \left( \sum_{k=1}^N x_k^2 \right) A + \left( \sum_{k=1}^N x_k \right) B + NC &= \sum_{k=1}^N y_k \end{aligned} \quad (8.26)$$

**Bukti** Dengan cara yang sama seperti pada Teorema 8.1, koefisien  $A$ ,  $B$ , dan  $C$  akan meminimumkan nilai dari

$$E(A, B, C) = \sum_{k=1}^N (Ax_k^2 + Bx_k + C - y_k)^2 \quad (8.27)$$

Hasil dari turunan parsial  $\partial E / \partial A$ ,  $\partial E / \partial B$ , dan  $\partial E / \partial C$  adalah

$$\begin{aligned} \frac{\partial E}{\partial A} &= 2 \sum_{k=1}^N (Ax_k^2 + Bx_k + C - y_k)x_k^2 = 0 \\ \frac{\partial E}{\partial B} &= 2 \sum_{k=1}^N (Ax_k^2 + Bx_k + C - y_k)x_k = 0 \\ \frac{\partial E}{\partial C} &= 2 \sum_{k=1}^N (Ax_k^2 + Bx_k + C - y_k) = 0 \end{aligned} \quad (8.28)$$

Dengan operasi pindah ruas, akan didapatkan sistem linear seperti pada Persamaan 8.26. ■

## Formula Matriks

Dari Persamaan 8.26, dapat dilihat bahwa diperlukan banyak operator penjumlahan  $\Sigma$  untuk menghitung nilai koefisien regresi. Hal ini akan menghasilkan waktu komputasi yang lama. Salah satu cara mengatasi masalah tersebut adalah dengan menuliskan sistem kedalam notasi matriks. Berikut ini adalah cara untuk mencari fungsi regresi polinomial berderajat dua (parabola) dan dapat dikembangkan untuk polinomial dengan derajat yang lebih tinggi.

Misalkan bahwa diberikan data  $\{(x_k, y_k)\}_{k=1}^N$  sebanyak  $N$  titik dengan nilai  $x_k$  yang berbeda dan akan dicari koefisien regresi dari persamaan

$$y = c_1 + c_2x + c_3x^2 \quad (8.29)$$

Matriks  $F$  didefinisikan dengan  $f_i(x) = x^{i-1}$  seperti berikut.

$$F = \begin{bmatrix} f_1(x_1) & f_2(x_1) & f_3(x_1) \\ f_1(x_2) & f_2(x_2) & f_3(x_2) \\ f_1(x_3) & f_2(x_3) & f_3(x_3) \\ \vdots & \vdots & \vdots \\ f_1(x_N) & f_2(x_N) & f_3(x_N) \end{bmatrix} = \begin{bmatrix} (x_1)^0 & (x_1)^1 & (x_1)^2 \\ (x_2)^0 & (x_2)^1 & (x_2)^2 \\ (x_3)^0 & (x_3)^1 & (x_3)^2 \\ \vdots & \vdots & \vdots \\ (x_N)^0 & (x_N)^1 & (x_N)^2 \end{bmatrix} \quad (8.30)$$

Ruas kanan dari sistem linear akan setara dengan nilai dari  $F^T Y$ , yaitu

$$F^T Y = \begin{bmatrix} (x_1)^0 & (x_2)^0 & (x_3)^0 & \dots & (x_N)^0 \\ (x_1)^1 & (x_2)^1 & (x_3)^1 & \dots & (x_N)^1 \\ (x_1)^2 & (x_2)^2 & (x_3)^2 & \dots & (x_N)^2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{bmatrix} \quad (8.31)$$

Sementara itu, ruas kiri dari sistem linear akan setara dengan nilai dari  $F^T F$ , yaitu

$$F^T F = \begin{bmatrix} (x_1)^0 & (x_2)^0 & (x_3)^0 & \dots & (x_N)^0 \\ (x_1)^1 & (x_2)^1 & (x_3)^1 & \dots & (x_N)^1 \\ (x_1)^2 & (x_2)^2 & (x_3)^2 & \dots & (x_N)^2 \end{bmatrix} \begin{bmatrix} (x_1)^0 & (x_1)^1 & (x_1)^2 \\ (x_2)^0 & (x_2)^1 & (x_2)^2 \\ (x_3)^0 & (x_3)^1 & (x_3)^2 \\ \vdots & \vdots & \vdots \\ (x_N)^0 & (x_N)^1 & (x_N)^2 \end{bmatrix} \quad (8.32)$$

Dengan demikian, nilai koefisien regresi  $C$  dapat dicari dengan menyelesaikan sistem linear

$$(F^T F) C = F^T Y \quad (8.33)$$

**Contoh 8.5 :** Carilah persamaan regresi parabola dari empat titik data  $(-3, 3)$ ,  $(0, 1)$ ,  $(2, 1)$ , dan  $(4, 3)$ .

**Solusi :** Tabel 8.7 berikut menunjukkan proses penjumlahan yang akan digunakan dalam sistem linear pada Persamaan 8.26.

**Tabel 8.7:** Penghitungan nilai koefisien regresi parabola.

$k$	$x_k$	$y_k$	$x_k^2$	$x_k^3$	$x_k^4$	$x_k y_k$	$x_k^2 y_k$
1	-3	3	9	-27	81	-9	27
2	0	1	0	0	0	0	0
3	2	1	4	8	16	2	4
4	4	3	16	64	256	12	48
$\Sigma$	3	8	29	45	353	5	79

Berdasarkan tabel di atas, sistem linear pada Persamaan 8.26 menjadi

$$353A + 45B + 29C = 79$$

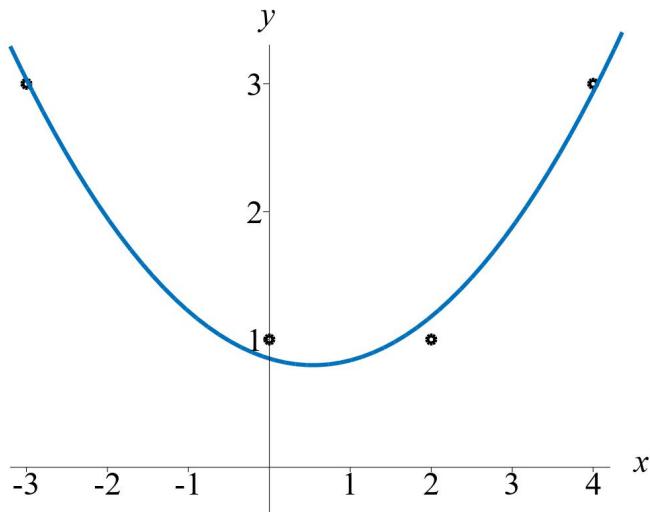
$$45A + 29B + 3C = 5$$

$$29A + 3B + 4C = 8$$

Dengan menyelesaikan sistem tersebut, dapat diperoleh koefisien regresi  $A = 0.178462$ ,  $B = -0.192495$ , dan  $C = 0.850519$ . Dengan demikian, persamaan regresi parabola yang didapatkan adalah

$$y = 0.178462x^2 - 0.192495x + 0.850519$$

Gambar 8.3 berikut menunjukkan grafik fungsi parabola yang diperoleh beserta titik-titik data.



**Gambar 8.3:** Grafik parabola  $y = 0.178462x^2 - 0.192495x + 0.850519$ . △

## 8.5 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Regresi Linear** berisi program untuk mencari persamaan regresi linear  $f(x) = Ax + B$  dari suatu kumpulan data. Program ini secara *default* berisi 2 masukan, yaitu kumpulan data terurut  $X$  dan  $Y$  serta 2 luaran, yaitu koefisien regresi  $A$  dan  $B$ . Berikut merupakan program untuk regresi linear.

**Algoritma 8.1:** Regresi linear

```
using Statistics
function reglin(X, Y)
    xmean = mean(X);
    ymean = mean(Y);
    # Hitung nilai jumlah dari xy dan x^2
    sumxy = (X.-xmean)'*(Y.-ymean)
    sumx2 = (X.-xmean)'*(X.-xmean)
    # Hitung nilai koefisien garis regresi linear Y=Ax+B
    A = sumxy/sumx2;
    B = ymean .- A*xmean;
    return A, B
end
```

**Regresi Pangkat** berisi program untuk mencari persamaan regresi pangkat  $f(x) = Ax^m$  dari suatu kumpulan data. Program ini secara *default* berisi 3 masukan, yaitu kumpulan data terurut  $X$ ,  $Y$ , dan nilai pangkat yang diinginkan  $m$  serta 1 luaran, yaitu koefisien regresi pangkat  $A$ . Berikut merupakan program untuk regresi pangkat.

### Algoritma 8.2: Regresi pangkat

```
function regpower(X, Y, m)
    sumxy = (X.^m)' * Y
    sumx2 = (X.^m)' * (X.^m)
    A = sumxy/sumx2
end
```

**Regresi Polinomial** berisi program untuk mencari koefisien persamaan regresi polinomial  $f(x) = a_m x^m + a_{m-1} x^{m-1} + \dots + a_1 x + a_0$  dari kumpulan data. Program ini secara *default* berisi 3 masukan, yaitu kumpulan data terurut  $X$ ,  $Y$ , dan nilai pangkat tertinggi yang diinginkan  $m$  serta 1 luaran, yaitu matriks yang berisi koefisien regresi polinomial  $C = [a_m, a_{m-1}, \dots, a_1, a_0]$ . Berikut merupakan program untuk regresi polinomial.

### Algoritma 8.3: Regresi polinomial

```
function regpoly(X, Y, m)
    F = zeros(length(X), m+1)
    for k = 1: m+1
        F[:, k] = X.^ (k-1);
    end
    A = F' * F;
    B = F' * Y;
    C = A \ B;
    C = reverse(C, dims=1)
end
```

## 8.5.1 Regresi Linear

Metode pencocokan kurva pertama yang akan dipelajari adalah metode regresi linear. Metode ini akan mencari persamaan linear dari kumpulan data, yaitu  $f(x) = Ax + B$  yang meminimumkan nilai RMSE.

**Contoh 8.6 :** Diberikan dataset  $(x_k, y_k)$  seperti berikut.

$x_k$	-1	0	1	2	3	4	5	6
$y_k$	10	9	7	5	4	3	0	-1

Berikut merupakan langkah-langkah untuk membentuk suatu garis regresi linear dari titik-titik data tersebut.

**Langkah 1:** Pendefinisian data  $x_k$  dan  $y_k$ , kemudian akan dicari koefisien garis regresi linear menggunakan Program 8.1.

```
Inp: x = [-1, 0, 1, 2, 3, 4, 5, 6]
     y = [10, 9, 7, 5, 4, 3, 0, -1];
     A, B = reglin(x, y)
```

```
Out: (-1.6071428571428572, 8.642857142857142)
```

**Langkah 2:** Pendefinisian fungsi regresi  $y = Ax + B$  dari koefisien yang diperoleh.

Inp:  $f(x) = A*x+B;$

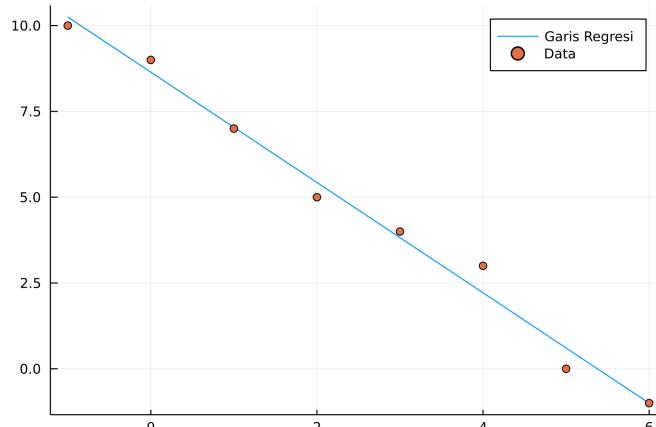
**Langkah 3:** Pembuatan grafik garis regresi yang diperoleh beserta titik-titik data.

```
Inp: using Plots
      xk = -1:0.1:6
      plot(xk,f.(xk),label = "Garis Regresi")
      scatter!(x,y,label = "Data")
```

**Langkah 4:** Hitung nilai RMSE dari garis regresi tersebut.

```
Inp: rmse(yk,yduga) = sqrt(mean((yk.-yduga).^2));
      yduga = f.(x);
      galat = rmse(y,yduga)
```

Out: 0.41726148019814



**Gambar 8.4:** Garis regresi linear  $y = -1.6071x + 8.6429$ .

Gambar 8.4 menunjukkan garis regresi linear terhadap titik titik data yang diberikan. Berdasarkan hasil yang diperoleh, garis persamaan regresi linear yang didapatkan adalah

$$y = -1.6071x + 8.6429$$

dengan galat RMSE sebesar 0.41726.

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 8.6 untuk membentuk garis regresi linear dari data berikut.

$x_k$	0.0	1.0	2.0	4.0	5.0
$y_k$	3.0	5.1	6.9	10.0	13.0

Gambarkan garis regresi beserta titik-titik data, kemudian hitung nilai RMSE dari garis regresi tersebut.

## 8.5.2 Regresi Pangkat

Metode pencocokan kurva selanjutnya yang akan dipelajari adalah metode regresi linear. Metode ini akan mencari persamaan pangkat dari kumpulan data, yaitu  $f(x) = Ax^m$  yang meminimumkan nilai RMSE.

**Contoh 8.7 :** Pada suatu praktikum, seorang mahasiswa melakukan percobaan dan mendapatkan data seperti berikut.

Waktu, $t_k$	Jarak, $d_k$
0.2	0.1960
0.4	0.7850
0.6	1.7665
0.8	3.1405
1.0	4.9075

Hubungan dari kedua data tersebut adalah  $d = \frac{1}{2}gt^2$  dengan  $g$  konstanta gravitasi. Jika akan dicari nilai konstanta gravitasi dengan regresi pangkat dari data percobaan di atas, maka bentuk persamaan garis regresi pangkatnya adalah  $d = At^2$  dengan  $A = \frac{1}{2}g$  seperti berikut.

```
Inp: tk = [0.2,0.4,0.6,0.8,1.0];
      dk = [0.1960,0.7850,1.7665,3.1405,4.9075];
      A = regpower(tk,dk,2)
Out: 4.907303370786516
```

```
Inp: g = 2*A
Out: 9.814606741573032
```

Jadi, nilai koefisien gravitasi pada percobaan tersebut adalah  $g = 9.8146 \text{ m/s}^2$ .

**Soal Latihan:** Diberikan data hasil pengamatan seperti berikut.

$d_k$	1.00	2.00	3.00	4.00	5.00
$t_k$	0.45	0.63	0.79	0.90	1.01

Hitunglah nilai koefisien grafitasi tempat pengamatan berdasarkan data tersebut dengan mengikuti cara pada Contoh 8.7.

### 8.5.3 Linearisasi Data

Sering kali kumpulan data tidak dapat didekati menggunakan garis linear ataupun pangkat. Jika hubungan data berupa persamaan tak-linear seperti eksponensial, maka pencocokan kurva persamaan tak-linear dapat dibentuk menggunakan metode linearisasi data.

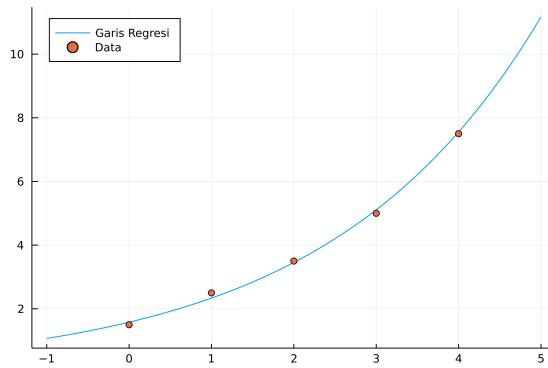
**Contoh 8.8 :** Diketahui bahwa transformasi untuk linearisasi data dari garis eksponensial  $y = Ce^{Ax}$  adalah

$$Y = AX + B$$

dengan  $X = x$ ,  $Y = \ln(y)$ , dan  $B = \ln(C)$ . Dengan metode linearisasi data, garis regresi eksponensial  $y = Ce^{Ax}$  dari lima dataset  $(0, 1.5)$ ,  $(1, 2.5)$ ,  $(2, 3.5)$ ,  $(3, 5.0)$ , dan  $(4, 7.5)$  dapat dicari dengan cara sebagai berikut.

```
Inp: x = [ 0,    1,    2,    3,    4]
      y = [1.5,  2.5,  3.5,  5.0,  7.5]
      Y = log.(y)
Inp: A,B = reglin(x,Y)
      C = exp(B)
      (A,C)
```

```
Out: (0.3912023005428146, 1.5799091528746356)
```



**Gambar 8.5:** Garis regresi eksponensial  $y = 1.5799e^{0.39120x}$ .

Gambar 8.5 menunjukkan garis regresi eksponensial terhadap titik titik data yang diberikan. Berdasarkan hasil yang diperoleh, garis regresi eksponensial yang didapatkan adalah

$$y = 1.5799e^{0.39120x}$$

dengan galat RMSE sebesar 0.10007.

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 8.8 untuk membentuk garis regresi eksponensial dari data berikut.

$x_k$	0.00	1.00	2.00	4.00	5.00
$y_k$	2.00	0.75	0.38	0.15	0.10

Gambarkan garis regresi beserta titik-titik data, kemudian hitung nilai RMSE dari garis regresi tersebut.

#### 8.5.4 Regresi Polinomial

Metode pencocokan kurva terakhir yang akan dipelajari adalah metode regresi polinomial. Metode ini akan mencari persamaan polinomial dari kumpulan data, yaitu

$$f(x) = a_m x^m + a_{m-1} x^{m-1} + \cdots + a_1 x + a_0$$

yang memminimumkan nilai RMSE.

**Contoh 8.9 :** Diketahui suatu dataset seperti berikut.

$x_k$	-3	0	2	4
$y_k$	3	1	1	3

Dari pasangan data terurut tersebut, dapat dibentuk suatu persamaan garis regresi parabola menggunakan Program 8.3 dengan cara seperti berikut.

```
Inp: x = [-3, 0, 2, 4]
      y = [ 3, 1, 1, 3]
      C = regpoly(x,y,2)
```

```
Out: 3-element Array{Float64,1}:
      0.17846247712019525
      -0.19249542403904824
      0.8505186089078707
```

```
Inp: f(x) = C[1]*x^2 + C[2]*x + C[3]
      xk = -3:0.1:4
      plot(xk,f.(xk),label = "Garis Regresi")
      scatter!(x,y,label = "Data")

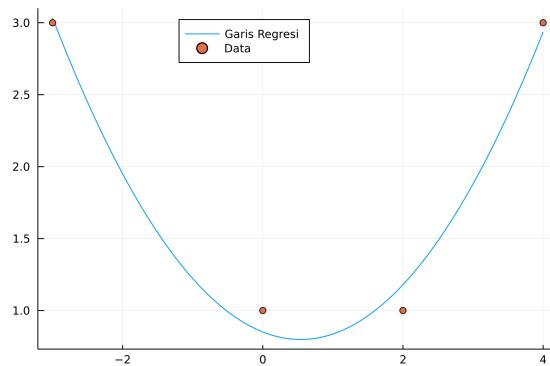
Inp: rmse(yk,yduga) = sqrt(mean((yk.-yduga).^2));
      yduga = f.(x);
      galat = rmse(y,yduga)

Out: 0.12226258262122926
```

Dari hasil yang diperoleh, garis regresi yang didapatkan adalah

$$f(x) = 0.17846x^2 - 0.19250x + 0.85052$$

dengan galat RMSE sebesar 0.12226. Gambar 8.6 di bawah menunjukkan garis regresi polinomial terhadap titik titik data yang diberikan.



**Gambar 8.6:** Garis regresi polinomial dari titik-titik data.

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 8.9 untuk membentuk garis regresi kuadratik dari data berikut.

$x_k$	0.00	1.00	2.00	4.00	5.00
$y_k$	7.10	3.00	3.60	11.00	24.00

Gambarkan garis regresi beserta titik-titik data, kemudian hitung nilai RMSE dari garis regresi tersebut.

## 8.6 Latihan-latihan

### 8.6.1 Ulasan Materi

1. Sebutkan dan jelaskan perbedaan antara intepolasi, regresi dan ekstrapolasi!
2. Sebutkan dan jelaskan tipe data atau kasus seperti apa yang cocok menggunakan metode intepolasi, regresi dan ekstrapolasi!
3. Jelaskan apa yang dimaksud dengan *maximum error*, *average error*, dan *root mean squared error*! Berikan contohnya.

4. Garis regresi linear  $y = Ax + B$  dapat dibentuk dengan meminimumkan nilai *root mean squared error* (RMSE). Buktikan bahwa nilai koefisien garis regresi linear tersebut dapat diperoleh dengan menyelesaikan sistem persamaan linear yang disebut *normal equation*, yaitu

$$\begin{aligned} \left( \sum_{k=1}^N x_k^2 \right) A + \left( \sum_{k=1}^N x_k \right) B &= \sum_{k=1}^N x_k y_k \\ \left( \sum_{k=1}^N x_k \right) A + NB &= \sum_{k=1}^N y_k \end{aligned}$$

5. Berikan suatu contoh penurunan rumus dari suatu persamaan tak-linear menjadi linear, atau disebut juga linearisasi data!
6. Manakah di antara pernyataan berikut yang benar/salah?
- Regresi harus melalui titik-titik data dengan tepat, sedangkan interpolasi tidak harus melalui titik-titik data dengan tepat.
  - Regresi memiliki galat (error) pada titik-titik data yang diketahui, sedangkan interpolasi tidak memiliki galat (error) pada titik-titik data yang diketahui.
  - Secara umum, metode interpolasi lebih baik dibandingkan regresi.
  - Interpolasi mengestimasi nilai data di antara titik-titik data yang diberikan, sedangkan regresi mengestimasi nilai data di luar titik-titik data yang diberikan.
  - Garis regresi diperoleh menggunakan metode kuadrat terkecil yang meminimumkan nilai galat kuadrat rata-rata.
  - Metode linearisasi data digunakan untuk mendekati suatu data yang tidak bisa dicocokkan menggunakan regresi linear atau pangkat (seperti persamaan tak-linear) agar memiliki galat yang lebih baik.

### 8.6.2 Soal Pemrograman

1. Carilah persamaan regresi linear  $y = Ax + B$  beserta nilai RMSE dari data berikut.

a.	$x_k$	$y_k$	b.	$x_k$	$y_k$	c.	$x_k$	$y_k$
	-6	7		-4	-3		-8	6.8
	-2	5		-1	-1		-2	5.0
	0	3		0	0		0	2.2
	2	2		2	1		4	0.5
	6	0		3	2		6	-1.3

2. Cari persamaan regresi  $y = Ax^2$ ,  $y = A/x$ ,  $y = A/x^2$ , dan  $y = Ax^3$  dari data di bawah ini, kemudian hitung nilai galat  $E_2(f)$  dari persamaan tersebut.

a.	$x_k$	$y_k$	b.	$x_k$	$y_k$
	2.0	5.1		2.0	5.9
	2.3	7.5		2.3	8.3
	2.6	10.6		2.6	10.7
	2.9	14.4		2.9	13.7
	3.2	19.0		3.2	17.0

3. Carilah beberapa persamaan regresi dengan terlebih dahulu melihat *scatter plot* data. Tentukan regresi yang memiliki nilai  $E_2(f)$  yang paling baik. Plot semua hasil regresi dalam satu grafik beserta titik-titik data.

a.	$x_k$	$y_k$
	-1	6.62
	0	3.94
	1	2.17
	2	1.35
	3	0.89

b.	$x_k$	$y_k$
	-3.0	-0.14
	-1.5	-2.16
	0.0	0.83
	1.5	2.28
	3.0	-0.51

4. Suatu masalah nilai awal  $dy/dt = f(t,y)$  mempunyai nilai solusi di beberapa titik sebagai berikut:

$x_k$	0	0.1	0.2	0.3	0.4	0.5
$y_k$	-1	-0.9	-0.79	-0.67	-0.54	-0.41

- (a) Buatlah *scatter plot* dari data di atas, amati pola hubungannya (linear, kuadratik, kubik, dan lain sebagainya).  
 (b) Buatlah garis regresi kemudian hitung nilai  $E_2(f)$ .  
 (c) Plot fungsi regresi yang anda dapat beserta titik-titik di atas.  
 (d) Gunakan fungsi regresi untuk menebak nilai  $y(0.26)$ ,  $y(0.35)$ , dan  $y(0.43)$ .
5. Berikut merupakan data yang diperoleh SHP Chen dan SC Saxena dalam percobaan mereka.

$T$	$e$	$T$	$e$	$T$	$e$
300	0.024	900	0.097	1500	0.186
400	0.035	1000	0.111	1600	0.202
500	0.046	1100	0.125	1700	0.219
600	0.056	1200	0.140	1800	0.235
700	0.067	1300	0.155	1900	0.252
800	0.083	1400	0.170	2000	0.269

Mereka menemukan bahwa persamaan

$$e(T) = 0.02424 \left( \frac{T}{303.16} \right)^{1.27591}$$

berkorelasi terhadap data dengan akurasi hingga 3 digit desimal.

- (a) Buatlah garis regresi  $p(T)$  yang menghubungkan data di atas dengan regresi. (Amati terlebih dahulu pola hubungannya)  
 (b) Plot  $p(T)$  dan  $e(T)$  dalam satu grafik beserta data pada interval [300,2000].  
 (c) Jelaskan gambar yang anda peroleh.
6. Diberikan data nilai kapasitas panas  $C$  dari titanium sebagai fungsi dari suhu  $T$  sebagai berikut.

$T$	605	685	725	765	825	855	875
$C$	0.622	0.655	0.668	0.679	0.73	0.907	1.336

- (a) Carilah garis regresi dari data di atas. (Amati dulu pola hubungannya)  
 (b) Perkirakan nilai  $C(645)$ ,  $C(795)$ , dan  $C(845)$ , serta bandingkan dengan nilai aslinya, yaitu  $C(645) = 0.639$ ,  $C(795) = 0.694$ , dan  $C(845) = 0.812$ .

**7. Rekor Dunia Sprint 100 m Putra**

Tahun	Rekor (s)	Tahun	Rekor (s)	Tahun	Rekor (s)
2009	9.58	1994	9.85	1964	10.06
2008	9.69	1991	9.86	1960	10.25
2008	9.72	1991	9.90	1958	10.29
2007	9.74	1988	9.92	1958	10.32
2006	9.77	1988	9.93	1958	10.32
2006	9.77	1987	9.93	1948	10.34
2005	9.77	1983	9.93	1932	10.38
2002	9.78	1968	9.95	1932	10.38
1999	9.79	1968	10.02	1932	10.53
1996	9.84	1968	10.03	1932	10.64

Tabel di atas menunjukkan catatan rekor dunia *sprint* dari tahun 1932 hingga 2009.

- (a) Buatlah *scatter plot* dari data di atas. Cari hubungan antara tahun dan rekor.
- (b) Buatlah persamaan regresi dari hubungan yang didapat dan hitung nilai  $E_2(f)$
- (c) Berdasarkan hasil regresi, tahun berapakah rekor baru akan terpecahkan?

**8. Kecepatan dan Hambatan Aerodinamik**

Data : <https://physics.info/curve-fitting/aerodynamic-drag.txt>

- (a) Buatlah *scatter plot* kecepatan ( $x$ ) dan hambatan aerodinamik ( $y$ ).
- (b) Hubungan apa yang diperoleh antara kecepatan dan hambatan aerodinamik.
- (c) Dari hubungan tersebut, buatlah persamaan regresi yang sesuai.
- (d) Berapa nilai *root mean squared error* dari hasil regresi tersebut.

**9. Suhu ruang penyimpanan dan umur simpan susu segar**

Data : <https://physics.info/curve-fitting/milk-freshness.txt>

- (a) Buatlah *scatter plot* suhu ruang penyimpanan ( $x$ ) dan umur simpan susu ( $y$ ).
- (b) Hubungan apa yang diperoleh antara suhu ruang penyimpanan dan umur simpan susu.
- (c) Dari hubungan tersebut, buatlah persamaan regresi yang sesuai.
- (d) Berapa nilai *root mean squared error* dari hasil regresi tersebut.

**10. Massa dan Akselerasi**

Data : <https://physics.info/curve-fitting/constant-force.txt>

- (a) Buatlah *scatter plot* massa (peubah  $x$ ) dan akselerasi (peubah  $y$ ).
- (b) Hubungan apa yang diperoleh antara dan akselerasi dan akselerasi.
- (c) Dari hubungan tersebut, buatlah persamaan regresi yang sesuai.
- (d) Berapa nilai *root mean squared error* dari hasil regresi tersebut.

**11. Hukum Hubble**

Data : <https://physics.info/curve-fitting/hubble-law.txt>

- (a) Buatlah *scatter plot* jarak (peubah  $x$ ) dan kecepatan radial (peubah  $y$ ).
- (b) Hubungan apa yang diperoleh antara dan akselerasi dan akselerasi.
- (c) Dari hubungan tersebut, buatlah persamaan regresi yang sesuai.
- (d) Berapa nilai *root mean squared error* dari hasil regresi tersebut.

**12. Tabung Resonansi**

Data : <https://physics.info/curve-fitting/resonance-tube.txt>

- (a) Buatlah *scatter plot* frekuensi (peubah  $x$ ) dan panjang gelombang (peubah  $y$ ).
- (b) Hubungan apa yang diperoleh antara dan akselerasi dan akselerasi.
- (c) Dari hubungan tersebut, buatlah persamaan regresi yang sesuai.
- (d) Berapa nilai *root mean squared error* dari hasil regresi tersebut.

**13. Iklim Antartika**

Data : <https://physics.info/curve-fitting/vostok.txt>

Keterangan :

Kolom 1 = Tahun (tahun sebelum sekarang)

Kolom 2 = Anomali Suhu ( $^{\circ}\text{C}$ )

Kolom 3 = Konsentrasi  $\text{CO}_2$

Kolom 4 = Konsentrasi debu

Pertanyaan:

- A. Hubungan anomali suhu dan konsentrasi  $\text{CO}_2$ 
  - (a) Scatter plot anomali suhu vs konsentrasi  $\text{CO}_2$
  - (b) Analisis hubungannya
  - (c) Buatlah persamaan regresi berdasarkan hubungan tersebut
  - (d) Hitung nilai RMSE
- B. Hubungan anomali suhu dan konsentrasi debu
  - (a) Scatter plot anomali suhu vs konsentrasi debu
  - (b) Analisis hubungannya
  - (c) Buatlah persamaan regresi berdasarkan hubungan tersebut
  - (d) Hitung nilai RMSE

**14. Suhu komputer**

Data : <https://physics.info/curve-fitting/computer-temperatures.txt>

Keterangan :

Kolom 1 = *Running time* (menit)

Kolom 2 = CPU A ( $^{\circ}\text{C}$ )

Kolom 3 = CPU B ( $^{\circ}\text{C}$ )

Kolom 4 = *Memory Controller Heatsink* ( $^{\circ}\text{C}$ )

Kolom 5 = *Drive Bay* ( $^{\circ}\text{C}$ )

Kolom 6 = *Hard Drive* ( $^{\circ}\text{C}$ )

Pertanyaan: bagaimana suhu dari komponen-komponen dalam komputer tersebut dipengaruhi oleh *running time*? Buatlah persamaan regresi berdasarkan hubungan antara *running time* terhadap suhu masing-masing komponen!

---

---

# BAB 9

---

## TURUNAN NUMERIK

Turunan merupakan masalah matematika yang sering muncul dalam berbagai bidang ilmu terapan. Namun, terdapat banyak persamaan matematika yang sulit untuk ditentukan nilai turunannya. Bahkan tak jarang terdapat kasus untuk menentukan turunan dari suatu data yang tidak diketahui persamaan fungsi  $f(x)$  secara eksplisit, melainkan hanya diketahui kumpulan data. Selain itu, meskipun diketahui fungsi eksplisit  $f(x)$ , persamaan fungsi tersebut terlalu rumit untuk dicari nilai turunannya secara analitik. Misalnya, akan sangat rumit jika harus mencari nilai  $f'(1)$  secara analitik dari persamaan berikut.

$$f(x) = \frac{\sqrt{\cos(2x^2) + x\tan(3x)}}{\sin(x) + e^x - 2x/\cos(x)}$$

Untuk mengatasi masalah tersebut, nilai turunan dari suatu fungsi dapat dihampiri menggunakan metode numerik. Dengan demikian, persamaan fungsi yang rumit dapat dihitung nilai turunannya dengan lebih sederhana. Turunan numerik ini akan sangat berguna untuk menyelesaikan masalah nilai batas dari persamaan differensial biasa dan parsial.

### 9.1 Hampiran Turunan

#### Rumus Limit Turunan

Dalam kalkulus, turunan dari  $f(x)$  didefinisikan sebagai

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h} \quad (9.1)$$

Untuk menghampiri nilai turunan numerik dari  $f(x)$ , akan dipilih barisan  $h_k$  sedemikian sehingga  $h_k \rightarrow 0$  dan hampiran  $f'(x)$  adalah

$$D_k = \frac{f(x+h_k) - f(x)}{h_k} \quad \text{untuk } k = 1, 2, \dots, n, \dots \quad (9.2)$$

Proses komputasi hanya akan menghitung nilai  $D_1, D_2, \dots, D_N$  dan menggunakan nilai  $D_N$  sebagai nilai turunan numerik  $f'(x)$ . Pertanyaan yang muncul dalam proses numerik ini adalah berapa nilai  $h_N$  yang harus dipilih, sehingga  $D_N$  menjadi nilai hampiran turunan  $f'(x)$  yang baik.

**Contoh 9.1 :** Diberikan  $f(x) = e^x$  dan  $x = 1$ . Hitunglah nilai hampiran turunan  $D_k$  menggunakan ukuran langkah  $h_k = 10^{-k}$  untuk  $k = 1, 2, \dots, 10$  dengan penghitungan 9 angka desimal.

**Solusi :** Dari Persamaan 9.2, hasil penghitungan  $D_k$  dapat dilihat pada Tabel 9.1 berikut.

**Tabel 9.1:** Penghitungan hampiran turunan  $D_k$ .

$h_k$	$f(1 + h_k)$	$f(1 + h_k) - f(1)$	$D_k$
0.100000000	3.004166024	0.285884196	2.858841960
0.010000000	2.745601015	0.027319187	2.731918700
0.001000000	2.721001470	0.002719642	2.719642000
0.000100000	2.718553670	0.000271842	2.718420000
0.000010000	2.718309011	0.000027183	2.718300000
0.000001000	2.718284547	0.000002719	2.719000000
0.000000100	2.718282100	0.000000272	2.720000000
0.000000010	2.718281856	0.000000028	2.800000000
0.000000001	2.718281831	0.000000003	3.000000000
0.000000000	2.718281828	0.000000000	0.000000000

Berdasarkan hasil yang didapatkan, jika nilai  $h_k$  terlalu besar, misalkan  $h_1 = 0.1$ , nilai hampiran yang didapatkan tidak bagus untuk menghampiri nilai  $f'(1)$ . Sementara itu, jika nilai  $h_k$  terlalu kecil misalkan  $h_9 = 10^{-9}$ , nilai  $f(x + h_k)$  dan  $f(x)$  saling berdekatan satu sama lain, sehingga nilai  $f(x + h_k) - f(x)$  memunculkan *lost of significant digit*. Berdasarkan Tabel 9.1, nilai hampiran ketika  $h_k = 10^{-5}$  merupakan hasil yang paling mendekati nilai  $f'(1) = 2.718281828459$ .  $\triangle$

## 9.2 Formula Beda-Pusat

Penghitungan turunan numerik menggunakan definisi turunan membutuhkan iterasi yang cukup banyak karena hanya memiliki kompleksitas komputasi  $O(h)$ . Oleh karena itu, diperlukan pengembangan formula yang dapat memberikan nilai akurasi yang lebih baik dengan nilai  $h$  yang lebih besar. Jika fungsi  $f(x)$  dapat dievaluasi pada nilai yang terletak di sebelah kanan dan kiri dari  $x$ , maka formula beda-pusat dapat digunakan untuk menghitung nilai turunan  $f'(x)$ .

**Teorema 9.1 Formula Beda-Pusat dengan Ordo  $O(h^2)$ :** Asumsikan bahwa  $f \in C^3[a, b]$  dan  $x - h, x, x + h \in [a, b]$ , maka

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} \quad (9.3)$$

Selain itu, terdapat nilai  $c = c(x) \in [a, b]$  yang menyebabkan

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} + E_{trunc}(f, h) \quad (9.4)$$

dengan

$$E_{trunc}(f, h) = -\frac{h^2 f^{(3)}(c)}{6} = O(h^2) \quad (9.5)$$

Bentuk  $E(f, h)$  disebut sebagai *truncation error*.

**Bukti** Pembuktian akan dimulai dari ekspansi Taylor derajat kedua  $f(x) = P_2(x) + E_2(x)$ , disekitar  $x$ , untuk  $f(x+h)$  dan  $f(x-h)$ , yaitu

$$f(x+h) = f(x) + f'(x)h + \frac{f^{(2)}(x)h^2}{2!} + \frac{f^{(3)}(c_1)h^3}{3!} \quad (9.6)$$

dan

$$f(x-h) = f(x) - f'(x)h + \frac{f^{(2)}(x)h^2}{2!} - \frac{f^{(3)}(c_1)h^3}{3!} \quad (9.7)$$

Selisih dari Persamaan 9.6 dan 9.7 adalah

$$f(x+h) - f(x-h) = 2f'(x)h + \frac{(f^{(3)}(c_1) + f^{(3)}(c_2))h^3}{3!} \quad (9.8)$$

Selama  $f^{(3)}(x)$  kontinu, teorema nilai antara dapat digunakan untuk mencari nilai  $c$ , sedemikian sehingga

$$\frac{f^{(3)}(c_1) + f^{(3)}(c_2)}{2} = f^{(3)}(c) \quad (9.9)$$

Substitusi Persamaan 9.9 ke Persamaan 9.8, sehingga didapatkan

$$f(x+h) - f(x-h) = 2f'(x)h + \frac{2f^{(3)}(c)h^3}{3!} \quad (9.10)$$

Dalam bentuk lain, Persamaan 9.10 dapat ditulis sebagai

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{f^{(3)}(c)h^2}{3!} \quad (9.11)$$

Pada ruas kanan Persamaan 9.11, bentuk pertama menunjukkan formula beda-pusat pada Persamaan 9.3 dan bentuk kedua menunjukkan *truncation error*. Bukti selesai. ■

Asumsikan bahwa nilai dari turunan ketiga  $f^{(3)}$  tidak berubah secara signifikan, maka *truncation error* pada Persamaan 9.5 akan bergerak menuju nol setara dengan  $h^2$ . Dengan demikian, ketika proses komputasi dilakukan, nilai  $h$  tidak terlalu kecil.

**Teorema 9.2 Formula Beda-Pusat dengan Ordo  $O(h^4)$ :** Asumsikan bahwa  $f \in C^5[a, b]$  dan  $x-2h, x-h, x, x+h, x+2h \in [a, b]$ , maka

$$f'(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} \quad (9.12)$$

Selain itu, terdapat nilai  $c = c(x) \in [a, b]$  yang menyebabkan

$$f'(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} + E_{trunc}(f, h) \quad (9.13)$$

dengan

$$E_{trunc}(f, h) = \frac{h^4 f^{(5)}(c)}{30} = O(h^4) \quad (9.14)$$

Bentuk  $E(f, h)$  disebut sebagai *truncation error*.

**Bukti** Pembuktian dimulai dari ekspansi Taylor derajat keempat  $f(x) = P_4(x) + E_4(x)$ , disekitar  $x$ , sehingga didapatkan selisih dari  $f(x+h)$  dan  $f(x-h)$  yaitu

$$f(x+h) - f(x-h) = 2f'(x)h + \frac{2f^{(3)}(x)h^3}{3!} + \frac{2f^{(5)}(c_1)h^5}{5!} \quad (9.15)$$

Serta, selisih ekspansi Taylor dari  $f(x+2h)$  dan  $f(x-2h)$ , yaitu

$$f(x+2h) - f(x-2h) = 4f'(x)h + \frac{16f^{(3)}(x)h^3}{3!} + \frac{64f^{(5)}(c_2)h^5}{5!} \quad (9.16)$$

Selanjutnya, dengan mengalikan Persamaan 9.15 dengan 8 dan menguranginya dengan Persamaan 9.16, diperoleh hasil seperti berikut.

$$\begin{aligned} & -f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h) \\ &= 12f'(x)h + \frac{(16f^{(5)}(c_1) - 64f^{(5)}(c_2))h^5}{120} \end{aligned} \quad (9.17)$$

Jika  $f^{(5)}(x)$  memiliki tanda yang sama dan nilainya tidak berubah secara tiba-tiba, maka dapat dicari nilai  $c$  yang terletak di  $[x-2h, x+2h]$ , sedemikian sehingga

$$16f^{(5)}(c_1) - 64f^{(5)}(c_2) = -48f^{(5)}(c) \quad (9.18)$$

Substitusi Persamaan 9.18 ke Persamaan 9.17, sehingga didapatkan

$$-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h) = 12f'(x)h - \frac{48f^{(5)}(c)h^5}{120} \quad (9.19)$$

Dalam bentuk lain, Persamaan 9.19 dapat ditulis sebagai

$$f'(x) = \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} + \frac{f^{(5)}(c)h^4}{30} \quad (9.20)$$

Pada ruas kanan Persamaan 9.20, bentuk pertama menunjukkan formula beda-pusat pada Persamaan 9.12 dan bentuk kedua menunjukkan *truncation error*. Bukti selesai. ■

**Contoh 9.2 :** Diberikan fungsi  $f(x) = \cos(x)$ . Gunakan formula beda pusat dengan ordo  $O(h^2)$  pada Persamaan 9.3 dan ordo  $O(h^4)$  pada Persamaan 9.12 dengan ukuran langkah  $h = 0.1, 0.01, 0.001$ , dan  $0.0001$  untuk menghitung nilai hampiran  $f'(0.8)$  menggunakan penghitungan 9 angka desimal. Bandingkan hasil yang didapatkan dengan nilai eksaknya, yaitu  $f'(0.8) = -\sin(0.8)$ .

**Solusi :** Untuk  $h = 0.1$ , hasil dari formula beda-pusat  $O(h^2)$  adalah

$$\begin{aligned} f'(0.8) &\approx \frac{f(0.9) - f(0.7)}{2(0.1)} \\ &\approx \frac{0.621609968 - 0.764842187}{0.2} \\ &\approx -0.716161095 \end{aligned} \quad (9.21)$$

Serta, hasil dari formula beda-pusat  $O(h^4)$  adalah

$$\begin{aligned} f'(0.8) &\approx \frac{-f(1.0) + 8f(0.9) - 8f(0.7) + f(0.6)}{12(0.1)} \\ &\approx \frac{-0.540302306 + 8(0.621609968) - 8(0.764842187) + 0.825335615}{1.2} \quad (9.22) \\ &\approx -0.717353703 \end{aligned}$$

Dengan cara yang sama, hasil dari formula beda pusat  $O(h^2)$  dan  $O(h^4)$  dengan ukuran langkah  $h = 0.1, 0.01, 0.001, 0.0001$  beserta galat turunan numerik terhadap nilai eksanya yaitu  $f'(0.8) = -\sin(0.8)$  dapat dilihat pada Tabel 9.2 berikut.

**Tabel 9.2:** Turunan numerik menggunakan formula beda pusat.

$h$	Hasil beda pusat $O(h^2)$	Galat beda pusat $O(h^2)$	Hasil beda pusat $O(h^4)$	Galat beda pusat $O(h^4)$
0.1000	-0.716161095	0.001194996	-0.717353703	0.000002388
0.0100	-0.717344150	0.000011941	-0.717356108	0.000000017
0.0010	-0.717356000	0.000000091	-0.717356167	0.000000076
0.0001	-0.717360000	0.000003909	-0.717360833	0.000004742

Berdasarkan hasil di atas, hasil turunan numerik  $f'(0.8)$  dengan formula beda-pusat ordo  $O(h^2)$  diperoleh pada  $h = 0.001$  dengan galat sebesar 0.000000091. Sementara itu, formula beda-pusat ordo  $O(h^4)$  memeroleh hasil turunan numerik lebih cepat yaitu pada  $h = 0.01$  dengan galat sebesar 0.000000017.  $\triangle$

### 9.3 Ekstrapolasi Richardson

Pada subbab ini, akan dipelajari hubungan antara formula beda pusat ordo  $O(h^2)$  dan  $O(h^4)$  pada Persamaan 9.3 dan 9.12. Misalkan,  $f_k = f(x_k) = f(x_0 + kh)$  serta  $D_0(h)$  dan  $D_0(2h)$  menunjukkan nilai hampiran dari  $f'(x_0)$  pada Persamaan 9.3 dengan ukuran langkah  $h$  dan  $2h$ . Dengan demikian, untuk ukuran langkah  $h$  didapatkan

$$f'(x_0) \approx D_0(h) + Ch^2 \quad (9.23)$$

dengan  $D_0(h) = \frac{f_1 - f_{-1}}{2h}$  dan  $C = -\frac{f^{(3)}(c)}{6}$ . Sementara itu, untuk ukuran langkah  $2h$  didapatkan

$$f'(x_0) \approx D_0(2h) + 4Ch^2 \quad (9.24)$$

dengan  $D_0(2h) = \frac{f_2 - f_{-2}}{2(2h)}$  dan  $C = -\frac{f^{(3)}(c)}{6}$ . Selanjutnya, eliminasi persamaan 9.23 dan 9.24 untuk menghilangkan nilai  $C$ , yaitu

$$\begin{array}{l|l|l} f'(x_0) \approx D_0(h) + Ch^2 & \times 4 & 4f'(x_0) \approx 4D_0(h) + 4Ch^2 \\ f'(x_0) \approx D_0(2h) + 4Ch^2 & \times 1 & f'(x_0) \approx D_0(2h) + 4Ch^2 \\ \hline & & 3f'(x_0) \approx 4D_0(h) - D_0(2h) \end{array} \quad (9.25)$$

Berdasarkan Persamaan 9.25, didapatkan hampiran  $f'(x_0)$ , yaitu

$$\begin{aligned} f'(x_0) &\approx \frac{4D_0(h) - D_0(2h)}{3} \\ &\approx \frac{4\left(\frac{f_1 - f_{-1}}{2h}\right) - \left(\frac{f_2 - f_{-2}}{2(2h)}\right)}{3} \\ &\approx \frac{-f_2 + 8f_1 - 8f_{-1} + f_{-2}}{12h} \end{aligned} \quad (9.26)$$

Persamaan 9.26 menunjukkan formula beda-pusat ordo  $O(h^4)$  pada Persamaan 9.12.

**Contoh 9.3 :** Diberikan  $f(x) = \cos(x)$ . Hitung nilai  $D_0(h)$  dan  $D_0(2h)$  dengan  $h = 0.01$  dan tunjukkan bahwa hasil kombinasi linear  $(4D_0(h) - D_0(2h))/3$  pada Persamaan 9.26 dapat digunakan untuk menghampiri nilai  $f'(0.8)$ . Gunakan 9 angka desimal dalam setiap proses penghitungan.

**Solusi :** Dengan Persamaan 9.23 dan 9.24, didapatkan

$$\begin{aligned} D_0(h) &\approx \frac{f(0.81) - f(0.79)}{0.02} \approx \frac{0.689498433 - 0.703845316}{0.02} \\ &\approx -0.717344150 \end{aligned} \quad (9.27)$$

dan

$$\begin{aligned} D_0(2h) &\approx \frac{f(0.82) - f(0.78)}{0.04} \approx \frac{0.682221207 - 0.710913538}{0.04} \\ &\approx -0.717308275 \end{aligned} \quad (9.28)$$

Selanjutnya, nilai kombinasi linear pada Persamaan 9.26 adalah

$$\begin{aligned} f'(0.8) &\approx \frac{4D_0(h) - D_0(2h)}{3} \approx \frac{4(-0.717344150) - (-0.717308275)}{3} \\ &\approx -0.717356108 \end{aligned} \quad (9.29)$$

Hasil yang didapatkan ini tepat sama seperti hasil formula beda-pusat ordo  $O(h^4)$  dengan  $h = 0.01$  pada Tabel 9.2 Contoh 9.2.  $\triangle$

Metode untuk memperoleh formula turunan numerik  $f'(x_0)$  dengan ordo yang lebih tinggi dari formula berordo rendah disebut dengan **ekstrapolasi**.

**Teorema 9.3 :** Misalkan bahwa terdapat dua hampiran  $f'(x_0)$  dengan ordo  $O(h^{2k})$  yaitu  $D_{k-1}(h)$  dan  $D_{k-1}(2h)$ , sehingga memenuhi

$$f'(x_0) = D_{k-1}(h) + Ch^{2k} + \dots \quad (9.30)$$

dan

$$f'(x_0) = D_{k-1}(2h) + 4^k Ch^{2k} + \dots \quad (9.31)$$

Selanjutnya, hampiran dengan ordo  $O(h^{2k+2})$  dapat diperoleh dengan formula

$$f'(x_0) = D_k(h) + O(h^{2k+2}) = \frac{4^k D_{k-1}(h) - D_{k-1}(2h)}{4^k - 1} + O(h^{2k+2}) \quad (9.32)$$



**Contoh 9.4 :** Misalkan  $f(x) = \cos(x)$ .

1. Hitung nilai hampiran dari  $f'(0.8)$  menggunakan formula beda-pusat ordo  $O(h^4)$  pada Persamaan 9.12 dengan  $h = 0.1$  dan  $h = 0.01$ . Gunakan 9 angka desimal dalam setiap penghitungan.
2. Gunakan ekstrapolasi Richardson untuk menghitung hampiran  $f'(0.8)$  berdasarkan hasil dari poin 1.

**Solusi :** Dari Contoh 9.2 pada Tabel 9.2, nilai formula beda-pusat ordo  $O(h^4)$  dengan  $h = 0.1$  dan  $h = 0.01$  masing-masing yaitu  $-0.717353703$  dan  $-0.717356108$ . Dalam notasi ekstrapolasi, hasil tersebut dapat ditulis sebagai berikut.

$$D_1(h) = -0.717353703 \quad \text{dan} \quad D_1(2h) = -0.717356108 \quad (9.33)$$

Berdasarkan hasil tersebut, dapat dihitung nilai hampiran dari  $f'(0.8)$  dengan ordo  $O(h^6)$  menggunakan ekstrapolasi pada Persamaan 9.32 seperti berikut.

$$\begin{aligned} f'(0.8) \approx D_2(h) &= \frac{4^2 D_1(h) - D_1(2h)}{4^2 - 1} \\ &= \frac{16(-0.717353703) - (-0.717356108)}{15} \\ &= -0.717353543 \end{aligned} \quad (9.34)$$

Dengan demikian, hasil ekstrapolasi Richardson yang didapatkan sebagai hampiran  $f'(0.8)$  adalah  $-0.717353543$ .  $\triangle$

## 9.4 Praktikum

**Metode Beda Pusat** berisi program untuk mencari turunan numerik suatu fungsi  $f(x)$  pada titik  $x$ . Program ini secara *default* berisi 2 masukan, yaitu fungsi  $f(x)$  dan titik yang akan dicari turunannya  $x$  serta 3 luaran, yaitu nilai turunan numerik  $sol$ , status solusi ( $flag$ ) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, dan matriks  $L$  yang berisi nilai turunan numerik dan galatnya pada setiap iterasi. Berikut merupakan program untuk metode beda pusat.

**Algoritma 9.1:** Metode beda-pusat untuk turunan numerik

```
function bedaPusat(f, a; delta=10^-9)
    maxi = 15;
    flag = 1;
    h = 1;
    D = (f(a+h)-f(a-h)) / (2*h);
    E = NaN;
    sol = NaN
    for k = 1:maxi
        h = h/10;
        D = [D (f(a+h)-f(a-h)) / (2*h)];
        E = [E abs(D[k+1]-D[k])];
        if E[k+1]<delta
            flag = 0;
            sol = D[end];
            break
        end
    end
```

```

if E[k+1]>E[k]
    sol = D[k];
    flag = 2;
    break
end
L = [D' E'];
return sol, flag, L
end

```

**Metode Ekstrapolasi Richardson** berisi program untuk mencari turunan numerik suatu fungsi  $f(x)$  pada titik  $x$ . Program ini secara *default* berisi 2 masukan, yaitu fungsi  $f(x)$  dan titik yang akan dicari turunannya  $x$  serta 3 luaran, yaitu nilai turunan numerik *sol*, status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, nilai galat akhir turunan numerik *err*, dan matriks *D* yang berisi tabel iterasi Richardson. Berikut merupakan program untuk metode ekstrapolasi richardson.

### Algoritma 9.2: Metode Richarson

```

function richardson(f, a; delta = 1e-9)
maxi = 50;
flag = 1;
h = 1;
D = (f(a+h)-f(a-h)) / (2*h);
err = NaN;
for j=1:maxi
    h = h/2;
    D = [D zeros(size(D,1),1);
        (f(a.+h).-f(a.-h))./(2.*h) zeros(1, size(D,1))];
    for k = 1:j
        D[j+1,k+1] = D[j+1,k] + (D[j+1,k]-D[j,k])/(4^(k-1));
    end
    err = abs(D[j+1,j+1]-D[j,j]);
    if err<delta
        flag=0;
        break
    end
end
sol = D[end,end];
return sol, flag, err, D
end

```

#### 9.4.1 Hampiran dari Turunan

Pada kalkulus, telah diketahui bahwa nilai turunan dari suatu fungsi dapat dicari menggunakan konsep limit, yaitu

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

Namun, untuk menghitung nilai turunan numerik, nilai  $h$  yang terlalu kecil akan mengakibatkan *lost of significant digit*, sehingga solusi yang dihasilkan menjadi tidak akurat. Tujuan utama dalam turunan numerik yaitu mencari nilai  $h$  sedemikian sehingga solusi numerik memiliki akurasi paling tinggi.

**Contoh 9.5 :** Diberikan fungsi  $f(x) = e^x$  dan  $x = 1$ . Berikut merupakan langkah-langkah untuk menghitung nilai hampiran  $f'(1)$ , yaitu  $D_k$  untuk  $k = 1, 2, 3, \dots, 10$ .

```
Inp: k = 1:10
    h = 10.0 .^ -k
    f(x) = exp.(x);
    a = 1;
    Dk = (f(a+h) .- f(a))./h;
    M = [k Dk abs.(Dk.-exp(a))]
```

**Tabel 9.3:** Hasil penghitungan nilai  $D_k = (e^{1+h_k} - e)/h_k$ .

$k$	$D_k$	$D_k - e$
1	2.85884195	0.14056013
2	2.73191866	0.01363683
3	2.71964142	0.00135959
4	2.71841775	0.00013592
5	2.71829542	0.00001359
6	2.71828319	0.00000136
7	2.71828197	0.00000014
8	2.71828182	0.00000001
9	2.71828204	0.00000022
10	2.71828338	0.00000155

Berdasarkan hasil yang didapatkan, jika nilai  $h_k$  terlalu besar (misalkan  $h_1 = 0.1$ ), maka nilai hampiran yang didapatkan tidak bagus untuk menghampiri nilai  $f'(1)$ . Se-mentara itu, jika nilai  $h_k$  terlalu kecil misalkan  $h_9 = 10^{-9}$ , nilai  $f(x + h_k)$  dan  $f(x)$  saling berdekatan satu sama lain, sehingga nilai  $f(x + h_k) - f(x)$  memunculkan *lost of significant digit*. Berdasarkan Tabel 9.3, nilai hampiran ketika  $h_k = 10^{-8}$  merupakan hasil yang paling mendekati nilai  $f'(1) = e$ .

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 9.5 untuk menghitung nilai turunan numerik  $f'(1)$  dari  $f(x) = \sin(x)$ , yaitu  $D_k$  untuk  $k = 1, 2, 3, \dots, 10$ .

#### 9.4.2 Formula Beda Pusat

Penghitungan turunan numerik menggunakan definisi turunan membutuhkan iterasi yang cukup banyak karena hanya memiliki kompleksitas komputasi  $O(h)$ . Salah satu cara untuk mempercepat penghitungan turunan numerik adalah menggunakan formula beda-pusat yang memiliki kompleksitas  $O(h^2)$ , yaitu

$$D_k = \frac{f(x + h_k) - f(x - h_k)}{2h_k}$$

dengan  $k = 0, 1, 2, 3, \dots$  dan  $h_k = 10^{-k}$ .

**Contoh 9.6 Turunan pada suatu titik fungsi kontinu:** Diberikan fungsi  $f(x) = \sin(x)$ . Hitung hampiran  $D_k$  dari turunan  $f'(x)$  ketika  $x = \pi/3$  menggunakan beda-pusat dengan mengatur toleransi yaitu  $\delta_1 = 10^{-9}$  dan  $\delta_1 = 10^{-12}$ .

**Langkah 1:** Penghitungan nilai turunan numerik  $f'(\pi/3)$  menggunakan metode beda pusat pada Program 9.1 dengan toleransi  $\delta_1 = 10^{-9}$ .

```
Inp: f(x) = sin(x); a = pi/3;
      sol,flag,L = bedaPusat(f,a)
      @show sol
      L
```

```
Inp: sol = 0.4999999999921733
      6x2 Array{Float64,2}:
      0.420735  NaN
      0.499167  0.0784316
      0.499992  0.000824583
      0.5       8.24996e-6
      0.5       8.24996e-8
      0.5       8.26006e-10
```

**Langkah 2:** Penghitungan nilai turunan numerik  $f'(\pi/3)$  menggunakan metode beda pusat pada Program 9.1 dengan toleransi  $\delta_2 = 10^{-12}$ .

```
Inp: f(x) = sin(x); a = pi/3;
      sol,flag,L = bedaPusat(f,a,delta = 10^-12)
      @show sol
      L
```

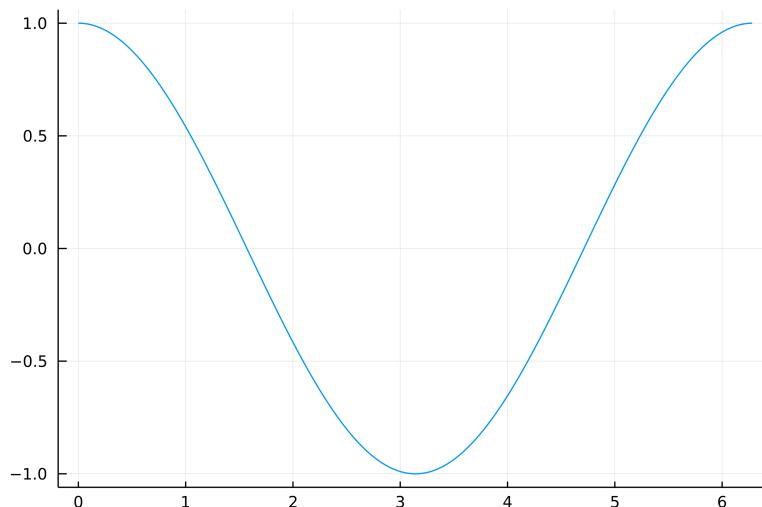
```
Out: sol = 0.4999999999588666
      8x2 Array{Float64,2}:
      0.420735  NaN
      0.499167  0.0784316
      0.499992  0.000824583
      0.5       8.24996e-6
      0.5       8.24996e-8
      0.5       8.26006e-10
      0.5       3.33067e-11
      0.5       3.33067e-10
```

Berdasarkan hasil yang diperoleh, pada langkah 1, iterasi keenam sudah melewati toleransi yang diberikan yaitu  $\delta_1 = 10^{-9}$  sehingga iterasi tidak dilanjutkan menuju nilai  $h$  yang lebih kecil. Sementara itu, pada langkah 2, iterasi kedelapan memberikan nilai galat yang lebih besar daripada nilai galat pada iterasi ketujuh, sehingga nilai toleransi  $\delta_2 = 10^{-12}$  tidak dapat dipenuhi dan solusi turunan numerik mengambil nilai turunan dengan galat yang paling kecil yaitu pada iterasi ketujuh.

**Soal Latihan:** Diketahui fungsi  $f(x) = \cos(x)$ . Gunakan metode beda pusat pada Program 9.1 untuk mencari turunan numerik dari  $f'(0.2)$  dengan galat paling minimum.

**Contoh 9.7 Turunan numerik dari fungsi pada interval tertentu:** Diberikan fungsi  $f(x) = \sin(x)$ . Hitung hampiran  $D_k$  dari turunan  $f'(x)$  ketika  $x = [0, 2\pi]$  menggunakan beda-pusat.

```
f(x) = sin.(x);
a = 0:0.01:2*pi
y = Array{Number}(undef,length(a),1)
for i in 1:length(a)
    sol,flag,L = bedaPusat(f,a[i]);
    y[i] = sol;
end
plot(a,y,legend=:false)
```



**Gambar 9.1:** Hasil turunan numerik  $f(x) = \sin(x)$  pada  $x \in [0, 2\pi]$  menggunakan beda-pusat.

Berdasarkan Gambar 9.1, grafik turunan numerik dari  $f(x) = \sin(x)$  pada  $x \in [0, 2\pi]$  sesuai dengan turunan analitiknya, yaitu  $f'(x) = \cos(x)$ .

**Soal Latihan:** Diketahui fungsi  $f(x) = \cos(x)$ . Gunakan metode beda pusat pada Program 9.1 untuk mencari solusi turunan numerik pada interval  $[0, 2\pi]$ .

#### 9.4.3 Ekstrapolasi Richardson

Metode alternatif lain yang dapat digunakan untuk mencari nilai turunan numerik adalah menggunakan ekstrapolasi Richardson. Metode ini merupakan modifikasi dari metode beda-pusat untuk menghasilkan kompleksitas yang lebih baik. Dengan memanfaatkan kompleksitas beda-pusat yaitu  $O(h^2)$ , ekstrapolasi Richardson dapat menghitung solusi numerik dengan kompleksitas  $O(h^4)$ ,  $O(h^6)$ , dan seterusnya.

**Contoh 9.8 Turunan pada suatu titik fungsi kontinu:** Diberikan fungsi  $f(x) = \sin(x)$ . Hitung hampiran dari turunan  $f'(x)$  ketika  $x = \pi/3$  menggunakan ekstrapolasi Richardson dengan mengatur toleransi yaitu  $\delta_1 = 10^{-9}$  dan  $\delta_1 = 10^{-16}$ .

**Langkah 1:** Penghitungan nilai turunan numerik  $f'(\pi/3)$  menggunakan ekstrapolasi Richardson pada Program 9.2 dengan toleransi  $\delta_1 = 10^{-9}$ .

```
Inp: f(x) = sin.(x);
      a = pi/3;
      sol,flag,err,D = richardson(f,a)
      @show sol
      @show err
      D

Out: sol = 0.4999999999998856
      err = 3.323475383787411e-10
      5x5 Array{Float64,2}:
      0.420735  0.0       0.0       0.0   0.0
      0.479426  0.498989  0.0       0.0   0.0
      0.494808  0.499935  0.499998  0.0   0.0
      0.498699  0.499996  0.5       0.5   0.0
      0.499675  0.5       0.5       0.5   0.5
```

**Langkah 2:** Penghitungan nilai turunan numerik  $f'(\pi/3)$  menggunakan ekstrapolasi Richardson pada Program 9.2 dengan toleransi  $\delta_l = 10^{-16}$ .

```
Inp: f(x) = sin.(x);
      a = pi/3;
      sol,flag,err,D = richardson(f,a,delta=10^-16)
      @show sol
      @show err

Out: sol = 0.5
      err = 0.0
```

Berdasarkan hasil yang diperoleh pada Contoh 9.6, metode beda-pusat hanya mampu menyelesaikan turunan dari  $f(x) = \sin(x)$  pada  $x = \pi/3$  sampai memperoleh estimasi galat sekitar  $3 \times 10^{-11}$ . Akan tetapi, dengan ekstrapolasi Richardson pada Contoh 9.8, nilai turunan yang dihasilkan mampu diatur toleransi galatnya. Dengan toleransi  $10^{-16}$ , metode Richardson dapat melampaui toleransi tersebut, sehingga hasil yang diperoleh lebih presisi daripada menggunakan metode beda-pusat.

**Soal Latihan:** Diketahui fungsi  $f(x) = \cos(x)$ . Gunakan ekstrapolasi Richardson pada Program 9.2 untuk mencari turunan numerik dari  $f'(0.2)$  dengan mengatur toleransi yaitu  $\delta = 10^{-k}$  dan  $k = 5, 6, 7, \dots, 16$ .

## 9.5 Latihan-Latihan

### 9.5.1 Ulasan Materi

1. Apakah dalam turunan numerik dapat menerapkan definisi turunan secara utuh (dalam arti mengambil nilai  $h$  sangat kecil mendekati nol)? Jelaskan!
2. Mengapa dalam mencari turunan numerik tidak bisa mengambil nilai  $h$  yang terlalu kecil?
3. Jelaskan bagaimana formula beda-pusat ordo  $O(h^2)$  diperoleh?
4. Sebutkan dan jelaskan sumber galat dari turunan numerik menggunakan formula beda-hingga!
5. Tuliskan beberapa (minimal 5) persamaan beda-hingga (beda-maju, beda-mundur, dan beda-pusat) beserta ordonya!
6. Manakah di antara beberapa pernyataan berikut yang benar/salah?
  - (a) Semakin kecil nilai  $h$  yang dipilih untuk menghitung turunan numerik, maka solusi yang dihasilkan akan semakin akurat.
  - (b) Sumber galat pada turunan numerik hanya berasal dari galat pemotongan (*truncation error*).
  - (c) Metode dengan order galat  $O(h^4)$  dapat menemukan solusi turunan numerik pada nilai  $h$  yang lebih besar dibandingkan metode dengan order lebih rendah  $O(h^2)$ .
  - (d) Formula beda-maju akan selalu optimal ketika  $h = 10^{-8}$ .
  - (e) Ekstrapolasi Richardson menuliskan formulasi yang mengaitkan metode-metode beda-pusat dengan order galat yang berbeda.
  - (f) Ekstrapolasi Richardson menghitung metode-metode beda-pusat yang ordernya lebih tinggi menggunakan metode beda-pusat yang lebih rendah ordernya.

### 9.5.2 Soal Pemrograman

1. Diberikan fungsi  $f(x) = \exp(x)$ .
  - (a) Hitung hampiran  $f'(2.3)$  menggunakan formula beda pusat dengan  $h = 0.1$ ,  $h = 0.01$ , dan  $h = 0.001$  (gunakan 9 angka desimal)
  - (b) Bandingkan hasil dengan  $f'(2.3) = \exp(2.3) = 9.974182454$
  - (c) Hitung nilai batas *truncation error* untuk setiap kasus, gunakan

$$\left| f^{(3)}(c) \right| \leq \exp(2.4) \approx 11.02317638$$

2. Diberikan fungsi  $f(x) = \exp(x)$ .
  - (a) Hitung hampiran  $f'(2.3)$  menggunakan formula beda pusat dengan  $h = 0.01$ , dan  $h = 0.02$  (gunakan 9 angka desimal)
  - (b) Gunakan ekstrapolasi richardson untuk menghitung  $f'(2.3)$  dengan menggunakan poin a.
  - (c) Bandingkan hasil dengan  $f'(2.3) = \exp(2.3) = 9.974182454$
  - (d) Hitung nilai batas *truncation error*, gunakan

$$E_{trunc}(f, h) = \frac{f^{(5)}(c)h^4}{30}$$

dan

$$\left| f^{(5)}(c) \right| \leq \exp(2.5) \approx 12.18249396$$

3. Diberikan fungsi

$$f(x) = \exp(x)$$

Gunakan ekstrapolasi Richardson untuk menghitung  $f'(1.8)$  hingga  $k = 3$  dengan nilai awal  $h = 1$ ,  $h = 0.8$ , dan  $h = 0.6$ . Bandingkan dengan  $\exp(1.8) = 6.049647464$ .

4. Diberikan data

$x$	1	2	3	4	5
$f(x)$	0.01	0.68	1.11	1.38	1.61

- (a) Hitunglah nilai  $f'(3)$  menggunakan beda-pusat  $h = 1$  dan  $h = 2$
- (b) Gunakan ekstrapolasi richardson untuk memperbaiki hasil beda-pusat
5. Voltase  $E = E(t)$  dalam ilmu fisika dirumuskan sebagai  $E(t) = L(dI/dt) + RI(t)$  dengan menggunakan nilai  $L = 0.05$  dan  $R = 2$  serta nilai  $I(t)$  seperti berikut

$t$	1.0	1.1	1.2	1.3	1.4
$I(t)$	8.2277	7.2428	5.9908	4.5260	2.9122

- (a) Hitunglah nilai  $I'(1.2)$  menggunakan beda-pusat  $h = 0.1$  dan  $h = 0.2$
- (b) Gunakan poin (a) untuk menghitung ekstrapolasi richardson.
- (c) Gunakan hasil  $I'(1.2)$  untuk menghitung  $E(1.2)$
- (d) Bandingkan jawaban kalian jika  $I(t) = 10e^{-t/10} \sin(2t)$
6. Gunakan ekstrapolasi richardson untuk menghitung
  - (a)  $f(x) = 3x^3 - 5x^2$  saat  $x = 2.6$ ,
  - (b)  $f(x) = \sqrt{x^2 + 1}$  saat  $x = 1.3$ ,
  - (c)  $f(x) = x \cos(x) - 3x$  saat  $x = 1.4$ ,
  - (d)  $f(x) = \sin(x) + x$  saat  $x = \pi/4$ ,
  - (e)  $f(x) = xe^x - x^2$  saat  $x = 2.3$ .

7. Diberikan suatu fungsi

$$f(x) = \sin(x^2)$$

- (a) Gunakan formula beda-pusat untuk mencari nilai  $f'(2\pi)$ .
- (b) Hitung nilai turunan  $f'(x)$  pada selang  $[0, 2\pi]$  dengan jeda 0.1
- (c) Hitung nilai RMSE dari error yang dihasilkan dari setiap titik.
- (d) Plot fungsi  $f(x)$  dan turunannya  $f'(x)$  dalam satu grafik.

8. Diberikan suatu fungsi

$$f(x) = x^{1/2} \sin(x^{-1})$$

- (a) Gunakan ekstrapolasi Richardson untuk mencari nilai  $f'(2\pi)$ .
- (b) Hitung nilai turunan  $f'(x)$  pada selang  $[0, 2\pi]$  dengan jeda 0.1
- (c) Hitung nilai RMSE dari error yang dihasilkan dari setiap titik.
- (d) Plot fungsi  $f(x)$  dan turunannya  $f'(x)$  dalam satu grafik.

9. Rancanglah program Julia untuk mencari turunan fungsi  $f$  dengan formula.

$$\begin{aligned} \text{i} \quad f'(x) &= \frac{f(x+h) - f(x)}{h} \\ \text{ii} \quad f'(x) &= \frac{f(x) - f(x-h)}{h} \\ \text{iii} \quad f'(x) &= \frac{f(x+h) - f(x-h)}{2h} \\ \text{iv} \quad f'(x) &= \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} \end{aligned}$$

Gunakan program yang telah kalian rancang untuk menghitung

- (a)  $f(x) = \frac{\ln(x)}{\sin(x)}, f'(2.3)$
- (b)  $f(x) = \tan^{-1}(x) - x, f'(1.5)$
- (c)  $f(x) = x \sec(x^2 - 1) \exp(x), f'(3.4)$

Bandingkan waktu komputasi yang diperlukan untuk menjalankan masing-masing program pada setiap kasus di atas. (Gunakan `BenchmarkTools.jl`)

---

---

# BAB 10

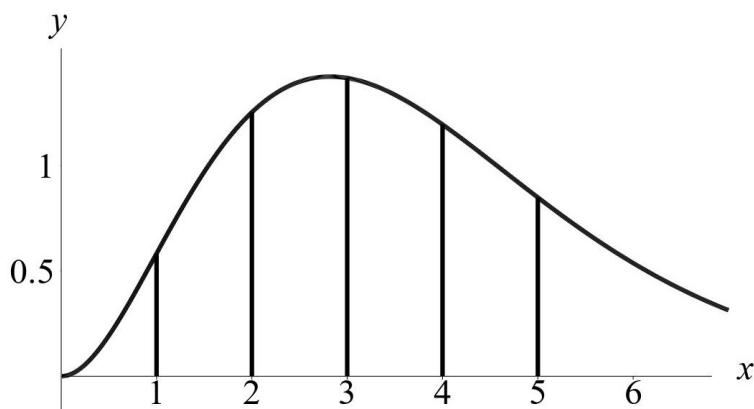
---

## INTEGRAL NUMERIK 1

Integral numerik merupakan alat utama yang dapat digunakan oleh para ilmuan untuk menentukan nilai solusi hampiran dari integral-tentu yang tidak dapat diselesaikan secara analitik. Misalnya, dalam termodinamika statistik, model Debye untuk menghitung kapasitas panas didefinisikan oleh fungsi berikut.

$$\Phi(x) = \int_0^x \frac{t^3}{e^t - 1} dt \quad (10.1)$$

Persamaan tersebut tidak memiliki solusi analitik untuk menuliskan  $\Phi(x)$ , sehingga nilai solusi hampiran dapat dicari dengan menggunakan integral numerik. Sebagai contoh, nilai dari  $\Phi(5)$  adalah luas area yang berada dibawah kurva  $y = f(t) = t^3/(e^t - 1)$  untuk  $0 \leq t \leq 5$ . (Perhatikan Gambar 10.1)



**Gambar 10.1:** Area di bawah kurva  $f(x)$  untuk  $0 \leq t \leq 5$ .

Solusi numerik dari  $\Phi(5)$  adalah

$$\Phi(5) = \int_0^5 \frac{t^3}{e^t - 1} dt \approx 4.8998922$$

Nilai tersebut dapat dicari menggunakan beberapa formula integral numerik yang akan dipelajari pada Bab ini.

## 10.1 Formula Kuadratur

Tujuan utama dari integral numerik adalah untuk mengaproksimasi/menhampiri integral-tentu dari  $f(x)$  pada interval  $[a, b]$  dengan mengevaluasi  $f(x)$  pada suatu *finite number* dari titik contoh.

**Definisi 10.1 :** Misalkan bahwa  $a = x_0 < x_1 < \dots < x_M = b$ . Suatu formula dengan bentuk

$$Q[f] = \sum_{k=0}^M f(x_k) = w_0 f(x_0) + w_1 f(x_1) + \dots + w_M f(x_M) \quad (10.2)$$

dengan ciri bahwa

$$\int_a^b f(x) dx = Q[f] + E[f] \quad (10.3)$$

disebut sebagai suatu integral numerik atau **formula kuadratur**. Bentuk  $E[f]$  merupakan *truncation error* dari integral. Nilai dari  $\{x_k\}_{k=0}^M$  disebut **node kuadratur** dan  $\{w_k\}_{k=0}^M$  disebut **bobot**. ■

Berdasarkan pengaplikasian definisi di atas, node  $\{x_k\}$  dapat dipilih dengan berbagai cara. Sebagai contoh, aturan trapesium, Simpson, dan Boole memilih node kuadratur dengan jarak yang sama. Berbagai aturan tersebut akan menghasilkan nilai integral numerik dan akurasi yang berbeda. Oleh karena itu, dibutuhkan pemahaman tentang akurasi dari solusi numerik yang akan dihasilkan masing-masing aturan.

**Definisi 10.2 : Derajat kepresisian** dari suatu formula kuadratur merupakan bilangan bulat positif  $N = n$  sedemikian sehingga  $E[P_i] = 0$  untuk setiap polinomial  $P_i(x)$  berderajat  $i \leq n$ , akan tetapi  $E[P_{n+1}] \neq 0$  untuk beberapa polinomial  $P_{n+1}(x)$  berderajat  $n+1$ . ■

Persamaan  $E[P_i]$  dapat diketahui dengan mempelajari akibat yang terjadi saat  $f(x)$  merupakan suatu polinomial. Misalkan terdapat sembarang polinomial berderajat  $i$  yaitu

$$P_i(x) = a_i x^i + a_{i-1} x^{i-1} + \dots + a_1 x + a_0$$

Jika  $i \leq n$ , maka  $P_i^{n+1}x = 0$  untuk setiap  $x$ , serta  $P_{n+1}^{n+1}x = (n+1)!a_{n+1}$  untuk setiap  $x$ . Dengan demikian, bentuk umum dari *truncation error* integral numerik adalah

$$E[f] = Kf^{n+1}(c) \quad (10.4)$$

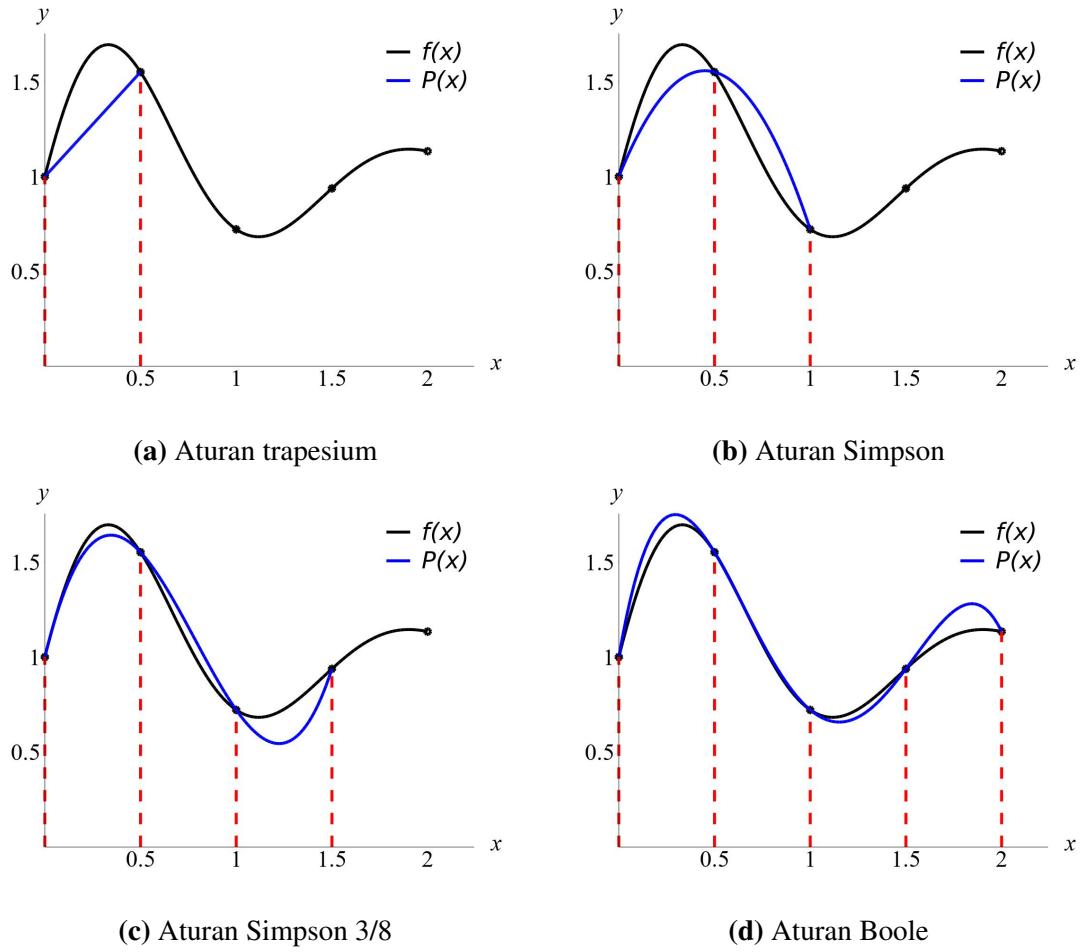
dengan  $K$  merupakan suatu konstanta dan  $n$  merupakan derajar kepresisian.

### Formula Kuadratur Newton-Cotes

Turunan dari formula kuadratur terkadang didasarkan pada interpolasi polinomial. Pada Bab sebelumnya tentang interpolasi, telah diketahui bahwa terdapat polinomial  $P_M(x)$  berderajat  $< M$  yang akan melewati  $M+1$  titik dengan jarak yang sama  $\{(x_k, y_k)\}_{k=0}^M$ . Polinomial ini akan digunakan untuk menghampiri  $f(x)$  pada  $[a, b]$ , kemudian integral dari  $f(x)$  dapat dihampiri menggunakan integral dari  $P_M(x)$ , formula yang dihasilkan disebut sebagai formula kuadratur Newton-Cotes.

Sebagai contoh, misalkan terdapat fungsi  $f(x) = 1 + e^{-x} \sin(4x)$ . Jika diketahui 2 titik  $(x_0, y_0)$  dan  $(x_1, y_1)$  dengan  $x_0 = 0$  dan  $x_1 = 0.5$ , maka dapat dicari hampiran polinomial  $P_1(x)$  dari titik-titik tersebut. Dengan demikian, nilai dari integral  $f(x)$  pada  $[0, 0.5]$  dapat

dihampiri dengan menghitung nilai integral dari  $P_1(x)$  pada  $[0, 0.5]$  seperti pada Gambar 10.2a. Aturan ini sering disebut sebagai **aturan trapesium**. Dengan cara yang sama, Jika diketahui 3 titik  $(x_0, y_0)$ ,  $(x_1, y_1)$  dan  $(x_2, y_2)$  dengan  $x_0 = 0$ ,  $x_1 = 0.5$  dan  $x_2 = 1.0$ , maka dapat dicari hampiran polinomial  $P_2(x)$  dari titik-titik tersebut. Dengan demikian, nilai dari integral  $f(x)$  pada  $[0, 1]$  dapat dihampiri dengan menghitung nilai integral dari  $P_1(x)$  pada  $[0, 1]$  seperti pada Gambar 10.2b. Aturan ini sering disebut sebagai **aturan Simpson**. Jika diketahui 4 dan 5 titik, nilai hampiran integral dapat dicari dengan cara yang sama serta aturannya disebut dengan **aturan Simpson 3/8** dan **aturan Boole** masing-masing digambarkan pada Gambar 10.2c dan 10.2d.



**Gambar 10.2:** Formula kuadratur Newton-Cotes.

**Teorema 10.1 :** Asumsikan bahwa  $x_k = x_0 + kh$  merupakan node dengan jarak yang sama dan  $f_k = f(x_k)$ . Empat formula kuadratur Newton-Cotes pertama adalah

a) Aturan trapesium

$$\int_{x_0}^{x_1} f(x) dx \approx \frac{h}{2}(f_0 + f_1) \quad (10.5)$$

b) Aturan Simpson

$$\int_{x_0}^{x_2} f(x) dx \approx \frac{h}{3}(f_0 + 4f_1 + f_2) \quad (10.6)$$

c) Aturan Simpson 3/8

$$\int_{x_0}^{x_3} f(x) dx \approx \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) \quad (10.7)$$

d) Aturan Boole

$$\int_{x_0}^{x_4} f(x) dx \approx \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) \quad (10.8)$$

**Bukti** Teorema 10.1.a. Pembuktian dimulai dengan menerapkan interpolasi Lagrange dengan  $M = 1$  sehingga didapatkan persamaan  $P_1(x)$  yaitu

$$P_1(x) = f_0 \frac{x - x_1}{x_0 - x_1} + f_1 \frac{x - x_0}{x_1 - x_0} \quad (10.9)$$

Karena  $f_0$  dan  $f_1$  konstan, nilai integral dari  $f(x)$  pada  $[x_0, x_1]$  dapat dihampiri menggunakan integral  $P_1(x)$  pada  $[x_0, x_1]$  seperti berikut.

$$\int_{x_0}^{x_1} f(x) dx \approx f_0 \int_{x_0}^{x_1} \frac{x - x_1}{x_0 - x_1} dx + f_1 \int_{x_0}^{x_1} \frac{x - x_0}{x_1 - x_0} dx \quad (10.10)$$

Misalkan  $x = x_0 + ht$  dan  $dx = h dt$ , sehingga batas integral menjadi  $t = 0$  sampai  $t = 1$ . Karena  $x_k = x_0 + kh$ , nilai  $x_k - x_j = (k - j)h$  dan  $x - x_k = (t - k)h$ . Dengan demikian, Persamaan 10.10 dapat ditulis ulang menjadi

$$\begin{aligned} \int_{x_0}^{x_1} f(x) dx &\approx f_0 \int_0^1 \frac{(t-1)h}{(-1)h} h dt + f_1 \int_0^1 \frac{(t)h}{h} h dt \\ &\approx -f_0 h \int_0^1 (t-1) dt + f_1 h \int_0^1 (t) dt \\ &\approx -f_0 h \left( \frac{t^2}{2} - t \right) \Big|_{t=0}^{t=1} + f_1 h \left( \frac{t^2}{2} \right) \Big|_{t=0}^{t=1} \\ &\approx -f_0 h \left( \frac{1}{2} - 1 \right) + f_1 h \left( \frac{1}{2} \right) \\ &\approx \frac{h}{2}(f_0 + f_1) \end{aligned} \quad (10.11)$$

Bukti selesai. Teorema 10.1.b sampai Teorema 10.1.d dapat dibuktikan menggunakan cara yang serupa. ■

**Contoh 10.1 :** Misalkan terdapat fungsi  $f(x) = 1 + e^{-x} \sin(4x)$  dengan node kadratur yang berjarak sama, yaitu  $x_0 = 0, x_1 = 0.5, x_2 = 1.0, x_3 = 1.5$ , dan  $x_4 = 2.0$ . Jika diketahui nilai  $f_0 = 1.0, f_1 = 1.55152, f_2 = 0.72159, f_3 = 0.93765$ , dan  $f_4 = 1.13390$ , terapkan formula kuadratur pada Persamaan 10.5 sampai Persamaan 10.8 untuk menghitung integral numerik pada Gambar 10.2a sampai Gambar 10.2d.

**Solusi :** Ukuran langkah antar node kuadratur adalah  $h = 0.5$ , dan komputasi masing-masing formula adalah

$$\begin{aligned}\int_0^{0.5} f(x) dx &\approx \frac{h}{2}(f_0 + f_1) = 0.63788 \\ \int_0^{1.0} f(x) dx &\approx \frac{h}{3}(f_0 + 4f_1 + f_2) = 1.32128 \\ \int_0^{1.5} f(x) dx &\approx \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) = 1.64193 \\ \int_0^{2.0} f(x) dx &\approx \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) = 2.29444\end{aligned}\quad \triangle$$

Pada Contoh 10.1, telah diaplikasikan formula kuadratur dengan  $h = 0.5$  serta batas integral yang tidak sama. Jika batas integral sama dan telah ditetapkan sebagai  $[a, b]$ , maka ukuran langkah  $h$  harus disesuaikan untuk masing-masing aturan, yaitu  $h = b - a$ ,  $h = (b - a)/2$ ,  $h = (b - a)/3$ , dan  $h = (b - a)/4$  masing-masing untuk aturan trapesium, Simpson, Simpson 3/8, dan Boole.

**Contoh 10.2 :** Misalkan terdapat fungsi  $f(x) = 1 + e^{-x} \sin(4x)$ . Hitunglah nilai  $\int_0^1 f(x) dx$  dengan formula kuadratur pada Persamaan 10.5 sampai Persamaan 10.8.

**Solusi :** Pada contoh ini, telah ditetapkan batas integral dari  $x = 0$  sampai  $x = 1$ , sehingga sebelum menghitung nilai integral, akan dihitung terlebih dulu ukuran langkah masing-masing formula kuadratur. Untuk aturan trapesium,  $h = 1$ , dan nilai integral numerik yang dihasilkan adalah

$$\begin{aligned}\int_0^1 f(x) dx &\approx \frac{1}{2}(f_0 + f_1) \\ &= \frac{1}{2}(f(0) + f(1)) = 0.86079\end{aligned}$$

Untuk aturan Simpson,  $h = 0.5$ , dan nilai integral numerik yang dihasilkan adalah

$$\begin{aligned}\int_0^1 f(x) dx &\approx \frac{0.5}{3}(f_0 + 4f_1 + f_2) \\ &= \frac{0.5}{3}(f(0) + 4f(0.5) + f(1)) = 1.32128\end{aligned}$$

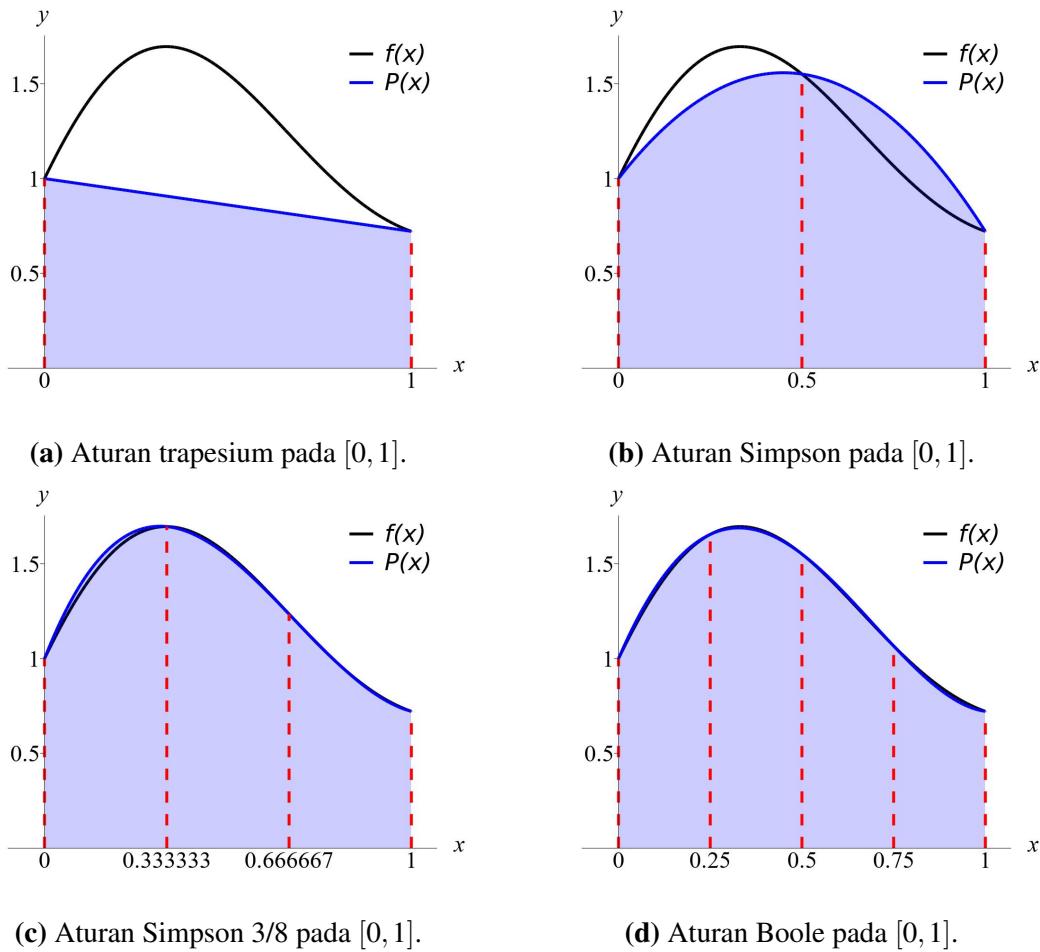
Untuk aturan Simpson 3/8,  $h = 1/3$ , dan nilai integral numerik yang dihasilkan adalah

$$\begin{aligned}\int_0^1 f(x) dx &\approx \frac{3(1/3)}{8}(f_0 + 3f_1 + 3f_2 + f_3) \\ &= \frac{3(1/3)}{8}(f(0) + 3f(\frac{1}{3}) + 3f(\frac{2}{3}) + f(1)) = 1.31440\end{aligned}$$

Untuk aturan Boole,  $h = 0.25$ , dan nilai integral numerik yang dihasilkan adalah

$$\begin{aligned}\int_0^1 f(x) dx &\approx \frac{2(0.25)}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) \\ &= \frac{2(0.25)}{45}(7f(0) + 32f(0.25) + 12f(0.5) + 32f(0.75) + 7f(1)) \\ &= 1.30859\end{aligned}$$

Nilai eksak dari integral-tentu yang diberikan adalah 1.3082506046... dan nilai integral numerik aturan Boole merupakan yang terbaik dibandingkan 3 formula lainnya. Luas area yang terletak di bawah masing-masing polinomial Lagrange  $P_1(x)$ ,  $P_2(x)$ ,  $P_3(x)$ , dan  $P_4(x)$  dapat dilihat pada Gambar 10.3 berikut.  $\triangle$



**Gambar 10.3:** Penerapan formula kuadratur Newton-Cotes untuk  $\int_0^1 1 + e^{-x} \sin(4x) dx$ .

Agar perbandingan hasil integral numerik dari aturan trapesium, Simpson dan Boole lebih adil, contoh selanjutnya akan menunjukkan perbandingan masing-masing aturan dengan jumlah evaluasi fungsi yang sama. Untuk membuat jumlah evaluasi fungsi yang sama, diperlukan pembagian interval aturan trapesium menjadi 4 sub-interval dan interval aturan Simpson menjadi 2 sub-interval. Untuk aturan trapesium, interval  $[x_0, x_4]$  akan dibagi menjadi 4 sub-interval  $[x_0, x_1]$ ,  $[x_1, x_2]$ ,  $[x_2, x_3]$ , dan  $[x_3, x_4]$ . Dengan demikian, diperoleh **aturan komposit trapesium** sebagai berikut.

$$\begin{aligned}
 \int_{x_0}^{x_4} f(x) dx &= \int_{x_0}^{x_1} f(x) dx + \int_{x_1}^{x_2} f(x) dx + \int_{x_2}^{x_3} f(x) dx + \int_{x_3}^{x_4} f(x) dx \\
 &\approx \frac{h}{2}(f_0 + f_1) + \frac{h}{2}(f_1 + f_2) + \frac{h}{2}(f_2 + f_3) + \frac{h}{2}(f_3 + f_4) \\
 &= \frac{h}{2}(f_0 + 2f_1 + 2f_2 + 2f_3 + f_4)
 \end{aligned} \tag{10.12}$$

Untuk aturan Simpson, interval  $[x_0, x_4]$  akan dibagi menjadi 2 sub-interval  $[x_0, x_2]$  dan  $[x_2, x_4]$ . Dengan demikian, diperoleh **aturan komposit Simpson** sebagai berikut.

$$\begin{aligned}\int_{x_0}^{x_4} f(x) dx &= \int_{x_0}^{x_2} f(x) dx + \int_{x_2}^{x_4} f(x) dx \\ &\approx \frac{h}{3}(f_0 + 4f_1 + f_2) + \frac{h}{3}(f_2 + 4f_3 + f_4) \\ &= \frac{h}{3}(f_0 + 4f_1 + 2f_2 + 4f_3 + f_4)\end{aligned}\quad (10.13)$$

**Contoh 10.3 :** Misalkan terdapat fungsi  $f(x) = 1 + e^{-x} \sin(4x)$ . Hitunglah nilai  $\int_0^1 f(x) dx$  menggunakan tepat 5 evaluasi fungsi dengan aturan komposit trapesium, aturan komposit Simpson, dan aturan Boole.

**Solusi :** Ukuran langkah untuk setiap aturan adalah  $h = 0.25$ . Aturan komposit trapesium pada Persamaan 10.12 menghasilkan

$$\int_0^1 f(x) dx \approx \frac{0.25}{2}(f(0) + 2f(0.25) + 2f(0.5) + 2f(0.75) + f(1)) = 1.28358$$

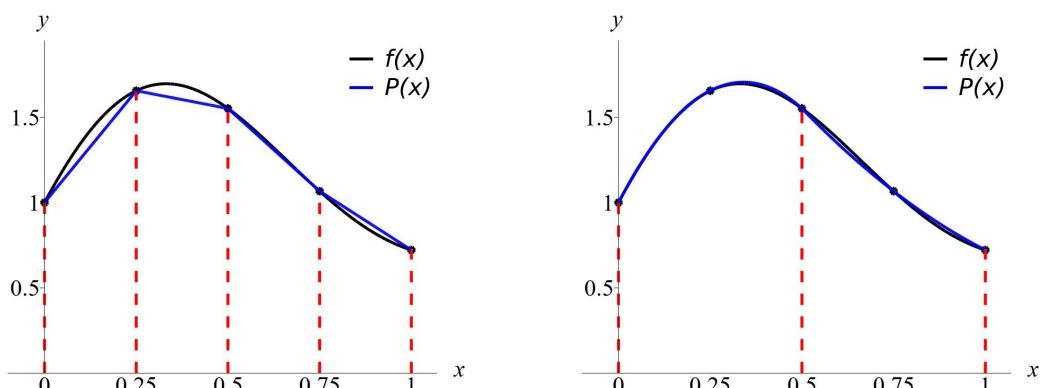
Sementara itu, aturan komposit Simpson pada Persamaan 10.13 menghasilkan

$$\int_0^1 f(x) dx \approx \frac{0.25}{3}(f(0) + 4f(0.25) + 2f(0.5) + 4f(0.75) + f(1)) = 1.30938$$

Serta, pada contoh sebelumnya, telah didapatkan hasil dari aturan Boole yaitu

$$\begin{aligned}\int_0^1 f(x) dx &\approx \frac{2(0.25)}{45}(7f(0) + 32f(0.25) + 12f(0.5) + 32f(0.75) + 7f(1)) \\ &= 1.30859\end{aligned}$$

Nilai eksak dari integral-tentu yang diberikan adalah 1.3082506046... Berdasarkan hasil yang diperoleh, nilai aturan Simpson jauh lebih baik dibandingkan dengan nilai aturan trapesium. Lebih jauh, aturan Boole tetap menjadi yang paling mendekati solusi eksak dibandingkan aturan trapesium ataupun Simpson. Luas area di bawah aturan trapesium dan Simpson dapat dilihat pada Gambar berikut.  $\triangle$



(a) Aturan komposit trapesium

(b) Aturan komposit Simpson

**Gambar 10.4:** Penerapan aturan komposit untuk aturan trapesium dan Simpson.

### Derajat Kepresision Formula Kuadratur Newton-Cotes.

Asumsikan bahwa  $f(x)$  terturunkan, maka  $E[f]$  untuk formula kuadratur Newton-Cotes melibatkan suatu turunan dengan derajat lebih tinggi. Aturan trapesium memiliki derajat kepresision  $n = 1$ . Jika  $f \in C^2[a,b]$ , maka

$$\int_{x_0}^{x_1} f(x) dx = \frac{h}{2}(f_0 + f_1) - \frac{h^3}{12}f^{(2)}(c)$$

Aturan Simpson memiliki derajat kepresision  $n = 3$ . Jika  $f \in C^4[a,b]$ , maka

$$\int_{x_0}^{x_2} f(x) dx = \frac{h}{3}(f_0 + 4f_1 + f_2) - \frac{h^5}{90}f^{(4)}(c)$$

Aturan Simpson  $\frac{3}{8}$  memiliki derajat kepresision  $n = 3$ . Jika  $f \in C^4[a,b]$ , maka

$$\int_{x_0}^{x_3} f(x) dx = \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) - \frac{3h^5}{80}f^{(4)}(c)$$

Aturan Boole memiliki derajat kepresision  $n = 5$ . Jika  $f \in C^6[a,b]$ , maka

$$\int_{x_0}^{x_4} f(x) dx = \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) - \frac{8h^7}{945}f^{(6)}(c)$$

**Contoh 10.4 :** Hitung nilai integral numerik menggunakan aturan Simpson  $\frac{3}{8}$  dari 5 fungsi percobaan, yaitu  $f(x) = 1, x, x^2, x^3, x^4$  pada  $[0,3]$ . Tentukan derajat kepresision dari aturan Simpson  $\frac{3}{8}$  berdasarkan hasil yang diperoleh.

**Solusi :** Untuk 4 fungsi pertama, integral numerik menggunakan aturan Simpson  $\frac{3}{8}$  menghasilkan solusi eksak, yaitu

$$\begin{aligned}\int_0^3 1 dx &= 3 = \frac{3}{8}(1 + 3(1) + 3(1) + 1) \\ \int_0^3 x dx &= \frac{9}{2} = \frac{3}{8}(0 + 3(1) + 3(2) + 3) \\ \int_0^3 x^2 dx &= 9 = \frac{3}{8}(0 + 3(1) + 3(4) + 9) \\ \int_0^3 x^3 dx &= \frac{81}{4} = \frac{3}{8}(0 + 3(1) + 3(8) + 27)\end{aligned}$$

Sementara itu, fungsi  $f(x) = x^4$  merupakan fungsi dengan pangkat  $x$  terkecil yang menghasilkan nilai integral numerik yang tidak eksak, yaitu

$$\int_0^3 x^4 dx = \frac{243}{5} \approx \frac{99}{2} = \frac{3}{8}(0 + 3(1) + 3(16) + 81)$$

Dengan demikian, derajat kepresision dari aturan Simpson  $\frac{3}{8}$  adalah  $n = 3$ .  $\triangle$

## 10.2 Aturan Komposit

Pada sub-bab sebelumnya, telah disinggung sedikit mengenai aturan komposit. Aturan komposit trapesium akan menghampiri nilai integral dari  $f(x)$  pada interval  $[a, b]$  menggunakan luas area dari beberapa trapesium yang terletak dibawah sub-interval  $\{(x_k, x_{k+1})\}$ .

**Teorema 10.2 Aturan Komposit Trapesium:** Misalkan bahwa interval  $[a, b]$  dipartisiakan menjadi  $M$  sub-interval  $(x_k, x_{k+1})$  dengan lebar  $h = (b - a)/M$  menggunakan node yang sama panjang yaitu  $x_k = a + kh$  untuk  $k = 0, 1, 2, \dots, M$ . Aturan komposit trapesium dapat dituliskan dengan 3 persamaan yang ekuivalen, yaitu

$$T(f, h) = \frac{h}{2} \sum_{k=1}^M (f(x_{k-1}) + f(x_k)) \quad (10.14a)$$

atau

$$T(f, h) = \frac{h}{2}(f_0 + 2f_1 + 2f_2 + \dots + 2f_{M-2} + 2f_{M-1} + f_M) \quad (10.14b)$$

atau

$$T(f, h) = \frac{h}{2}(f(a) + f(b)) + h \sum_{k=1}^{M-1} f(x_k) \quad (10.14c)$$

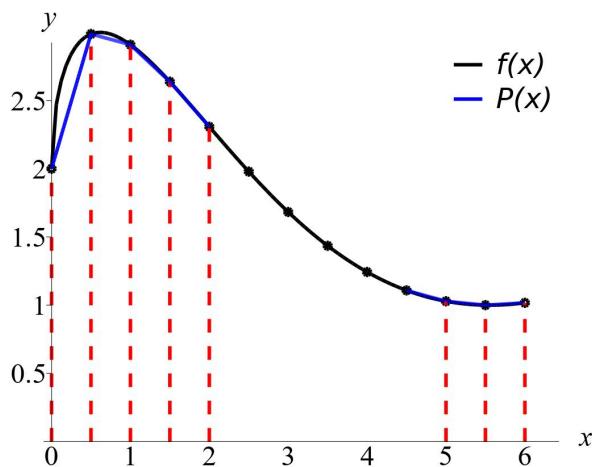
Persamaan tersebut merupakan hampiran dari integral  $f(x)$  pada selang  $[a, b]$ , sehingga dapat dituliskan sebagai

$$\int_a^b f(x) dx \approx T(f, h) \quad (10.15)$$

**Bukti** Dengan menerapkan aturan trapesium untuk setiap sub-interval  $(x_k, x_{k+1})$  (lihat Gambar 10.5), sifat penjumlahan untuk integral akan menghasilkan

$$\int_a^b f(x) dx = \sum_{k=1}^M \int_{x_{k-1}}^{x_k} f(x) dx \approx \sum_{k=1}^M \frac{h}{2}(f(x_{k-1}) + f(x_k)) \quad (10.16)$$

Karena  $h/2$  konstan, hukum distributif penjumlahan dapat diterapkan pada persamaan di atas, sehingga didapatkan Persamaan 10.14a. Persamaan 10.14b merupakan ekspansi dari Persamaan 10.14a. Persamaan 10.14c menunjukkan pengelompokan semua deret yang berada di tengah Persamaan 10.14b yang dikalikan 2. Bukti selesai. ■



**Gambar 10.5:** Aturan komposit trapesium untuk  $y = 2 + \sin(2\sqrt{x})$ .

Berdasarkan Gambar 10.5, hampiran polinomial untuk  $y = 2 + \sin(2\sqrt{x})$  dengan fungsi linear menghasilkan hampiran yang rapat terhadap fungsi asli pada beberapa sub-interval dan terdapat pula yang masih jauh terhadap fungsi asli pada sub-interval lainnya. Untuk mendapatkan akurasi yang baik dari aturan komposit trapesium, contoh berikutnya akan menghitung nilai integral numerik dari fungsi  $y = 2 + \sin(2\sqrt{x})$  pada interval  $[1, 6]$ .

**Contoh 10.5 :** Misalkan terdapat fungsi  $f(x) = 2 + \sin(2\sqrt{x})$ . Gunakan aturan komposit trapesium dengan 11 node untuk menghitung nilai integral dari  $f(x)$  pada interval  $[1, 6]$ .

**Solusi :** Untuk mendapatkan 11 titik contoh, interval integral akan dipartisi menjadi  $M = 10$  sub-interval dengan  $h = (6 - 1)/10 = 0.5$ . Dengan menerapkan Persamaan 10.14b, didapatkan hasil dari aturan komposit trapesium, yaitu

$$\begin{aligned} T(f, 0.5) &= \frac{0.5}{2}(f(1) + 2f(1.5) + 2f(2) + 2f(2.5) + \cdots + 2f(5) + 2f(5.5) + f(6)) \\ &= 8.19385457 \end{aligned} \quad \triangle$$

**Teorema 10.3 Aturan Komposit Simpson:** Misalkan bahwa interval  $[a, b]$  dipartisi menjadi  $2M$  sub-interval  $[x_k, x_{k+1}]$  dengan lebar  $h = (b - a)/(2M)$  menggunakan node yang sama panjang yaitu  $x_k = a + kh$  untuk  $k = 0, 1, 2, \dots, 2M$ . Aturan komposit Simpson untuk  $2M$  sub-interval dapat dituliskan dengan 3 persamaan yang ekuivalen, yaitu

$$S(f, h) = \frac{h}{3} \sum_{k=1}^{M} (f(x_{2k-2}) + 4f(x_{2k-1}) + f(x_k)) \quad (10.17a)$$

atau

$$S(f, h) = \frac{h}{3}(f_0 + 4f_1 + 2f_2 + 4f_3 + \cdots + 2f_{2M-2} + 4f_{2M-1} + f_{2M}) \quad (10.17b)$$

atau

$$S(f, h) = \frac{h}{3}(f(a) + f(b)) + \frac{2h}{3} \sum_{k=1}^{M-1} f(x_{2k}) + \frac{4h}{3} \sum_{k=1}^{M} f(x_{2k-1}) \quad (10.17c)$$

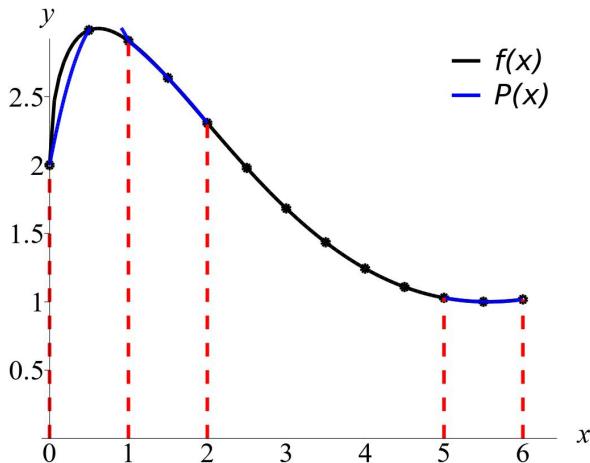
Persamaan tersebut merupakan hampiran dari integral  $f(x)$  pada selang  $[a, b]$ , sehingga dapat dituliskan sebagai

$$\int_a^b f(x) dx \approx S(f, h) \quad (10.18)$$

**Bukti** Dengan menerapkan aturan Simpson untuk setiap sub-interval  $[x_k, x_{k+1}]$  (lihat Gambar 10.6), sifat penjumlahan untuk integral akan menghasilkan

$$\int_a^b f(x) dx = \sum_{k=1}^{M} \int_{x_{2k-2}}^{x_{2k}} f(x) dx \approx \sum_{k=1}^{M} \frac{h}{3} (f(x_{2k-2}) + 4f(x_{2k-1}) + f(x_k)) \quad (10.19)$$

Karena  $h/2$  konstan, hukum distributif penjumlahan dapat diterapkan pada persamaan di atas, sehingga didapatkan Persamaan 10.17a. Persamaan 10.17b merupakan ekspansi dari Persamaan 10.17a. Persamaan 10.17c menunjukkan pengelompokan semua deret yang berada di tengah Persamaan 10.17b yang dikalikan 2 dan dikalikan 4. Bukti selesai. ■



**Gambar 10.6:** Aturan komposit Simpson untuk  $y = 2 + \sin(2\sqrt{x})$ .

**Contoh 10.6 :** Misalkan terdapat fungsi  $f(x) = 2 + \sin(2\sqrt{x})$ . Gunakan aturan komposit Simpson dengan 11 node untuk menghitung nilai integral dari  $f(x)$  pada interval  $[1, 6]$ .

**Solusi :** Untuk mendapatkan 11 titik contoh, interval integral akan dipartisi menjadi  $M = 5$  sub-interval dengan  $h = (6 - 1)/10 = 0.5$ . Dengan menerapkan Persamaan 10.17b, didapatkan hasil dari aturan komposit Simpson, yaitu

$$\begin{aligned} S(f, 0.5) &= \frac{0.5}{3}(f(1) + 4f(1.5) + 2f(2) + 4f(2.5) + \cdots + 2f(5) + 4f(5.5) + f(6)) \\ &= 8.18301550 \end{aligned} \quad \triangle$$

### 10.3 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Aturan Komposit** berisi program untuk mencari integral numerik dari fungsi  $f(x)$  pada interval  $[a, b]$ . Aturan komposit yang dipelajari adalah aturan komposit trapesium dan Simpson. Program ini secara *default* berisi 4 masukan, yaitu fungsi  $f(x)$ , titik ujung interval  $[a, b]$  dan banyaknya sub-interval  $M$ , serta 1 luaran, yaitu nilai integral numerik  $y$ . Berikut merupakan program untuk aturan komposit trapesium dan Simpson.

**Algoritma 10.1:** Aturan komposit trapesium

```
function komptrap(f, a, b, M)
    h = (b-a)/M;
    s = 0;
    for k = 1:M-1
        x = a+k*h;
        s = s+f(x);
    end
    y = h/2*(f(a)+f(b)+2*s)
end
```

### Algoritma 10.2: Aturan komposit Simpson

```

function kompsimp(f,a,b,M)
    h = (b-a)/(2*M);
    s1 = 0;    s2 = 0;
    for k = 1:M
        x = a + (2*k-1)*h;
        s1 = s1 + f(x);
    end
    for k = 1:M-1
        x = a + (2*k)*h;
        s2 = s2 + f(x);
    end
    y = h/3*(f(a)+f(b)+4*s1+2*s2)
end

```

#### 10.3.1 Formula Kuadratur

Terdapat 4 formula kuadratur Newton-Cotes yang akan dipelajari untuk menghitung nilai integral numerik yaitu aturan trapesium, Simpson, Simpson 3/8 dan Boole.

**Contoh 10.7 :** Nilai dari

$$\int_1^6 2 + \sin(2\sqrt{x}) dx$$

dapat dihampiri menggunakan 4 formula kuadratur dengan cara seperti berikut.

```

Inp: f = @(x) 2 + sin(2*sqrt(x))
      a = 1;
      b = 6;
Inp: % Trapesium
      h = (b-a)/1
      ytrap = h/2*(f(a)+f(a+h))
Inp: % Simpson
      h = (b-a)/2
      ysimp = h/3*(f(a)+4*f(a+h)+f(a+2*h))
Inp: % Simpson 3/8
      h = (b-a)/3
      ysimp3 = 3*h/8*(f(a)+3*f(a+h)+3*f(a+2*h)+f(a+3*h))
Inp: % Boole
      h = (b-a)/4
      yb = 2*h/45*(7*f(a)+32*f(a+h)+12*f(a+2*h)+32*f(a+3*h)+7*f(a+4*h))

Out: 9.81663746814197 # Solusi Aturan Trapesium
     8.056559493332967 # Solusi Aturan Simpson
     8.122115320629923 # Solusi Aturan Simpson 3/8
     8.178034766424261 # Solusi Aturan Boole

```

Jika diketahui nilai eksak dari masalah integral di atas adalah 8.183479207, maka hampiran integral yang paling mendekati adalah aturan Boole. Namun, solusi aturan Boole masih memiliki galat yang besar yaitu 0.005.

**Soal Latihan:** Hitunglah nilai hampiran dari

$$\int_0^3 1 + e^{-x} \tan^{-1}(4x) dx$$

menggunakan 4 formula kuadratur dengan cara seperti pada Contoh 10.7.

### 10.3.2 Aturan Komposit

Aturan kuadratur Newton-Cotes pada sub-bab sebelumnya memiliki galat yang cukup tinggi terhadap nilai solusi eksaknya dan aturan tersebut tidak dapat mengatur galat yang dihasilkan. Untuk mengatasinya, dilakukan pembagian interval integral menjadi beberapa sub-interval. Metode ini disebut dengan aturan komposit. Pada praktikum ini, hanya akan dipelajari 2 aturan komposit, yaitu aturan komposit trapesium dan Simpson.

**Contoh 10.8 :** Diberikan fungsi

$$f(x) = 2 + \sin(2\sqrt{x})$$

Untuk menghitung nilai dari  $\int_1^6 f(x) dx$  dengan 11 titik contoh menggunakan aturan komposit **trapesium** dan **Simpson**, diperlukan jumlah sub-interval sebanyak  $M = 10$ . Dengan demikian, hasil hampiran integral menggunakan aturan komposit trapesium dan Simpson dengan 11 titik contoh adalah sebagai berikut.

```
Inp: a = 1; b = 6;
f(x) = 2+sin(2*sqrt(x));
y = komptrap(f,a,b,10) # Komposit Trapesium
y = kompsimp(f,a,b,10) # Komposit Simpson

Out: 8.193854565172531 # Komposit Trapesium
     8.183447496636239 # Komposit Simpson
```

Jika diketahui nilai eksak dari masalah integral di atas adalah 8.183479207, maka hampiran integral dengan aturan komposit trapesium dan Simpson dengan 10 sub-interval memiliki galat masing-masing sebesar 0.010375 dan 0.0000317.

**Soal Latihan:** Gunakan aturan komposit **trapesium** dan **Simpson** untuk menyelesaikan

$$\int_0^3 \tan^{-1}(x) dx$$

dengan 21 titik contoh pada selang  $[0, 3]$  menggunakan cara seperti pada Contoh 10.8.

### 10.3.3 Analisis Galat Aturan Komposit

Untuk mengetahui kompleksitas komputasi dari aturan komposit trapesium dan Simpson, akan dipelajari analisis galat dari masing masing aturan komposit sebagai berikut.

**Contoh 10.9 (Analisis Galat Aturan Komposit Trapesium):** Diberikan integral-tentu

$$\int_1^6 2 + \sin(2\sqrt{x}) dx$$

Berikut merupakan langkah-langkah melakukan analisis galat aturan trapesium untuk menyelesaikan integral-tentu tersebut dan menentukan kompleksitasnya menggunakan sub-interval  $M = 5, 10, 20, 40, 80$ , dan  $160$ , jika diketahui  $\int_1^6 f(x) dx \approx 8.1834792077$ .

**Langkah 1:** Penghitungan nilai  $\int_1^6 f(x) dx$  menggunakan aturan komposit trapesium pada Program 10.1 dengan sub-interval  $M = 5, 10, 20, 40, 80$ , dan 160.

```
Inp: a = 1; b = 6;
f(x) = 2+sin(2*sqrt(x));
M = [5,10,20,40,80,160];
h = (b-a). / M;
y = Array{Float64} (undef, length(M), 1)
for i = 1:length(M)
    y[i] = komptrap(f, a, b, M[i]);
end
```

**Langkah 2:** Penghitungan nilai galat aturan komposit trapesium.

```
Inp: yeks = 8.1834792077;
err = abs.(y.-yeks);
P = [M h y err]
```

**Tabel 10.1:** Aturan komposit trapesium untuk  $f(x) = 2 + \sin(2\sqrt{x})$  pada selang  $[1, 6]$ .

$M$	$h$	Nilai integral, $y$	Galat, $E$
5	1.00000	8.226371779	0.042892571
10	0.50000	8.193854565	0.010375357
20	0.25000	8.186049264	0.002570056
40	0.12500	8.184120192	0.000640984
80	0.06250	8.183639357	0.000160150
160	0.03125	8.183519239	0.000040031

Berdasarkan hasil yang diperoleh, berkurangnya nilai  $h$  menjadi  $1/2$  kali dari nilai  $h$  sebelumnya mengakibatkan perubahan galat menjadi sekitar  $1/4$  kali dari galat sebelumnya. Dengan demikian, aturan komposit trapesium memiliki kompleksitas sebesar  $O(h^2)$ .  $\triangle$

**Contoh 10.10 Analisis Galat Aturan Komposit Simpson:** Diberikan integral-tentu

$$\int_1^6 2 + \sin(2\sqrt{x}) dx$$

Berikut merupakan langkah-langkah melakukan analisis galat aturan komposit Simpson untuk menghampiri integral-tentu dan menentukan kompleksitasnya menggunakan sub-interval  $M = 5, 10, 20, 40, 80$ , dan 160, jika diketahui  $\int_1^6 f(x) dx \approx 8.1834792077$ .

**Langkah 1:** Penghitungan nilai  $\int_1^6 f(x) dx$  menggunakan aturan komposit Simpson pada Program 10.2 dengan sub-interval  $M = 5, 10, 20, 40, 80$ , dan 160.

```
Inp: a = 1; b = 6;
f(x) = 2+sin(2*sqrt(x));
M = [5,10,20,40,80,160];
h = (b-a). / M;
y = Array{Float64} (undef, length(M), 1)
for i = 1:length(M)
    y[i] = kompsimp(f, a, b, M[i]);
end
```

**Langkah 2:** penghitung nilai galat.

```
Inp: yeks = 8.1834792077;
err = abs.(y.-yeks);
P = [M h y err]
```

**Tabel 10.2:** Aturan komposit Simpson untuk  $f(x) = 2 + \sin(2\sqrt{x})$  pada selang  $[1, 6]$ .

M	h	Nilai integral, y	Galat, E
5	1.00000	8.183015494	0.000463714
10	0.50000	8.183447497	0.000031711
20	0.25000	8.183477168	0.000002040
40	0.12500	8.183479079	0.000000129
80	0.06250	8.183479200	0.000000008
160	0.03125	8.183479207	0.000000001

Berdasarkan hasil yang diperoleh, berkurangnya nilai  $h$  menjadi  $1/2$  kali dari nilai  $h$  sebelumnya akan mengakibatkan perubahan galat menjadi sekitar  $1/16$  kali dari galat sebelumnya. Dengan demikian, aturan komposit Simpson memiliki kompleksitas sebesar  $O(h^4)$ .

**Soal Latihan:** Gunakan aturan komposit trapesium dan Simpson untuk menyelesaikan

$$\int_0^3 \tan^{-1}(x) dx$$

dengan sub-interval  $M = 10, 20, \dots, 100$ . Ulangi langkah-langkah pada Contoh 10.9 dan Contoh 10.10 untuk mengetahui kompleksitas dari aturan komposit trapesium dan Simpson, jika diketahui  $\int_0^3 \tan^{-1}(x) dx \approx 2.59584477$ .

## 10.4 Latihan-latihan

### 10.4.1 Ulasan Materi

1. Jelaskan ide dasar dari pencarian integral melalui pendekatan numerik menggunakan formula kuadratur!
2. Jelaskan apa yang dimaksud dengan derajat kepresision dalam integral numerik! Berikan contohnya.
3. Sebutkan dan jelaskan 4 formula kuadratur Newton-Cotes pertama!
4. Jelaskan penurunan rumus intergral menjadi formula kuadratur berupa aturan trapesium menggunakan pendekatan interpolasi titik!
5. Sebutkan dan jelaskan derajat kepresision serta galat pemotongan (*truncation error*) dari keempat formula kuadratur Newton-Cotes pertama!
6. Jelaskan ide dasar dari aturan komposit trapesium dan Simpson!
7. Tunjukkan dan jelaskan bahwa kompleksitas komputasi dari aturan komposit trapesium dan Simpson masing-masing adalah  $O(h^2)$  dan  $O(h^4)$ !

### 10.4.2 Soal Pemrograman

1. Gunakan aturan komposit **trapesium** dan **Simpson** untuk menyelesaikan

$$\int_0^2 \ln\left(\frac{e^x+2}{\cos(x)+2}\right) dx$$

Untuk subinterval  $M = 5, 7, 9$ . Tuliskan hasil yang diperoleh dalam suatu tabel kemudian bandingkan.

2. Gunakan aturan komposit **trapesium** dan **Simpson** untuk menyelesaikan

$$\int_{\pi/4}^{\pi/2} \frac{\cos(x) \log(\sin x)}{\sin^2(x) + 1} dx$$

Untuk subinterval  $M = 2, 4, \dots, 20$ . Tuliskan hasil yang diperoleh dalam suatu tabel kemudian bandingkan.

3. Gunakan aturan komposit **trapesium** dan **Simpson** untuk menyelesaikan

$$\int_{-\pi}^{\pi} \frac{250x^2}{\cosh^2(500[x-t])} dx$$

Untuk  $t = 0.5$  dan  $t = 1$  serta subinterval  $M = 4, 8, 12, \dots, 80$ . Tuliskan hasil yang diperoleh dalam suatu tabel. bandingkan.

4. Berapakah nilai ukuran langkah  $h$  yang diperlukan atutan komposit trapesium dan Simpson untuk menghitung

$$\int_0^{\pi} x \sin(x) dx$$

hingga mempunyai akurasi  $10^{-6}$ .

5. Berapakah nilai ukuran langkah  $h$  yang diperlukan atutan komposit trapesium dan Simpson untuk menghitung

$$\int_{0.2}^{1.1} e^{2x} dx$$

hingga mempunyai akurasi  $10^{-4}$ .

6. Berapakah nilai ukuran langkah  $h$  yang diperlukan atutan komposit trapesium dan Simpson untuk menghitung

$$\int_0^3 x^2 \cos(x^2 - 1) dx$$

hingga mempunyai akurasi  $10^{-3}$ .

7. Perhatikan data berikut

$x$	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8
$f(x)$	1.544	1.667	1.811	1.972	2.152	2.351	2.576	2.828	3.107

- (a) Gunakan aturan komposit **trapesium** dengan  $h = 0.1, h = 0.2$  dan  $h = 0.4$  untuk menghitung  $\int_1^{1.8} f(x) dx$ .
- (b) Gunakan aturan komposit **Simpson** dengan  $h = 0.1, h = 0.2$  dan  $h = 0.4$  untuk menghitung  $\int_1^{1.8} f(x) dx$ .

8. Perhatikan data berikut

$x$	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0
$f(x)$	4.953	6.050	7.389	9.025	11.023	13.464	16.445	20.086

(a) Gunakan aturan komposit **trapesium** dengan  $h = 0.2$  untuk menghitung

$$\int_{1.6}^3 f(x) dx$$

(b) Gunakan aturan komposit **Simpson** dengan  $h = 0.2$  untuk menghitung

$$\int_{1.6}^3 f(x) dx$$

(c) Bandingkan hasil yang diperoleh dengan solusi eksaknya yaitu 15.1325045.

9. Aturan komposit dapat diterapkan untuk menghitung integral dari suatu fungsi yang hanya diketahui barisan titik-titik. Gunakan pendekatan aturan komposit **trapesium** untuk menghitung nilai integral dari suatu pasangan terurut.

$$(k, \sqrt{k^2 + 1}) , k = 0, 1, 2, \dots, 13$$

10. Aturan komposit dapat diterapkan untuk menghitung integral dari suatu fungsi yang hanya diketahui barisan titik-titik. Gunakan pendekatan aturan komposit **Simpson** untuk menghitung nilai integral dari suatu pasangan terurut.

$$(k, \sqrt{k^2 + 1}) , k = 0, 1, 2, \dots, 13$$

11. Aturan komposit dapat diterapkan untuk menghitung integral dari suatu fungsi yang hanya diketahui barisan titik-titik. Gunakan pendekatan aturan komposit **trapesium** untuk menghitung nilai integral dari suatu pasangan terurut.

$$(\sqrt{k^2 + 1}, k^{1/3}) , k = 0, 1, 2, \dots, 13$$

*Catatan:* subinterval memiliki ukuran lebar  $h$  yang berbeda sehingga hitunglah masing-masing subinterval.

12. Hitung nilai integral numerik untuk sejumlah fungsi  $f(x)$  berikut ini.

(a)  $\int_{-1}^2 x^2 dx$

(b)  $\int_0^\pi \sin(2x) dx$

(c)  $\int_0^4 xe^{x^2} dx$

Gunakan beberapa formula kuadratur yang berbeda, yaitu Trapesium, Simpson dan Boole komposit dan tentukan berapa nilai ukuran langkah  $h$  yang harus digunakan untuk masing-masing formula, agar nilai integral numerik tersebut memiliki nilai galat absolut yang tak lebih dari  $10^{-8}$ .

13. Diberikan integral berikut.

$$A = \int_{-1}^1 \frac{1}{x^2 + 1} dx$$

- (a) Hitung nilai integral tersebut secara numerik menggunakan aturan komposit trapesium, untuk beberapa ukuran langkah  $h = \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ , dan  $\frac{1}{64}$ .
- (b) Bila diketahui nilai eksak integral tersebut adalah  $\pi/2$ , hitung nilai absolut galat untuk masing-masing nilai integral yang diperoleh pada poin (a).
- (c) Dengan mengamati perubahan nilai  $h$  terhadap perubahan absolut galat yang diperoleh, tentukan berapakah orde galat komposit trapesium?

14. Diberikan integral berikut.

$$A = \int_{-1}^1 \frac{1}{x^2 + 1} dx$$

- (a) Hitung nilai integral tersebut secara numerik menggunakan aturan komposit Simpson, untuk beberapa ukuran langkah  $h = \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ , dan  $\frac{1}{64}$ .
- (b) Bila diketahui nilai eksak integral tersebut adalah  $\pi/2$ , hitung nilai absolut galat untuk masing-masing nilai integral yang diperoleh pada poin (a).
- (c) Dengan mengamati perubahan nilai  $h$  terhadap perubahan absolut galat yang diperoleh, tentukan berapakah orde galat komposit Simpson?

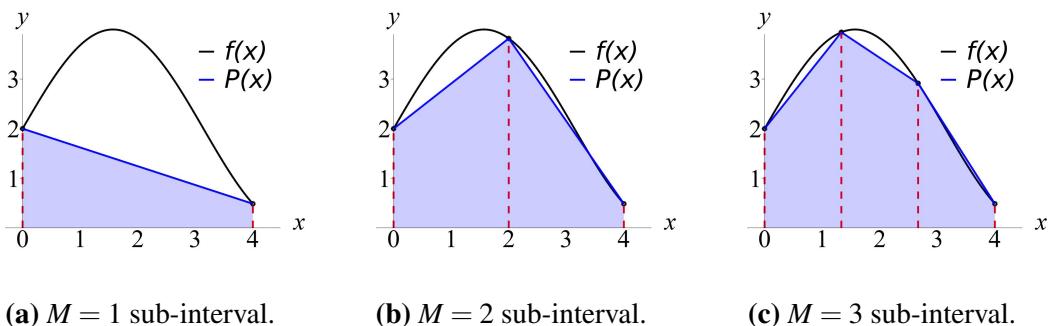
---

# BAB 11

---

## INTEGRAL NUMERIK 2

Pada Bab sebelumnya, telah dibahas aturan kuadratur yang menjadi dasar dari penghitungan integral numerik. Pada Bab tersebut, telah diketahui bahwa aturan komposit akan menghasilkan nilai dan akurasi yang berbeda, apabila banyaknya sub-interval  $[x_{k-1}, x_k]$  yang digunakan berbeda. Sebagai contoh, misalkan bahwa akan dicari solusi dari integral-tentu  $\int_1^5 2 + 2 \sin(x) dx$  menggunakan aturan komposit trapesium dengan beberapa sub-interval yaitu  $M = 1, 2, 3$ . Luas area trapesium dari masing-masing sub-interval dapat dilihat pada Gambar 11.1 berikut.



(a)  $M = 1$  sub-interval.      (b)  $M = 2$  sub-interval.      (c)  $M = 3$  sub-interval.

**Gambar 11.1:** Perbandingan luas area aturan komposit Simpson dengan sub-interval yang berbeda untuk  $2 + 2 \sin(x)$  pada  $[0, 4]$ .

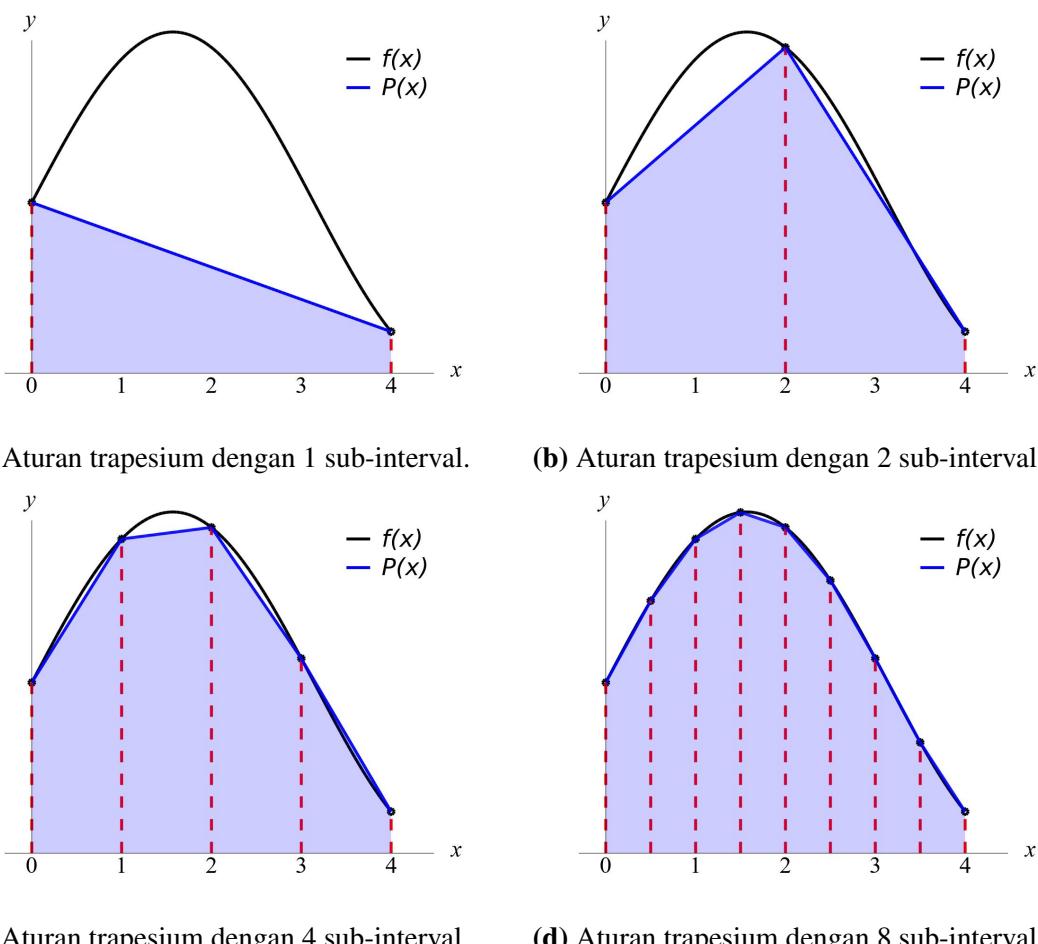
Hasil integral numerik untuk setiap sub-interval  $M = 1, 2, 3$  masing-masing adalah  $T = 4.97279$ ,  $T = 10.12358$ , dan  $T = 10.80216$ . Jika solusi eksak dari integral-tentu adalah  $\int_1^5 2 + 2 \sin(x) dx = 11.307287243\dots$ , maka nilai galat mutlak dari masing-masing hampiran adalah  $E = 6.33449$ ,  $E = 1.18370$ , dan  $E = 0.50513$ .

Berdasarkan hasil yang diperoleh, semakin banyak sub-interval  $M$  yang digunakan, maka nilai galat hampiran semakin kecil, dengan kata lain hampiran memiliki nilai akurasi yang semakin baik. Oleh karena itu, pertanyaan selanjutnya untuk aturan komposit adalah seberapa banyak sub-interval yang dibutuhkan agar nilai hampiran memiliki galat yang cukup kecil dan dapat diterima sebagai solusi dari integral-tentu. Kriteria suatu solusi dapat diterima biasanya ditentukan menggunakan nilai toleransi  $\delta$ . Semakin kecil nilai  $\delta$  yang ditetapkan, maka semakin akurat pula nilai hampiran solusi yang dihasilkan. Untuk menjawab pertanyaan tersebut, Bab ini akan membahas beberapa aturan integral numerik yang dapat digunakan untuk menghasilkan solusi numerik dengan nilai galat yang memenuhi toleransi  $\delta$  yang ditetapkan.

## 11.1 Aturan Rekursif dan Integral Romberg

Pada sub-bab ini, akan dibahas cara untuk menghitung hampiran Simpson menggunakan kombinasi linear khusus dari aturan trapesium. Nilai hampiran akan menghasilkan akurasi yang lebih baik, apabila hampiran memiliki banyak sub-interval. Untuk menentukan banyaknya sub-interval yang akan digunakan, proses sekuensial akan melakukan integral numerik menggunakan 2 sub-interval, 4 sub-interval, dan seterusnya hingga nilai akurasi yang dibutuhkan dapat tercapai.

Pertama-tama, akan dibentuk suatu barisan  $\{T(J)\}$  dari aturan trapesium. Dengan menggandakan sub-interval yang digunakan, banyaknya evaluasi fungsi yang diperlukan akan menjadi sekitar 2 kali lipat dari sebelumnya. Perubahan banyaknya evaluasi fungsi tersebut dapat dilihat pada Gambar 11.2 berikut.



**Gambar 11.2:** Luas area di bawah kurva pada aturan trapesium dengan sub-interval yang berbeda.

Berdasarkan Gambar 11.2 di atas, jika akan dihitung nilai aturan trapesium untuk setiap sub-interval, maka akan diperoleh cara penghitungan seperti berikut.

Untuk sub-interval sebanyak  $M = 1$ , nilai aturan trapesium adalah

$$T(f, 4) = \frac{4}{2}(f(0) + f(4))$$

Untuk sub-interval sebanyak  $M = 2$ , nilai aturan trapesium adalah

$$T(f, 2) = \frac{2}{2}(f(0) + 2f(2) + f(4))$$

Untuk sub-interval sebanyak  $M = 4$ , nilai aturan trapesium adalah

$$T(f, 1) = \frac{1}{1}2(f(0) + 2f(1) + 2f(2) + 2f(3) + f(4))$$

Untuk sub-interval sebanyak  $M = 8$ , nilai aturan trapesium adalah

$$\begin{aligned} T(f, 0.5) = & \frac{0.5}{2}(f(0) + 2f(0.5) + 2f(1) + 2f(1.5) + 2f(2) + 2f(2.5) + 2f(3) \\ & + 2f(3.5) + f(4)) \end{aligned}$$

Berdasarkan hasil di atas, penghitungan aturan trapesium pada setiap sub-interval akan terjadi pengulangan evaluasi fungsi  $f(x)$ . Misalkan, untuk jumlah sub-interval  $M = 1$ , aturan trapesium mengevaluasi fungsi  $f(0)$  dan  $f(4)$ . Hal ini terulang pada aturan trapesium dengan  $M = 2$ , yaitu terjadi evaluasi fungsi  $f(0)$  dan  $f(4)$ . Begitu pula pada  $M = 4$  dan  $M = 8$ . Pengulangan ini akan mengakibatkan banyaknya waktu komputasi jika fungsi yang dievaluasi rumit dan rekursif  $M$  bernilai banyak. Untuk mengatasi evaluasi fungsi yang berlebihan tersebut, berikut merupakan hubungan aturan trapesium yang berurutan.

**Teorema 11.1 :** Misalkan bahwa  $J \geq 1$  dan titik  $\{x_k = a + kh\}$  membagi interval  $[a, b]$  menjadi  $2^J = 2M$  sub-interval dengan langkah  $h = (b - a)/2^J$ . Aturan trapesium  $T(f, h)$  dan  $T(f, 2h)$  akan memenuhi hubungan berikut.

$$T(f, h) = \frac{T(f, 2h)}{2} + h \sum_{k=1}^M f(x_{2k-1}) \quad (11.1)$$

■

### Aturan Rekursif Trapesium

**Definisi 11.1 Barisan aturan trapesium:** Defisikan  $T(0) = \frac{h}{2}(f(a) + f(b))$  yaitu aturan trapesium dengan ukuran langkah  $h = b - a$ . Selanjutnya untuk setiap  $J \geq 1$ , definisikan  $T(J) = T(f, h)$  yaitu aturan trapesium dengan ukuran langkah  $h = (b - a)/2^J$ . ■

**Corollary 11.1 Aturan Rekursif Trapesium:** Dimulai dengan  $T(0) = \frac{h}{2}(f(a) + f(b))$ , kemudian suatu barisan dari aturan trapesium  $\{T(J)\}$  dapat dihasilkan oleh formula rekursif

$$T(J) = \frac{T(J-1)}{2} + h \sum_{k=1}^M f(x_{2k-1}) \quad \text{untuk } J = 1, 2, \dots \quad (11.2)$$

dengan  $h = (b - a)/2^J$  dan  $\{x_k = a + kh\}$ .

**Bukti** Untuk node genap yaitu  $x_0 < x_2 < \dots < x_{2M-2} < x_{2M}$ , gunakan aturan trapesium dengan ukuran langkah  $2h$ , yaitu

$$T(J-1) = \frac{2h}{2}(f_0 + 2f_2 + 2f_4 + \dots + 2f_{2M-2} + 2f_{2M}) \quad (11.3)$$

Untuk semua node yaitu  $x_0 < x_1 < x_2 < \dots < x_{2M-2} < x_{2M-1} < x_{2M}$ , gunakan aturan trapesium dengan ukuran langkah  $h$ , yaitu

$$T(J) = \frac{h}{2}(f_0 + 2f_1 + 2f_2 + \dots + 2f_{2M-1} + 2f_{2M}) \quad (11.4)$$

Kumpulkan evaluasi fungsi node ganjil pada Persamaan 11.4,yaitu

$$T(J) = \frac{h}{2}(f_0 + 2f_2 + \dots + 2f_{2M-2} + 2f_{2M}) + h \sum_{k=1}^M f_{2k-1} \quad (11.5)$$

Substitusi Persamaan 11.3 ke Persamaan 11.5, sehingga didapatkan hasil Persamaan 11.2. Bukti selesai. ■

**Contoh 11.1 :** Gunakan aturan rekursif trapesium untuk menghitung nilai hampiran  $T(0)$ ,  $T(1)$ ,  $T(2)$ , dan  $T(3)$  untuk integral-tentu  $\int_1^5 dx/x$ .

**Solusi :** Dimulai dengan mendefinisikan  $T(0)$  yaitu aturan trapesium dengan  $h = 4$ . Ingat bahwa  $f_k = f(x_k)$  dengan  $x_k = a + kh$ .

$$T(0) = \frac{h}{2}(f_0 + f_1) = \frac{4}{2}(f(1) + f(5)) = 2.4$$

Hitung nilai  $T(1)$ ,  $T(2)$ , dan  $T(3)$  menggunakan aturan rekursif trapesium, yaitu

$$T(1) \text{ dengan } h = 2: \quad T(1) = \frac{T(0)}{2} + hf_1 = \frac{2.4}{2} + 2f(3) = 1.86666$$

$$\begin{aligned} T(2) \text{ dengan } h = 1: \quad T(2) &= \frac{T(1)}{2} + h(f_1 + f_3) \\ &= \frac{1.86666}{2} + 1(f(2) + f(4)) = 1.68333 \end{aligned}$$

$$\begin{aligned} T(3) \text{ dengan } h = 0.5: \quad T(3) &= \frac{T(2)}{2} + h(f_1 + f_3 + f_5 + f_7) \\ &= \frac{1.68333}{2} + 0.5(f(1.5) + f(2.5) + f(3.5) + f(4.5)) \\ &= 1.628968 \end{aligned}$$

△

### Aturan Rekursif Simpson.

Selanjutnya, ketika aturan trapesium dihitung dengan ukuran langkah  $2h$  dan  $h$  masing-masing akan menghasilkan nilai  $T(f, 2h)$  dan  $T(f, h)$ . Aturan Simpson dapat dihasilkan dari kombinasi linear dari kedua nilai tersebut, yaitu

$$S(f, h) = \frac{4T(f, h) - T(f, 2h)}{3}$$

**Teorema 11.2 Aturan rekursif Simpson:** Misalkan bahwa  $\{T(J)\}$  merupakan barisan aturan trapesium yang dihasilkan oleh Corollary 11.1. Jika  $J \geq 1$  dan  $S(J)$  merupakan aturan Simpson untuk  $2^J$  sub-interval dari  $[a, b]$ , maka  $S(J)$  dan aturan trapesium  $T(J-1)$  dan  $T(J)$  memenuhi hubungan

$$S(J) = \frac{4T(J) - T(J-1)}{3} \quad \text{untuk } J = 1, 2, \dots \quad (11.6)$$

**Bukti** Aturan trapesium  $T(J)$  dengan ukuran langkah  $h$  adalah

$$T(J) = \frac{h}{2}(f_0 + 2f_1 + 2f_2 + \cdots + 2f_{2M-2} + 2f_{2M-1} + f_{2M}) \quad (11.7)$$

dan aturan trapesium  $T(J-1)$  dengan ukuran langkah  $2h$  adalah

$$T(J-1) = h(f_0 + 2f_2 + 2f_4 + \cdots + 2f_{2M-2} + f_{2M}) \quad (11.8)$$

Kalikan Persamaan 11.7 dengan 4, kemudian kurangkan dengan Persamaan 11.8, sehingga didapatkan

$$4T(J) - T(J-1) = h(f_0 + 4f_1 + 2f_2 + \cdots + 2f_{2M-2} + 4f_{2M-1} + f_{2M}) \quad (11.9)$$

Bagi masing-masing ruas dengan 3, sehingga diperoleh

$$\frac{4T(J) - T(J-1)}{3} = \frac{h}{3}(f_0 + 4f_1 + 2f_2 + \cdots + 2f_{2M-2} + 4f_{2M-1} + f_{2M}) \quad (11.10)$$

Ruas kanan merupakan aturan Simpson dengan ukuran langkah  $h$  yang dapat dinotasikan sebagai  $S(J)$ . Bukti selesai. ■

**Contoh 11.2 :** Gunakan aturan rekursif Simpson untuk menghitung nilai hampiran  $S(1)$ ,  $S(2)$ , dan  $S(3)$  dari integral-tentu pada Contoh 11.1.

**Solusi :** Dengan aturan rekursif Simpson pada Persamaan 11.6 dan hasil Contoh 11.1, nilai hampiran aturan simpson adalah

$$\begin{aligned} S(1) &= \frac{4T(1) - T(0)}{3} = \frac{4(1.866666) - 2.4}{3} = 1.688888 \\ S(2) &= \frac{4T(2) - T(1)}{3} = \frac{4(1.683333) - 1.866666}{3} = 1.622222 \\ S(3) &= \frac{4T(3) - T(2)}{3} = \frac{4(1.628968) - 1.683333}{3} = 1.610846 \end{aligned} \quad \triangle$$

### Aturan Rekursif Boole.

Teorema selanjutnya akan menunjukkan hubungan antara aturan Simpson terhadap aturan Boole.

**Teorema 11.3 Aturan rekursif Boole:** Misalkan bahwa  $\{S(J)\}$  merupakan barisan aturan Simpson yang dihasilkan oleh Teorema 11.2. Jika  $J \geq 2$  dan  $B(J)$  merupakan aturan Boole untuk  $2^J$  sub-interval dari  $[a, b]$ , maka  $B(J)$  dan aturan Simpson  $S(J-1)$  dan  $S(J)$  memenuhi hubungan

$$B(J) = \frac{16S(J) - S(J-1)}{15} \quad \text{untuk } J = 2, 3, \dots \quad (11.11)$$

■

**Contoh 11.3 :** Gunakan aturan rekursif Boole untuk menghitung nilai hampiran  $B(2)$  dan  $B(3)$  dari integral-tentu pada Contoh 11.1.

**Solusi :** Dengan aturan rekursif Boole pada Persamaan 11.11 dan hasil Contoh 11.2, nilai hampiran aturan Boole adalah

$$\begin{aligned} B(2) &= \frac{16S(2) - S(1)}{15} = \frac{16(1.622222) - 1.688888}{15} = 1.617777 \\ B(3) &= \frac{16S(3) - S(2)}{15} = \frac{16(1.610846) - 1.622222}{15} = 1.610088 \end{aligned} \quad \triangle$$

### Integral Romberg

Berdasarkan aturan rekursif trapesium, Simpson, dan Boole, masing-masing terdapat hubungan dari satu aturan ke aturan yang lainnya. Dilihat dari kompleksitas komputasi, pola hubungan ini mengembangkan aturan trapesium yang memiliki ordo  $O(h^2)$  menjadi aturan Simpson yang memiliki ordo  $O(h^4)$ . Lebih lanjut lagi, pola hubungan ini juga mengembangkan aturan Simpson yang memiliki ordo  $O(h^4)$  menjadi aturan Boole yang memiliki ordo  $O(h^6)$ . Berdasarkan pola tersebut, setiap tingkatan pengembangan akan meningkatkan ordo galat dari  $O(h^{2N})$  menjadi  $O(h^{2N+2})$ . Proses ini dinamakan **integral Romberg**. Pola umum integral Romberg dapat dilihat pada Lemma 11.1 berikut

**Lema 11.1 Ekstrapolasi Richardson untuk integral Romberg:** Diberikan 2 hampiran  $R(2h, K - 1)$  dan  $R(h, K - 1)$  untuk nilai  $Q$  yang memenuhi persamaan

$$Q = R(h, K - 1) + O(h^{2K}) \quad (11.12)$$

dan

$$Q = R(2h, K - 1) + O(h^{2K}) \quad (11.13)$$

Dengan demikian, pengembangan hampiran tersebut adalah

$$Q = \frac{4^K R(h, K - 1) - R(h, K - 1)}{4^K - 1} + O(h^{2K+2}) \quad (11.14)$$

■

Untuk mempermudah penulisan, integral Romberg biasanya dituliskan dalam bentuk tabel. Berikut merupakan urutan langkah untuk membentuk tabel integral Romberg tersebut.

**Definisi 11.2 :** Definisikan barisan  $\{R(J, K) : J \geq K\}_{k=0}^{\infty}$  dari formula kuadratur untuk  $f(x)$  pada interval  $[a, b]$  sebagai berikut.

- |                  |   |
|------------------|---|
| $R(J, 0) = T(J)$ | untuk $J \geq 0$ , adalah barisan aturan rekursif trapesium |
| $R(J, 1) = S(J)$ | untuk $J \geq 1$ , adalah barisan aturan rekursif Simpson   |
| $R(J, 2) = B(J)$ | untuk $J \geq 2$ , adalah barisan aturan rekursif Boole     |

■

Pembentukan tabel dimulai dengan mendefinisikan  $\{R(J, 0)\}$  yang akan digunakan untuk membentuk  $\{R(J, 1)\}$ . Dari  $\{R(J, 1)\}$  tersebut, akan dibentuk barisan  $\{R(J, 2)\}$  berdasarkan Persamaan 11.14, yaitu

$$\begin{aligned} R(J, 1) &= \frac{4^1 R(J, 0) - R(J - 1, 0)}{4^1 - 1} \quad \text{untuk } J \geq 1 \\ R(J, 2) &= \frac{4^2 R(J, 1) - R(J - 1, 1)}{4^2 - 1} \quad \text{untuk } J \geq 2 \end{aligned}$$

yang merupakan aturan rekursif Simpson dan Boole yang terdapat pada Persamaan 11.6 dan 11.11 dengan notasi integral Romberg. Selanjutnya, aturan umum untuk mengembangkan nilai  $R(J, 2)$  adalah

$$R(J, K) = \frac{4^K R(h, K - 1) - R(h, K - 1)}{4^K - 1} \quad \text{untuk } J \geq K$$

Untuk mempermudah penghitungan, nilai  $R(J, K)$  dapat ditulis seperti pada Tabel 11.1 berikut.

**Tabel 11.1:** Tabel integral Romberg.

$J$	$R(J, 0)$	$R(J, 1)$	$R(J, 2)$	$R(J, 3)$	$R(J, 4)$
	Aturan Trapesium	Aturan Simpson	Aturan Boole	Perbaikan Ketiga	Perbaikan Keempat
0	$R(0, 0)$				
1	$R(1, 0)$	$R(1, 1)$			
2	$R(2, 0)$	$R(2, 1)$	$R(2, 2)$		
3	$R(3, 0)$	$R(3, 1)$	$R(3, 2)$	$R(3, 3)$	
4	$R(4, 0)$	$R(4, 1)$	$R(4, 2)$	$R(4, 3)$	$R(4, 4)$

**Contoh 11.4 :** Gunakan integral Romberg untuk mencari nilai hampiran dari integral tentu

$$\int_0^{\pi/2} (x^2 + x + 1) \cos(x) dx = -2 + \frac{\pi}{2} + \frac{\pi}{4} = 2.038197427067\dots$$

**Solusi :** Hasil penghitungan tabel integral Romberg dapat dilihat pada Tabel 11.2 berikut.

**Tabel 11.2:** Hasil integral Romberg Contoh 11.4.

$J$	$R(J, 0)$	$R(J, 1)$	$R(J, 2)$	$R(J, 3)$	$R(J, 4)$
0	0.78539816				
1	1.72681266	2.04061749			
2	1.96053417	2.03844134	2.03829626		
3	2.01879395	2.03821388	2.03819871	2.03819716	
4	2.03334734	2.03819847	2.03819745	2.03819743	2.03819743
5	2.03698495	2.03819749	2.03819743	2.03819743	2.03819743

Berdasarkan hasil yang diperoleh, setiap kolom menghasilkan nilai yang konvergen menuju solusi eksaknya, yaitu 2.038197427067.... Secara umum, hasil kolom sebelah kanan lebih cepat konvergen dibandingkan hasil kolom di sebelah kirinya. Contohnya, hasil dari kolom aturan Simpson lebih cepat untuk konvergen dibandingkan kolom aturan trapesium.  $\triangle$

**Teorema 11.4 Kepresisan integral Romberg:** Asumsikan bahwa  $f \in C^{2K+2}[a, b]$ . Truncation error untuk hampiran Romberg dapat dihitung dengan formula

$$\int_a^b f(x) dx = R(J, K) + O(h^{2K+2}) \quad (11.15)$$

dengan  $h = (b - a)/2^J$ .  $\blacksquare$

## 11.2 Kuadratur Adaptif

Aturan komposit kuadratur mengharuskan menggunakan titik node yang berjarak sama. Khususnya, penggunaan ukuran langkah  $h$  yang kecil akan digunakan pada seluruh interval  $[a, b]$  untuk menentukan akurasi akhir dari hampiran. Hal ini tidak mempertimbangkan bahwa sebagian sub-interval dari kurva  $f(x)$  memiliki nilai fungsi yang lebih bervariasi dan memerlukan lebih banyak perhatian dibandingkan dengan sebagian sub-interval yang lain. Dengan demikian, penggunaan metode yang dapat menyesuaikan ukuran langkah  $h$  bergantung pada variasi nilai fungsi  $f(x)$  menjadi sangat diperlukan. Teknik Metode ini sering disebut dengan **kuadratur adaptif**. Kuadratur adaptif yang akan dipelajari berbasis pada aturan Simpson.

Aturan Simpson pada sub-interval  $[a_k, b_k]$  didefinisikan dengan

$$S(a_k, b_k) = \frac{h}{3}(f(a_k) + 4f(c_k) + f(b_k)) \quad (11.16)$$

dengan  $c_k$  adalah titik tengah  $[a_k, b_k]$  dan  $h = (b_k - a_k)/2$ . Selanjutnya, jika  $f \in C^4[a_k, b_k]$ , maka terdapat nilai  $d_1 \in [a_k, b_k]$  sedemikian sehingga

$$\int_{a_k}^{b_k} f(x) dx = S(a_k, b_k) - h^5 \frac{f^{(4)}(d_1)}{90} \quad (11.17)$$

### Perbaikan Aturan Simpson

Perbaikan dilakukan dengan cara membagi sub-interval  $[a_k, b_k]$  menjadi 2 sub-interval  $[a_{k1}, b_{k1}]$  dan  $[a_{k2}, b_{k2}]$ , hitung masing-masing sub-interval tersebut dengan aturan komposit Simpson. Dengan dibaginya sub-interval menjadi 2 sub-interval, ukuran langkah aturan Simpson menjadi  $h/2$ . Jadi, dengan menerapkan Persamaan 11.16 untuk masing-masing sub-interval maka didapatkan

$$\begin{aligned} \int_{a_{k1}}^{b_{k1}} f(x) dx &= S(a_{k1}, b_{k1}) - (h/2)^5 \frac{f^{(4)}(d_2)}{90} \\ &= \frac{h}{6}(f(a_{k1}) + 4f(c_{k1}) + f(b_{k1})) - \frac{h^5}{32} \frac{f^{(4)}(d_2)}{90} \end{aligned} \quad (11.18)$$

dan

$$\begin{aligned} \int_{a_{k2}}^{b_{k2}} f(x) dx &= S(a_{k2}, b_{k2}) - (h/2)^5 \frac{f^{(4)}(d_2)}{90} \\ &= \frac{h}{6}(f(a_{k2}) + 4f(c_{k2}) + f(b_{k2})) - \frac{h^5}{32} \frac{f^{(4)}(d_2)}{90} \end{aligned} \quad (11.19)$$

dengan  $c_{k1}$  adalah nilai tengah  $[a_{k1}, b_{k1}]$  dan  $c_{k2}$  adalah nilai tengah  $[a_{k2}, b_{k2}]$ . Dengan hasil tersebut, integral pada sub-interval  $[a_k, b_k]$  menjadi

$$\begin{aligned} \int_{a_k}^{b_k} f(x) dx &= \int_{a_{k1}}^{b_{k1}} f(x) dx + \int_{a_{k2}}^{b_{k2}} f(x) dx \\ &= S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) - \frac{h^5}{16} \frac{f^{(4)}(d_2)}{90} \end{aligned} \quad (11.20)$$

Asumsikan bahwa  $f^{(4)}(d_1) \approx f^{(4)}(d_2)$ , maka dari Persamaan 11.17 dan 11.20 didapatkan relasi

$$S(a_k, b_k) - h^5 \frac{f^{(4)}(d_2)}{90} \approx S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) - \frac{h^5}{16} \frac{f^{(4)}(d_2)}{90} \quad (11.21)$$

Persamaan 11.21 di atas dapat ditulis dengan

$$-h^5 \frac{f^{(4)}(d_2)}{90} \approx \frac{16}{15} (S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) - S(a_k, b_k)) \quad (11.22)$$

Berdasarkan persamaan 11.20 dan 11.22, didapatkan estimasi nilai *error*

$$\left| \int_{a_k}^{b_k} f(x) dx - S(a_{k1}, b_{k1}) - S(a_{k2}, b_{k2}) \right| \approx \frac{1}{15} |S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) - S(a_k, b_k)| \quad (11.23)$$

Karena asumsi  $f^{(4)}(d_1) \approx f^{(4)}(d_2)$ , koefisien  $\frac{1}{15}$  akan diganti dengan  $\frac{1}{10}$  pada saat implementasi metode.

### **Uji Akurasi**

Asumsikan bahwa nilai toleransi  $\varepsilon_k > 0$  telah ditentukan untuk interval  $[a_k, b_k]$ . Jika

$$\frac{1}{10} |S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) - S(a_k, b_k)| < \varepsilon_k \quad (11.24)$$

maka dapat disimpulkan bahwa

$$\left| \int_{a_k}^{b_k} f(x) dx - S(a_{k1}, b_{k1}) - S(a_{k2}, b_{k2}) \right| < \varepsilon_k \quad (11.25)$$

dengan aturan komposit Simpson digunakan untuk menghampiri integral-tentu, yaitu

$$\int_{a_k}^{b_k} f(x) dx \approx S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2}) \quad (11.26)$$

dan batas *error* dari hampiran pada interval  $[a_k, b_k]$  adalah  $\varepsilon_k$ .

### **Langkah Integral Numerik Menggunakan Adaptif Kuadratur.**

Dimulai dengan  $\{[a_0, b_0], \varepsilon_0\}$  dengan  $\varepsilon_0$  merupakan toleransi dari integral numerik pada  $[a_0, b_0]$ . Bagi interval tersebut menjadi dua sub-interval dan diberikan label masing-masing  $[a_{01}, b_{01}]$  dan  $[a_{02}, b_{02}]$ . Hitung nilai integral numerik, kemudian cek nilai akurasi pada Persamaan 11.24. Jika nilai akurasi terpenuhi, maka nilai integral numerik memakai Persamaan 11.26. Jika nilai akurasi tidak terpenuhi, maka lanjutkan proses integral numerik menggunakan 2 sub-interval. Dengan demikian, diperoleh 2 sub-interval dengan label  $\{[a_1, b_1], \varepsilon_1\}$  dan  $\{[a_2, b_2], \varepsilon_2\}$  dengan  $\varepsilon_1 = \varepsilon_2 = \frac{1}{2}\varepsilon_0$ . Selanjutnya, lakukan tahap kedua kuadratur adaptif.

Pada tahap kedua, mulai dengan  $\{[a_1, b_1], \varepsilon_1\}$  dengan  $\varepsilon_1$  merupakan toleransi dari integral numerik pada  $[a_1, b_1]$ . Bagi interval tersebut menjadi dua sub-interval dan diberikan label masing-masing  $[a_{11}, b_{11}]$  dan  $[a_{12}, b_{12}]$ . Hitung nilai integral numerik, kemudian

cek nilai akurasi pada Persamaan 11.24. Jika nilai akurasi terpenuhi, maka nilai integral numerik memakai Persamaan 11.26. Jika nilai akurasi tidak terpenuhi, maka lanjutkan proses integral numerik menggunakan 2 sub-interval dengan nilai toleransi  $\frac{1}{2}\varepsilon_1$ . Selanjutnya, mulai dengan  $\{[a_2, b_2], \varepsilon_2\}$  dengan  $\varepsilon_2$  merupakan toleransi dari integral numerik pada  $[a_2, b_2]$ . Bagi interval tersebut menjadi dua sub-interval dan diberikan label masing-masing  $[a_{21}, b_{21}]$  dan  $[a_{22}, b_{22}]$ . Hitung nilai integral numerik, kemudian cek nilai akurasi pada Persamaan 11.24. Jika nilai akurasi terpenuhi, maka nilai integral numerik memakai Persamaan 11.26. Jika nilai akurasi tidak terpenuhi, maka lanjutkan proses integral numerik menggunakan 2 sub-interval dengan nilai toleransi  $\frac{1}{2}\varepsilon_2$ . Dengan demikian, terdapat 2 skenario yang bisa terjadi pada tahap 2 ini, yaitu kedua sub-interval tidak memenuhi tes akurasi atau salah satu sub-interval memenuhi tes akurasi. Jika salah satu sub-interval memenuhi tes akurasi, maka akan didapatkan 3 sub-interval baru dan berikan label  $\{[a_1, b_1], \varepsilon_1\}$ ,  $\{[a_2, b_2], \varepsilon_2\}$ , dan  $\{[a_3, b_3], \varepsilon_3\}$  dengan  $\varepsilon_1 + \varepsilon_2 + \varepsilon_3 = \varepsilon_0$ . Jika kedua sub-interval tidak memenuhi tes akurasi, maka akan didapatkan 4 sub-interval baru dan berikan label  $\{[a_1, b_1], \varepsilon_1\}$ ,  $\{[a_2, b_2], \varepsilon_2\}$ ,  $\{[a_3, b_3], \varepsilon_3\}$ , dan  $\{[a_4, b_4], \varepsilon_4\}$  dengan  $\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4 = \varepsilon_0$ . Lanjutkan tahap iterasi kuadratur adaptif hingga semua sub-interval memenuhi tes akurasi.

**Contoh 11.5 :** Gunakan kuadratur adaptif untuk menghitung nilai integral numerik dari integral-tentu

$$\int_0^4 13(x - x^2) e^{-3x/2} dx$$

dengan nilai toleransi awal  $\varepsilon_0 = 0.1$ .

**Solusi :** Mulai tahap pertama dengan  $\{[a_0, b_0], \varepsilon_0\} = \{[0, 4], 0.1\}$  dengan  $\varepsilon_0 = 0.1$  adalah toleransi dari integral numerik pada  $[0, 4]$ . Bagi interval tersebut menjadi  $[a_{01}, b_{01}] = [0, 2]$  dan  $[a_{02}, b_{02}] = [2, 4]$ . Hitung nilai integral numerik masing-masing interval, yaitu

$$\begin{aligned} S(0, 4) &= \frac{2}{3}(f(0) + 4f(2) + f(4)) = -3.7097 \\ S(0, 2) &= \frac{1}{3}(f(0) + 4f(1) + f(2)) = -0.43149 \\ S(2, 4) &= \frac{1}{3}(f(2) + 4f(3) + f(4)) = -1.7157 \end{aligned}$$

Selanjutnya, cek nilai akurasi pada Persamaan 11.24, yaitu

$$e = \frac{1}{10} |S(0, 2) + S(2, 4) - S(0, 4)| = 0.15625$$

Karena nilai akurasi tidak terpenuhi  $e > \varepsilon_0$ , maka lanjutkan proses integral numerik menggunakan 2 sub-interval, yaitu  $\{[0, 2], 0.05\}$  dan  $\{[2, 4], 0.05\}$ .

Selanjutnya, Mulai tahap kedua dengan interval  $\{[a_{11}, b_{11}], \varepsilon_1\} = \{[0, 2], 0.05\}$ . Bagi interval tersebut menjadi  $[a_{11}, b_{11}] = [0, 1]$  dan  $[a_{12}, b_{12}] = [1, 2]$ . Hitung nilai integral numerik masing-masing interval, yaitu

$$\begin{aligned} S(0, 2) &= \frac{1}{3}(f(0) + 4f(1) + f(2)) = -0.43149 \\ S(0, 1) &= \frac{0.5}{3}(f(0) + 4f(0.5) + f(1)) = 1.0235 \\ S(1, 2) &= \frac{0.5}{3}(f(1) + 4f(1.5) + f(2)) = -0.90084 \end{aligned}$$

Selanjutnya, cek nilai akurasi pada Persamaan 11.24, yaitu

$$e = \frac{1}{10} |S(0, 1) + S(1, 2) - S(0, 2)| = 0.055411$$

Karena nilai akurasi tidak terpenuhi  $e > \varepsilon_1 = 0.05$ , maka lanjutkan proses integral numerik menggunakan 2 sub-interval, yaitu  $\{[0, 1], 0.025\}$  dan  $\{[1, 2], 0.025\}$ . Lanjutkan tahap kedua dengan interval  $\{[a_2, b_2], \varepsilon_1\} = \{[2, 4], 0.05\}$ . Bagi interval tersebut menjadi  $[a_{21}, b_{21}] = [2, 3]$  dan  $[a_{22}, b_{22}] = [3, 4]$ . Hitung nilai integral numerik masing-masing interval, kemudian cek nilai akurasi pada Persamaan 11.24, yaitu

$$\begin{aligned} e &= \frac{1}{10} |S(0, 1) + S(1, 2) - S(0, 2)| \\ &= \frac{1}{10} (-1.1245 - 0.60680 - (-1.7157)) = 0.0015570 \end{aligned}$$

Nilai akurasi terpenuhi, yaitu  $e < \varepsilon_2 = 0.05$ . Berdasarkan hasil tahap kedua, tahap tiga dilanjutkan dengan 3 sub-interval, yaitu  $\{[0, 1], 0.025\}$ ,  $\{[1, 2], 0.025\}$ , dan  $\{[2, 4], 0.05\}$ .

Dengan cara yang sama, lanjutkan tahap tiga, sehingga didapatkan nilai cek akurasi masing-masing sub-interval sebagai berikut.

$$\begin{aligned} e(0, 1) &= \frac{1}{10} (0.68636 + 0.39171 - (1.0235)) = 0.0054607 \\ e(1, 2) &= \frac{1}{10} (-0.2933 - 0.60551 - (-0.9008)) = 0.0002023 \\ e(2, 4) &= \frac{1}{10} (-1.1245 - 0.60680 + (-1.7157)) = 0.0015570 \end{aligned}$$

Ketiga sub-interval telah memenuhi akurasi pada masing-masing akurasi yang ditetapkan. Dengan demikian, nilai integral-tentu masing-masing sub-interval adalah

$$\begin{aligned} \int_0^1 f(x) dx &= S(0, 0.5) + S(0.5, 1) = 0.68636 + 0.39171 = 1.07807 \\ \int_1^2 f(x) dx &= S(1, 1.5) + S(1.5, 2) = -0.2933 - 0.60551 = -0.89882 \\ \int_2^4 f(x) dx &= S(2, 3) + S(3, 4) = -1.1245 - 0.60680 = -1.73129 \end{aligned}$$

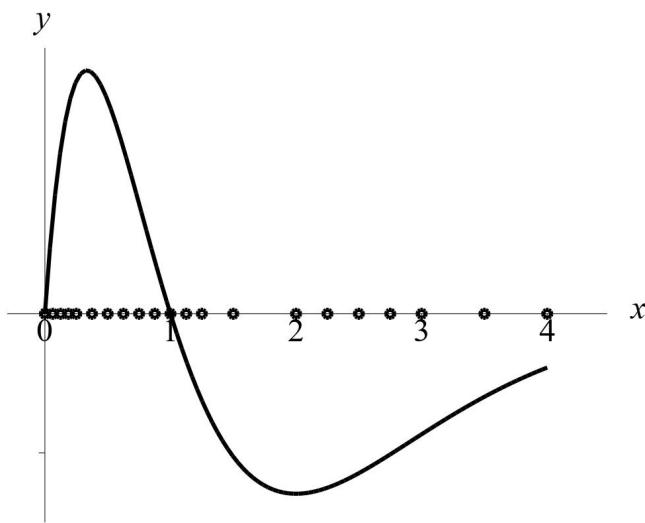
Jadi, hasil akhir metode kuadratur adaptif adalah hasil penjumlahan dari integral tentu masing-masing sub-interval, yaitu  $-1.5520$ .  $\triangle$

Apabila kasus pada Contoh 11.5 memiliki nilai toleransi  $\varepsilon_0 = 0.00001$ , maka diperoleh hasil pembagian interval beserta nilai integral numerik dan galat setiap sub-interval seperti pada Tabel 11.3 berikut.

**Tabel 11.3:** Kuadratur adaptif untuk Contoh 11.5 dengan  $\varepsilon_0 = 0.00001$ .

$a_k$	$b_k$	$S(a_{k1}, b_{k1}) + S(a_{k2}, b_{k2})$	estimasi error	toleransi, $\varepsilon_k$
0.0000	0.0625	0.022872	0.00000002	0.00000016
0.0625	0.1250	0.059487	0.00000001	0.00000016
0.1250	0.1875	0.084342	0.00000001	0.00000016
0.1875	0.2500	0.099699	0.00000001	0.00000016
0.2500	0.3750	0.216721	0.00000025	0.00000031
0.3750	0.5000	0.206464	0.00000018	0.00000031
0.5000	0.6250	0.171506	0.00000013	0.00000031
0.6250	0.7500	0.124334	0.00000010	0.00000031
0.7500	0.8750	0.073245	0.00000007	0.00000031
0.8750	1.0000	0.023529	0.00000005	0.00000031
1.0000	1.1250	-0.021660	0.00000003	0.00000031
1.1250	1.2500	-0.060651	0.00000002	0.00000031
1.2500	1.5000	-0.210808	0.00000032	0.00000063
1.5000	2.0000	-0.605510	0.00000003	0.00000125
2.0000	2.2500	-0.319857	0.00000008	0.00000063
2.2500	2.5000	-0.300617	0.00000008	0.00000063
2.5000	2.7500	-0.270100	0.00000007	0.00000063
2.7500	3.0000	-0.234747	0.00000005	0.00000063
3.0000	3.5000	-0.363898	0.00000104	0.00000125
3.5000	4.0000	-0.243138	0.00000041	0.00000125
Jumlah		-1.548788	0.00000297	0.00001000

Pembagian interval dapat dilihat lebih jelas menggunakan grafik pada Gambar 11.3 berikut.

**Gambar 11.3:** Sub-interval kuadratur adaptif untuk Contoh 11.5 dengan  $\varepsilon_0 = 0.00001$ .

Berdasarkan Gambar 11.3 tersebut, kuadratur adaptif akan menyesuaikan sub-interval berdasarkan grafik  $f(x)$ . Pada grafik yang curam atau memiliki variasi nilai fungsi yang tinggi, sub-interval pada daerah tersebut akan berjumlah banyak dan sebaliknya.

## 11.3 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Aturan Rekursif Trapesium** berisi program untuk mencari integral numerik dari fungsi  $f(x)$  pada interval  $[a, b]$ . Program ini secara *default* berisi 4 masukan, yaitu fungsi  $f(x)$ , titik ujung interval  $[a, b]$  dan banyaknya iterasi  $n$ , serta 1 luaran, yaitu solusi integral numerik pada setiap iterasi  $T$ . Berikut merupakan program untuk aturan rekursif trapesium.

**Algoritma 11.1:** Aturan rekursif trapesium

```
function rekursif(f,a,b,n)
    M = 1;
    h = b-a;
    T = Array{Float64}(undef,n+1,1)
    T[1] = h/2*(f(a)+f(b))
    for j = 1:n
        h = h/2;
        s = 0;
        for k = 1:M
            x = a+h*(2*k-1);
            s = s+f(x);
        end
        M = 2*M;
        T[j+1] = T[j]/2+h*s;
    end
    return T
end
```

**Aturan Romberg** berisi program untuk mencari integral numerik dari fungsi  $f(x)$  pada interval  $[a, b]$ . Program ini secara *default* berisi 3 masukan, yaitu fungsi  $f(x)$ , dan titik ujung interval  $[a, b]$  serta 4 luaran, yaitu nilai solusi integral numerik *sol*, status solusi (*flag*) berisi keputusan apakah solusi hampiran dapat diterima atau tidak, hampiran galat *err* dari solusi integral numerik, dan matriks *R* yaitu matriks iterasi Romberg. Berikut merupakan program untuk aturan Romberg.

**Algoritma 11.2:** Metode Romberg

```
function romberg(f, a, b; maxi = 10, delta = 10^-9)
    flag = 1;
    M = 1;
    h = b-a;
    err = 1;
    R = h/2*(f(a)+f(b));
    for J = 1:maxi
        # Rekursif Trapesium
        h = h/2;
        M = 2*M;
        s = 0;
        for p = 1:M/2
            x = a+h*(2*p-1);
            s = s+f(x);
```

```

end
R = [R zeros(size(R,1),1) ; R[J,1]/2 + h*s zeros(1,size(R,1)) ]
# Aturan Romberg
for K=1:J
    R[J+1,K+1]=R[J+1,K]+(R[J+1,K]-R[J,K])/(4^K-1);
end
err = abs.(R[J,J]-R[J+1,J+1]);
if err<delta; flag=0; break; end
end
sol = R[end,end];
return sol, flag, err, R
end

```

**Aturan Kuadratur Adaptif** berisi program untuk mencari integral numerik dari fungsi  $f(x)$  pada interval  $[a,b]$ . Program ini secara *default* berisi 4 masukan, yaitu fungsi  $f(x)$ , titik ujuang interval  $[a,b]$ , dan toleransi galat yang dapat diterima  $\delta$ , serta 3 luaran, yaitu nilai solusi integral numerik  $sol$ , hampiran galat  $err$  dari solusi integral numerik, dan matriks  $SRmat$  yaitu matriks iterasi kuadratur adaptif yang berisi sub-interval  $[a_k, b_k]$ , solusi integral, galat, dan toleransi galat pada sub-interval tersebut. Berikut merupakan program untuk aturan kuadratur adaptif.

### Algoritma 11.3: Aturan kuadratur adaptif

```

function adaptif(f, a, b; delta=10^-9)
    iterating = 0;
    done = 1;
    SRvec = srule(f,a,b,delta);
    SRmat = SRvec;
    m = 1;
    state = iterating;
    while(state==iterating)
        n = m;
        for j = n:-1:1
            p = j;
            SR0vec = SRmat[p,:];
            err = SR0vec[5];
            delta = SR0vec[6];
            if (delta<=err)
                state=done;
                SR1vec=SR0vec;
                SR2vec=SR0vec;
                a = SR0vec[1];
                b = SR0vec[2];
                c =(a+b)/2;
                err=SR0vec[5];
                delta=SR0vec[6];
                delta2=delta/2;
                SR1vec=srule(f,a,c,delta2);
                SR2vec=srule(f,c,b,delta2);
                err = abs(SR0vec[3]-SR1vec[3]-SR2vec[3])/10;
                if err<delta
                    SRmat[p,:]=SR0vec;
                    SRmat[p,4]=SR1vec[3]+SR2vec[3];
                    SRmat[p,5]=err;
                else
                    SRmat = [SRmat; zeros(1,6)]
                
```

```

SRmat [p+1:m+1, :] = SRmat [p:m, :];
m=m+1;
SRmat [p, :] = SR1vec;
SRmat [p+1, :] = SR2vec;
state=iterating;
end
end
end
end
sol = sum(SRmat [:, 4]);
err = sum(abs.(SRmat [:, 5]));
SRmat = SRmat [1:m, [1, 2, 4, 5, 6]];
return sol, err, SRmat
end

function srule(f, a0, b0, delta0)
h = (b0-a0)/2;
C = zeros(1, 3)
C = f.([a0 (a0+b0)/2 b0]);
S = h*(C[1]+4*C[2]+C[3])/3;
S2= S;
delta1=delta0;
err=delta0;
Z = [a0 b0 S S2 err delta1];
return Z
end

```

### 11.3.1 Aturan Rekursif

Pada praktikum ini, akan dipelajari hubungan dari aturan trapesium dan Simpson untuk menghasilkan solusi integral numerik dengan akurasi yang baik. Praktikum ini dimulai dengan mempelajari aturan rekursif trapesium sebagai berikut.

**Contoh 11.6 :** Berikut merupakan langkah-langkah untuk menghitung nilai hampiran  $T(J)$  dengan  $J = 0, 1, 2, \dots, 5$  dari

$$\int_1^5 \frac{dx}{x}$$

serta nilai galat dari masing-masing hampiran, jika diketahui nilai solusi eksak integral adalah 1.609437912 menggunakan aturan rekursif trapesium.

**Langkah 1:** Penghitungan nilai hampiran  $T(0)$  hingga  $T(5)$ .

Inp:  $f(x) = 1/x;$   
 $a = 1;$   
 $b = 5;$   
 $T = \text{rekursif}(f, a, b, 5)$

Out: 6x1 Array{Float64,2}:
2.4
1.8666666666666667
1.6833333333333333
1.628968253968254
1.6144063238103485
1.6106858960792332

**Langkah 2:** Penghitungan nilai galat masing-masing hampiran.

```
Inp: eksak = 1.609437912;
      galat = T.-eksak
```

```
Out: 6x1 Array{Float64,2}:
      0.7905620879999999
      0.2572287546666667
      0.07389542133333338
      0.019530341968253984
      0.004968411810348572
      0.0012479840792332109
```

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 11.6 untuk menghitung nilai  $T(J)$  untuk  $J = 0, 1, 2, 3, 4, 5$  dari

$$\int_0^{\pi/2} (x^2 + x + 1) \cos(x) dx$$

Hitung pula nilai galat  $T(J)$ , jika diketahui nilai solusi eksak integral adalah 2.038197427.

### 11.3.2 Aturan Romberg

Aturan Romberg adalah cara untuk menghubungkan aturan rekursif trapesium, Simpson, Boole, dan seterusnya dalam suatu matriks. Hubungan masing-masing aturan dapat dituliskan dalam suatu persamaan, yaitu

$$R(J, K) = R(J, K - 1) + \frac{R(J, K - 1) - R(J - 1, K - 1)}{4^K - 1}$$

dengan  $R(J, 0)$  merupakan aturan rekursif trapesium.

**Contoh 11.7 :** Diberikan fungsi

$$f(x) = (x^2 + x + 1) \cos(x)$$

Berikut merupakan cara untuk menghitung nilai hampiran dari  $\int_0^{\pi/2} f(x) dx$  beserta lama waktu komputasi menggunakan metode Romberg (Gunakan: BenchmarkTools untuk menghitung waktu komputasi dan nilai  $\delta = 10^{-9}$  sebagai toleransi).

```
Inp: using BenchmarkTools
Inp: f(x) = (x^2+x+1)*cos(x)
      a = 0;
      b = pi/2;
      @btime sol, flag, err, R = romberg(f, a, b)
      sol, flag, err, R = romberg(f, a, b)
      @show (sol,flag,err)
      R
```

```
Out: 150.099 μs (605 allocations: 21.44 KiB)
      (sol, flag, err) = (2.0381974270672245, 0, 1.213065203842234e-10)
      6x6 Array{Float64,2}:
      0.785398  0.0       0.0       0.0       0.0
      1.72681   2.04062  0.0       0.0       0.0
```

1.96053	2.03844	2.0383	0.0	0.0	0.0
2.01879	2.03821	2.0382	2.0382	0.0	0.0
2.03335	2.0382	2.0382	2.0382	2.0382	0.0
2.03698	2.0382	2.0382	2.0382	2.0382	2.0382

Berdasarkan hasil di atas, *flag* bernilai 0, artinya solusi telah memenuhi nilai toleransi galat yang diberikan. Proses komputasi integral numerik dari  $\int_0^{\pi/2} f(x) dx$  menghasilkan solusi 2.038197427 dengan galat  $1.213 \times 10^{-10}$  dalam waktu 150 mikrodetik. Waktu komputasi mungkin berbeda pada satu komputer dengan komputer lain, akan tetapi perbedaan waktu komputasi tidak akan berbeda signifikan.

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 11.7 untuk menghitung nilai dari

$$\int_0^2 \sqrt{4x - x^2} dx$$

menggunakan integral Romberg hingga mendapatkan galat  $\epsilon = 10^{-10}$ . Hitunglah waktu komputasi yang diperlukan menggunakan macro `@btime` pada BenchmarkTools.

*Catatan.* Hal ini akan memungkinkan untuk mengubah nilai `maxi` dan `delta` pada Program 11.2. Perhatikan pula nilai galat dan *flag* yang dihasilkan. Pastikan bahwa *flag* bernilai 0. Jika *flag* bernilai 1, kemungkinan besar dikarenakan nilai maksimum iterasi terlalu kecil, sehingga sesuaikan nilai `maxi` agar *flag* bernilai 0.

### 11.3.3 Aturan Kuadratur Adaptif

Aturan romberg memiliki waktu komputasi yang relatif lama untuk menghasilkan akurasi yang baik pada banyak kasus. Salah satu aturan yang dapat digunakan untuk mengatasinya adalah aturan kuadratur adaptif. Metode ini akan menyesuaikan ukuran langkah  $h$  sesuai bentuk fungsi, sehingga menghemat waktu komputasi.

**Contoh 11.8 :** Diberikan integral tentu

$$\int_0^4 13(x-x^2)e^{-3x/2} dx$$

Berikut merupakan langkah-langkah untuk menghitung integral di atas menggunakan aturan kuadratur adaptif dengan toleransi awal  $\epsilon = 0.00001$  dan menggambarkan pembagian selang  $[a_k, b_k]$ .

**Langkah 1:** Hitung nilai hampiran integral dengan metode kuadratur adaptif.

```
Inp: f(x) = 13*(x-x.^2).*exp(-3*x/2);
a = 0;
b = 4;
@btime sol,err,SRmat = adaptif(f,a,b,delta=0.00001)
sol,err,SRmat = adaptif(f,a,b,delta=0.00001)
@show sol
@show err
SRmat
```

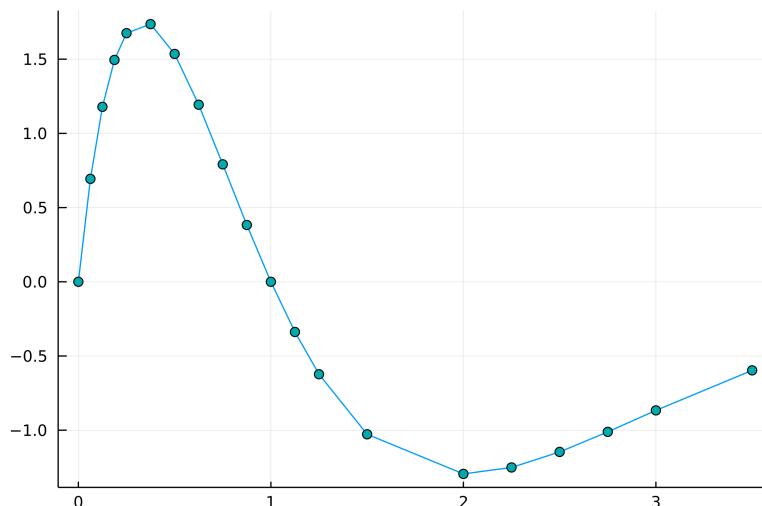
```
Out: 304.000 μs (2588 allocations: 105.48 KiB)
sol = -1.5487882341253174
err = 2.9680861581356417e-6
```

**Tabel 11.4:** Komputasi kuadratur adaptif untuk  $f(x) = 13(x - x^2)e^{-3x/2}$ .

$a_k$	$b_k$	Aturan Simpson selang $[a_k, b_k]$	Batas galat	Toleransi $\varepsilon_k$ pada selang $[a_k, b_k]$
0.0000	0.0625	0.022871848	0.000000015	0.000000156
0.0625	0.1250	0.059486865	0.000000013	0.000000156
0.1250	0.1875	0.084342136	0.000000011	0.000000156
0.1875	0.2500	0.099698715	0.000000010	0.000000156
0.2500	0.3750	0.216721368	0.000000251	0.000000313
0.3750	0.5000	0.206463916	0.000000184	0.000000313
0.5000	0.6250	0.171506172	0.000000134	0.000000313
0.6250	0.7500	0.124333638	0.000000096	0.000000313
0.7500	0.8750	0.073245151	0.000000068	0.000000313
0.8750	1.0000	0.023528832	0.000000047	0.000000313
1.0000	1.1250	-0.021660390	0.000000032	0.000000313
1.1250	1.2500	-0.060650794	0.000000021	0.000000313
1.2500	1.5000	-0.210808238	0.000000317	0.000000625
1.5000	2.0000	-0.605509650	0.000000032	0.000001250
2.0000	2.2500	-0.319857202	0.000000081	0.000000625
2.2500	2.5000	-0.300617492	0.000000083	0.000000625
2.5000	2.7500	-0.270099624	0.000000071	0.000000625
2.7500	3.0000	-0.234747212	0.000000054	0.000000625
3.0000	3.5000	-0.363897997	0.000001037	0.000001250
3.5000	4.0000	-0.243138278	0.000000411	0.000001250
Total		-1.548788234	0.000002968	0.000010000

**Langkah 2:** Gambarkan pembagian selang  $[a_k, b_k]$  metode kuadratur adaptif untuk fungsi  $f(x) = 13(x - x^2)e^{-3x/2}$ .

```
Inp: using Plots
x = [SRmat[:,1],b]
plot(x,f,legend=:false)
scatter!(x,f)
```

**Gambar 11.4:** Sub-interval dari integral kuadratur adaptif.

---

**Soal Latihan:** Diberikan fungsi

$$f(x) = \sqrt{4x - x^2}$$

Gunakan metode kuadratur adaptif dengan toleransi awal  $\epsilon = 10^{-10}$  untuk menghitung solusi integral  $\int_0^2 f(x) dx$ , kemudian gambarkan pembagian selang  $[a_k, b_k]$  metode tersebut seperti pada Contoh 11.8. Hitunglah waktu komputasi yang diperlukan dengan paket BenchmarkTools. Bandingkan lama waktu komputasi yang diperlukan dengan soal latihan pada aturan Romberg. Jelaskan!

---

**Soal Latihan:** Diberikan fungsi

$$f(x) = \sqrt{4x - x^2}$$

Bandingkan waktu komputasi yang diperlukan oleh integral Romberg dan kuadratur adaptif untuk menyelesaikan  $\int_0^2 f(x) dx$ . Gunakan BenchmarkTools untuk menghitung waktu komputasi dan pastikan bahwa *flag* bernilai 0 dengan toleransi yang digunakan adalah

- a.  $\epsilon = 10^{-13}$
- b.  $\epsilon = 10^{-16}$

Berdasarkan hasil yang diperoleh,

1. Metode manakah yang lebih cepat mendapatkan hasil integralnya?
  2. Jika solusi dari integral tersebut adalah  $\pi$ , berapakah nilai *absolute error* dari kedua metode? Apakah sudah mencapai toleransi yang ditentukan?
  3. Berapakah percepatan yang diperoleh dari metode yang lebih cepat untuk kedua nilai toleransi? Jelaskan!
- 

## 11.4 Latihan-latihan

### 11.4.1 Ulasan Materi

1. Jelaskan apa yang dimaksud dengan aturan rekursif pada formula kuadratur trapezium! berikan contoh dan cara menghitungnya!
2. Jelaskan apa yang dimaksud dengan aturan rekursif pada formula kuadratur Simpson! berikan contoh dan cara menghitungnya!
3. Jelaskan apa yang dimaksud dengan aturan rekursif pada formula kuadratur Boole! berikan contoh dan cara menghitungnya!
4. Apa yang dimaksud integral Romberg? Jelaskan hubungan integral Romberg dengan aturan rekursif!
5. Jelaskan kegunaan dari ekstrapolasi Richardson dalam integral Romberg!
6. Jelaskan bagaimana cara mengetahui kepresisan (*truncation error*) dari integral Romberg!
7. Apa yang dimaksud dengan aturan kuadratur adaptif? Jelaskan bagaimana cara kerjanya!
8. Jelaskan langkah-langkah yang harus dilalui jika suatu integral numerik dikerjakan menggunakan aturan kuadratur adaptif!

### 11.4.2 Soal Pemrograman

1. Untuk setiap integral berikut, bentuklah matriks integral Romberg hingga  $K = 3$ .
  - (a)  $\int_0^3 \frac{\sin(2x)}{1+x^2} dx$ .
  - (b)  $\int_0^3 \sin(4x)e^{-2x} dx$ .
  - (c)  $\int_{0.04}^1 \frac{1}{\sqrt{x}} dx$ .
  - (d)  $\int_0^2 \frac{1}{x^2 + \frac{1}{10}} dx$ .
  - (e)  $\int_{1/(2\pi)}^2 \sin\left(\frac{1}{x}\right) dx$ .
  - (f)  $\int_0^2 \sqrt{4-x^2} dx$ .
2. Tentukan berapakah nilai  $K$  terkecil yang memenuhi
  - (a)  $\int_0^2 8x^7 dx = 256 \equiv R(K, K)$
  - (b)  $\int_0^2 11x^{10} dx = 2048 \equiv R(K, K)$
3. Gunakan integral Romberg untuk menghitung  $R(4, 4)$  dari integral berikut:
  - (a)  $\int_0^{\pi/3} x \sin(x) dx$
  - (b)  $\int_{\pi/7}^{\pi/5} \sin(x) dx$
4. Gunakan integral Romberg dengan  $n = 4$  untuk menghampiri integral

$$\int_1^3 \frac{1}{x} dx$$

Bandingkan dengan nilai sebenarnya yaitu  $\int_1^3 \frac{1}{x} dx = \ln(3)$ .

5. Diberikan data seperti berikut.

$x$	1	2	3	4	5
$f(x)$	-1	-0.307	0.099	0.386	0.609

Gunakan metode integral romberg untuk menghitung  $\int_1^5 f(x) dx$

6. Diberikan data seperti berikut.

$x$	0	0.5	1	1.5	2	2.5	3	3.5	4
$f(x)$	-4271	-2522	-499	1795	4358	7187	10279	13633	17246

Gunakan metode integral romberg untuk menghitung  $\int_0^4 f(x) dx$ .

7. Diberikan data seperti berikut.

$x$	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00
$f(x)$	4.00	3.91	3.63	3.20	2.62	1.95	1.21	0.47	-0.25

Gunakan metode integral romberg untuk menghitung  $\int_0^2 f(x) dx$ .

8. Selesaikan soal berikut menggunakan kuadratur adaptif hingga memperoleh galat  $\delta < 10^{-7}$

$$\int_0^2 x^4 \log(x + \sqrt{x^2 + 1}) dx$$

9. Gunakan integral romberg dan kuadratur adaptif untuk menghitung dua integral-tentu berikut dengan akurasi 5, 8, dan 10 angka desimal. Nilai eksak dari kedua soal adalah  $\pi$ . Bandingkan dan jelaskan kecepatan adaptif kuadratur dan integral Romberg dalam menyelesaikan kedua soal berikut.

(a)  $\int_0^2 \sqrt{4x - x^2} dx$

(b)  $\int_0^1 \frac{4}{1+x^2} dx$

10. Diketahui suatu fungsi dalam teori medan listrik yaitu

$$H(x, r) = \frac{60r}{r^2 - x^2} \int_0^{2\pi} \left[ 1 - \left( \frac{x}{r} \right)^2 \sin^2 \theta \right]^{0.5} d\theta$$

Hitung solusi persamaan tersebut menggunakan kuadratur adaptif hingga memiliki galat  $\delta = 10^{-5}$  dan bandingkan kecepatan komputasi dengan integral romberg jika diketahui.

- (a)  $r = 110, x = 75$   
 (b)  $r = 100, x = 70$   
 (c)  $r = 90, x = 65$

11. Gunakan kuadratur adaptif untuk menyelesaikan integral-tentu berikut, kemudian gambarkan pembagian selang ukuran langkah  $h$  dari metode tersebut.

(a)  $\int_0^{2\pi} e^{-x} \sin(x) dx$  dengan akurasi  $10^{-6}$

(b)  $\int_0^{\pi} e^x \sin(x^2) dx$  dengan akurasi  $10^{-6}$

(c)  $\int_0^4 13(x - x^2)e^{-3x/2} dx$  dengan akurasi  $10^{-6}$

12. Gunakan metode kuadratur adaptif dengan toleransi awal  $\epsilon = 0.00001$  untuk menghitung solusi integral berikut, kemudian gambarkan pembagian selang  $[a_k, b_k]$  metode tersebut.

(a)  $\int_0^2 \frac{1}{x^2 + \frac{1}{10}} dx$

(b)  $\int_0^3 \sin(4x)e^{-2x} dx$

(c)  $\int_{1/(2\pi)}^2 \sin\left(\frac{1}{x}\right) dx$

(d)  $\int_{0.04}^1 \frac{1}{\sqrt{x}} dx$

(e)  $\int_0^2 \sqrt{4x - x^2} dx$

13. Fungsi galat  $\text{erf}(x)$  didefinisikan sebagai

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

Hitunglah nilai hampiran untuk  $\text{erf}(0.5)$ .

14. Sebuah batang dengan panjang  $l$  yang terletak di sepanjang sumbu  $x$  memiliki muatan seragam per satuan panjang dan muatan total  $Q$ . Potensial listrik pada titik  $P$  sepanjang sumbu  $y$  pada jarak  $d$  dari titik asal diberikan oleh

$$V = k \frac{Q}{l} \int_0^l \frac{1}{\sqrt{x^2 + d^2}} dx$$

dimana  $k = 9 \times 10^9$ . Hitung hampiran  $V$  jika  $Q = 10^{-6}$ ,  $l = 5$ , dan  $d = 5$

15. Diberikan data pencatatan suhu per jam pada suatu hari di sebuah daerah seperti berikut

Pukul( $x$ )	8:00	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00
Suhu( $T^\circ C$ )	24.1	25.0	28.4	29.3	30.3	30.2	27.3	25.9	25.1

Hitunglah rata-rata suhu dari pukul 8:00 hingga 16:00 yaitu

$$\frac{1}{b-a} \int_a^b T(x) dx$$

---

---

## BAB 12

---

# PERSAMAAN DIFFERENSIAL BIASA 1

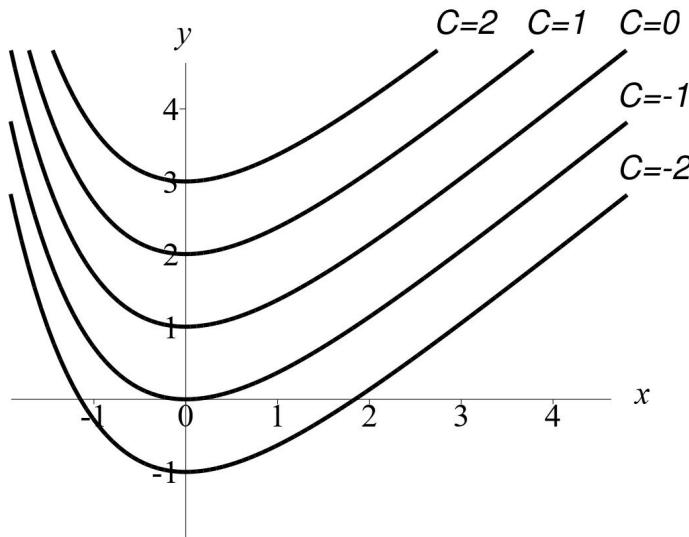
Persamaan differensial merupakan bentuk persamaan matematika yang sering muncul di berbagai bidang ilmu pengetahuan. Contoh dari persamaan differensial adalah

$$\frac{dy}{dt} = 1 - e^{-t} \quad (12.1)$$

Persamaan tersebut merupakan persamaan differensial karena melibatkan turunan  $dy/dt$  dari fungsi  $y = y(t)$  yang tidak diketahui. Fungsi  $y(t)$  dapat dicari menggunakan aturan integral, yaitu

$$y(t) = t + e^{-t} + C \quad (12.2)$$

dengan  $C$  konstan. Dengan demikian, terdapat tak-hingga banyak solusi yang dapat memenuhi persamaan  $y'(t) = 1 - e^{-t}$  dan membentuk suatu keluarga kurva seperti pada Gambar 12.1 berikut.



**Gambar 12.1:** Kurva solusi  $y(t) = t + e^{-t} + C$ .

Meskipun beberapa persamaan differensial mampu untuk diselesaikan secara analitik, masih terdapat banyak sekali persamaan differensial yang sulit (bahkan tidak bisa) untuk

diselesaikan secara analitik dan membutuhkan penyelesaian secara numerik. Pada bab ini, akan dipelajari beberapa metode yang digunakan untuk menyelesaikan persamaan differensial biasa secara numerik.

## 12.1 Pendahuluan Persamaan Diferensial

Secara analitik, persamaan differensial memiliki dua jenis penyelesaian, yaitu solusi umum dan solusi khusus. Pada persamaan differensial 12.1, solusi yang dihasilkan merupakan solusi umum, yaitu persamaan 12.2. Solusi umum akan menampilkan banyak solusi yang memungkinkan untuk memenuhi persamaan differensial yang diberikan. Sementara itu, solusi khusus akan menampilkan tepat satu solusi dari persamaan differensial jika diberikan nilai awal. Berikut ini merupakan contoh dari persamaan differensial dengan nilai awal.

Jika  $A$  merupakan suhu rata-rata ruangan dan  $y(t)$  merupakan suhu suatu benda pada waktu  $t$ , maka hukum Newton tentang pendinginan mengatakan bahwa

$$\frac{dy}{dt} = -k(y - A) \quad (12.3)$$

dengan  $k$  suatu konstanta positif. Apabila diketahui suhu benda tersebut pada waktu  $t = 0$  sebesar  $y_0$  dan informasi tersebut dimasukkan ke Persamaan 12.3, maka bentuk persamaan differensial menjadi

$$\frac{dy}{dt} = -k(y - A) \quad \text{dengan} \quad y(0) = y_0 \quad (12.4)$$

Dengan teknik pemisahan *variable*, solusi dari umum dari persamaan tersebut adalah

$$\begin{aligned} \int \frac{dy}{y-A} &= \int -k dt \\ \ln(y-A) &= -kt + C \\ y &= A + e^{-kt+C} \end{aligned}$$

Solusi tersebut merupakan solusi umum dari persamaan differensial 12.4 karena masih bergantung pada koefisien  $C$ . Solusi khusus dari persamaan differensial dapat diperoleh dengan cara mengevaluasi nilai kondisi awal yang diberikan yaitu  $y(0) = y_0$ . Dengan demikian, diperoleh nilai koefisien  $C$ , yaitu

$$\begin{aligned} y(0) = y_0 &\Leftrightarrow y_0 = A + e^{-k(0)+C} \\ &\Leftrightarrow C = \ln(y_0 - A) \end{aligned}$$

Jadi, diperoleh solusi khusus persamaan differensial 12.4 yaitu

$$y(t) = A + (y_0 - A)e^{-kt}$$

Suatu persamaan differensial yang telah diketahui kondisi awal seperti pada Persamaan 12.4 disebut dengan **masalah nilai awal**.

### Masalah Nilai Awal

**Definisi 12.1 :** Suatu solusi masalah nilai awal (MNA)

$$y' = f(t, y) \quad \text{dengan} \quad y(t_0) = y_0 \quad (12.5)$$

pada interval  $[t_0, b]$  merupakan suatu fungsi terturunkan  $y = y(t)$  sedemikian sehingga

$$y(t_0) = y_0 \quad \text{dan} \quad y'(t) = f(t, y(t)) \quad \text{untuk setiap } t \in [t_0, b] \quad (12.6)$$

Perhatikan bahwa kurva solusi  $y = y(t)$  harus melewati titik awal  $(t_0, y_0)$ . ■

## 12.2 Metode Euler

Telah diketahui bahwa tidak semua masalah nilai awal dapat diselesaikan secara eksplisit menggunakan metode analitik dan sebagian besar masalah nilai awal tidak mungkin untuk dicari solusi  $y(t)$ . Sebagai contoh, masalah nilai awal  $y' = t^3 + y^2$  dengan  $y(0) = 0$  tidak memiliki solusi berupa persamaan. Oleh karena itu, diperlukan suatu metode numerik yang dapat menghampiri solusi dari masalah nilai awal yang sulit (bahkan tidak dapat) dicari persamaan solusinya eksplisitnya.

Pendekatan numerik pertama yang sering dipakai untuk menjelaskan konsep dasar teknik hampiran masalah nilai awal adalah metode Euler. Metode ini sangat jarang digunakan karena memiliki nilai galat akumulasi yang besar. Namun, metode Euler penting untuk dipelajari karena analisis galat metode ini cukup mudah untuk dipahami.

Misalkan, diberikan masalah nilai awal  $y' = f(t, y)$  dengan  $y(a) = y_0$  yang akan dicari nilai solusinya pada  $[a, b]$ . Serta, diketahui bahwa tidak ada fungsi terturunkan yang memenuhi masalah nilai awal tersebut. Sebagai gantinya, suatu barisan titik  $\{(t_k, y_k)\}$  digunakan sebagai hampiran, yaitu  $y(t_k) \approx y_k$ . Berikut merupakan cara untuk membentuk barisan titik tersebut yang memenuhi masalah nilai awal secara numerik.

Langkah pertama adalah menentukan absis dari solusi hampiran. Untuk memudahkan penghitungan, interval  $[a, b]$  akan dibagi menjadi  $M$  sub-interval dengan lebar  $h$  yang sama, yaitu

$$t_k = a + kh \quad \text{untuk } k = 0, 1, \dots, M \quad \text{dengan} \quad h = \frac{b - a}{M}$$

Selanjutnya, akan dicari nilai hampiran yang memenuhi masalah nilai awal

$$y' = f(t, y) \quad \text{pada } [t_0, t_M] \quad \text{dengan} \quad y(t_0) = y_0$$

Asumsikan bahwa fungsi  $y(t)$ ,  $y'(t)$ , dan  $y''(t)$  kontinu dan gunakan teorema Taylor untuk ekspansi  $y'(t)$  disekitar  $t = t_0$ . Dengan demikian untuk setiap nilai  $t$ , terdapat nilai  $c_1 \in [t_0, t]$  sedemikian sehingga

$$y(t) = y(t_0) + y'(t_0)(t - t_0) + \frac{y''(c_1)(t - t_0)^2}{2} \quad (12.7)$$

Ketika  $y'(t_0) = f(t_0, y(t_0))$  dan  $h = t_1 - t_0$  disubstitusikan ke Persamaan 12.7, akan dihasilkan persamaan untuk  $y(t_1)$ , yaitu

$$y(t_1) = y(t_0) + hf(t_0, y(t_0)) + y''(c_1) \frac{h^2}{2} \quad (12.8)$$

Jika ukuran langkah  $h$  yang dipilih cukup kecil, maka ordo kedua ekspansi Taylor dapat diabaikan, sehingga didapatkan

$$y_1 = y_0 + hf(t_0, y_0) \quad (12.9)$$

Persamaan 12.9 disebut dengan **hampiran Taylor**. Proses diulangi hingga mendapatkan barisan titik-titik yang menghampiri solusi  $y = y(t)$ . Jadi, bentuk umum **metode Euler** adalah

$$t_{k+1} = t_k + h, \quad y_{k+1} = y_k + hf(t_k, y_k) \quad \text{untuk } k = 0, 1, \dots, M-1 \quad (12.10)$$

**Contoh 12.1 :** Gunakan metode Euler untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1$ .

**Solusi :** Diketahui bahwa nilai awal  $t_0 = 0$  dan  $y_0 = 1$ . Berdasarkan Persamaan 12.10, didapatkan nilai  $t_1 = 1$  dan

$$\begin{aligned} y_1 &= y_0 + hf(t_0, y_0) \\ &= 1 + 1 \frac{0-1}{2} = 0.5 \end{aligned}$$

Dengan mengulangi Persamaan 12.10, akan didapatkan nilai  $y_2$  dan  $y_3$  serta nilai galat masing-masing hampiran terhadap nilai eksak solusi  $y(t) = 3e^{-t/2} - 2 + t$  yang dapat dilihat pada Tabel 12.1 berikut.

**Tabel 12.1:** Metode Euler untuk  $y' = (t-y)/2$  dengan  $y(0) = 1$  dan  $h = 1$ .

$k$	$t_k$	$y_k$	Nilai eksak, $y(t_k)$	galat, $e$
0	0	1.00000	1.00000	0.00000
1	1	0.50000	0.81959	0.31959
2	2	0.75000	1.10364	0.35364
3	3	1.37500	1.66939	0.29439

△

### Kepresian Metode Euler

Metode yang digunakan untuk menghampiri solusi dari suatu masalah nilai awal disebut sebagai metode beda (*difference methods*) atau metode peubah diskret (*discrete variable methods*). Suatu metode langkah-tunggal akan memiliki bentuk  $y_{k+1} = y_k + h \Phi(t_k, y_k)$  dengan fungsi  $\Phi$  yang disebut sebagai fungsi inkremen. Ketika menggunakan suatu metode peubah diskret untuk menghampiri solusi masalah nilai awal, akan mengakibatkan dua sumber galat, yaitu diskretisasi dan *round-off*.

**Definisi 12.2 :** Asumsikan bahwa  $\{(t_k, y_k)\}_{k=0}^M$  merupakan barisan dari hampiran diskret dan  $y(t)$  adalah solusi tunggal dari masalah nilai awal.

**Galat diskretisasi global**  $e_k$  adalah selisih antara solusi tunggal dan solusi yang didapatkan dari metode peubah diskret. Galat diskretisasi global didefinisikan oleh

$$e_k = y(t_k) - y_k \quad \text{untuk } k = 0, 1, 2, \dots, M \quad (12.11)$$

**Galat diskretisasi lokal**  $\varepsilon_{k+1}$  adalah galat yang terjadi pada langkah-tunggal dari  $t_k$  ke  $t_{k+1}$ . Galat diskretisasi lokal didefinisikan oleh

$$\varepsilon_{k+1} = y(t_{k+1}) - y_k - h \Phi(t_k, y_k) \quad \text{untuk } k = 0, 1, 2, \dots, M-1 \quad (12.12)$$

■

**Teorema 12.1 Kepresisan Metode Euler:** Asusmsikan bahwa  $y(t)$  merupakan solusi dari suatu masalah nilai awal. Jika  $y(t) \in C^2[t_0, b]$  dan  $\{(t_k, y_k)\}_{k=0}^M$  adalah barisan hampiran yang dihasilkan oleh metode Euler, maka

$$\begin{aligned} |e_k| &= |y(t_k) - y_k| = O(h) \\ |\varepsilon_{k+1}| &= |y(t_{k+1}) - y_k - h f(t_k, y_k)| = O(h^2) \end{aligned} \quad (12.13)$$

Serta, galat pada akhir interval yang disebut dengan galat global akhir (*final global error*) adalah

$$E(y(b), h) = |y(b) - y_M| = O(h) \quad (12.14)$$

■

**Contoh 12.2 :** Bandingkan galat global akhir yang dihasilkan oleh metode Euler untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1, \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{64}$ .

**Solusi :** Untuk  $h = 1$ , berdasarkan hasil yang diperoleh pada Contoh 12.1, didapatkan nilai hampiran untuk  $y(3)$  yaitu  $y_3 = 1.375$  dan galat global akhir yaitu 0.29439. Dengan mengulangi langkah-langkah pada Contoh 12.1 untuk ukuran langkah  $h = \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{64}$ , akan didapatkan nilai hampiran untuk  $y(3)$  beserta galat global akhir masing-masing hampiran seperti pada Tabel 12.2 berikut.

**Tabel 12.2:** Perbandingan nilai galat global akhir metode Euler.

Ukuran langkah, $h$	Banyak sub-interval, $M$	Hampiran $y(3), y_M$	Galat global akhir, $E(y(3), h)$
1	3	1.37500	0.29439
$\frac{1}{2}$	6	1.53394	0.13545
$\frac{1}{4}$	12	1.60425	0.06514
$\frac{1}{8}$	24	1.63743	0.03196
$\frac{1}{16}$	48	1.65356	0.01583
$\frac{1}{32}$	96	1.66151	0.00788
$\frac{1}{64}$	192	1.66546	0.00393

Berdasarkan hasil yang diperoleh pada Tabel 12.2, setiap perubahan ukuran langkah sebesar  $\frac{1}{2}h$  akan mengakibatkan perubahan galat global akhir sekitar  $\frac{1}{2}E$ . Jadi, hubungan antara ukuran langkah dan galat global akhir adalah  $O(h)$ .  $\triangle$

## 12.3 Metode Heun

Pendekatan selanjutnya adalah metode Heun. Berikut merupakan ide dasar dari metode Heun untuk memperoleh suatu algoritma untuk menyelesaikan masalah nilai awal

$$y' = f(t, y) \quad \text{pada } [t_0, t_M] \quad \text{dengan} \quad y(t_0) = y_0$$

Untuk mendapatkan solusi  $(t_1, y_1)$ , ide dasar metode Heun yaitu menggunakan teorema kalkulus dan menghitung integral dari  $y'(t)$  pada  $[t_0, t_1]$ , yaitu

$$\int_{t_0}^{t_1} y'(t) dt = y(t_1) - y(t_0)$$

dengan  $y(t)$  merupakan anti-turunan yang diinginkan dari  $y'(t)$ . Karena  $y'(t) = f(t, y(t))$ , maka didapatkan solusi untuk  $y(t_1)$  yaitu

$$y(t_1) = y(t_0) + \int_{t_0}^{t_1} f(t, y(t)) dt$$

Selanjutnya, solusi integral-tentu dapat dihampiri menggunakan aturan trapesium dengan ukuran langkah  $h = t_1 - t_0$ , sehingga akan didapatkan

$$y(t_1) \approx y(t_0) + \frac{h}{2}(f(t_0, y(t_0)) + f(t_1, y(t_1)))$$

Perhatikan bahwa ruas kanan terdapat nilai yang belum diketahui, yaitu  $y(t_1)$ . Nilai  $y(t_1)$  tersebut dapat diperkirakan menggunakan metode Euler. Dengan demikian, hampiran metode Heun adalah

$$y_1 = y_0 + \frac{h}{2}(f(t_0, y_0) + f(t_1, y_0 + h f(t_0, y_0))) \quad (12.15)$$

Proses pada Persamaan 12.15 diulangi hingga diperoleh barisan titik yang menghampiri kurva solusi  $y = y(t)$ . Untuk setiap langkah, metode Euler digunakan sebagai prediktor dan aturan trapesium digunakan sebagai korektor untuk memeroleh hasil akhir. Jadi, bentuk umum **metode Heun** adalah

$$\begin{aligned} p_{k+1} &= y_k + h f(t_k, y_k), \quad t_{k+1} = t_k + h \\ y_{k+1} &= y_k + \frac{h}{2}(f(t_k, y_k) + f(t_{k+1}, p_{k+1})) \end{aligned} \quad (12.16)$$

**Contoh 12.3 :** Gunakan metode Heun untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1$ .

**Solusi :** Diketahui bahwa nilai awal  $t_0 = 0$  dan  $y_0 = 1$ . Berdasarkan Persamaan 12.16, didapatkan nilai  $t_1 = 1$  dan

$$p_1 = y_0 + h f(t_0, y_0) = 1 + 1 \frac{0-1}{2} = 0.5$$

$$y_1 = y_0 + \frac{h}{2}(f(t_0, y_0) + f(t_1, p_1)) = 1 + \frac{1}{2} \left( \frac{0-1}{2} + \frac{1-0.5}{2} \right) = 0.875$$

Dengan mengulangi Persamaan 12.16, akan didapatkan nilai  $y_2$  dan  $y_3$  serta nilai galat masing-masing hampiran terhadap nilai eksak solusi  $y(t) = 3e^{-t/2} - 2 + t$  yang dapat dilihat pada Tabel 12.3 berikut.

**Tabel 12.3:** Metode Heun untuk  $y' = (t - y)/2$  dengan  $y(0) = 1$  dan  $h = 1$ .

$k$	$t_k$	$y_k$	Nilai eksak, $y(t_k)$	galat, $e$
0	0	1.00000	1.00000	0.00000
1	1	0.87500	0.81959	0.05541
2	2	1.17188	1.10364	0.06824
3	3	1.73242	1.66939	0.06303

△

### Kepresian Metode Heun

**Teorema 12.2 Kepresian Metode Heun:** Asusmsikan bahwa  $y(t)$  merupakan solusi dari suatu masalah nilai awal. Jika  $y(t) \in C^3[t_0, b]$  dan  $\{(t_k, y_k)\}_{k=0}^M$  adalah barisan hampiran yang dihasilkan oleh metode Heun, maka

$$\begin{aligned}|e_k| &= |y(t_k) - y_k| = O(h^2) \\ |\varepsilon_{k+1}| &= |y(t_{k+1}) - y_k - h \Phi(t_k, y_k)| = O(h^3)\end{aligned}\quad (12.17)$$

dengan  $\Phi(t_k, y_k) = y_k + (h/2)(f(t_k, y_k) + f(t_{k+1}, y_k + h f(t_k, y_k)))$ . Serta, galat global akhir (*final global error*) pada akhir interval akan memenuhi

$$E(y(b), h) = |y(b) - y_M| = O(h^2) \quad (12.18)$$

■

**Contoh 12.4 :** Bandingkan galat global akhir yang dihasilkan oleh metode Heun untuk menyelesaikan masalah nilai awal

$$y' = \frac{t - y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1, \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{64}$ .

**Solusi :** Untuk  $h = 1$ , berdasarkan hasil yang diperoleh pada Contoh 12.3, didapatkan nilai hampiran untuk  $y(3)$  yaitu  $y_3 = 1.73242$  dan galat global akhir yaitu 0.06303. Dengan mengulangi langkah-langkah pada Contoh 12.3 untuk ukuran langkah  $h = \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{64}$ , akan didapatkan nilai hampiran untuk  $y(3)$  beserta galat global akhir masing-masing hampiran seperti pada Tabel 12.4 berikut.

**Tabel 12.4:** Perbandingan nilai galat global akhir metode Heun.

Ukuran langkah, $h$	Banyak sub-interval, $M$	Hampiran $y(3), y_M$	Galat global akhir, $E(y(3), h)$
1	3	1.73242	0.06303
$\frac{1}{2}$	6	1.68212	0.01273
$\frac{1}{4}$	12	1.67227	0.00288
$\frac{1}{8}$	24	1.67008	0.00069
$\frac{1}{16}$	48	1.66956	0.00017
$\frac{1}{32}$	96	1.66943	0.00004
$\frac{1}{64}$	192	1.66940	0.00001

Berdasarkan hasil yang diperoleh pada Tabel 12.4, setiap perubahan ukuran langkah sebesar  $\frac{1}{2}h$  akan mengakibatkan perubahan galat global akhir sekitar  $\frac{1}{4}E$ . Jadi, hubungan antara ukuran langkah dan galat global akhir adalah  $O(h^2)$ .  $\triangle$

## 12.4 Metode Deret Taylor

Metode deret Taylor dapat diterapkan secara umum dan dapat dijadikan standar untuk membandingkan akurasi dari metode numerik lain dalam menyelesaikan suatu masalah nilai awal. Metode ini dapat dirancang untuk memiliki akurasi sesuai kebutuhan. Berikut merupakan teorema ekspansi Taylor yang telah disesuaikan untuk menyelesaikan persamaan differensial.

**Teorema 12.3 :** Asumsikan bahwa  $y(t) \in C^{N+1}[t_0, b]$ . Berikut merupakan ekspansi Taylor ordo  $N$  dari  $y(t)$  di sekitar  $t = t_k \in [t_0, b]$ .

$$y(t_k + h) = y(t_k) + hT_N(t_k, y(t_k)) + O(h^{N+1}) \quad (12.19)$$

dengan

$$T_N(t_k, y(t_k)) = \sum_{j=1}^N \frac{y^{(j)}(t_k)}{j!} h^{j-1} \quad (12.20)$$

dan  $y^{(j)}(t) = f^{(j-1)}(t, y(t))$ .  $\blacksquare$

Solusi numerik dari masalah nilai awal  $y'(t) = f(t, y)$  pada  $[t_0, t_M]$  diturunkan menggunakan Persamaan 12.19 untuk setiap sub-interval  $[t_k, t_{k+1}]$ . Bentuk umum dari metode Taylor ordo  $N$  adalah

$$y_{k+1} = y_k + d_1 h + \frac{d_2 h^2}{2!} + \frac{d_3 h^3}{3!} + \cdots + \frac{d_N h^N}{N!} \quad (12.21)$$

dengan  $d_j = y^{(j)}(t_k)$  untuk  $j = 1, 2, \dots, N$  pada setiap langkah  $k = 0, 1, \dots, M - 1$ .

**Contoh 12.5 :** Gunakan metode deret Taylor ordo 4 untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1$ .

**Solusi :** Langkah pertama untuk menyelesaikan masalah nilai awal menggunakan metode deret Taylor ordo 4 adalah mendefinisikan turunan dari  $y(t)$  hingga turunan ke-4 sebagai berikut.

$$\begin{aligned} y'(t) &= \frac{t-y}{2} \\ y^{(2)}(t) &= \frac{d}{dt} \left( \frac{t-y}{2} \right) = \frac{1-y'}{2} = \frac{2-t+y}{4} \\ y^{(3)}(t) &= \frac{d}{dt} \left( \frac{2-t+y}{4} \right) = \frac{0-1+y'}{4} = \frac{-2+t-y}{8} \\ y^{(4)}(t) &= \frac{d}{dt} \left( \frac{-2+t-y}{8} \right) = \frac{-0+1-y'}{8} = \frac{2-t+y}{16} \end{aligned}$$

Diketahui bahwa nilai awal  $t_0 = 0$  dan  $y_0 = 1$ . Selanjutnya, akan dihitung nilai  $d_1, d_2, d_3$ , dan  $d_4$  seperti berikut.

$$\begin{aligned}d_1 &= y'(t_0) = -0.5 \\d_2 &= y^{(2)}(t_0) = 0.75 \\d_3 &= y^{(3)}(t_0) = -0.375 \\d_4 &= y^{(4)}(t_0) = 0.1875\end{aligned}$$

Berdasarkan Persamaan 12.21 dengan  $N = 4$ , didapatkan nilai  $t_1 = 1$  dan

$$y_1 = y_0 + d_1 h + \frac{d_2 h^2}{2!} + \frac{d_3 h^3}{3!} + \frac{d_4 h^4}{4!}$$

Tulis ulang persamaan, sehingga operasi pangkat terhadap  $h$  dapat dihilangkan untuk mengurangi galat akhir seperti berikut.

$$\begin{aligned}y_1 &= y_0 + h \left( d_1 + h \left( \frac{d_2}{2!} + h \left( \frac{d_3}{3!} + h \frac{d_4}{4!} \right) \right) \right) \\&= 1 + 1 \left( -0.5 + 1 \left( \frac{0.75}{2!} + 1 \left( \frac{-0.375}{3!} + 1 \frac{0.1875}{4!} \right) \right) \right) \\&= 0.82031\end{aligned}$$

Dengan mengulangi Persamaan 12.16, akan didapatkan nilai  $y_2$  dan  $y_3$  serta nilai galat masing-masing hampiran terhadap nilai eksak solusi  $y(t) = 3e^{-t/2} - 2 + t$  yang dapat dilihat pada Tabel 12.5 berikut.

**Tabel 12.5:** Metode deret Taylor ordo 4 untuk  $y' = (t - y)/2$  dengan  $y(0) = 1$  dan  $h = 1$ .

$k$	$t_k$	$y_k$	Nilai eksak, $y(t_k)$	galat, $e$
0	0	1.00000	1.00000	0.00000
1	1	0.82031	0.81959	0.00072
2	2	1.10451	1.10364	0.00087
3	3	1.67019	1.66939	0.00080

△

### Kepresisan Metode Deret Taylor

**Teorema 12.4 Kepresisan Deret Taylor:** Asusmsikan bahwa  $y(t)$  merupakan solusi dari suatu masalah nilai awal. Jika  $y(t) \in C^{N+1}[t_0, b]$  dan  $\{(t_k, y_k)\}_{k=0}^M$  adalah barisan hampiran yang dihasilkan oleh metode Taylor ordo  $N$ , maka

$$\begin{aligned}|e_k| &= |y(t_k) - y_k| = O(h^{N+1}) \\|\varepsilon_{k+1}| &= |y(t_{k+1}) - y_k - h T_N(t_k, y_k)| = O(h^N)\end{aligned}\tag{12.22}$$

Serta, galat global akhir (*final global error*) pada akhir interval akan memenuhi

$$E(y(b), h) = |y(b) - y_M| = O(h^N)\tag{12.23}$$

■

**Contoh 12.6 :** Bandingkan galat global akhir yang dihasilkan oleh metode deret Taylor ordo 4 untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ .

**Solusi :** Untuk  $h = 1$ , berdasarkan hasil yang diperoleh pada Contoh 12.5, didapatkan nilai hampiran untuk  $y(3)$  yaitu  $y_3 = 1.67019$  dan galat global akhir yaitu 0.0008. Dengan mengulangi langkah-langkah pada Contoh 12.5 untuk ukuran langkah  $h = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ , akan didapatkan nilai hampiran untuk  $y(3)$  beserta galat global akhir masing-masing hampiran seperti pada Tabel 12.6 berikut.

**Tabel 12.6:** Perbandingan nilai galat global akhir metode deret Taylor ordo 4.

Ukuran langkah, $h$	Banyak sub-interval, $M$	Hampiran $y(3), y_M$	Galat global akhir, $E(y(3), h)$
1	3	1.6701860	0.0007955
$\frac{1}{2}$	6	1.6694308	0.0000403
$\frac{1}{4}$	12	1.6693927	0.0000023
$\frac{1}{8}$	24	1.6693906	0.0000001

Berdasarkan hasil yang diperoleh pada Tabel 12.6, setiap perubahan ukuran langkah sebesar  $\frac{1}{2}h$  akan mengakibatkan perubahan galat global akhir sekitar  $\frac{1}{16}E$ . Jadi, hubungan antara ukuran langkah dan galat global akhir adalah  $O(h^4)$ .  $\triangle$

## 12.5 Metode Runge-Kutta

Metode memiliki keunggulan yaitu galat global akhir (*final global error*) memiliki ordo  $O(h^N)$  dan semakin besar  $N$  yang dipilih akan mengakibatkan nilai galat semakin kecil. Namun, penggunaan metode Taylor memiliki kelemahan yaitu untuk mendapatkan ordo  $N$  harus memerlukan turunan dari  $y(t)$  hingga turunan ke- $N$ . Hal ini mengakibatkan metode Taylor sulit untuk diterapkan meskipun memiliki ordo galat yang bagus. Untuk mengatasi masalah tersebut, metode Runge-Kutta dikembangkan dari metode Taylor yang sesuai, sehingga metode ini mampu mendapatkan galat global akhir dengan ordo  $O(h^N)$ , tanpa harus mencari turunan tingkat tinggi dari  $y(t)$ .

Metode Runge-Kutta dapat dikembangkan untuk setiap ordo  $N$ . Namun, metode Runge-Kutta dengan ordo  $N = 4$  merupakan metode yang populer dan sering digunakan, karena memiliki akurasi yang cukup baik, stabil dan mudah untuk pemrogramannya. Metode Runge-Kutta dengan ordo yang lebih tinggi akan memiliki akurasi yang lebih baik, tetapi dibutuhkan pemrograman yang lebih kompleks. Dengan demikian, jika dibutuhkan nilai akurasi yang lebih baik, maka cukup memperkecil ukuran langkah  $h$  metode Runge-Kutta ordo 4 atau menggunakan metode adaptif daripada memilih untuk menggunakan metode Runge-Kutta dengan ordo yang lebih tinggi.

Bentuk umum metode Runge-Kutta ordo 4 untuk menyelesaikan masalah nilai awal  $y'(t) = f(t, y)$  pada  $[t_0, t_M]$  adalah

$$y_{k+1} = y_k + \frac{h}{6}(f_1 + 2f_2 + 2f_3 + f_4) \quad (12.24)$$

dengan

$$\begin{aligned}
 f_1 &= f(t_k, y_k) \\
 f_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2}f_1\right) \\
 f_3 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2}f_2\right) \\
 f_4 &= f(t_k + h, y_k + h f_3)
 \end{aligned} \tag{12.25}$$

**Contoh 12.7 :** Gunakan metode Runge-Kutta ordo 4 untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1$ .

**Solusi :** Diketahui bahwa  $t_0 = 0$  dan  $y_0 = 1$ . Langkah pertama untuk menyelesaikan masalah nilai awal menggunakan metode Runge-Kutta ordo 4 adalah menghitung nilai  $f_1, f_2, f_3$ , dan  $f_4$  yang terdapat pada Persamaan 12.25.

$$\begin{aligned}
 f_1 &= f(t_0, y_0) = f(0, 1) = \frac{0-1}{2} = -0.5 \\
 f_2 &= f\left(t_0 + \frac{h}{2}, y_0 + \frac{h}{2}f_1\right) = f(0.5, 0.75) = \frac{0.5-0.75}{2} = -0.125 \\
 f_3 &= f\left(t_0 + \frac{h}{2}, y_0 + \frac{h}{2}f_2\right) = f(0.5, 0.9375) = \frac{0.5-0.9375}{2} = -0.21875 \\
 f_4 &= f(t_0 + h, y_0 + h f_3) = f(1, 0.78125) = \frac{1-0.78125}{2} = 0.10938
 \end{aligned}$$

Berdasarkan Persamaan 12.24, didapatkan nilai  $t_1 = 1$  dan

$$\begin{aligned}
 y_1 &= y_0 + \frac{h}{6}(f_1 + 2f_2 + 2f_3 + f_4) \\
 &= 1 + \frac{1}{6}(-0.5 + 2(-0.125) + 2(-0.21875) + 0.10938) \\
 &= 0.82031
 \end{aligned}$$

Dengan mengulangi langkah di atas, akan didapatkan nilai  $y_2$  dan  $y_3$  serta nilai galat masing-masing hampiran terhadap nilai eksak solusi  $y(t) = 3e^{-t/2} - 2 + t$  yang dapat dilihat pada Tabel 12.7 berikut. Hasil yang diperoleh tersebut sama dengan hasil yang diperoleh metode Taylor ordo 4 pada Contoh 12.5.

**Tabel 12.7:** Metode Runge-Kutta ordo 4 untuk  $y' = (t-y)/2$  dengan  $y(0) = 1$  dan  $h = 1$ .

$k$	$t_k$	$y_k$	Nilai eksak, $y(t_k)$	galat, $e$
0	0	1.00000	1.00000	0.00000
1	1	0.82031	0.81959	0.00072
2	2	1.10451	1.10364	0.00087
3	3	1.67019	1.66939	0.00080

△

### Kepresisan Metode Runge-Kutta

**Teorema 12.5 Kepresisan metode Runge-Kutta ordo 4:** Asusmsikan bahwa  $y(t)$  merupakan solusi dari suatu masalah nilai awal. Jika  $y(t) \in C^{N+1}[t_0, b]$  dan  $\{(t_k, y_k)\}_{k=0}^M$  adalah barisan hampiran yang dihasilkan oleh metode Runge-Kutta ordo 4, maka

$$\begin{aligned}|e_k| &= |y(t_k) - y_k| = O(h^4) \\ |\varepsilon_{k+1}| &= |y(t_{k+1}) - y_k - h T_N(t_k, y_k)| = O(h^5)\end{aligned}\quad (12.26)$$

Serta, galat global akhir (*final global error*) pada akhir interval akan memenuhi

$$E(y(b), h) = |y(b) - y_M| = O(h^4) \quad (12.27)$$

■

**Contoh 12.8 :** Bandingkan galat global akhir yang dihasilkan oleh metode Runge-Kutta ordo 4 untuk menyelesaikan masalah nilai awal

$$y' = \frac{t-y}{2} \quad \text{dengan } y(0) = 1$$

pada interval  $[0, 3]$  menggunakan ukuran langkah  $h = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ .

**Solusi :** Untuk  $h = 1$ , berdasarkan hasil yang diperoleh pada Contoh 12.7, didapatkan nilai hampiran untuk  $y(3)$  yaitu  $y_3 = 1.67019$  dan galat global akhir yaitu 0.0008. Dengan mengulangi langkah-langkah pada Contoh 12.7 untuk ukuran langkah  $h = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ , akan didapatkan nilai hampiran untuk  $y(3)$  beserta galat global akhir masing-masing hampiran seperti pada Tabel 12.8 berikut.

**Tabel 12.8:** Perbandingan nilai galat global akhir metode Runge-Kutta ordo 4.

Ukuran langkah, $h$	Banyak sub-interval, $M$	Hampiran $y(3), y_M$	Galat global akhir, $E(y(3), h)$
1	3	1.6701860	0.0007955
$\frac{1}{2}$	6	1.6694308	0.0000403
$\frac{1}{4}$	12	1.6693927	0.0000023
$\frac{1}{8}$	24	1.6693906	0.0000001

Berdasarkan hasil yang diperoleh pada Tabel 12.8, setiap perubahan ukuran langkah sebesar  $\frac{1}{2}h$  akan mengakibatkan perubahan galat global akhir sekitar  $\frac{1}{16}E$ . Jadi, hubungan antara ukuran langkah dan galat global akhir adalah  $O(h^4)$ . △

## 12.6 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Euler** berisi program untuk mencari solusi persamaan differensial  $y' = f(t,y)$  dengan masalah nilai awal  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $f(t,y)$ , titik ujung interval penyelesaian  $[a,b]$ , nilai awal  $y_0$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai awal  $sol$ . Berikut merupakan program untuk metode Euler.

### Algoritma 12.1: Metode Euler

```
function euler(f,a,b,y0,M)
    M = Int(M)
    h = (b-a)/M;
    T = a:h:b;
    Y = Array{Float64}(undef,length(T),1)
    Y[1] = y0;
    # Mulai langkah iterasi euler
    for k = 1:M
        # Rumus iterasi euler
        Y[k+1] = Y[k]+h*f(T[k],Y[k])
    end
    sol = [T Y]
end
```

**Metode Heun** berisi program untuk mencari solusi persamaan differensial  $y' = f(t,y)$  dengan masalah nilai awal  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $f(t,y)$ , titik ujung interval penyelesaian  $[a,b]$ , nilai awal  $y_0$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai awal  $sol$ . Berikut merupakan program untuk metode Heun.

### Algoritma 12.2: Metode Heun

```
function heun(f,a,b,y0,M)
    M = Int(M)
    h = (b-a)/M;
    T = a:h:b;
    Y = Array{Float64}(undef,length(T),1)
    P = Array{Float64}(undef,length(T),1)
    Y[1] = y0;
    # Mulai langkah iterasi Heun
    for k = 1:M
        # Rumus iterasi Heun
        P[k+1]=Y[k]+h*f(T[k],Y[k]);
        Y[k+1]=Y[k]+h/2*(f(T[k],Y[k])+f(T[k+1],P[k+1]));
    end
    sol = [T Y];
end
```

**Metode Deret Taylor Orde-4** berisi program untuk mencari solusi persamaan differensial  $y' = f(t,y)$  dengan masalah nilai awal  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $df$  (berisi nilai dari  $y'$ ,  $y''$ ,  $y'''$ ,  $y^{(4)}$ ), titik ujung interval penyelesaian  $[a,b]$ , nilai awal  $y_0$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai awal  $sol$ . Berikut merupakan program untuk metode deret Taylor orde-4.

**Algoritma 12.3:** Metode deret Taylor orde-4

```
function taylor(df,a,b,y0,M)
M = Int(M)
h = (b-a)/M;
T = a:h:b;
Y = Array{Float64}(undef,length(T),1)
Y[1]=y0;
for j = 1:M
    D = df(T[j],Y[j]);
    Y[j+1]=Y[j]+h*(D[1]+h*(D[2]/2+h*(D[3]/6+h*D[4]/24)));
end
sol = [T Y];
end
```

**Metode Runge-Kutta Orde-4** berisi program untuk mencari solusi persamaan differensial  $y' = f(t, y)$  dengan masalah nilai awal  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $f(t, y)$ , titik ujung interval penyelesaian  $[a, b]$ , nilai awal  $y_0$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai awal  $sol$ . Berikut merupakan program untuk metode Runge-Kutta orde-4.

**Algoritma 12.4:** Metode Runge-Kutta orde-4

```
function rungekutta(f,a,b,y0,M)
M = Int(M)
h = (b-a)/M;
T = a:h:b;
Y = Array{Float64}(undef,length(T),1)
Y[1] = y0;
for k = 1:M
    f1 = f(T[k] ,Y[k] );
    f2 = f(T[k]+h/2 ,Y[k]+f1*h/2 );
    f3 = f(T[k]+h/2 ,Y[k]+f2*h/2 );
    f4 = f(T[k]+h ,Y[k]+f3*h );
    Y[k+1] = Y[k] + h/6*(f1+2*f2+2*f3+f4);
end
sol = [T Y];
end
```

**12.6.1 Metode Euler**

Metode Euler merupakan metode paling klasik yang digunakan untuk menghitung solusi numerik dari persamaan differensial biasa dengan masalah nilai awal. Metode Euler memiliki kompleksitas komputasi  $O(h)$ .

**Contoh 12.9 :** Diberikan persamaan differensial

$$\frac{dy}{dt} = \frac{t-y}{2}, \text{ dengan } y(0) = 1$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari persamaan di atas dengan  $h = 1/2$  dan  $h = 1/4$  dan galat hampiran terhadap solusi eksaknya, yaitu  $y(t) = 3e^{-t/2} - 2 + t$  menggunakan metode Euler.

**Langkah 1:** Penghitungan solusi PD pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1/2$  dan  $1/4$  menggunakan metode Euler.

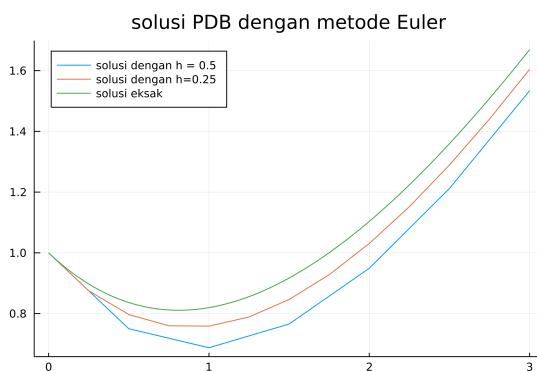
```
Inp: a = 0;
    b = 3;
    f(t,y) = (t-y)/2;
    y0 = 1;
Inp: h = 1/2;
    M = (b-a)/h;
    sol1 = euler(f,a,b,y0,M)
Inp: h = 1/4;
    M = (b-a)/h;
    sol2 = euler(f,a,b,y0,M)
```

**Langkah 2:** Pembuatan gambar solusi hampiran terhadap solusi eksak.

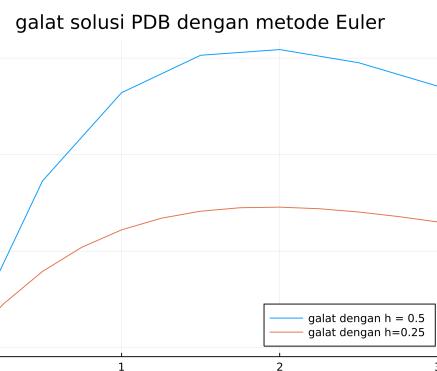
```
Inp: y(t) = 3*exp(-t/2)-2+t;
    t1 = sol1[:,1];
    y1 = sol1[:,2];
    t2 = sol2[:,1];
    y2 = sol2[:,2];
    t = a:0.01:b
Inp: plot(t1,y1,label="solusi dengan h = 0.5")
    plot!(t2,y2,label="solusi dengan h=0.25")
    plot!(t,y.(t),label="solusi eksak")
    title!("solusi PDB dengan metode Euler")
```

**Langkah 3:** Penghitungan dan pembuatan gambar nilai galat hampiran.

```
Inp: y(t) = 3*exp(-t/2)-2+t;
    e1 = abs.(y1 - y.(t1));
    e2 = abs.(y2 - y.(t2));
Inp: plot(t1,e1,label="galat dengan h = 0.5", legend=:bottomright)
    plot!(t2,e2,label="galat dengan h=0.25")
    title!("galat solusi PDB dengan metode Euler")
```



(a) Perbandingan solusi hampiran.



(b) Perbandingan galat hampiran.

**Gambar 12.2:** Perbandingan solusi dan galat hampiran metode Euler PD  $y' = (t - y)/2$  dengan  $y(0) = 1$  menggunakan ukuran langkah  $h = 1/2$  dan  $1/4$ .

Berdasarkan hasil penghitungan galat, nilai galat akhir global untuk hampiran dengan  $h = 1/2$  dan  $h = 1/4$  masing-masing adalah 0.13545 dan 0.065139. Dengan demikian, kompleksitas komputasi untuk metode Euler adalah  $O(h)$ .

**Soal Latihan:** Diberikan persamaan differensial

$$\frac{dy}{dt} = -y - 2t - 1, \text{ dengan } y(0) = 2$$

Gunakan metode Euler untuk menyelesaikan PD di atas untuk  $h = 1/2$ ,  $h = 1/4$  dan  $h = 1/8$ , kemudian bandingkan dengan solusi eksaknya adalah  $y(t) = e^{-t} - 2t + 1$  dan hitung nilai galat akhir global masing-masing hampiran dengan langkah-langkah seperti pada Contoh 12.9.

## 12.6.2 Metode Heun

Metode Heun merupakan pengembangan dari metode Euler. Metode Heun memiliki kompleksitas komputasi yang lebih baik daripada metode Euler, yaitu  $O(h^2)$ .

**Contoh 12.10 :** Diberikan persamaan differensial

$$\frac{dy}{dt} = \frac{t-y}{2}, \text{ dengan } y(0) = 1$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari persamaan di atas dengan  $h = 1/2$  dan  $h = 1/4$  dan galat hampiran terhadap solusi eksaknya, yaitu  $y(t) = 3e^{-t/2} - 2 + t$  menggunakan metode Heun.

**Langkah 1:** Penghitungan solusi PD pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1/2$  dan  $1/4$  menggunakan metode Heun.

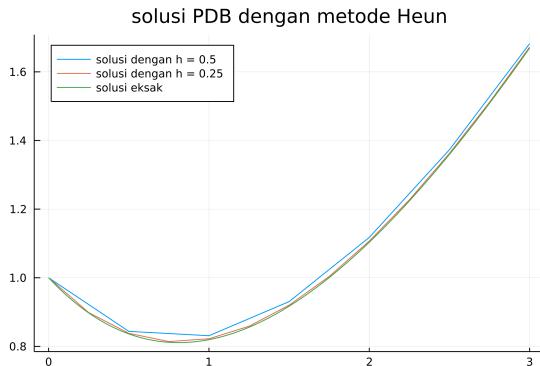
```
Inp: a = 0;
      b = 3;
      f(t,y) = (t-y)/2;
      y0 = 1;
Inp: h = 1/2;
      M = (b-a)/h;
      sol1 = heun(f,a,b,y0,M)
Inp: h = 1/4;
      M = (b-a)/h;
      sol2 = heun(f,a,b,y0,M)
```

**Langkah 2:** Pembuatan gambar solusi hampiran terhadap solusi eksak.

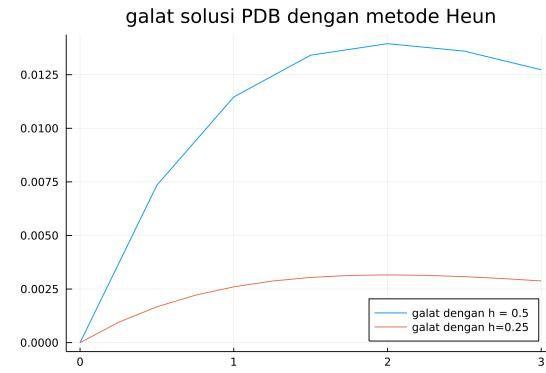
```
Inp: y(t) = 3*exp(-t/2)-2+t;
      t1 = sol1[:,1];
      y1 = sol1[:,2];
      t2 = sol2[:,1];
      y2 = sol2[:,2];
      t = a:0.01:b
Inp: plot(t1,y1,label="solusi dengan h = 0.5")
      plot!(t2,y2,label="solusi dengan h = 0.25")
      plot!(t,y.(t),label="solusi eksak")
      title!("solusi PDB dengan metode Heun")
```

**Langkah 3:** Penghitungan dan pembuatan gambar nilai galat hampiran.

```
Inp: y(t) = 3*exp(-t/2)-2+t;
e1 = abs.(y1 - y.(t1));
e2 = abs.(y2 - y.(t2));
Inp: plot(t1,e1,label="galat dengan h = 0.5",legend=:bottomright)
plot!(t2,e2,label="galat dengan h=0.25")
title!("galat solusi PDB dengan metode Heun")
```



(a) Perbandingan solusi hampiran.



(b) Perbandingan galat hampiran.

**Gambar 12.3:** Perbandingan solusi dan galat hampiran metode Heun PD  $y' = (t - y)/2$  dengan  $y(0) = 1$  menggunakan ukuran langkah  $h = 1/2$  dan  $1/4$ .

Berdasarkan hasil penghitungan galat, nilai galat akhir global untuk hampiran dengan  $h = 1/2$  dan  $h = 1/4$  masing-masing adalah 0.012731 dan 0.0028783. Dengan demikian, kompleksitas komputasi untuk metode Heun adalah  $O(h^2)$ .

**Soal Latihan:** Diberikan persamaan differensial

$$\frac{dy}{dt} = -y - 2t - 1, \text{ dengan } y(0) = 2$$

Gunakan metode Heun untuk menyelesaikan PD di atas untuk  $h = 1/2$ ,  $h = 1/4$  dan  $h = 1/8$ , kemudian bandingkan dengan solusi eksaknya adalah  $y(t) = e^{-t} - 2t + 1$  dan hitung nilai galat akhir global masing-masing hampiran dengan langkah-langkah seperti pada Contoh 12.10.

### 12.6.3 Metode Deret Taylor Orde-4

Metode Euler dan Heun yang telah dipelajari masih kompleksitas yang rendah yaitu  $O(h)$  dan  $O(h^2)$ . Dengan memanfaatkan teorema deret Taylor, kompleksitas komputasi untuk mencari solusi numerik persamaan differensial dapat ditingkatkan sesuai dengan orde yang dipilih. Pada praktikum ini, akan digunakan deret Taylor orde-4 untuk mencari solusi numerik persamaan differensial dengan kompleksitas komputasi  $O(h^4)$ .

**Contoh 12.11 :** Diberikan persamaan differensial

$$\frac{dy}{dt} = \frac{t - y}{2}, \text{ dengan } y(0) = 1$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari persamaan di atas dengan  $h = 1/2$  dan  $h = 1/4$  dan galat hampiran terhadap solusi eksaknya, yaitu  $y(t) = 3e^{-t/2} - 2 + t$  menggunakan metode deret Taylor orde-4.

**Langkah 1:** Pendefinisian fungsi df yang berisi  $y'$ ,  $y''$ ,  $y^{(3)}$ , dan  $y^{(4)}$ . Perhatikan bahwa

$$\begin{aligned}y'(t) &= \frac{t-y}{2} \\y^{(2)}(t) &= \frac{d}{dt}\left(\frac{t-y}{2}\right) = \frac{1-y'}{2} = \frac{2-t+y}{4} \\y^{(3)}(t) &= \frac{d}{dt}\left(\frac{2-t+y}{4}\right) = \frac{0-1+y'}{4} = \frac{-2+t-y}{8} \\y^{(4)}(t) &= \frac{d}{dt}\left(\frac{-2+t-y}{8}\right) = \frac{-0+1-y'}{8} = \frac{2-t+y}{16}\end{aligned}$$

Dengan demikian diperoleh fungsi df yaitu

```
Inp: function df(t,y)
      y1 = (t-y)/2;
      y2 = (2-t+y)/4;
      y3 = (-2+t-y)/8;
      y4 = (2-t+y)/16;
      z = [y1,y2,y3,y4];
end
```

**Langkah 2:** Penghitungan solusi PD pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1/2$  dan  $1/4$  menggunakan metode deret Taylor orde-4.

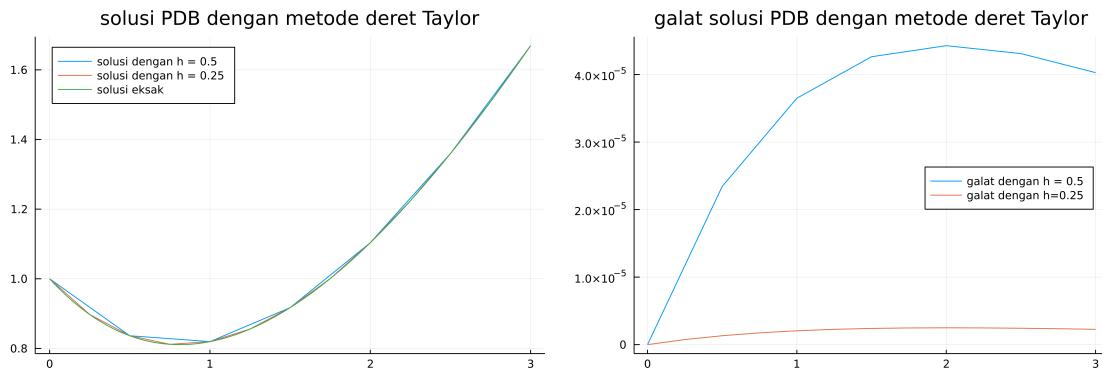
```
Inp: a = 0;
     b = 3;
     y0 = 1;
Inp: h = 1/2;
     M = (b-a)/h;
     sol1 = taylor(df,a,b,y0,M)
Inp: h = 1/4;
     M = (b-a)/h;
     sol2 = taylor(df,a,b,y0,M)
```

**Langkah 3:** Pembuatan gambar solusi hampiran terhadap solusi eksak.

```
Inp: y(t) = 3*exp(-t/2)-2+t;
     t1 = sol1[:,1];
     y1 = sol1[:,2];
     t2 = sol2[:,1];
     y2 = sol2[:,2];
     t = a:0.01:b
Inp: plot(t1,y1,label="solusi dengan h = 0.5")
     plot!(t2,y2,label="solusi dengan h = 0.25")
     plot!(t,y.(t),label="solusi eksak")
     title!("solusi PDB dengan metode deret Taylor")
```

**Langkah 4:** Penghitungan dan pembuatan gambar nilai galat hampiran.

```
Inp: y(t) = 3*exp(-t/2)-2+t;
     e1 = abs.(y1 - y.(t1));
     e2 = abs.(y2 - y.(t2));
Inp: plot(t1,e1,label="galat dengan h = 0.5",legend=:right)
     plot!(t2,e2,label="galat dengan h=0.25")
     title!("galat solusi PDB dengan metode deret Taylor")
```



(a) Perbandingan solusi hampiran.

(b) Perbandingan galat hampiran.

**Gambar 12.4:** Perbandingan solusi dan galat hampiran metode deret Taylor Orde-4 PD  $y' = (t - y)/2$  dengan  $y(0) = 1$  menggunakan ukuran langkah  $h = 1/2$  dan  $1/4$ .

Berdasarkan hasil penghitungan galat, nilai galat akhir global untuk hampiran dengan  $h = 1/2$  dan  $h = 1/4$  masing-masing adalah 0.000040281 dan 0.0000022674. Dengan demikian, kompleksitas komputasi untuk metode deret Taylor Orde-4 adalah  $O(h^4)$ .

**Soal Latihan:** Diberikan persamaan differensial

$$\frac{dy}{dt} = -y - 2t - 1, \text{ dengan } y(0) = 2$$

Gunakan metode deret Taylor Orde-4 untuk menyelesaikan PD di atas untuk ukuran langkah  $h = 1/2$ ,  $h = 1/4$  dan  $h = 1/8$ , kemudian bandingkan dengan solusi eksaknya yaitu  $y(t) = e^{-t} - 2t + 1$  dan hitung pula nilai galat akhir global masing-masing hampiran dengan langkah-langkah seperti pada Contoh 12.11.

#### 12.6.4 Metode Runge-Kutta Orde-4

Kelemahan terbesar mencari solusi numerik menggunakan deret Taylor adalah untuk menghasilkan metode dengan kompleksitas  $O(h^4)$ , dibutuhkan persamaan fungsi turunan hingga orde ke-4. Sementara itu, banyak persamaan differensial yang sulit untuk dicari turunannya. Salah satu metode yang dapat mengatasinya adalah metode Runge-Kutta orde-4. Metode ini memiliki kompleksitas komputasi  $O(h^4)$  tanpa harus mendefinisikan fungsi turunan hingga orde-4.

**Contoh 12.12 :** Diberikan persamaan differensial

$$\frac{dy}{dt} = \frac{t - y}{2}, \text{ dengan } y(0) = 1$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari persamaan di atas dengan  $h = 1/2$  dan  $h = 1/4$  dan galat hampiran terhadap solusi eksaknya, yaitu  $y(t) = 3e^{-t/2} - 2 + t$  menggunakan metode Runge-Kutta orde-4.

**Langkah 1:** Penghitungan solusi PD pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1/2$  dan  $1/4$  menggunakan metode Runge-Kutta orde-4.

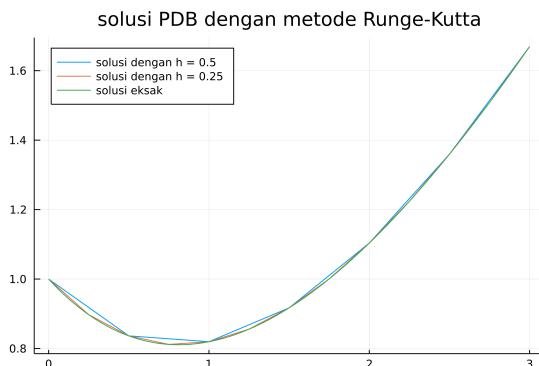
```
Inp: a = 0;
      b = 3;
      f(t,y) = (t-y)/2;
      y0 = 1;
Inp: h = 1/2;
      M = (b-a)/h;
      sol1 = rungekutta(f,a,b,y0,M)
Inp: h = 1/4;
      M = (b-a)/h;
      sol2 = rungekutta(f,a,b,y0,M)
```

**Langkah 2:** Pembuatan gambar solusi hampiran terhadap solusi eksak.

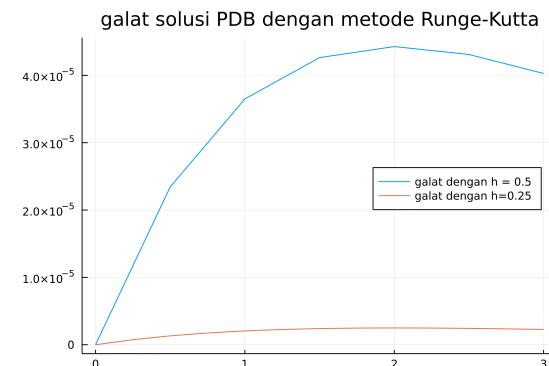
```
Inp: y(t) = 3*exp(-t/2)-2+t;
      t1 = sol1[:,1];
      y1 = sol1[:,2];
      t2 = sol2[:,1];
      y2 = sol2[:,2];
      t = a:0.01:b
Inp: plot(t1,y1,label="solusi dengan h = 0.5")
      plot!(t2,y2,label="solusi dengan h = 0.25")
      plot!(t,y.(t),label="solusi eksak")
      title!("solusi PDB dengan metode Runge-Kutta")
```

**Langkah 3:** Penghitungan dan pembuatan gambar nilai galat hampiran.

```
Inp: y(t) = 3*exp(-t/2)-2+t;
      e1 = abs.(y1 - y.(t1));
      e2 = abs.(y2 - y.(t2));
Inp: plot(t1,e1,label="galat dengan h = 0.5",legend=:right)
      plot!(t2,e2,label="galat dengan h=0.25")
      title!("galat solusi PDB dengan metode Runge-Kutta")
```



(a) Perbandingan solusi hampiran.



(b) Perbandingan galat hampiran.

**Gambar 12.5:** Perbandingan solusi dan galat hampiran metode Runge-Kutta Orde-4 PD  $y' = (t - y)/2$  dengan  $y(0) = 1$  menggunakan ukuran langkah  $h = 1/2$  dan  $1/4$ .

Berdasarkan hasil penghitungan galat, nilai galat akhir global untuk hampiran dengan  $h = 1/2$  dan  $h = 1/4$  masing-masing adalah  $0.000040281$  dan  $0.0000022674$ . Dengan demikian, kompleksitas komputasi untuk metode Rungke-Kutta Orde-4 adalah  $O(h^4)$ .

---

**Soal Latihan:** Diberikan persamaan differensial

$$\frac{dy}{dt} = -y - 2t - 1, \text{ dengan } y(0) = 2$$

Gunakan metode Runge-Kutta Orde-4 untuk menyelesaikan PD di atas untuk ukuran langkah  $h = 1/2$ ,  $h = 1/4$  dan  $h = 1/8$ , kemudian bandingkan dengan solusi eksaknya yaitu  $y(t) = e^{-t} - 2t + 1$  dan hitung nilai galat akhir global masing-masing hampiran dengan langkah-langkah seperti pada Contoh 12.12.

---

## 12.7 Latihan-latihan

### 12.7.1 Ulasan Materi

1. Mengapa pendekatan solusi persamaan diferensial secara numerik dibutuhkan?
2. Jelaskan ide dasar dalam menyelesaikan suatu persamaan diferensial menggunakan pendekatan numerik!
3. Jelaskan apa yang dimaksud dengan solusi umum dan solusi khusus pada persamaan diferensial! Berikan contohnya!
4. Jelaskan apa yang dimaksud dengan masalah nilai awal pada persamaan diferensial! Berikan contohnya!
5. Mengapa metode Euler tetap dipelajari meskipun galat (*error*) yang dihasilkan cukup tinggi?
6. Jelaskan langkah-langkah untuk memperoleh solusi numerik dari masalah nilai awal menggunakan metode Euler!
7. Jelaskan hubungan antara metode Euler dan metode Heun! Apa kegunaan formula pada metode Euler untuk mendapatkan solusi persamaan diferensial dengan metode Heun?
8. Jelaskan ide dasar dan cara memperoleh formula metode Heun!
9. Jelaskan langkah-langkah untuk memperoleh solusi numerik dari masalah nilai awal menggunakan metode Heun!
10. Jelaskan apa yang dimaksud dengan galat diskritisasi lokal dan global! Berikan contohnya!
11. Jelaskan bagaimana deret Taylor dapat digunakan untuk menyelesaikan persamaan diferensial biasa dengan masalah nilai batas!
12. Sebutkan dan jelaskan kelebihan serta kekurangan deret Taylor dalam menyelesaikan persamaan diferensial biasa!
13. Jelaskan kepresisian dari metode deret Taylor!
14. Sebutkan dan jelaskan kelebihan serta kekurangan metode Runge-Kutta ordo 4 dalam menyelesaikan persamaan diferensial biasa!
15. Sebutkan dan bandingkan ordo galat dari metode Euler, Heun, Taylor ordo 4 dan Runge-Kutta ordo 4! Jelaskan pula kelebihan masing-masing metode dibandingkan yang lain!

### 12.7.2 Soal Pemrograman

Untuk soal nomor 1 sampai 6, kerjakan menggunakan langkah-langkah berikut.

- Gunakan metode Euler untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1, 1/2, 1/4, 1/8$  dan  $1/16$ .
- Jika diketahui solusi eksak dari persamaan tersebut adalah  $y(t)$ . Bandingkan setiap solusi numerik dengan solusi eksak dalam satu grafik.
- Hitung nilai galat masing-masing hampiran terhadap solusi eksak  $y(t)$ , kemudian plot nilai grafik masing-masing hampiran dalam satu grafik.
- Hitung nilai galat akhir global masing-masing hampiran kemudian sajikan dalam bentuk tabel.
- Hitung dan periksa ordo galat metode Euler berdasarkan hasil yang diperoleh.

- $y' = t^2 - y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = -e^{-t} + t^2 - 2t + 2$

- $y' = 3y + 3t$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{4}{3}e^{3t} - t - \frac{1}{3}$

- $y' = -ty$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = e^{-t^2/2}$

- $y' = 2ty^2$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1-t^2}$

- $y' = -3ysin(t)$  dengan  $y(0) = 1/2$  dan solusi eksak  $y(t) = \frac{1}{2}\exp(3\cos(t) - 3)$

- $y' = -\frac{1}{1+t}y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1+t}$

Untuk soal nomor 7 sampai 12, kerjakan menggunakan langkah-langkah berikut.

- Gunakan metode Heun untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1, 1/2, 1/4, 1/8$  dan  $1/16$ .
- Jika diketahui solusi eksak dari persamaan tersebut adalah  $y(t)$ . Bandingkan setiap solusi numerik dengan solusi eksak dalam satu grafik.
- Hitung nilai galat masing-masing hampiran terhadap solusi eksak  $y(t)$ , kemudian plot nilai grafik masing-masing hampiran dalam satu grafik.
- Hitung nilai galat akhir global masing-masing hampiran kemudian sajikan dalam bentuk tabel.
- Hitung dan periksa ordo galat metode Heun berdasarkan hasil yang diperoleh.

- $y' = t^2 - y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = -e^{-t} + t^2 - 2t + 2$

- $y' = 3y + 3t$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{4}{3}e^{3t} - t - \frac{1}{3}$

- $y' = -ty$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = e^{-t^2/2}$

- $y' = 2ty^2$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1-t^2}$

- $y' = -3ysin(t)$  dengan  $y(0) = 1/2$  dan solusi eksak  $y(t) = \frac{1}{2}\exp(3\cos(t) - 3)$

- $y' = -\frac{1}{1+t}y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1+t}$

Untuk soal nomor 13 sampai 18, kerjakan menggunakan langkah-langkah berikut.

- Gunakan metode deret Taylor Orde-4 untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1, 1/2, 1/4, 1/8$  dan  $1/16$ .
- Jika diketahui solusi eksak dari persamaan tersebut adalah  $y(t)$ . Bandingkan setiap solusi numerik dengan solusi eksak dalam satu grafik.
- Hitung nilai galat masing-masing hampiran terhadap solusi eksak  $y(t)$ , kemudian plot nilai grafik masing-masing hampiran dalam satu grafik.
- Hitung nilai galat akhir global masing-masing hampiran kemudian sajikan dalam bentuk tabel.
- Hitung dan periksa ordo galat metode deret Taylor Orde-4 berdasarkan hasil yang diperoleh.

13.  $y' = t^2 - y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = -e^{-t} + t^2 - 2t + 2$

14.  $y' = 3y + 3t$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{4}{3}e^{3t} - t - \frac{1}{3}$

15.  $y' = -ty$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = e^{-t^2/2}$

16.  $y' = 2ty^2$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1-t^2}$

17.  $y' = 2 + \sqrt{y-2t+3}$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = 1 + 4t + \frac{1}{4}t^2$

18.  $y' = -y - 2t - 1$  dengan  $y(0) = 2$  dan solusi eksak  $y(t) = e^{-t} - 2t + 1$

Untuk soal nomor 19 sampai 24, kerjakan menggunakan langkah-langkah berikut.

- Gunakan metode Runge-Kutta Orde-4 untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan ukuran langkah  $h = 1, 1/2, 1/4, 1/8$  dan  $1/16$ .
- Jika diketahui solusi eksak dari persamaan tersebut adalah  $y(t)$ . Bandingkan setiap solusi numerik dengan solusi eksak dalam satu grafik.
- Hitung nilai galat masing-masing hampiran terhadap solusi eksak  $y(t)$ , kemudian plot nilai grafik masing-masing hampiran dalam satu grafik.
- Hitung nilai galat akhir global masing-masing hampiran kemudian sajikan dalam bentuk tabel.
- Hitung dan periksa ordo galat metode Runge-Kutta Orde-4 berdasarkan hasil yang diperoleh.

19.  $y' = t^2 - y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = -e^{-t} + t^2 - 2t + 2$

20.  $y' = 3y + 3t$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{4}{3}e^{3t} - t - \frac{1}{3}$

21.  $y' = -ty$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = e^{-t^2/2}$

22.  $y' = 2ty^2$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1-t^2}$

23.  $y' = -3ysin(t)$  dengan  $y(0) = 1/2$  dan solusi eksak  $y(t) = \frac{1}{2}\exp(3\cos(t) - 3)$

24.  $y' = -\frac{1}{1+t}y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1+t}$

25. Diberikan persamaan diferensial biasa dengan masalah nilai awal seperti berikut.

$$y' = -ty \text{ dengan nilai awal } y(0) = 1$$

Solusi eksak persamaan diferensial tersebut adalah  $y(t) = e^{-t^2/2}$

- (a) Bandingkan penggunaan metode Euler, metode Heun dan metode Runge-Kutta orde 4 untuk memperoleh solusi numerik  $y(0.75)$  dengan menggunakan beberapa ukuran langkah  $h = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32},$  dan  $\frac{1}{64}$ .
- (b) Hitung galat masing-masing metode dan ukuran langkah, kemudian lengkapi lah tabel galat akhir berikut ini.

$h$	$e_{\text{Euler}}$	$e_{\text{Heun}}$	$e_{\text{RK4}}$
$1/4$			
$1/8$			
$1/16$			
$1/32$			
$1/64$			

- (c) Hitung dan periksa ordo galat dari masing-masing metode berdasarkan hasil yang diperoleh.

26. Diberikan persamaan diferensial biasa dengan masalah nilai awal seperti berikut.

$$y' = 2ty^2 \text{ dengan nilai awal } y(0) = 1$$

Solusi eksak persamaan diferensial tersebut adalah  $y(t) = \frac{1}{1-t^2}$

- (a) Bandingkan penggunaan metode Euler, metode Heun dan metode Runge-Kutta orde 4 untuk memperoleh solusi numerik  $y(0.75)$  dengan menggunakan beberapa ukuran langkah  $h = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32},$  dan  $\frac{1}{64}$ .
- (b) Hitung galat masing-masing metode dan ukuran langkah, kemudian lengkapi lah tabel galat akhir berikut ini.

$h$	$e_{\text{Euler}}$	$e_{\text{Heun}}$	$e_{\text{RK4}}$
$1/4$			
$1/8$			
$1/16$			
$1/32$			
$1/64$			

- (c) Hitung dan periksa ordo galat dari masing-masing metode berdasarkan hasil yang diperoleh.

---

---

## BAB 13

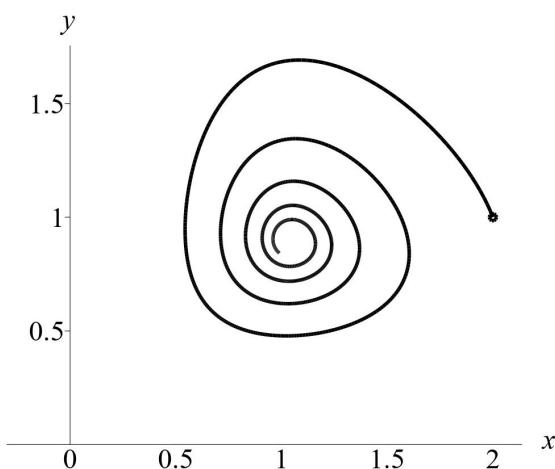
---

### PERSAMAAN DIFFERENSIAL BIASA 2

Sistem persamaan differensial merupakan salah satu formula yang sering digunakan untuk pemodelan matematika pada bidang sains dan teknik. Namun, sering kali suatu sistem persamaan differensial tidak dapat diselesaikan secara analitik dan membutuhkan penyelesaian secara numerik. Sebagai contoh, berikut merupakan model dinamika populasi berupa modifikasi dari persamaan Lotka-Volterra:

$$\begin{aligned}x' &= f(t, x, y) = x - xy - 0.1x^2 \\y' &= g(t, x, y) = xy - y - 0.05y^2\end{aligned}$$

dengan kondisi awal  $x(0) = 2$  dan  $y(0) = 1$  untuk  $t \in [0, 30]$ . Meskipun solusi numerik hanya berupa daftar dari kumpulan angka, hal ini akan sangat berguna apabila digambarkan dalam suatu garis yang menghubungkan titik-titik hampiran  $\{(x_k, y_k)\}$  seperti yang dapat dilihat pada Gambar 13.1 berikut.



**Gambar 13.1:** Lintasan sistem persamaan differensial  $x'$  dan  $y'$ .

Pada bab ini, akan dipelajari beberapa metode yang digunakan untuk menyelesaikan sistem persamaan differensial biasa dan masalah nilai batas secara numerik. Akan tetapi, sebelum membahas materi tersebut, akan dibahas terlebih dulu modifikasi dari Runge-Kutta Orde 4, yaitu Runge-Kutta-Fehlberg 4/5 (RKF45), yang menggunakan nilai  $h$  adaptif untuk menyelesaikan suatu persamaan diferensial biasa dengan masalah nilai awal.

### 13.1 Metode Runge-Kutta-Fehlberg 4/5

Salah satu cara untuk menjamin akurasi dalam solusi persamaan diferensial biasa dengan masalah nilai awal adalah memecahkan masalah menjadi dua, yaitu menggunakan ukuran langkah  $h$  dan  $h/2$ , kemudian jawaban pada titik yang sesuai dengan ukuran langkah yang lebih besar dibandingkan. Namun hal ini membutuhkan perhitungan yang cukup banyak untuk ukuran langkah yang lebih kecil dan harus diulang jika toleransi yang diberikan tidak terpenuhi.

Metode Runge-Kutta-Fehlberg (RKF45) adalah salah satu cara untuk mencoba menyelesaikan masalah ini. Metode ini memiliki prosedur untuk menentukan ukuran langkah yang tepat  $h$  sedang digunakan. Pada setiap langkah, dua pendekatan berbeda untuk solusi dibuat dan dibandingkan. Jika kedua jawaban itu sangat cocok, maka pendekatannya diterima. Jika dua jawaban tidak sesuai dengan toleransi yang ditentukan, maka ukuran langkah dikurangi. Jika jawaban memiliki akurasi yang jauh lebih baik daripada toleransi yang ditentukan, maka ukuran langkah ditingkatkan.

Setiap langkah metode RKF45 memerlukan penggunaan enam nilai berikut.

$$\begin{aligned} k_1 &= hf(t_k, y_k) \\ k_2 &= hf\left(t_k + \frac{1}{4}h, y_k + \frac{1}{4}k_1\right) \\ k_3 &= hf\left(t_k + \frac{3}{8}h, y_k + \frac{3}{32}k_1 + \frac{9}{32}k_2\right) \\ k_4 &= hf\left(t_k + \frac{12}{13}h, y_k + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right) \\ k_5 &= hf\left(t_k + h, y_k + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right) \\ k_6 &= hf\left(t_k + \frac{1}{2}h, y_k - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right) \end{aligned}$$

Selanjutnya, pendekatan solusi dari persamaan diferensial biasa dengan masalah nilai awal dibuat menggunakan metode Runge-Kutta ordo 4, yaitu

$$y_{k+1} = y_k + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4101}k_4 - \frac{1}{5}k_5 \quad (13.1)$$

Perhatikan bahwa persamaan 13.1 hanya menggunakan nilai  $k_1, k_3, k_4$ , dan  $k_5$ , tetapi nilai  $k_2$  tidak digunakan. Nilai yang lebih baik untuk solusi ditentukan menggunakan metode Runge-Kutta orde 5, yaitu

$$z_{k+1} = y_k + \frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6 \quad (13.2)$$

Ukuran langkah optimal  $sh$  dapat ditentukan dengan mengalikan skalar  $s$  dengan ukuran langkah saat ini  $h$  dengan skalar  $s$  adalah

$$s = \left( \frac{\text{tol } h}{2|z_{k+1} - y_{k+1}|} \right)^{1/4} \approx 0.84 \left( \frac{\text{tol } h}{|z_{k+1} - y_{k+1}|} \right)^{1/4} \quad (13.3)$$

dengan tol adalah toleransi kontrol yang ditentukan.

Berdasarkan metode ini, penting untuk dipelajari bahwa ukuran langkah yang tetap bukanlah strategi yang terbaik meskipun dapat memberikan tabel nilai yang tampak lebih bagus. Jika diperlukan nilai yang tidak ada dalam tabel, nilai tersebut dapat ditentukan menggunakan interpolasi polinomial.

**Contoh 13.1 :** Bandingkan solusi metode RKF45 dan RK4 untuk suatu masalah persamaan diferensial biasa

$$y' = 1 + y^2 \text{ dengan } y(0) = 0 \text{ pada } [0, 1.4]$$

Program RKF45 menggunakan nilai  $\text{tol} = 2 \times 10^{-5}$  sebagai toleransi kontrol kesalahan. Metode ini mengubah ukuran langkah  $h$  secara automatis dan menghasilkan 10 nilai hampiran untuk solusi pada (Tabel 13.1). Sementara itu, Program RK4 menggunakan ukuran langkah  $h = 0.1$ , yang mengharuskan komputer untuk menghasilkan 14 nilai hampiran pada titik-titik yang berjarak sama (Tabel 13.2). Nilai hampiran di titik  $x = 1.4$  yang dihasilkan oleh program RKF45 dan RK 4 secara berturut-turut adalah

$$y(1.4) \approx y_{10} = 5.7985045 \text{ dan } y(1.4) \approx y_{14} = 5.7919748$$

dengan galat absolut yaitu  $E_{10} = 0.0006208$  dan  $E_{14} = 0.0059089$ . Berdasarkan hasil tersebut, metode RKF45 memiliki *error* yang lebih kecil serta membutuhkan titik hampiran yang lebih sedikit dibandingkan metode RK4.  $\triangle$

**Tabel 13.1:** Solusi metode RKF45 untuk Contoh 13.1.

$k$	$t_k$	Hampiran RKF45, $y_k$	Solusi eksak, $\tan(t_k)$	Error, $ y(t_k) - \tan(t_k) $
0	0.0	0.0000000	0.0000000	0.0000000
1	0.2	0.2027100	0.2027100	0.0000000
2	0.4	0.4227933	0.4227931	0.0000002
3	0.6	0.6841376	0.6841368	0.0000008
4	0.8	1.0296434	1.0296386	0.0000048
5	1.0	1.5574398	1.5774077	0.0000321
6	1.1	1.9648085	1.9647597	0.0000488
7	1.2	2.5722408	2.5721516	0.0000892
8	1.3	3.6023295	3.6021024	0.0002271
9	1.35	4.4555714	4.4552218	0.0003496
10	1.4	5.7985045	5.7978837	0.0006208

**Tabel 13.2:** Solusi metode RK4 untuk Contoh 13.1.

$k$	$t_k$	Hampiran RK4, $y_k$	Solusi eksak, $\tan(t_k)$	Error, $ y(t_k) - \tan(t_k) $
0	0.0	0.0000000	0.0000000	0.0000000
1	0.1	0.1003346	0.1003347	0.0000001
2	0.2	0.2027099	0.2027100	0.0000001
3	0.3	0.3093360	0.3093362	0.0000002
4	0.4	0.4227930	0.4227932	0.0000002
5	0.5	0.5463023	0.5463025	0.0000002
6	0.6	0.6841368	0.6841368	0.0000000
7	0.7	0.8422886	0.8422884	0.0000002
8	0.8	1.0296391	1.0296386	0.0000005
9	0.9	1.2601588	1.2601582	0.0000006
10	1.0	1.5574064	1.5574077	0.0000013
11	1.1	1.9647466	1.9647597	0.0000131
12	1.2	2.5720718	2.5721516	0.0000798
13	1.3	3.6015634	3.6021024	0.0005390
14	1.4	5.7919748	5.7978837	0.0059089

## 13.2 Sistem Persamaan Diferensial

Sebagai ilustrasi konsep dasar sistem persamaan differensial, berikut merupakan masalah nilai awal  $dx/dt$  dan  $dy/dt$ .

$$\begin{aligned} \frac{dx}{dt} &= f(t, x, y) \\ \frac{dy}{dt} &= g(t, x, y) \end{aligned} \quad \text{dengan} \quad \begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \end{cases} \quad (13.4)$$

Solusi dari persamaan di atas adalah suatu pasangan dari fungsi terturunkan  $x(t)$  dan  $y(t)$  dengan syarat, yaitu ketika  $t$ ,  $x(t)$ , dan  $y(t)$  disubstitusikan ke  $f(t, x, y)$  dan  $g(t, x, y)$  akan menghasilkan turunan  $x'(t)$  dan  $y'(t)$ . Sebagai contoh, diberikan suatu sistem persamaan differensial

$$\begin{aligned} \frac{dx}{dt} &= f(t, x, y) = x + 2y \\ \frac{dy}{dt} &= g(t, x, y) = 3x + 2y \end{aligned} \quad \text{dengan} \quad \begin{cases} x(0) = 6 \\ y(0) = 4 \end{cases} \quad (13.5)$$

Solusi dari sistem persamaan differensial 13.5 adalah

$$\begin{aligned} x(t) &= 4e^{4t} + 2e^{-t} \\ y(t) &= 6e^{4t} - 2e^{-t} \end{aligned}$$

maka

$$\begin{aligned} f(t, x, y) &= x(t) + 2y(t) = 4e^{4t} + 2e^{-t} + 2(6e^{4t} - 2e^{-t}) = 16e^{4t} - 2e^{-t} = x'(t) \\ g(t, x, y) &= 3x(t) + 2y(t) = 3(4e^{4t} + 2e^{-t}) + 2(6e^{4t} - 2e^{-t}) = 24e^{4t} + 2e^{-t} = y'(t) \end{aligned}$$

### Solusi Numerik

Solusi numerik dari sistem persamaan differensial 13.4 pada interval  $t \in [a, b]$  dapat dicari menggunakan metode-metode yang digunakan untuk menyelesaikan persamaan differensial biasa. Metode yang paling mudah untuk diformulasikan adalah menggunakan metode Euler. Diawali dengan membagi interval  $[a, b]$  menjadi sebanyak  $M$  sub-interval dengan lebar  $h = (b - a)/M$ . Dengan metode Euler, solusi numerik sistem persamaan differensial 13.4 dapat dicari dengan formula rekursif

$$\begin{aligned} t_{k+1} &= t_k + h \\ x_{k+1} &= x_k + h f(t_k, x_k, y_k) \\ y_{k+1} &= y_k + h g(t_k, x_k, y_k) \end{aligned} \quad (13.6)$$

**Contoh 13.2 :** Gunakan metode Euler pada Persamaan 13.6 untuk menyelesaikan sistem persamaan differensial 13.5 pada  $[0, 0.2]$  dengan ukuran langkah  $h = 0.02$ .

**Solusi :** Diketahui kondisi awal sistem adalah  $t_0 = 0$ ,  $x_0 = 6$ , dan  $y_0 = 4$ . Berdasarkan Persamaan 13.6, didapatkan nilai

$$\begin{aligned} t_1 &= t_0 + h = 0 + 0.02 = 0.02 \\ x_1 &= x_0 + h f(t_0, x_0, y_0) = 6 + 0.02 f(0, 6, 4) = 6.28 \\ y_1 &= y_0 + h g(t_0, x_0, y_0) = 4 + 0.02 g(0, 6, 4) = 4.52 \end{aligned}$$

Dengan mengulangi langkah di atas, akan diperoleh nilai  $t_k$ ,  $x_k$ , dan  $y_k$  untuk  $k = 2, \dots, 10$  seperti pada Tabel 13.3.  $\triangle$

**Tabel 13.3:** Solusi sistem persamaan differensial 13.5 menggunakan metode Euler dan Runge-Kutta ordo 4.

k	t <sub>k</sub>	Metode Euler		Metode RK-4	
		x <sub>k</sub>	y <sub>k</sub>	x <sub>k</sub>	y <sub>k</sub>
0	0.00	6.00000	4.00000	6.00000	4.00000
1	0.02	6.28000	4.52000	6.29355	4.53932
2	0.04	6.58640	5.07760	6.61562	5.11949
3	0.06	6.92123	5.67589	6.96853	5.74397
4	0.08	7.28669	6.31820	7.35474	6.41653
5	0.10	7.68515	7.00813	7.77697	7.14127
6	0.12	8.11918	7.74956	8.23814	7.92260
7	0.14	8.59155	8.54669	8.74141	8.76532
8	0.16	9.10525	9.40406	9.29021	9.67460
9	0.18	9.66351	10.32653	9.88827	10.65606
10	0.20	10.26985	11.31940	10.53962	11.71578

Telah diketahui bahwa metode Euler memiliki galat yang besar untuk menyelesaikan persamaan differensial. Metode dengan ordo yang lebih tinggi dapat pula digunakan untuk menyelesaikan sistem persamaan differensial, seperti metode Runge-Kutta ordo 4. Bentuk umum metode Runge-Kutta ordo 4 untuk menyelesaikan sistem persamaan differensial adalah

$$\begin{aligned} t_{k+1} &= t_k + h \\ x_{k+1} &= x_k + \frac{h}{6} (f_1 + 2f_2 + 2f_3 + f_4) \\ y_{k+1} &= y_k + \frac{h}{6} (g_1 + 2g_2 + 2g_3 + g_4) \end{aligned} \quad (13.7)$$

dengan

$$\begin{aligned} f_1 &= f(t_k, x_k, y_k) & g_1 &= g(t_k, x_k, y_k) \\ f_2 &= f\left(t_k + \frac{h}{2}, x_k + \frac{h}{2}f_1, y_k + \frac{h}{2}g_1\right) & g_2 &= g\left(t_k + \frac{h}{2}, x_k + \frac{h}{2}f_1, y_k + \frac{h}{2}g_1\right) \\ f_3 &= f\left(t_k + \frac{h}{2}, x_k + \frac{h}{2}f_2, y_k + \frac{h}{2}g_2\right) & g_3 &= g\left(t_k + \frac{h}{2}, x_k + \frac{h}{2}f_2, y_k + \frac{h}{2}g_2\right) \\ f_4 &= f(t_k + h, x_k + hf_3, y_k + hg_3) & g_4 &= g(t_k + h, x_k + hf_3, y_k + hg_3) \end{aligned}$$

**Contoh 13.3 :** Gunakan metode Runge-Kutta ordo 4 pada Persamaan 13.7 untuk menyelesaikan sistem persamaan differensial 13.5 pada  $[0, 0.2]$  dengan ukuran langkah  $h = 0.02$ .

**Solusi :** Diketahui kondisi awal sistem adalah  $t_0 = 0$ ,  $x_0 = 6$ , dan  $y_0 = 4$ . Berdasarkan Persamaan 13.7, untuk  $k = 0$  didapatkan nilai

$$\begin{aligned} f_1 &= f(0.00, 6.0000, 4.0000) = 14 & g_1 &= g(0.00, 6.0000, 4.0000) = 26 \\ f_2 &= f(0.01, 6.1400, 4.2600) = 14.66 & g_2 &= g(0.01, 6.1400, 4.2600) = 26.94 \\ f_3 &= f(0.01, 6.1466, 4.2694) = 14.6854 & g_3 &= g(0.01, 6.1466, 4.2694) = 26.9786 \\ f_4 &= f(0.02, 6.2937, 4.5396) = 15.372852 & g_4 &= g(0.02, 6.2937, 4.5396) = 27.960268 \end{aligned}$$

Dengan demikian, diperoleh solusi  $t_1$ ,  $x_1$ , dan  $y_1$ , yaitu

$$t_1 = t_0 + h = 0.02$$

$$x_1 = x_0 + \frac{0.02}{6} (14 + 2(14.66) + 2(14.6854) + 15.372852) = 6.29354551$$

$$y_1 = y_0 + \frac{0.02}{6} (26 + 2(26.94) + 2(26.9786) + 27.960268) = 4.53932490$$

Dengan mengulangi langkah di atas, akan diperoleh nilai  $t_k$ ,  $x_k$ , dan  $y_k$  untuk  $k = 2, \dots, 10$  seperti pada Tabel 13.3.  $\triangle$

### Persamaan Diferensial Ordo Tinggi

Suatu persamaan diferensial dengan ordo tinggi akan mengandung turunan tingkat tinggi, seperti  $x''(t)$ ,  $x'''(t)$ , dan seterusnya. Sebagai contoh, persamaan

$$mx''(t) + cx'(t) + kx(t) = g(t)$$

merepresentasikan suatu sistem mekanik dari suatu pegas dengan  $k$  konstanta pegas dan  $m$  massa benda yang dikaitkan pada pegas.

Dengan menyelesaikan turunan kedua, suatu masalah nilai awal dengan ordo 2 dapat dituliskan dalam bentuk

$$x''(t) = f(t, x(t), x'(t)) \quad \text{dengan } x(t_0) = x_0 \text{ dan } x'(t_0) = y_0 \quad (13.8)$$

Masalah nilai awal tersebut dapat diselesaikan secara numerik dengan cara mengubah bentuk persamaan diferensial ordo 2 menjadi sistem persamaan diferensial ordo 1. Diawali dengan memisalkan  $y(t) = x'(t)$ , sehingga persamaan diferensial 13.8 dapat dituliskan menjadi

$$\begin{aligned} x' &= y \\ y' &= f(t, x, y) \end{aligned} \quad \text{dengan} \quad \begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \end{cases} \quad (13.9)$$

Selanjutnya, solusi numerik  $x_k$  dan  $y_k$  dapat dicari menggunakan metode Runge-Kutta ordo 4 pada Persamaan 13.7.

#### Contoh 13.4 : Selesaikan masalah nilai awal ordo 2

$$x''(t) + 4x'(t) + 5x(t) = 0 \quad \text{dengan } x(0) = 3 \text{ dan } x'(0) = -5$$

pada interval  $[0, 5]$  dengan ukuran langkah  $h = 0.1$ . Bandingkan solusi yang diperoleh, jika solusi eksak persamaan diferensial tersebut adalah  $x(t) = 3e^{-2t} \cos(t) + e^{-2t} \sin(t)$ .

**Solusi :** Masalah nilai awal memiliki bentuk persamaan diferensial

$$x''(t) = -4x'(t) - 5x(t)$$

Misalkan  $y = x'$ , maka masalah nilai awal dapat dituliskan menjadi sistem persamaan diferensial seperti berikut.

$$\begin{aligned} x' &= y \\ y' &= -5x - 4y \end{aligned} \quad \text{dengan} \quad \begin{cases} x(0) = 3 \\ y(0) = -5 \end{cases}$$

Dengan metode Runge-Kutta, didapatkan solusi  $x_k$  dari sistem persamaan tersebut seperti pada Tabel 13.4 berikut.  $\triangle$

**Tabel 13.4:** Solusi masalah nilai awal  $x''(t) + 4x'(t) + 5x(t) = 0$  dengan  $x(0) = 3$  dan  $x'(0) = -5$ .

$k$	$t_k$	sol. hamp., $x_k$	sol. eksak, $x(t_k)$
0	0.0	3.0000000	3.0000000
1	0.1	2.5256458	2.5256582
2	0.2	2.1040278	2.1040469
3	0.3	1.7350627	1.7350843
4	0.4	1.4165337	1.4165551
5	0.5	1.1448851	1.1449045
10	1.0	0.3332430	0.3332466
20	2.0	-0.0062068	-0.0062116
30	3.0	-0.0070108	-0.0070120
40	4.0	-0.0009116	-0.0009117
48	4.8	-0.0000497	-0.0000497
49	4.9	-0.0000235	-0.0000235
50	5.0	-0.0000049	-0.0000049

### 13.3 Masalah Nilai Batas

Masalah nilai batas merupakan bentuk lain dari persamaan differensial. Masalah nilai batas memiliki bentuk umum

$$x'' = f(t, x, y) \quad \text{untuk } a \leq t \leq b \quad (13.10)$$

dengan kondisi batas  $x(a) = \alpha$  dan  $x(b) = \beta$ . Sebelum menerapkan metode-metode untuk menyelesaikan masalah nilai batas, eksistensi solusi masalah nilai batas harus diperiksa. Eksistensi solusi masalah nilai batas dapat diperiksa menggunakan Teorema 13.1 berikut.

**Teorema 13.1 Masalah Nilai Batas:** Asumsikan bahwa  $f(t, x, y)$  kontinu pada daerah  $R = \{(t, x, y) : t \in [a, b], x, y \in \mathbb{R}\}$  serta  $\partial f / \partial x = f_x(t, x, y)$  dan  $\partial f / \partial y = f_y(t, x, y)$  kontinu di  $\mathbb{R}$ . Jika terdapat konstanta  $M > 0$  untuk  $f_x$  dan  $f_y$  yang memenuhi

$$\begin{aligned} f_x(t, x, y) &> 0 \\ |f_y(t, x, y)| &\leq M \end{aligned} \quad \text{untuk setiap } (t, x, y) \in \mathbb{R}$$

maka masalah nilai batas

$$x'' = f(t, x, y) \quad \text{dengan } x(a) = \alpha \text{ dan } x(b) = \beta$$

memiliki solusi tunggal  $x = x(t)$  untuk  $a \leq t \leq b$ . ■

Notasi  $y = x'(t)$  digunakan untuk menuliskan variabel ketiga pada fungsi  $f(t, x, x')$ . Pada bab ini hanya akan dibahas masalah nilai batas dengan persamaan differensial linear.

### Masalah Nilai Batas Linear

Asumsikan bahwa  $f$  pada Teorema 13.1 memiliki bentuk  $f(t, x, y) = p(t)y + q(t)x + r(t)$  serta  $f$  dan turunan parsial  $\partial f / \partial x = q(t)$  dan  $\partial f / \partial y = p(t)$  kontinu pada  $\mathbb{R}$ . Jika terdapat konstanta  $M = \max_{a \leq t \leq b} \{|p(t)|\} > 0$  untuk  $p(t)$  dan  $q(t)$  yang memenuhi

$$\begin{aligned} q(t) &> 0 \quad \text{untuk semua } t \in [a, b] \\ |p(t)| &\leq M = \max_{a \leq t \leq b} \{|p(t)|\} \end{aligned}$$

maka masalah nilai batas linear

$$x'' = p(t)x'(t) + q(t)x(t) + r(t) \quad \text{dengan } x(a) = \alpha \text{ dan } x(b) = \beta \quad (13.11)$$

memiliki solusi tunggal  $x = x(t)$  untuk  $a \leq t \leq b$ .

#### 13.3.1 Metode *Linear Shooting*.

Metode *linear shooting* akan mereduksi masalah nilai batas menjadi 2 masalah nilai awal. Misalkan bahwa  $u(t)$  merupakan solusi tunggal dari masalah nilai awal

$$u'' = p(t)u'(t) + q(t)u(t) + r(t) \quad \text{dengan } u(a) = \alpha \text{ dan } u'(a) = 0$$

Misalkan pula bahwa  $v(t)$  merupakan solusi tunggal dari masalah nilai awal

$$v'' = p(t)v'(t) + q(t)v(t) \quad \text{dengan } v(a) = 0 \text{ dan } v'(a) = 1$$

Dengan demikian, kombinasi linear

$$x(t) = u(t) + Cv(t) \quad (13.12)$$

merupakan solusi dari  $x'' = p(t)x'(t) + q(t)x(t) + r(t)$  sebagaimana seperti yang dapat dilihat pada komputasi berikut.

$$\begin{aligned} x'' &= u'' + Cv'' = p(t)u'(t) + q(t)u(t) + r(t) + C(p(t)v'(t) + q(t)v(t)) \\ &= p(t)(u'(t) + Cv'(t)) + q(t)(u(t) + Cv(t)) + r(t) \\ &= p(t)x'(t) + q(t)x(t) + r(t) \end{aligned}$$

Solusi  $x(t)$  dapat diperoleh melalui nilai batas

$$\begin{aligned} x(a) &= u(a) + Cv(a) = \alpha + 0 = \alpha \\ x(b) &= u(b) + Cv(b) \end{aligned}$$

Dengan kondisi batas  $x(b) = \beta$ , didapatkan  $C = (\beta - u(b))/v(b)$ . Jadi, jika  $v(b) \neq 0$ , maka solusi masalah nilai batas pada Persamaan 13.11 adalah

$$x(t) = u(t) + \frac{\beta - u(b)}{v(b)}v(t) \quad (13.13)$$

**Contoh 13.5 :** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1$$

dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  pada interval  $[0, 4]$ .

1. Tunjukkan hasil reduksi masalah nilai batas menjadi 2 masalah nilai awal  $u(t)$  dan  $v(t)$ , kemudian selesaikan masing-masing masalah nilai awal menggunakan metode Runge-Kutta orde 4 dengan  $h = 0.2$ .
2. Hitung solusi masalah nilai batas  $x(t)$  menggunakan Persamaan 13.20.

**Solusi :**

1. Hasil reduksi masalah nilai batas adalah

$$\begin{aligned} u''(t) &= \frac{2t}{1+t^2}u'(t) - \frac{2}{1+t^2}u(t) + 1 && \text{dengan } \begin{cases} u(0) = 1.25 \text{ dan } u'(0) = 0 \\ v(0) = 0 \text{ dan } v'(0) = 1 \end{cases} \\ v''(t) &= \frac{2t}{1+t^2}v'(t) - \frac{2}{1+t^2}v(t) \end{aligned}$$

Untuk menyelesaikan masing-masing masalah nilai awal tersebut, kedua persamaan akan ditulis ulang menjadi sistem persamaan differensial ordo 1, yaitu

$$\begin{aligned} u'(t) &= w(t) \\ w'(t) &= \frac{2t}{1+t^2}w(t) - \frac{2}{1+t^2}u(t) + 1 && \text{dengan } \begin{cases} u(0) = 1.25 \\ w(0) = 0 \end{cases} \end{aligned}$$

dan

$$\begin{aligned} v'(t) &= y(t) \\ y'(t) &= \frac{2t}{1+t^2}y(t) - \frac{2}{1+t^2}v(t) && \text{dengan } \begin{cases} v(0) = 0 \\ y(0) = 1 \end{cases} \end{aligned}$$

Dengan metode Runge-Kutta ordo 4, akan didapatkan solusi  $u(t)$  dan  $v(t)$  dari masing-masing sistem seperti pada Tabel 13.5.

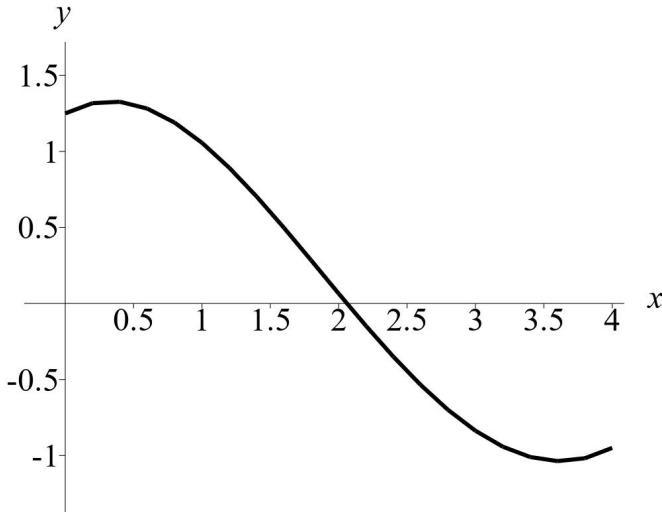
2. Seletah didapatkan nilai solusi  $u(t)$  dan  $v(t)$ , solusi  $x(t)$  dapat dicari menggunakan Persamaan 13.20, yaitu

$$x(t) = u(t) + \frac{-0.95 - u(4)}{v(4)}v(t) \approx u(t) + 0.48588v(t)$$

Solusi masalah nilai batas  $x(t)$  dapat dilihat pada Tabel 13.5 dan Gambar 13.2.  $\triangle$

**Tabel 13.5:** Solusi masalah nilai batas  $x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1$ .

$t_k$	$u_k$	$v_k$	$x_k$
0.0	1.2500000	0.00	1.2500000
0.2	1.2201314	0.20	1.3173081
0.4	1.1320726	0.40	1.3264261
0.6	0.9901221	0.60	1.2816523
0.8	0.8005687	0.80	1.1892756
1.0	0.5708440	1.00	1.0567277
2.8	-2.0609039	2.80	-0.7004296
3.0	-2.2949157	3.00	-0.8372646
3.2	-2.4968416	3.20	-0.9420138
3.4	-2.6620042	3.40	-1.0099997
3.6	-2.7859602	3.60	-1.0367789
3.8	-2.8644811	3.80	-1.0181230
4.0	-2.8935347	4.00	-0.9500000



**Gambar 13.2:** Solusi masalah nilai batas  $x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1$ .

### 13.3.2 Metode *Finite-Difference*

Metode lain yang dapat digunakan untuk menyelesaikan masalah nilai batas adalah menggunakan formula beda-pusat. Misalkan, persamaan differensial linear ordo 2

$$x'' = p(t)x'(t) + q(t)x(t) + r(t) \quad (13.14)$$

pada  $[a, b]$  dengan  $x(a) = \alpha$  dan  $x(b) = \beta$ . Partisikan interval  $[a, b]$  menjadi  $N$  sub-interval menggunakan titik  $a = t_0 < t_1 < \dots < t_N = b$  dengan  $h = (b - a)/N$  dan  $t_j = a + jh$  untuk  $j = 0, 1, 2, \dots, N$ . Dengan formula beda-pusat, nilai turunan pertama dan kedua dari  $x(t_j)$  dapat diperoleh, yaitu

$$x'(t_j) = \frac{x(t_{j+1}) - x(t_{j-1})}{2h}$$

dan

$$x''(t_j) = \frac{x(t_{j+1}) - 2x(t_j) + x(t_{j-1})}{h^2}$$

Substitusikan bentuk formula  $x'(t_j)$  dan  $x''(t_j)$  di atas ke Persamaan 13.14, sehingga akan diperoleh persamaan

$$\frac{x_{j+1} - 2x_j + x_{j-1}}{h^2} = p_j \frac{x_{j+1} - x_{j-1}}{2h} + q_j x_j + r_j$$

dengan  $x_j = x(t_j)$ ,  $p_j = p(t_j)$ ,  $q_j = q(t_j)$ , dan  $r_j = r(t_j)$ . Kalikan kedua ruas dengan  $h^2$  dan kumpulkan persamaan yang memiliki  $x_{j+1}$ ,  $x_j$ , dan  $x_{j-1}$ , sehingga diperoleh bentuk persamaan seperti berikut.

$$\left(\frac{-h}{2}p_j - 1\right)x_{j-1} + (2 + h^2q_j)x_j + \left(\frac{h}{2}p_j - 1\right)x_{j+1} = -h^2r_j \quad (13.15)$$

untuk  $j = 1, 2, \dots, N - 1$  dengan  $x_0 = \alpha$  dan  $x_N = \beta$ . Dalam matriks, persamaan 13.15 dapat dituliskan seperti berikut.

$$\begin{bmatrix} 2 + h^2 q_1 & \frac{h}{2} p_1 - 1 & 0 & 0 & 0 \\ \frac{-h}{2} p_2 - 1 & 2 + h^2 q_2 & \frac{h}{2} p_2 - 1 & 0 & 0 \\ 0 & \frac{-h}{2} p_j - 1 & 2 + h^2 q_j & \frac{h}{2} p_j - 1 & 0 \\ 0 & 0 & \frac{-h}{2} p_{N-2} - 1 & 2 + h^2 q_{N-2} & \frac{h}{2} p_{N-2} - 1 \\ 0 & 0 & 0 & \frac{-h}{2} p_{N-1} - 1 & 2 + h^2 q_{N-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_j \\ x_{j-2} \\ x_{j-1} \end{bmatrix} = \begin{bmatrix} -h^2 r_1 + e_0 \\ -h^2 r_2 \\ -h^2 r_j \\ -h^2 r_{j-2} \\ -h^2 r_{j-1} + e_N \end{bmatrix}$$

dengan

$$e_0 = \left( \frac{h}{2} p_1 + 1 \right) \alpha \quad \text{dengan} \quad e_N = \left( \frac{-h}{2} p_{N-1} + 1 \right) \beta$$

**Contoh 13.6 :** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = \frac{2t}{1+t^2} x'(t) - \frac{2}{1+t^2} x(t) + 1$$

dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  pada interval  $[0, 4]$ .

1. Gunakan metode *finite-difference* untuk menyelesaikan masalah nilai batas tersebut dengan  $h = 1$ .
2. Gunakan metode *finite-difference* untuk menyelesaikan masalah nilai batas tersebut dengan  $h = 0.2$ .

**Solusi :** Diketahui bahwa fungsi  $p_j$ ,  $q_j$ , dan  $r_j$  adalah

$$p_j = \frac{2t_j}{1+t_j^2} \quad q_j = -\frac{2}{1+t_j^2} \quad r_j = 1$$

Karena  $h = 1$ , banyak sub-interval yang akan digunakan adalah  $N = (b - a)/h = 4$ . Bagi interval  $[0, 4]$  menjadi 4 sub-interval menggunakan titik  $t_j = j$  dengan  $j = 0, 1, \dots, 4$ . Berdasarkan Persamaan 13.15, maka didapatkan sistem persamaan

$$\begin{aligned} -1.5x_0 + 1x_1 - 0.5x_2 &= -1 \\ -1.4x_1 + 1.6x_2 - 0.6x_3 &= -1 \\ -1.3x_2 + 1.8x_3 - 0.7x_4 &= -1 \end{aligned}$$

Karena nilai  $x_0 = 1.25$  dan  $x_4 = -0.95$ , sistem persamaan di atas dapat dituliskan menjadi

$$\begin{bmatrix} 1 & -0.5 & 0 \\ -1.4 & 1.6 & -0.6 \\ 0 & -1.3 & 1.8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0.875 \\ -1 \\ -1.665 \end{bmatrix}$$

Dengan menyelesaikan SPL tersebut, solusi masalah nilai batas yang dihasilkan adalah  $x_1 = 0.52143$ ,  $x_2 = -0.70714$  dan  $x_3 = -1.43571$ .

Dengan cara yang sama, hasil solusi numerik masalah nilai batas dengan  $h = 0.2$  dapat dilihat pada Tabel 13.6 berikut. Solusi masalah nilai batas yang dihasilkan oleh metode *finite-difference* sama dengan solusi yang dihasilkan oleh metode *linear shooting*.  $\triangle$

**Tabel 13.6:** Solusi masalah nilai batas dengan  $h = 0.2$  menggunakan metode *finite-difference*.

$t_j$	$x_j$
0.0	1.2500000
0.2	1.3145034
0.4	1.3206069
0.6	1.2727548
0.8	1.1773987
1.0	1.0421063
2.0	0.0423983
3.0	-0.8549878
3.2	-0.9572499
3.4	-1.0222209
3.6	-1.0454566
3.8	-1.0227272
4.0	-0.9500000

## 13.4 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Runge-Kutta-Fehlberg 4/5 (RKF45)** berisi program untuk mencari solusi persamaan diferensial  $y' = f(t, y)$  dengan  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 6 masukan, yaitu fungsi  $f(t, y)$ , titik ujung interval penyelesaian  $[a, b]$ , nilai awal  $y_0$ , jumlah sub-interval  $M$  dan nilai toleransi *error*  $\delta$ , serta 1 luaran, yaitu solusi masalah nilai awal *sol*. Berikut merupakan program untuk metode Runge-Kutta-Fehlberg (RKF45).

### Algoritma 13.1: Metode Runge-Kutta-Fehlberg (RKF45)

```
function rkf45(f, a, b, y0, M, delta)
M = round(M)
a2=1/4;b2=1/4;a3=3/8;b3=3/32;c3=9/32;
a4=12/13;b4=1932/2197;c4=-7200/2197;d4=7296/2197;
a5=1;b5=439/216;c5=-8;d5=3680/513;e5=-845/4104;
a6=1/2;b6=-8/27;c6=2;d6=-3544/2565;e6=1859/4104;f6=-11/40;
r1=1/360;r3=-128/4275;r4=-2197/75240;r5=1/50;r6=2/55;
n1=25/216;n3=1408/2565;n4=2197/4104;n5=-1/5;
big=1e15;
h=(b-a)/M;
hmin=h/64;
hmax=h*64;
maxi=200;
j=1;
Y = y0;
T = a;
br= b-0.001*abs(b);
```

```

err=NaN
while T[j]<b
    if (T[j]+h)>br;h=b-T[j];end
    #% Hitung koefisien
    k1=h*f(T[j],Y[j]);
    y2=Y[j]+b2*k1;
    k2=h*f(T[j]+a2*h,y2);
    y3=Y[j]+b3*k1+c3*k2;
    k3=h*f(T[j]+a3*h,y3);
    y4=Y[j]+b4*k1+c4*k2+d4*k3;
    k4=h*f(T[j]+a4*h,y4);
    y5=Y[j]+b5*k1+c5*k2+d5*k3+e5*k4;
    k5=h*f(T[j]+a5*h,y5);
    y6=Y[j]+b6*k1+c6*k2+d6*k3+e6*k4+f6*k5;
    k6=h*f(T[j]+a6*h,y6);
    err=abs(r1*k1+r3*k3+r4*k4+r5*k5+r6*k6);
    ynew=Y[j]+n1*k1+n3*k3+n4*k4+n5*k5;
    #% Perbarui ukuran langkah
    if (err<delta) || (h<2*hmin)
        Y = [Y; ynew];
        if (T[j]+h)>br
            T = [T; b];
        else
            T = [T; T[j]+h];
        end
        j=j+1;
    end
    if (err==0)
        s=0;
    else
        s=0.84*(delta*h/err)^0.25;
    end
    if (s<0.75) && (h>2*hmin); h = h/2;end
    if (s>1.50) && (2*h<hmax); h = 2*h;end
    if abs(Y[j])>big || maxi==j;break;end
end
sol = [T Y];
return sol
end

```

**Metode Runge-Kutta untuk Sistem Persamaan Differensial** berisi program untuk mencari solusi sistem persamaan differensial  $x' = f(t, z)$  dan  $y' = g(t, z)$  dengan masalah nilai awal  $x(a) = x_0$  dan  $y(a) = y_0$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi yang berisi  $f(t, z)$  dan  $g(t, z)$ , titik ujung interval  $[a, b]$ , nilai awal  $z_0$  yang berisi  $[x_0, y_0]$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai awal  $sol$ . Berikut merupakan program untuk metode Runge-Kutta untuk sistem persamaan differensial.

**Algoritma 13.2:** Metode Runge-Kutta untuk sistem persamaan differensial

```

function rungekuttasistem(f,a,b,y0,M)
M = Int(M)
h = (b-a)/M;
T = a:h:b;
Y = Array{Float64}(undef,M+1,length(y0))
Y[1,:] = y0;

```

```

for k = 1:M
    f1 = f(T[k] ,Y[k,:]);
    f2 = f(T[k]+h/2 ,Y[k,:]+f1*h/2 );
    f3 = f(T[k]+h/2 ,Y[k,:]+f2*h/2 );
    f4 = f(T[k]+h ,Y[k,:]+f3*h );
    Y[k+1,:] = Y[k,:]+ h/6*(f1+2*f2+2*f3+f4);
end
sol = [T Y];
end

```

**Metode Linear Shooting** berisi program untuk mencari solusi persamaan differensial  $x'' = f(t, x, x')$  dengan masalah nilai batas  $x(a) = \alpha$  dan  $x(b) = \beta$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $F1$  dan  $F2$  (sistem persamaan differensial hasil transformasi dari  $x''$ ), titik ujung interval  $[a, b]$ , nilai batas  $[\alpha, \beta]$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai batas  $sol$ . Berikut merupakan program untuk metode *linear shooting*.

#### Algoritma 13.3: Metode *linear shooting*

```

function linearshooting(F1,F2,a,b,alpha,beta,M)
M = Int(M)
Za = [alpha,0];
sol = rungekuttasistem(F1,a,b,Za,M);
U = sol[:,2];

Za = [0,1];
sol = rungekuttasistem(F2,a,b,Za,M);
V = sol[:,2];

T = sol[:,1];
X = U + (beta-U[M+1])*V/V[M+1];
solusi = [T X];
end

```

**Metode Finite-Difference** berisi program untuk mencari solusi persamaan differensial  $x'' = f(t, x, x')$  dengan masalah nilai batas  $x(a) = \alpha$  dan  $x(b) = \beta$  pada interval  $a \leq t \leq b$ . Program ini secara *default* berisi 5 masukan, yaitu fungsi  $p(t)$ ,  $q(t)$  dan  $r(t)$  dari persamaan  $x'' = p(t)x' + q(t)x + r(t)$ , titik ujung interval  $[a, b]$ , nilai batas  $[\alpha, \beta]$ , dan jumlah sub-interval  $M$ , serta 1 luaran, yaitu solusi masalah nilai batas  $sol$ . Berikut merupakan program untuk metode *finite-difference*.

#### Algoritma 13.4: Metode *finite-difference*

```

using LinearAlgebra
function findiff(p,q,r,a,b,alpha,beta,M)
h = (b-a)/M;
T = a:h:b;
T = T[2:end-1];
<% Bangun Matriks B
B = -h^2*r.(T);
B[1] = B[1] + (1+h/2*p(T[1]))*alpha;
B[end] = B[end] + (1-h/2*p(T[end]))*beta;
<% Bangun Matriks A - Bagian Diagonal
Ad = 2 .+h^2*q.(T);
<% Bangun Matriks A - Bagian Bawah Diagonal

```

```

Tbawah = T[2:end];
Abawah = -1 .-h/2*p.(Tbawah);
<% Bangun Matriks A - Bagian Atas Diagonal
Tatas = T[1:end-1];
Aatas = -1 .+h/2*p.(Tatas);
A = Tridiagonal(ABawah, Ad, Aatas)
<% Selesaikan AX=B
X = A\B;
T = [a; T; b];
X = [alpha; X ;beta];
solusi = [T X];
end

```

### 13.4.1 Metode RKF45

RKF45 merupakan modifikasi metode RK4 untuk memperoleh metode dengan ukuran langkah  $h$  yang adaptif, artinya ukuran langkah akan menyesuaikan bentuk fungsi ( $h$  kecil jika fungsi curam,  $h$  besar jika fungsi landai). Solusi pada setiap iterasinya dihitung sebanyak dua kali, masing-masing menggunakan metode berorde 4 dan metode berorde 5. Selisih kedua solusi pada setiap iterasinya digunakan untuk menentukan ukuran langkah  $h$  pada iterasi berikutnya. Ukuran langkah  $h$  dapat membesar atau mengecil sesuai kebutuhan.

**Contoh 13.7 :** Diberikan persamaan differensial biasa

$$y' = 1 + y^2$$

dengan masalah nilai awal  $y(0) = 0$ . Berikut merupakan solusi numerik dari PDB tersebut menggunakan metode RKF45 pada  $t \in [0, 1.4]$  dengan toleransi kesalahan  $2 \times 10^{-5}$  dan ukuran langkah awal  $h_0 = 0.2$ . Serta, gambar dari pembagian ukuran langkah  $h$  yang berbeda.

**Langkah 1:** Selesaikan masalah nilai awal menggunakan Program 13.1.

```

Inp: f(t,y) = 1+y^2
      a = 0
      b = 1.4
      y0 = 0
      h0 = 0.2
      M = (b-a)/h0
      delta = 2*10^-5
      sol = rkf45(f,a,b,y0,M,delta)

```

**Langkah 2:** Gambarkan solusi yang diperoleh.

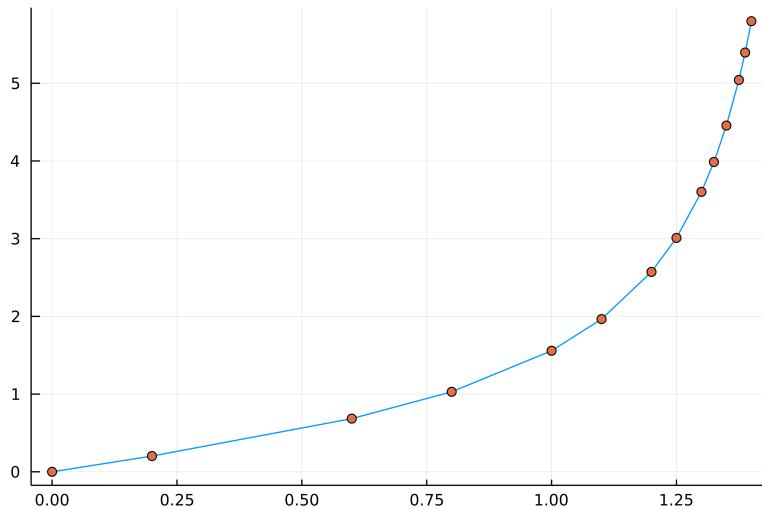
```

Inp: using Plots
      t = sol[:,1];
      y = sol[:,2];
      plot(t,y,legend = :false)
      scatter!(t,y)

```

Solusi numerik masalah nilai awal pada Contoh 13.7 menggunakan metode Runge-Kutta-Fehlberg (RKF45) divisualisasikan pada Gambar 13.3 berikut. Gamber tersebut menunjukkan ukuran langkah  $h$  yang berbeda untuk membentuk sub-interval dalam menyelesaikan masalah nilai awal. Semakin landai fungsi yang dibentuk, maka semakin besar

ukuran langkah yang digunakan. Sebaliknya, semakin curam fungsi yang dibentuk, maka semakin kecil ukuran langkah yang digunakan.



**Gambar 13.3:** Solusi numerik masalah nilai awal pada Contoh 13.7 menggunakan RKF45.

**Soal Latihan:** Diberikan persamaan differensial

$$\frac{dy}{dt} = -y - 2t - 1, \text{ dengan } y(0) = 2$$

Gunakan metode Runge-Kutta-Fehlberg 4/5 (RKF45) untuk menyelesaikan PD di atas dengan nilai toleransi  $\delta = 10^{-6}, 10^{-12}$  dan  $10^{-16}$ , kemudian bandingkan dengan solusi eksaknya yaitu  $y(t) = e^{-t} - 2t + 1$  dan gambarkan solusi beserta titik-titik pembagian ukuran langkah seperti pada Contoh 13.7.

### 13.4.2 Solusi Sistem Persamaan Diferensial

Sistem persamaan differensial merupakan masalah yang sering ditemui dalam pemodelan matematika. Sistem persamaan differensial biasanya digambarkan dalam suatu bidang fase dan bidang solusi yang dapat diselesaikan secara numerik menggunakan metode Runge-Kutta orde-4 seperti berikut.

**Contoh 13.8 :** Diberikan sistem persamaan differensial seperti berikut.

$$\begin{aligned} \dot{x} &= x + 2y & \text{dengan } x(0) = 6 \\ \dot{y} &= 3x + 2y & y(0) = 4 \end{aligned} \tag{13.16}$$

Berikut merupakan langkah-langkah untuk menggambarkan plot bidang fase dan bidang solusi dari sistem persamaan differensial di atas pada interval  $t \in [0, 0.2]$  dengan ukuran langkah  $h = 0.02$ .

**Langkah 1:** Penghitungan solusi penyelesaian sistem persamaan differensial di atas menggunakan metode Runge-Kutta sistem orde-4 pada Program 13.2.

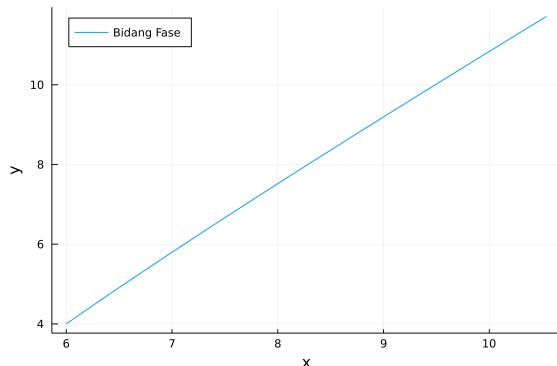
```
Inp: f(t,z) = [ z[1]+2*z[2] , 3*z[1]+2*z[2] ]
      a = 0
      b = 0.2
      y0 = [6, 4]
      h = 0.02
      M = (b-a)/h
      sol = rungekuttasistem(f,a,b,y0,M)
```

**Langkah 2:** Pembuatan plot bidang fase dari solusi yang dihasilkan, yaitu plot solusi  $y$  terhadap  $x$ .

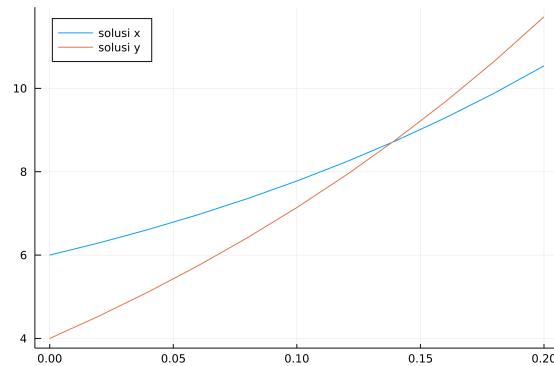
```
Inp: t = sol[:,1];
      x = sol[:,2];
      y = sol[:,3];
      plot(x,y,label="Bidang Fase",legend=:topleft)
```

**Langkah 3:** Pembuatan plot bidang solusi  $x$  dan  $y$  dari solusi yang dihasilkan, yaitu plot  $x$  terhadap  $t$  dan plot  $y$  terhadap  $t$ .

```
Inp: plot(t,x,label="solusi x",legend=:topleft)
      plot!(t,y,label="solusi y")
```



(a) Plot bidang fase  $y(x)$ .



(b) Plot bidang solusi  $x(t)$  dan  $y(t)$ .

**Gambar 13.4:** Plot bidang fase dan solusi dari sistem  $x' = x + 2y$  dan  $y' = 3x + 2y$  dengan  $x(0) = 6$  dan  $y(0) = 4$ .

Gambar 13.4 menunjukkan solusi numerik dari masalah sistem persamaan diferensial biasa pada Contoh 13.8 berupa plot bidang fase (Gambar 13.4a) dan bidang solusi (Gambar 13.4b) dari sistem persamaan differensial pada Persamaan 13.16.

**Soal Latihan:** Ulangi langkah-langkah pada Contoh 13.8 untuk menggambarkan plot bidang fase dan bidang solusi dari sistem persamaan berikut.

$$\begin{aligned} \dot{x} &= -2x - y && \text{dengan } x(0) = 6 \\ \dot{y} &= x - y && y(0) = 0 \end{aligned}$$

pada interval  $t \in [0, 5]$  dengan ukuran langkah  $h = 0.1$ .

### 13.4.3 Solusi Masalah Nilai Awal Orde Tinggi

Selain persamaan diferensial orde pertama, sering kali dijumpai masalah matematika dalam bentuk persamaan diferensial orde ke-2 atau lebih tinggi. Masalah persamaan diferensial orde ke-2 atau lebih tinggi dapat diselesaikan dengan cara mentransformasi persamaan tersebut kedalam bentuk sistem persamaan diferensial orde pertama, kemudian diselesaikan menggunakan metode Runge-Kutta orde-4.

**Contoh 13.9 :** Diketahui suatu persamaan diferensial orde-2 seperti berikut.

$$x''(t) + 4x'(t) + 5x(t) = 0 \quad (13.17)$$

dengan nilai awal  $x(0) = 3$  dan  $x'(0) = -5$ . Berikut merupakan penyelesaian masalah tersebut menggunakan metode Runge-Kutta pada interval  $[0, 5]$  dengan 50 sub-interval dan perbandingan solusi numerik terhadap solusi eksaknya, yaitu

$$x(t) = 3e^{-2t} \cos(t) + e^{-2t} \sin(t)$$

**Langkah 1:** Transformasi persamaan diferensial orde-2 pada Persamaan 13.17 menjadi sistem persamaan diferensial orde-1.

Misalkan,  $x'(t) = y(t)$  sehingga persamaan diferensial memiliki bentuk

$$\begin{aligned} x''(t) + 4x'(t) + 5x(t) &= 0 \\ \Leftrightarrow y'(t) + 4y(t) + 5x(t) &= 0 \\ \Leftrightarrow y'(t) &= -4y(t) - 5x(t) \end{aligned}$$

dan nilai awal  $x'(0) = -5$  menjadi  $y(0) = 5$ . Secara lengkap, sistem baru yang equivalen dengan persamaan diferensial di atas dapat dituliskan sebagai berikut.

$$\begin{aligned} x' &= y && \text{dengan } x(0) = 3 \\ y' &= -4y - 5x && y(0) = -5 \end{aligned} \quad (13.18)$$

**Langkah 2:** Penyelesaian sistem persamaan diferensial pada Persamaan 13.18 menggunakan metode Runge-Kutta.

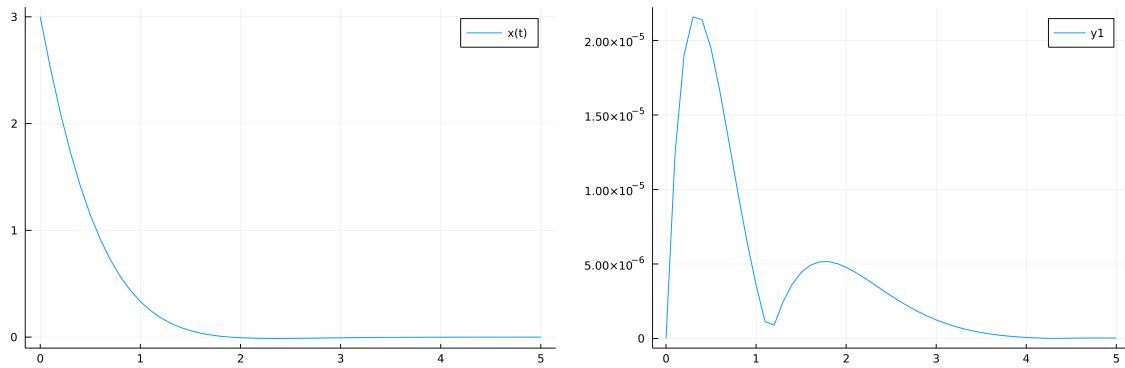
```
Inp: f(t,z) = [ z[2] , -4*z[2]-5*z[1] ]
      a = 0
      b = 5
      y0 = [3, -5]
      M = 50
      sol = rungekuttasistem(f,a,b,y0,M)
```

**Langkah 3:** Gambar plot nilai hampiran solusi dan galatnya pada setiap langkah  $t$ .

```
Inp: # Plot Solusi
      t = sol[:,1];
      x = sol[:,2];
      plot(t,x,label="x(t)")

Inp: # Hitung Galat
      xt(t) = 3*exp(-2*t)*cos(t)+exp(-2*t)*sin(t)
      galat = abs.(x .- xt.(t))

Inp: # Plot Galat
      plot(t,galat)
```

(a) Plot bidang solusi  $x$ .

(b) Plot galat pada setiap iterasi.

**Gambar 13.5:** Plot bidang solusi  $x$  dan galat pada setiap iterasi dari masalah persamaan differensial orde-2  $x''(t) + 4x'(t) + 5x(t) = 0$  dengan nilai awal  $x(0) = 3$  dan  $x'(0) = -5$ .

**Soal Latihan:** Diketahui suatu persamaan differensial orde-2 seperti berikut.

$$2x''(t) - 5x'(t) - 3x(t) = 45e^{2t}$$

dengan nilai awal  $x(0) = 2$  dan  $x'(0) = 1$ . Selesaikan masalah tersebut menggunakan Runge-Kutta pada interval  $[0, 2]$  dengan  $h = 0.05$ , kemudian bandingkan solusi numerik terhadap solusi eksaknya yaitu  $x(t) = 4e^{-t/2} + 7e^{3t} - 9e^{2t}$  dengan langkah-langkah seperti pada Contoh 13.9.

### 13.4.4 Solusi Masalah Nilai Batas

Masalah nilai batas merupakan bentuk lain dari masalah persamaan differensial biasa. Masalah nilai batas memiliki bentuk umum berupa persamaan differensial orde 2 yaitu  $x'' = f(x', x, t)$  dengan nilai batas pada selang  $[a, b]$ , yaitu  $x(a) = \alpha$  dan  $x(b) = \beta$ . Akan dipelajari dua metode untuk mencari solusi numerik masalah nilai batas, yaitu metode *linear shooting* dan *finite-difference*.

#### Metode *Linear Shooting*

Ide dasar dari metode *linear shooting* adalah melakukan transformasi masalah nilai batas menjadi dua persamaan differensial orde-2 dengan masalah nilai awal. Misalkan, diketahui masalah nilai batas  $x'' = p(t)x' + q(t)x + r(t)$  dengan nilai batas  $x(a) = \alpha$  dan  $x(b) = \beta$ . Transformasi dari masalah nilai batas tersebut adalah

$$\begin{aligned} u'' &= p(t)u' + q(t)u + r(t) & \text{dengan} & \quad u(a) = \alpha \text{ dan } u'(a) = 0 \\ v'' &= p(t)v' + q(t)v & & \quad v(a) = 0 \text{ dan } v'(a) = 1 \end{aligned} \quad (13.19)$$

Persamaan 13.19 dapat diselesaikan menggunakan metode Runge-Kutta orde-4 dengan mengubahnya menjadi bentuk sistem persamaan differensial, sehingga akan didapatkan solusi  $u(t)$  dan  $v(t)$ . Selanjutnya, solusi masalah nilai batas  $x(t)$  dapat dicari dengan Persamaan 13.20 berikut.

$$x(t) = u(t) + \frac{\beta - u(b)}{v(b)}v(t) \quad (13.20)$$

**Contoh 13.10 :** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1 \quad (13.21)$$

dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  pada interval  $[0, 4]$ . Berikut merupakan langkah-langkah untuk menyelesaikan masalah nilai batas tersebut menggunakan metode *linear shooting*.

**Langkah 1:** Transformasi masalah nilai batas menjadi masalah nilai awal orde-2.

Berdasarkan Persamaan 13.19, dua masalah nilai awal orde-2 yang setara dengan masalah nilai batas pada Persamaan 13.21 adalah

$$u'' = \frac{2t}{1+t^2}u' - \frac{2}{1+t^2}u + 1 \quad (13.22)$$

dengan  $u(0) = 1.25$  dan  $u'(0) = 0$ . Serta,

$$v'' = \frac{2t}{1+t^2}v' - \frac{2}{1+t^2}v \quad (13.23)$$

dengan  $v(0) = 0$  dan  $v'(0) = 1$ .

**Langkah 2:** Transformasi masalah nilai awal orde-2 menjadi sistem persamaan differensial orde-1.

Persamaan 13.22 dan 13.23 dapat ditulis dalam bentuk sistem persamaan linear orde pertama, yaitu

$$\begin{aligned} u' &= u_2 \\ u'_2 &= \frac{2t}{1+t^2}u_2 - \frac{2}{1+t^2}u + 1 \end{aligned} \quad (13.24)$$

dengan  $u(0) = 1.25$  dan  $u_2(0) = 0$ . Serta,

$$\begin{aligned} v' &= v_2 \\ v'_2 &= \frac{2t}{1+t^2}v_2 - \frac{2}{1+t^2}v \end{aligned} \quad (13.25)$$

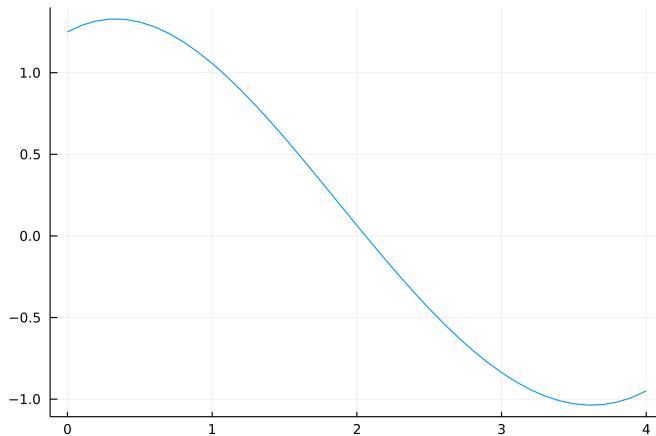
dengan  $v(0) = 0$  dan  $v_2(0) = 1$ .

Dalam fungsi octave, sistem persamaan differensial pada Persamaan 13.24 dan 13.25 dapat ditulis sebagai berikut.

```
Inp: F1(t, z) = [ z[2], 2*t/(1+t^2)*z[2]-2/(1+t^2)*z[1]+1 ]
      F2(t, z) = [ z[2], 2*t/(1+t^2)*z[2]-2/(1+t^2)*z[1] ]
```

**Langkah 3:** Penghitungan solusi masalah nilai batas pada Persamaan 13.21 menggunakan Program 13.3 serta penggambaran plot solusi penyelesaiannya.

```
Inp: a = 0
      b = 4
      alpha = 1.25
      beta = -0.95
      M = 40
      solusi = linearshooting(F1,F2,a,b,alpha,beta,M)
Inp: # Plot Solusi
      t = solusi[:,1];
      x = solusi[:,2];
      plot(t,x)
```



**Gambar 13.6:** Solusi persamaan  $x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1$  dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  menggunakan metode *linear shooting*.

**Soal Latihan:** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = -\frac{2}{t}x'(t) + \frac{2}{t^2}x(t) + \frac{\sin(t)}{t^2}$$

dengan nilai batas  $x(1) = -0.02$  dan  $x(6) = 0.02$  pada interval  $[1, 6]$ . Gunakan metode *linear shooting* untuk menyelesaikan masalah nilai batas di atas dengan langkah-langkah pada Contoh 13.10.

### Metode *Finite Difference*

Ide dasar dari metode *finite-difference* adalah mengubah persamaan differensial dengan masalah nilai batas menjadi formula beda-hingga.

**Contoh 13.11 :** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = \frac{2t}{1+t^2}x'(t) - \frac{2}{1+t^2}x(t) + 1$$

dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  pada interval  $[0, 4]$ . Berikut merupakan langkah-langkah untuk mencari solusi penyelesaian dari masalah nilai batas tersebut menggunakan metode beda-hingga (*finite-difference*) dengan ukuran langkah  $h = 0.2$ .

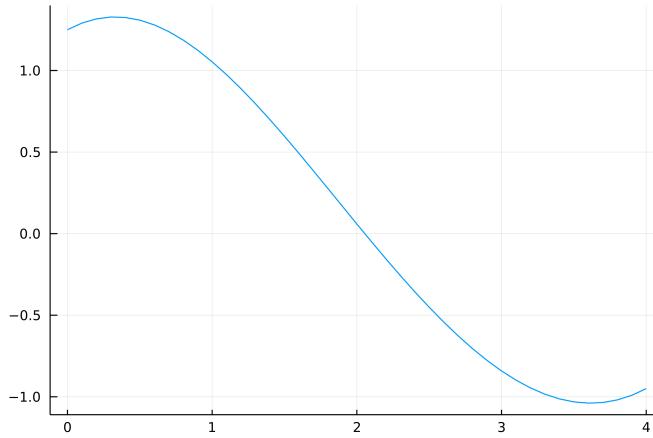
**Langkah 1:** Pendefinisian fungsi koefisien  $p(t)$ ,  $q(t)$ , dan  $r(t)$ .

```
Inp: p(t) = 2*t/(1+t^2);
     q(t) = -2/(1+t^2);
     r(t) = 1 + 0*t;
```

**Langkah 2:** Penghitungan solusi penyelesaian masalah nilai batas menggunakan metode *finite-difference* pada Program 13.4 dengan ukuran langkah  $h = 0.2$  dan penggambaran grafik solusinya.

```
Inp: a = 0
     b = 4
     alpha = 1.25
     beta = -0.95
     M = 40
     solusi = findiff(p, q, r, a, b, alpha, beta, M)
```

```
Inp: # Plot Solusi
t = solusi[:,1];
x = solusi[:,2];
plot(t,x)
```



**Gambar 13.7:** Solusi persamaan  $x''(t) = \frac{2t}{1+t^2} x'(t) - \frac{2}{1+t^2} x(t) + 1$  dengan nilai batas  $x(0) = 1.25$  dan  $x(4) = -0.95$  menggunakan metode beda-hingga.

**Soal Latihan:** Diketahui persamaan masalah nilai batas seperti berikut.

$$x''(t) = -\frac{2}{t} x'(t) + \frac{2}{t^2} x(t) + \frac{\sin(t)}{t^2}$$

dengan nilai batas  $x(1) = -0.02$  dan  $x(6) = 0.02$  pada interval  $[1, 6]$ . Gunakan metode *finite difference* untuk menyelesaikan masalah nilai batas di atas dengan langkah-langkah pada Contoh 13.11.

## 13.5 Latihan-latihan

### 13.5.1 Ulasan Materi

1. Jelaskan ide dasar metode Runge-Kutta-Fehlberg 4/5 (RKF45) dalam menyelesaikan masalah nilai awal!
2. Jelaskan keunggulan dan kelemahan metode Runge-Kutta-Fehlberg 4/5 (RKF45) dibandingkan metode penyelesaian masalah nilai awal yang lain!
3. Jelaskan tahapan-tahapan yang dilakukan dalam menyelesaikan masalah nilai awal menggunakan metode Runge-Kutta-Fehlberg 4/5 (RKF45)!
4. Tuliskan formula Runge-Kutta ordo 4 dan ordo 5 yang digunakan dalam menyelesaikan persamaan diferensial menggunakan metode Runge-Kutta-Fehlberg 4/5 (RKF45)!
5. Jelaskan fungsi/kegunaan dari metode Runge-Kutta ordo 4 dan ordo 5 dalam penyelesaian persamaan diferensial menggunakan metode RKF45!
6. Jelaskan bagaimana cara untuk menyelesaikan suatu sistem persamaan diferensial biasa dengan masalah nilai awal!

7. Jelaskan perbedaan metode Euler dalam menyelesaikan suatu sistem persamaan diferensial biasa dengan metode Euler yang digunakan untuk menyelesaikan persamaan diferensial biasa!
8. Jelaskan bagaimana cara untuk menyelesaikan persamaan diferensial dengan ordo 2, 3, dan seterusnya!
9. Apa yang dimaksud dengan masalah nilai batas? Jelaskan perbedaannya dengan masalah nilai awal!
10. Sebutkan dan jelaskan minimal tiga metode yang dapat digunakan untuk menyelesaikan suatu persamaan diferensial biasa dengan masalah nilai batas!
11. Jelaskan ide dasar metode *linear shooting* untuk menyelesaikan suatu persamaan diferensial biasa dengan masalah nilai batas!
12. Jelaskan bagaimana suatu masalah nilai batas dapat ditransformasi menjadi masalah nilai awal!
13. Jelaskan ide dasar untuk menyelesaikan suatu persamaan diferensial biasa dengan masalah nilai batas menggunakan beda-hingga (*finite-difference*)!
14. Jelaskan bagaimana suatu masalah nilai batas dapat ditransformasi menjadi suatu sistem persamaan linear!

### 13.5.2 Soal Pemrograman

#### Metode Runge-Kutta-Fehlberg (RKF45)

Untuk soal nomor 1 sampai 6, kerjakan menggunakan langkah-langkah berikut.

- a. Gunakan metode Runge-Kutta-Fehlberg (RKF45) untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan toleransi  $\delta_1 = 10^{-3}$ ,  $\delta_2 = 10^{-6}$ , dan  $\delta_3 = 10^{-12}$ .
- b. Gunakan metode Runge-Kutta ordo 4 (RK4) untuk menyelesaikan persamaan di bawah ini pada selang  $[0, 3]$  dengan mengatur ukuran langkah  $h$  sedemikian sehingga solusi yang dihasilkan memiliki galat akhir yang mendekati solusi RKF45 dengan toleransi  $\delta_1 = 10^{-3}$ ,  $\delta_2 = 10^{-6}$ , dan  $\delta_3 = 10^{-12}$ .
- c. Bandingkan banyaknya sub-interval yang terbentuk dari solusi yang dihasilkan oleh metode RKF45 dan RK4.
- d. Gambarkan grafik solusi beserta titik-titik sub-interval yang terbentuk oleh kedua metode dalam satu *figure*.

1.  $y' = t^2 - y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = -e^{-t} + t^2 - 2t + 2$

2.  $y' = 3y + 3t$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{4}{3}e^{3t} - t - \frac{1}{3}$

3.  $y' = -ty$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = e^{-t^2/2}$

4.  $y' = 2ty^2$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1-t^2}$

5.  $y' = -3y\sin(t)$  dengan  $y(0) = 1/2$  dan solusi eksak  $y(t) = \frac{1}{2}\exp(3\cos(t) - 3)$

6.  $y' = -\frac{1}{1+t}y$  dengan  $y(0) = 1$  dan solusi eksak  $y(t) = \frac{1}{1+t}$

### Sistem Persamaan Diferensial

Untuk soal nomor 7 sampai 9, kerjakan langkah-langkah berikut.

- Selesaikan menggunakan metode Runge-Kutta Orde-4 untuk sistem persamaan differensial,
- Buatlah grafik bidang fase (solusi  $y$  terhadap  $x$ ),
- Buatlah grafik bidang solusi  $x$  (solusi  $x$  terhadap  $t$ ) dan bidang solusi  $y$  (solusi  $y$  terhadap  $t$ ) dalam satu *figure*.

7. Diketahui sistem persamaan differensial :

$$\begin{aligned}\dot{x} &= -x + 0.04y + x^2y \\ \dot{y} &= 0.2 - 0.04y - x^2y\end{aligned}$$

dengan nilai awal  $x(0) = 0.5$  dan  $y(0) = 2$ . Hitung solusi  $t \in [0, 100]$  dengan  $h = 0.5$ .

8. Diketahui sistem persamaan differensial :

$$\begin{aligned}\dot{x} &= x(1-x) - \frac{1.1xy}{x+y} \\ \dot{y} &= y\left(-0.1 + \frac{0.65x}{x+y}\right)\end{aligned}$$

dengan  $x(0) = 0.2$  dan  $y(0) = 0.5$ . Hitung solusi  $t \in [0, 1000]$  dengan  $h = 2$ .

9. Diketahui sistem persamaan differensial :

$$\begin{aligned}\dot{x} &= 2x + 3y \\ \dot{y} &= 2x + y\end{aligned}$$

dengan  $x(0) = -2.7$  dan  $y(0) = 2.8$ . Hitung solusi  $t \in [0, 1]$  dengan  $h = 0.05$ .

### Persamaan Diferensial Orde-2

Untuk soal nomor 10 sampai 12, kerjakan langkah-langkah berikut.

- Formulakan masalah persamaan differensial orde-2 menjadi sistem persamaan differensial orde-1 yang ekuivalen dengan masalah awal,
- Gunakan metode Runge-Kutta untuk mencari penyelesaian sistem persamaan differensial pada selang  $[0, 2]$  dengan ukuran langkah  $h = 0.05$ ,
- Hitung nilai galat solusi numerik yang didapatkan terhadap solusi eksak yang diberikan pada setiap langkah.
- Gambarkan plot nilai hampiran solusi yang dihasilkan dan plot galat pada setiap langkah.

10. Diketahui persamaan

$$x''(t) + 6x'(t) + 9x(t) = 0$$

dengan  $x(0) = 4$  dan  $x'(0) = -4$  serta solusi eksak  $x(t) = 4e^{-3t} + 8te^{-3t}$ .

11. Diketahui persamaan

$$x''(t) + x(t) = 6\cos(t)$$

dengan  $x(0) = 2$  dan  $x'(0) = 3$  serta solusi eksak  $x(t) = 2\cos(t) + 3\sin(t) + 3t\sin(t)$ .

12. Diketahui persamaan

$$x''(t) + 3x'(t) = 12$$

dengan  $x(0) = 5$  dan  $x'(0) = 1$  serta solusi eksak  $x(t) = 4 + 4t + e^{-3t}$ .

**Masalah Nilai Batas**

Untuk soal nomor 13 sampai 16, selesaikan persamaan differensial dengan masalah nilai batas yang diberikan dengan metode *linear shooting* atau *finite difference*, kemudian buatlah plot solusi  $x$  terhadap  $t$ .

13. Diketahui  $x(0) = 5$  dan  $x(1) = 10$  dengan  $h = 0.05$  pada  $[0, 1]$  dari masalah nilai batas persamaan differensial berikut

$$x'' = 2x' - x + t^2 - 1$$

14. Diketahui  $x(1) = 1$  dan  $x(6) = 0$  dengan  $h = 0.06$  dari masalah nilai batas persamaan differensial berikut

$$x'' + \frac{1}{t}x' + \left(1 - \frac{1}{4t^2}\right)x = 0$$

15. Diketahui  $x(1) = 1$  dan  $x(3) = -1$  dengan  $h = 0.01$  pada  $[1, 3]$  dari masalah nilai batas persamaan differensial berikut

$$x'' = -\frac{2}{t}x' + \frac{2}{t^2}x + \frac{10\cos(\ln(t))}{t^2}$$

16. Diketahui  $x(0) = 0.6$  dan  $x(4) = -0.1$  dengan  $h = 0.01$  dari masalah nilai batas persamaan differensial berikut

$$x'' + 2x' + 2x = e^{-t} + \sin(2t)$$

**Soal Terapan**

17. Dalam studi tentang aliran non-isotermal dalam fluida Newton diantara plat paralel, muncul persamaan differensial

$$y'' = -t^2 e^y$$

dengan masalah nilai awal  $y(0) = 1$  dan  $y'(0) = 1$ . Gunakan metode Runge-Kutta orde 4 untuk menghitung nilai tebakan bagi  $y(1)$  dengan ukuran langkah  $h = 0.05$ .

18. Pada tahun 1926, Volterra menemukan model matematika untuk sistem mangsa-pemangsa. Jika  $r(t)$  dan  $f(t)$  masing-masing menunjukkan populasi kelinci dan rubah pada waktu  $t$ , maka model Volterra untuk pertumbuhan populasi dinotasikan dalam sistem persamaan differensial

$$\begin{aligned} r'(t) &= r(t)(a - bf(t)) \\ f'(t) &= f(t)(dr(t) - c) \end{aligned}$$

Jika diketahui  $a = 0.03$ ,  $b = 0.01$ ,  $c = 0.01$ , dan  $d = 0.01$ , hitung solusi numerik dari sistem persamaan differensial tersebut pada interval  $t \in [0, 500]$  dengan  $h = 1$  dan nilai awal

(a)  $r(0) = 1$ ,  $f(0) = 2$

(b)  $r(0) = 1$ ,  $f(0) = 4$

Plot solusi numerik dari fungsi  $r(t)$  dan  $f(t)$  dalam satu grafik.

19. Potensial elektrostatik  $u(t)$  antara dua bidang konsentris dengan jari-jari  $r = 1$  dan  $r = 4$  ditentukan dari persamaan differensial

$$\frac{d^2u}{dr^2} + \frac{2}{r} \frac{du}{dr} = 0$$

dengan nilai batas  $u(0) = 50$  dan  $u(4) = 100$ . Gunakan metode linear shooting untuk menentukan solusi masalah nilai batas tersebut.

20. Pada tahun 2018, Lajmiri menganalisis stabilitas dan bifurkasi dari model mangsa-pemangsa berikut.

$$\begin{aligned}\dot{x} &= x(1-x) - \frac{axy}{x+y} \\ \dot{y} &= y\left(-c + \frac{bx}{x+y}\right)\end{aligned}$$

dimana  $x(t)$  adalah populasi mangsa pada waktu  $t$  dan  $y(t)$  adalah populasi pemangsa pada waktu  $t$  serta  $a, b$ , dan  $c$  adalah koefisien. Dengan nilai awal  $x(0) = 0.2$  dan  $y(0) = 0.5$  serta koefisien  $b = 0.65$  dan  $c = 0.1$ , buatlah grafik bidang fase dan bidang solusi sistem tersebut jika

- (a)  $a = 0.6$  pada selang  $t \in [0, 100]$
- (b)  $a = 1.1$  pada selang  $t \in [0, 1000]$
- (c)  $a = 1.12$  pada selang  $t \in [0, 1000]$
- (d)  $a = 1.18$  pada selang  $t \in [0, 100]$

Jelaskan perilaku solusi  $x$  dan  $y$  dari masing-masing kasus tersebut.

21. Persamaan Matthieu didefinisikan dalam bentuk seperti berikut.

$$y'' + (\lambda - 2q \cos(2x))y = 0$$

pada selang  $[0, \pi]$ , dengan syarat batas  $y'(0) = 0$  dan  $y'(\pi) = 0$  untuk  $q = 5$ . Solusi dinormalisasi dengan cara menetapkan solusi memenuhi  $y(0) = 1$ . Carilah nilai  $\lambda$  yang memenuhi syarat batas  $y'(\pi) = 0$ . Gunakan nilai tebakan awal  $\lambda_{(0)} = 15$ .

---

---

# BAB 14

---

## PERSAMAAN DIFFERENSIAL PARSIAL

Selain persamaan differensial biasa (PDB), persamaan matematika yang sering digunakan untuk pemodelan adalah persamaan differensial parsial. Berbeda dengan PDB, persamaan differensial parsial (PDP) memiliki lebih dari satu peubah bebas. Solusi numerik dari beberapa PDP dapat dicari menggunakan metode beda-hingga.

Suatu persamaan differensial parsial dengan bentuk umum

$$A\Phi_{xx} + B\Phi_{xy} + C\Phi_{yy} = f(x, y, \Phi, \Phi_x, \Phi_y) \quad (14.1)$$

dengan  $A$ ,  $B$ , dan  $C$  merupakan suatu konstanta. Persamaan 14.1 disebut juga sebagai persamaan *quasi-linear*. Terdapat 3 kategori dari persamaan quasi-linear, yaitu

Jika  $B^2 - 4AC < 0$ , persamaan disebut eliptik

Jika  $B^2 - 4AC = 0$ , persamaan disebut parabolik

Jika  $B^2 - 4AC > 0$ , persamaan disebut hiperbolik

Pada bab ini akan dipelajari metode untuk menyelesaikan beberapa bentuk persamaan differensial parsial.

### 14.1 Persamaan Hiperbolik

Salah satu contoh bentuk persamaan hiperbolik adalah persamaan gelombang. Persamaan gelombang dapat berupa model persamaan dari seutas tali elastis yang digetarkan, yaitu  $u(x, t)$  dengan  $u$  merupakan tinggi titik ketika tali memiliki jarak  $x$  dari titik ujung tali pada waktu  $t$  setelah digetarkan.

#### Persamaan Gelombang

Suatu persamaan gelombang memiliki bentuk umum, yaitu

$$u_{tt}(x, t) = c^2 u_{xx}(x, t) \quad \text{untuk } 0 < x < a \text{ dan } 0 < t < b \quad (14.2)$$

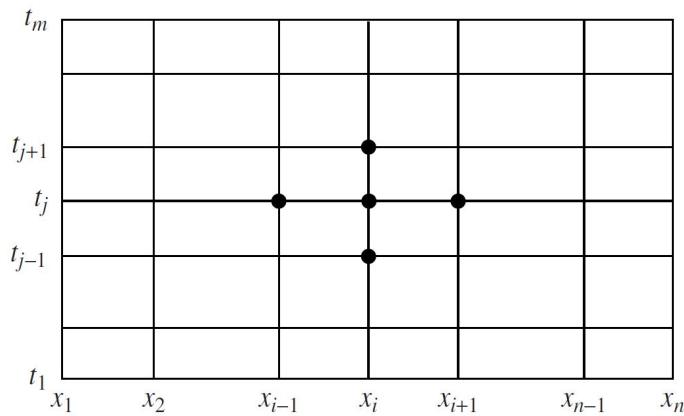
dengan kondisi batas

$$\begin{aligned} u(0, t) &= 0 \text{ dan } u(a, t) = 0 && \text{untuk } 0 \leq t \leq b \\ u(x, 0) &= f(x) && \text{untuk } 0 \leq x \leq a \\ u_t(x, 0) &= g(x) && \text{untuk } 0 < x < a \end{aligned} \quad (14.3)$$

Persamaan gelombang memodelkan suatu tali elastik yang digetarkan  $u$  dengan nilai tetap pada setiap ujung tali, yaitu  $x = 0$  dan  $x = a$ . Persamaan ini dapat diselesaikan secara analitik menggunakan deret Fourier. Sementara itu, secara numerik, solusi persamaan gelombang dapat dihampiri menggunakan metode beda-hingga.

### Solusi Numerik Persamaan Gelombang

Partisiakan persegi  $R = \{(x, t) : x \in [0, a], t \in [0, b]\}$  menjadi suatu grid yang mengandung persegi sebanyak  $n - 1 \times m - 1$  dengan panjang sisi  $\Delta x = h$  dan  $\Delta t = k$  seperti yang dapat dilihat pada Gambar 14.1 berikut.



**Gambar 14.1:** Pembagian grid untuk persamaan gelombang.

Solusi numerik dimulai dari bawah dengan  $t = t_1 = 0$  yaitu menggunakan kondisi batas  $u(x_i, t_1) = f(x_i)$ . Selanjutnya, akan digunakan metode beda-hingga untuk mencari hampiran solusi persamaan differensial untuk

$$\{u_{i,j} : i = 1, 2, \dots, n\} \quad \text{untuk } j = 2, 3, \dots, m$$

dengan  $u_{i,j}$  merupakan hampiran dari solusi eksak persamaan differensial  $u(x_i, t_j)$ .

Formula beda-pusat untuk hampiran  $u_{tt}(x, t)$  dan  $u_{xx}(x, t)$  adalah

$$u_{tt}(x, t) = \frac{u(x, t+k) - 2u(x, t) + u(x, t-k)}{k^2} + O(k^2) \quad (14.4)$$

dan

$$u_{xx}(x, t) = \frac{u(x+h, t) - 2u(x, t) + u(x-h, t)}{h^2} + O(h^2) \quad (14.5)$$

Selanjutnya, dengan menghilangkan  $O(k^2)$  dan  $O(h^2)$  serta menggunakan  $u_{i,j}$  untuk menghampiri  $u(x_i, t_j)$ , hasil substitusi Persamaan 14.4 dan 14.5 ke Persamaan 14.2 adalah

$$\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = c^2 \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \quad (14.6)$$

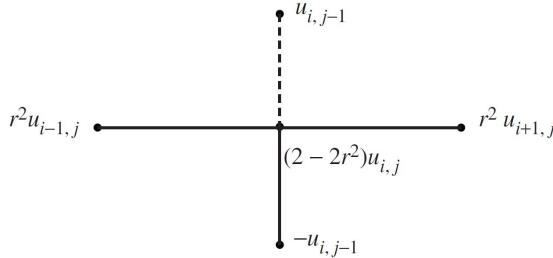
Supaya lebih mudah, substitusi nilai  $r = ck/h$  pada Persamaan 14.6, sehingga diperoleh persamaan

$$u_{i,j+1} - 2u_{i,j} + u_{i,j-1} = r^2 (u_{i+1,j} - 2u_{i,j} + u_{i-1,j}) \quad (14.7)$$

Asumsikan bahwa hampiran untuk baris ke- $j$  dan baris ke- $j-1$  telah diketahui, sehingga hampiran baris ke- $j+1$  adalah

$$u_{i,j+1} = (2 - 2r^2)u_{i,j} + r^2(u_{i+1,j} + u_{i-1,j}) - u_{i,j-1} \quad (14.8)$$

untuk  $i = 2, 3, \dots, n-1$ . Empat titik yang harus diketahui nilainya pada Persamaan 14.8 dapat dilihat pada Gambar 14.2 berikut.



**Gambar 14.2:** Stensil persamaan gelombang.

Hal yang perlu diperhatikan dalam penggunaan metode beda-hingga untuk mencari nilai hampiran solusi persamaan gelombang adalah solusi yang dihasilkan tidak selalu stabil. Syarat yang diperlukan agar mendapatkan solusi yang stabil adalah nilai  $r = ck/h \leq 1$ .

### Nilai Awal

Perlu diperhatikan bahwa Persamaan 14.8 dapat digunakan untuk mencari nilai solusi hampiran baris ke-3, apabila baris pertama dan ke-2 telah diketahui nilai hampirannya. Pada baris pertama, nilainya sudah diketahui menggunakan kondisi batas yaitu  $u_{i,1} = f_i$ . Sementara itu, baris kedua dapat dihampiri menggunakan cara seperti berikut.

Diawali menggunakan persamaan gelombang dan hubungannya dengan turunan kedua, sehingga didapatkan

$$\begin{aligned} u_{tt}(x_i, 0) &= c^2 u_{xx}(x_i, 0) \\ &= c^2 f''(x_i) = c^2 \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2} + O(h^2) \end{aligned} \quad (14.9)$$

Dengan formula Taylor ordo 2, fungsi  $u(x, k)$  dapat dituliskan menjadi

$$u(x, k) = u(x, 0) + u_t(x, 0)k + \frac{u_{tt}(x, 0)k^2}{2} + O(k^3) \quad (14.10)$$

Selanjutnya, substitusi kondisi batas persamaan gelombang, yaitu  $u(x_i, 0) = f(x_i) = f_i$  dan  $u_t(x_i, 0) = g(x_i) = g_i$  serta Persamaan 14.9 ke Persamaan 14.10, sehingga diperoleh persamaan

$$u(x_i, k) = f_i + kg_i + \frac{c^2 k^2}{2h^2} (f_{i+1} - 2f_i + f_{i-1})$$

Gunakan  $r = ck/h$ , maka diperoleh nilai persamaan untuk menghampiri baris kedua yaitu

$$u_{i,2} = (1 - r^2)f_i + kg_i + \frac{r^2}{2} (f_{i+1} + f_{i-1}) \quad (14.11)$$

untuk  $i = 2, 3, \dots, n-1$ .

**Contoh 14.1 :** Gunakan metode beda-hingga untuk mencari solusi persamaan gelombang dari suatu senar yang bergetar.

$$u_{tt}(x, t) = 4u_{xx}(x, t) \quad \text{untuk } 0 < x < 1 \text{ dan } 0 < t < 0.5$$

dengan kondisi batas

$$\begin{aligned} u(0, t) &= 0 \text{ dan } u(1, t) = 0 && \text{untuk } 0 \leq t \leq 0.5 \\ u(x, 0) &= f(x) = \sin(\pi x) + \sin(2\pi x) && \text{untuk } 0 \leq x \leq 1 \\ u_t(x, 0) &= g(x) = 0 && \text{untuk } 0 \leq x \leq 1 \end{aligned}$$

dan panjang sisi  $h = 0.1$  dan  $k = 0.05$ .

**Solusi :** Dengan  $c = 2$ , didapatkan nilai  $r = ck/h = 2(0.05)/0.1 = 1$ , sehingga solusi yang dihasilkan metode beda-hingga akan stabil.

Solusi numerik diawali dengan menghitung baris pertama, yaitu

$$u_{i,1} = u(x_i, 0) = f(x_i)$$

Selanjutnya, hitung solusi numerik untuk baris ke-dua menggunakan Persamaan 14.11 dengan  $r = 1$  dan  $g(x_i) = 0$ , yaitu

$$u_{i,2} = \frac{1}{2}(f_{i+1} + f_{i-1}) \quad \text{untuk } i = 2, 3, \dots, 10$$

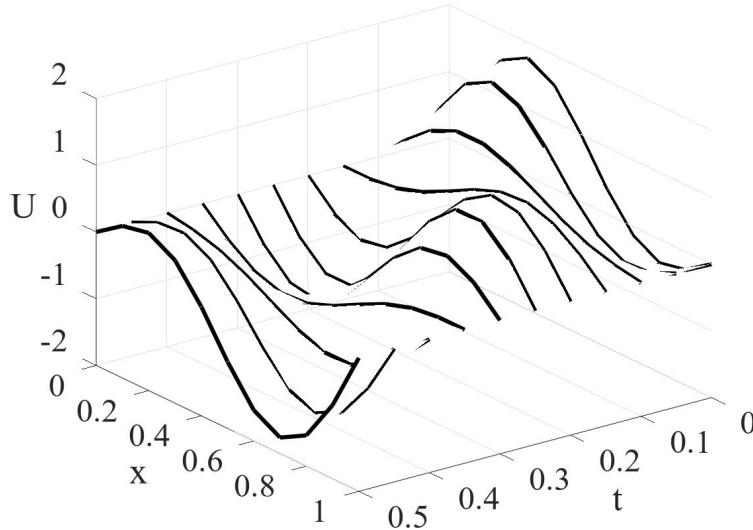
Dengan substitusi  $r = 1$  pada Persamaan 14.8, diperoleh persamaan hampiran untuk baris ke-3 sampai baris ke-10, yaitu

$$u_{i,j+1} = u_{i+1,j} + u_{i-1,j} - u_{i,j-1}$$

untuk  $j = 2, 3, \dots, 10$  dan  $i = 2, 3, \dots, 10$ . Hasil penghitungan solusi numerik dapat dilihat pada Tabel 14.1 dan Gambar 14.3 berikut.  $\triangle$

**Tabel 14.1:** Hasil penghitungan solusi numerik persamaan gelombang.

$t_j$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$	$x_{11}$
0.00	0.0	0.897	1.539	1.760	1.539	1.000	0.363	-0.142	-0.363	-0.279	0.0
0.05	0.0	0.769	1.328	1.539	1.380	0.951	0.429	-0.000	-0.210	-0.182	0.0
0.10	0.0	0.432	0.769	0.948	0.951	0.809	0.588	0.361	0.182	0.068	0.0
0.15	0.0	-0.000	0.052	0.182	0.377	0.588	0.741	0.769	0.639	0.363	0.0
0.20	0.0	-0.380	-0.588	-0.519	-0.182	0.309	0.769	1.019	0.951	0.571	0.0
0.25	0.0	-0.588	-0.951	-0.951	-0.588	-0.000	0.588	0.951	0.951	0.588	0.0
0.30	0.0	-0.571	-0.951	-1.019	-0.769	-0.309	0.182	0.519	0.588	0.380	0.0
0.35	0.0	-0.363	-0.639	-0.769	-0.741	-0.588	-0.377	-0.182	-0.052	-0.000	0.0
0.40	0.0	-0.068	-0.182	-0.361	-0.588	-0.809	-0.951	-0.948	-0.769	-0.432	0.0
0.45	0.0	0.182	0.210	0.000	-0.429	-0.951	-1.380	-1.539	-1.328	-0.769	0.0
0.50	0.0	0.279	0.363	0.142	-0.363	-1.000	-1.539	-1.760	-1.539	-0.897	0.0



**Gambar 14.3:** Getaran senar pada persamaan gelombang.

## 14.2 Persamaan Parabolik

Salah satu contoh bentuk persamaan parabolik adalah persamaan panas. Persamaan panas berupa model persamaan dari suhu suatu pipa terinsulasi dengan suhu konstan di setiap ujung pipa yaitu  $c_1$  dan  $c_2$  serta distribusi suhu awal sepanjang pipa adalah  $f(x)$ .

### Persamaan Panas

Suatu persamaan panas memiliki bentuk umum, yaitu

$$u_t(x, t) = c^2 u_{xx}(x, t) \quad \text{untuk } 0 \leq x < a \text{ dan } 0 < t < b \quad (14.12)$$

dengan kondisi awal

$$u(x, 0) = f(x) \quad \text{untuk } t = 0 \text{ dan } 0 \leq x \leq a \quad (14.13)$$

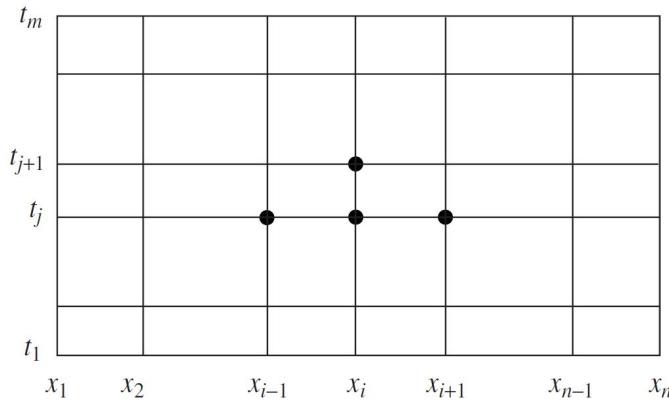
dan kondisi batas

$$\begin{aligned} u(0, t) &= g_1(t) \equiv c_1 && \text{untuk } x = 0 \text{ dan } 0 \leq t \leq b \\ u(a, t) &= g_2(t) \equiv c_2 && \text{untuk } x = a \text{ dan } 0 \leq t \leq b \end{aligned} \quad (14.14)$$

Suatu persamaan panas dapat diselesaikan secara analitik menggunakan deret Fourier. Sementara itu, secara numerik, solusi persamaan panas dapat dihampiri menggunakan metode beda-maju.

### Solusi Numerik Persamaan Panas

Partisiakan persegi  $R = \{(x, t) : x \in [0, a], t \in [0, b]\}$  menjadi suatu grid yang mengandung persegi sebanyak  $n - 1 \times m - 1$  dengan panjang sisi  $\Delta x = h$  dan  $\Delta t = k$  seperti yang dapat dilihat pada Gambar 14.4 berikut.



**Gambar 14.4:** Pembagian grid untuk persamaan panas.

Solusi numerik dimulai dari bawah dengan  $t = t_1 = 0$  yaitu menggunakan kondisi batas  $u(x_i, t_1) = f(x_i)$ . Selanjutnya, akan digunakan metode beda-hingga untuk mencari hampiran solusi persamaan differensial untuk

$$\{u_{i,j} : i = 1, 2, \dots, n\} \quad \text{untuk } j = 2, 3, \dots, m$$

Formula beda-maju dapat digunakan untuk menghampiri  $u_t(x, t)$  dan formula beda-tengah untuk  $u_{xx}(x, t)$ , yaitu

$$u_t(x, t) = \frac{u(x, t+k) - u(x, t)}{k} + O(k) \quad (14.15)$$

dan

$$u_{xx}(x, t) = \frac{u(x-h, t) - 2u(x, t) + u(x+h, t)}{h^2} + O(h^2) \quad (14.16)$$

Selanjutnya, dengan menghilangkan  $O(k)$  dan  $O(h^2)$  serta menggunakan  $u_{i,j}$  untuk menghampiri  $u(x_i, t_j)$ , hasil substitusi Persamaan 14.15 dan 14.16 ke Persamaan 14.12 adalah

$$\frac{u_{i,j+1} - u_{i,j}}{k} = c^2 \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2} \quad (14.17)$$

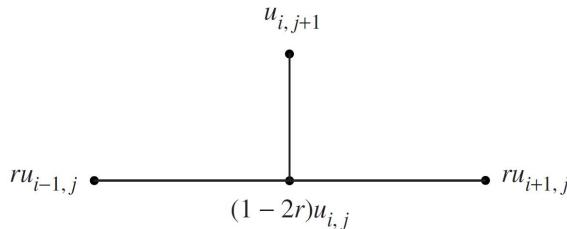
Supaya lebih mudah, substitusi  $r = c^2 k / h^2$  pada Persamaan 14.17, sehingga diperoleh persamaan

$$u_{i,j+1} - u_{i,j} = r(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) \quad (14.18)$$

Asumsikan bahwa hampiran untuk baris ke- $j$  telah diketahui, sehingga hampiran baris ke- $j+1$  adalah

$$u_{i,j+1} = (1 - 2r)u_{i,j} + r(u_{i-1,j} + u_{i+1,j}) \quad (14.19)$$

untuk  $i = 2, 3, \dots, n-1$ . Tiga titik yang harus diketahui nilainya pada Persamaan 14.19 dapat dilihat pada Gambar 14.5 berikut.



**Gambar 14.5:** Stensil persamaan panas.

Hal yang perlu diperhatikan dalam penggunaan metode beda-hingga untuk mencari nilai hampiran solusi persamaan panas adalah solusi yang dihasilkan tidak selalu stabil. Syarat yang diperlukan agar solusi yang dihasilkan stabil adalah nilai  $0 \leq r = c^2 k / h^2 \leq \frac{1}{2}$ .

**Contoh 14.2 :** Gunakan metode beda-maju untuk mencari solusi persamaan panas

$$u_t(x, t) = u_{xx}(x, t) \quad \text{untuk } 0 < x < 1 \text{ dan } 0 < t < 0.20$$

dengan kondisi awal

$$u(x, 0) = f(x) = 4x - 4x^2 \quad \text{untuk } t = 0 \text{ dan } 0 \leq x \leq 1$$

dan kondisi batas

$$\begin{aligned} u(0, t) &= g_1(t) \equiv 0 && \text{untuk } x = 0 \text{ dan } 0 \leq t \leq 0.20 \\ u(a, t) &= g_2(t) \equiv 0 && \text{untuk } x = a \text{ dan } 0 \leq t \leq 0.20 \end{aligned}$$

dan panjang sisi  $h = 0.2$  dan  $k = 0.02$ .

**Solusi :** Dengan  $c = 1$ , didapatkan nilai  $r = c^2 k / h^2 = 1^2 (0.02) / 0.2^2 = 0.5$ , sehingga solusi yang dihasilkan metode beda-maju akan stabil.

Solusi numerik diawali dengan menghitung baris pertama, yaitu

$$u_{i,1} = u(x_i, 0) = f(x_i)$$

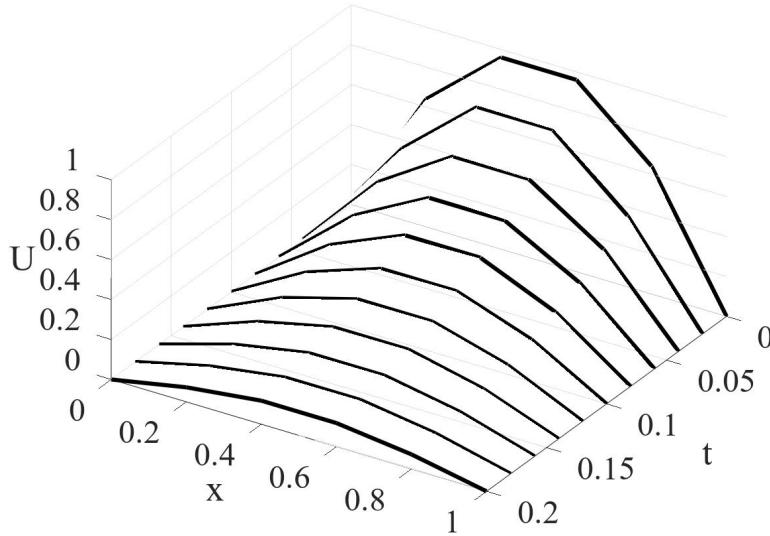
Selanjutnya, hitung solusi numerik untuk baris ke-dua hingga baris ke-11 menggunakan Persamaan 14.19 dengan  $r = \frac{1}{2}$ , yaitu

$$u_{i,j+1} = \frac{u_{i-1,j} + u_{i+1,j}}{2}$$

untuk  $j = 1, 2, \dots, 10$  dan  $i = 2, 3, \dots, 5$ . Hasil penghitungan solusi numerik dapat dilihat pada Tabel 14.2 dan Gambar 14.6 berikut.  $\triangle$

**Tabel 14.2:** Hasil penghitungan solusi numerik persamaan panas.

$t_j$	$x_1 = 0$	$x_2 = 0.2$	$x_3 = 0.4$	$x_4 = 0.6$	$x_5 = 0.8$	$x_6 = 1$
0.00	0.000	0.64000	0.96000	0.96000	0.64000	0.000
0.02	0.000	0.48000	0.80000	0.80000	0.48000	0.000
0.04	0.000	0.40000	0.64000	0.64000	0.40000	0.000
0.06	0.000	0.32000	0.52000	0.52000	0.32000	0.000
0.08	0.000	0.26000	0.42000	0.42000	0.26000	0.000
0.10	0.000	0.21000	0.34000	0.34000	0.21000	0.000
0.12	0.000	0.17000	0.27500	0.27500	0.17000	0.000
0.14	0.000	0.13750	0.22250	0.22250	0.13750	0.000
0.16	0.000	0.11125	0.18000	0.18000	0.11125	0.000
0.18	0.000	0.09000	0.14563	0.14563	0.09000	0.000
0.20	0.000	0.07281	0.11781	0.11781	0.07281	0.000



**Gambar 14.6:** Getaran senar pada persamaan panas.

### 14.3 Persamaan Eliptik

Beberapa contoh dari persamaan eliptik antara lain persamaan Laplace, Poisson, dan Helmholtz. Laplacian dari fungsi  $u(x, y)$  didefinisikan sebagai

$$\nabla^2 u = u_{xx} + u_{yy} \quad (14.20)$$

Dengan notasi tersebut, persamaan Laplace, Poisson, dan Helmholtz dapat dituliskan dalam bentuk

$\nabla^2 u = 0$	Persamaan Laplace
$\nabla^2 u = g(x, y)$	Persamaan Poisson
$\nabla^2 u + f(x, y)u = g(x, y)$	Persamaan Helmholtz

(14.21)

#### Solusi Persamaan Laplace

Solusi persamaan Laplace dapat dicari menggunakan metode beda-hingga. Telah diketahui bahwa hampiran turunan ke-2 dari fungsi  $f$  adalah

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(h^2)$$

Ketika formula tersebut diaplikasikan pada fungsi  $u(x, y)$  untuk menghampiri nilai  $u_{xx}(x, y)$  dan  $u_{yy}(x, y)$ , akan didapatkan Laplacian sebagai berikut.

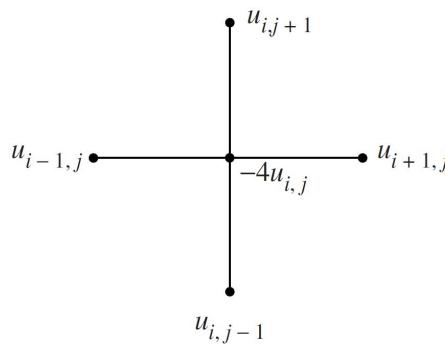
$$\begin{aligned} \nabla^2 u &= u_{xx} + u_{yy} \\ &= \frac{u(x+h, y) - 2u(x, y) + u(x-h, y)}{h^2} + \frac{u(x, y+h) - 2u(x, y) + u(x, y-h)}{h^2} + O(h^2) \\ &= \frac{u(x+h, y) + u(x-h, y) + u(x, y+h) + u(x, y-h) - 4u(x, y)}{h^2} + O(h^2) \end{aligned}$$

Asumsikan bahwa persegi  $R = \{(x, y) : 0 \leq x \leq a, 0 \leq y \leq b\}$  dengan  $b/a = m/n$  dapat dipartisiakan menjadi sebanyak  $n - 1 \times m - 1$  persegi dengan panjang sisi  $h$ . Gunakan notasi  $u_{i,j}$  sebagai nilai hampiran untuk  $u(x_i, y_j)$ , maka akan didapatkan hampiran persamaan Laplace yaitu

$$\nabla^2 u_{i,j} \approx \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2} = 0 \quad (14.22)$$

Persamaan tersebut dikenal sebagai formula beda 5-titik (*five-point difference formula*). Titik-titik yang digunakan pada formula tersebut dapat dilihat pada stensil yang terdapat pada Gambar 14.7 berikut. Dengan mengeliminasi  $h^2$ , akan diperoleh fomula komputasi Laplacian seperti berikut.

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = 0 \quad (14.23)$$



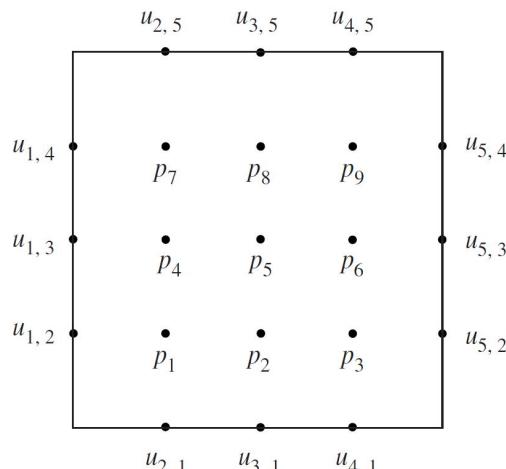
**Gambar 14.7:** Stensil persamaan Laplace.

### Pengaturan Sistem Linear

Asumsikan bahwa diketahui suatu persamaan Laplace dengan nilai batas grid, yaitu

$$\begin{aligned} u(x_1, y_j) &= u_{1,j} && \text{untuk } 2 \leq j \leq m-1 \text{ sebagai batas kiri} \\ u(x_i, y_1) &= u_{i,1} && \text{untuk } 2 \leq i \leq n-1 \text{ sebagai batas bawah} \\ u(x_n, y_j) &= u_{n,j} && \text{untuk } 2 \leq j \leq m-1 \text{ sebagai batas kanan} \\ u(x_i, y_m) &= u_{i,m} && \text{untuk } 2 \leq i \leq n-1 \text{ sebagai batas atas} \end{aligned}$$

Misalkan bahwa persegi  $R$  dipartisiakan dengan  $m = n = 5$  dan terdapat 9 nilai  $u(x_i, y_j)$  yang tidak diketahui dengan label  $p_1, p_2, \dots, p_9$  seperti pada Gambar berikut.



**Gambar 14.8:** Pembagian grid menjadi  $5 \times 5$ .

Dengan menerapkan Persamaan 14.23, jika dipilih  $p_1$  sebagai pusat stensil yaitu  $u_{2,2}$ , maka akan diperoleh persamaan  $p_2 + u_{1,2} + p_4 + u_{2,1} - 4p_1 = 0$ . Selanjutnya, jika Persamaan 14.23 diterapkan dengan pusat stensil mulai dari  $p_1, p_2$ , hingga  $p_9$ , maka akan diperoleh sistem persamaan berikut.

$$\begin{aligned} p_2 + u_{1,2} + p_4 + u_{2,1} - 4p_1 &= 0 \\ p_3 + p_1 + p_5 + u_{3,1} - 4p_2 &= 0 \\ u_{5,2} + p_2 + p_6 + u_{4,1} - 4p_3 &= 0 \\ p_5 + u_{1,3} + p_7 + p_1 - 4p_4 &= 0 \\ p_6 + p_4 + p_8 + p_2 - 4p_5 &= 0 \\ u_{5,3} + p_5 + p_9 + p_3 - 4p_6 &= 0 \\ p_8 + u_{1,4} + u_{2,5} + p_4 - 4p_7 &= 0 \\ p_9 + p_7 + u_{3,5} + p_5 - 4p_8 &= 0 \\ u_{5,4} + p_8 + u_{4,5} + p_6 - 4p_9 &= 0 \end{aligned}$$

Dalam notasi matriks, sistem persamaan linear tersebut dapat ditulis menjadi

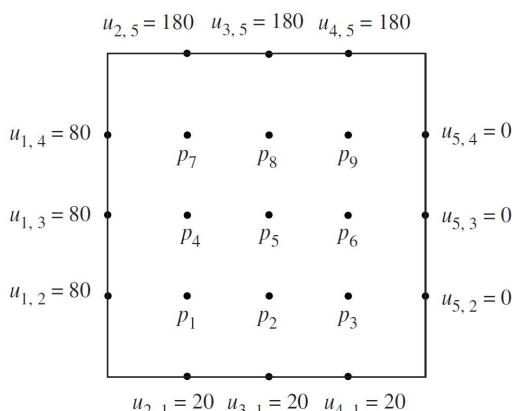
$$\left[ \begin{array}{ccccccccc} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{array} \right] \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \end{bmatrix} = \begin{bmatrix} -u_{2,1} - u_{1,2} \\ -u_{3,1} \\ -u_{4,1} - u_{5,2} \\ -u_{1,3} \\ 0 \\ -u_{5,3} \\ -u_{2,5} - u_{1,4} \\ -u_{3,5} \\ -u_{4,5} - u_{5,4} \end{bmatrix} \quad (14.24)$$

**Contoh 14.3 :** Carilah solusi hampiran untuk persamaan Laplace  $\nabla^2 u = 0$  pada persegi  $R = \{(x,y) : 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan  $u(x,y)$  menunjukkan suhu pada titik  $(x,y)$  dan memiliki nilai batas

$$\begin{aligned} u(x,0) &= 20 & \text{dan} & \quad u(x,4) = 180 & \text{untuk } 0 < x < 4 \\ u(0,y) &= 80 & \text{dan} & \quad u(4,y) = 0 & \text{untuk } 0 < y < 4 \end{aligned}$$

yang dipartisiakan menjadi  $5 \times 5$  grid dengan panjang sisi  $h = 1$ .

**Solusi :** Berdasarkan Gambar 14.8, pembagian grid beserta nilai batas dan nilai yang tidak diketahui dapat dilihat pada Gambar 14.9 berikut.



**Gambar 14.9:** Pembagian grid untuk Contoh 14.3.

Dengan menerapkan SPL pada Persamaan 14.24, akan didapatkan sistem  $AP = B$  seperti berikut.

$$\begin{bmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \end{bmatrix} = \begin{bmatrix} -100 \\ -20 \\ -20 \\ -80 \\ 0 \\ -0 \\ -260 \\ -180 \\ -180 \end{bmatrix}$$

Dengan menyelesaikan SPL di atas, akan diperoleh solusi matriks  $P$  yaitu

$$\begin{aligned} P &= [p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9]^T \\ &= [55.714, 43.214, 27.143, 79.643, 70.000, 45.357, 112.857, 111.786, 84.286]^T \end{aligned}$$

△

### Kondisi Batas Turunan

Kondisi batas Neumann menentukan turunan berarah dari  $u(x,y)$  ke titik tepi. Untuk ilustrasi, akan digunakan kondisi turunan normal bernilai nol, yaitu

$$\frac{\partial}{\partial N} u(x,y) = 0 \quad (14.25)$$

Untuk aplikasi di area aliran panas, kondisii ini menunjukkan bahwa daerah tepi terisolasi secara termal dan fluks panas di seluruh tepi adalah nol.

Misalkan bahwa  $x = x_n$  dianggap tetap dan hanya diperhatikan tepi kanan  $x = a$  dari persegi panjang  $R = \{(x,y) : 0 \leq x \leq a, 0 \leq y \leq b\}$ . Kondisi batas normal yang digunakan sepanjang sisi ini adalah

$$\frac{\partial}{\partial x} u(x_n, y_j) = u_x(x_n, y_j) = 0 \quad (14.26)$$

Oleh karena itu, persamaan beda Laplace pada titik  $(x_n, y_j)$  adalah

$$u_{n+1,j} + u_{n-1,j} + u_{n,j+1} + u_{n,j-1} - 4u_{n,j} = 0 \quad (14.27)$$

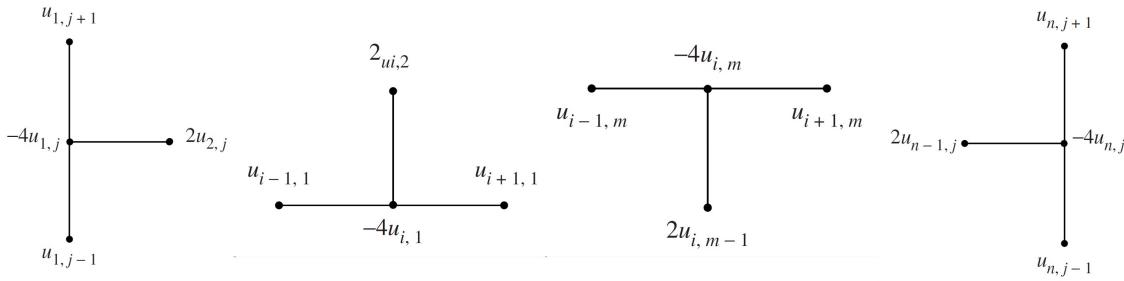
Dari Persamaan 14.27, nilai  $u_{n+1,j}$  tidak diketahui karena terletak di luar daerah  $R$ . Akan tetapi, berdasarkan formula beda hingga

$$\frac{u_{n+1,j} - u_{n-1,j}}{2h} \approx u_x(x_n, y_j) = 0 \quad (14.28)$$

diperoleh nilai hampiran  $u_{n+1,j} \approx u_{n-1,j}$ , dengan ordo akurasi  $O(h^2)$ . Ketika nilai hampiran tersebut digunakan pada Persamaan 14.27, hasilnya adalah

$$2u_{n-1,j} + u_{n,j+1} + u_{n,j-1} - 4u_{n,j} = 0 \quad (14.29)$$

Formula tersebut menghubungkan nilai  $u_{n,j}$  ke tiga nilai tetangganya, yaitu  $u_{n-1,j}$ ,  $u_{n,j+1}$ , dan  $u_{n,j-1}$ .



Gambar 14.10: Stensil komputasi batas Neumann.

Stensil komputasi untuk tepi lainnya dapat diturunkan dengan cara yang sama (lihat Gambar 14.10). Empat kasus untuk stensil komputasi dari kondisi batas Neumanm dapat dirangkum menjadi seperti berikut.

$$\text{Tepi Bawah : } 2u_{i,2} + u_{i-1,1} + u_{i+1,1} - 4u_{i,1} = 0 \quad (14.30)$$

$$\text{Tepi Atas : } 2u_{i,m-1} + u_{i-1,m} + u_{i+1,m} - 4u_{i,m} = 0 \quad (14.31)$$

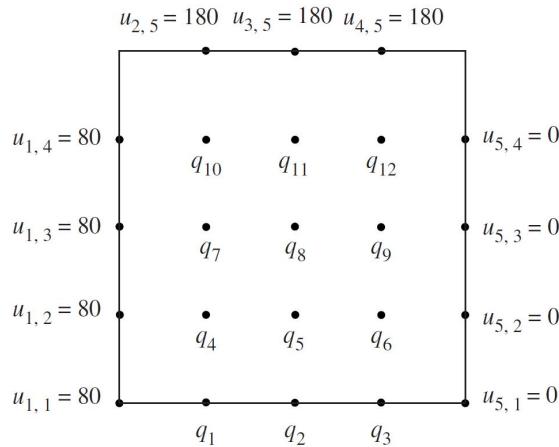
$$\text{Tepi Kiri : } 2u_{2,j} + u_{1,j-1} + u_{1,j+1} - 4u_{1,j} = 0 \quad (14.32)$$

$$\text{Tepi Kanan : } 2u_{n-1,j} + u_{n,j-1} + u_{n,j+1} - 4u_{n,j} = 0 \quad (14.33)$$

Misalkan bahwa kondisi turunan  $\partial u(x, y) / \partial N = 0$  digunakan di sepanjang bagian dari batas  $R$ , dan nilai batas yang diketahui dari  $u(x, y)$  digunakan pada bagian lain dari batas, maka terdapat masalah campuran pada kondisi batas. Persamaan 14.29 digunakan untuk menentukan hampiran  $u(x_i, y_j)$  pada titik batas yang melibatkan turunan. Sementara itu, formula komputasi Laplace pada Persamaan 14.27 masih digunakan untuk menentukan aproksimasi untuk  $u(x_i, y_j)$  pada titik-titik interior  $R$ .

**Contoh 14.4 :** Carilah solusi hampiran untuk persamaan Laplace  $\nabla^2 u = 0$  pada persegi  $R = \{(x, y) : 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan  $u(x, y)$  menunjukkan suhu pada titik  $(x, y)$  dan memiliki nilai batas seperti pada Gambar 14.11.

$$\begin{aligned} u_y(x, 0) &= 0 \quad \text{dan} \quad u(x, 4) = 180 \quad \text{untuk } 0 < x < 4 \\ u(0, y) &= 80 \quad \text{dan} \quad u(4, y) = 0 \quad \text{untuk } 0 \leq y < 4 \end{aligned}$$



Gambar 14.11: Pembagian grid untuk Contoh 14.4.

**Solusi :** Formula komputasi Neumann untuk tepi bawah (Persamaan 14.30) digunakan pada titik batas  $q_1, q_2$ , dan  $q_3$ , serta stensil komputasi Laplace (Persamaan 14.27) digunakan untuk titik-titik lainnya, yaitu  $q_4, q_5, \dots, q_{12}$ . Berdasarkan formulasi tersebut, suatu sistem linear  $AQ = B$  diperoleh yaitu

$$\left[ \begin{array}{cccccccccccc} -4 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{array} \right] \left[ \begin{array}{c} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \\ q_7 \\ q_8 \\ q_9 \\ q_{10} \\ q_{11} \\ q_{12} \end{array} \right] = \left[ \begin{array}{c} -8 \\ 0 \\ 0 \\ -80 \\ 0 \\ 0 \\ -80 \\ 0 \\ 0 \\ -260 \\ -180 \\ -180 \end{array} \right]$$

Nilai solusi dari  $Q$  dapat diperoleh dengan cara menyelesaikan sistem linear di atas menggunakan eliminasi Gauss (atau metode lain yang lebih efisien). Suhu pada titik-titik interior dan sepanjang tepi bagian bawah diekspresikan dalam bentuk vektor, yaitu

$$\begin{aligned} Q &= [q_1 \ q_2 \ q_3 \ q_4 \ q_5 \ q_6 \ q_7 \ q_8 \ q_9 \ q_{10} \ q_{11} \ q_{12}]^T \\ &= [71.8218 \ 56.8543 \ 32.2342 \ 75.2165 \ 61.6806 \ 36.0412 \\ &\quad 87.3636 \ 78.6103 \ 50.2502 \ 115.628 \ 115.147 \ 86.3492]^T \quad \triangle \end{aligned}$$

## 14.4 Praktikum

Pada bagian ini, praktikan diwajibkan untuk mempelajari, menulis ulang, dan melengkapi beberapa program yang akan digunakan pada praktikum ini serta mengecek kebenaran program tersebut. Berikut merupakan beberapa program yang akan digunakan pada praktikum ini.

**Metode Beda-Hingga untuk Persamaan Gelombang** berisi program untuk mencari solusi persamaan differensial parsial yaitu persamaan gelombang. Program ini secara *default* berisi 7 masukan, yaitu fungsi nilai batas  $f(x)$  dan  $g(x)$ , batas solusi  $a$  dan  $b$ , koefisien persamaan gelombang  $c$ , dan jumlah grid solusi  $x$  dan  $t$  yaitu  $m$  dan  $n$ , serta 1 luaran, yaitu solusi persamaan gelombang  $U$ . Berikut merupakan program untuk metode beda-hingga untuk persamaan gelombang.

**Algoritma 14.1:** Metode beda-hingga untuk persamaan gelombang.

```
function gelombang(f,g,a,b,c,m,n)
    h = a/(m-1);
    x = 0:h:1;
    k = b/(n-1);
    r = c*k/h;
    U = zeros(m,n);
    for i = 2:m-1
        U[i,1] = f(x[i]);
        U[i,2] = (1-r^2)*f(x[i]) + k*g(x[i]) + r^2/2*(f(x[i+1])+f(x[i-1]));
    end
    for j = 3:n
        for i = 2:(m-1)
```

```

    U[i,j] = (2-2*r^2)*U[i,j-1]+r^2*(U[i-1,j-1]+U[i+1,j-1])-U[i,j-2];
end
end
U = U';
end

```

**Metode Beda-Maju untuk Persamaan Panas** berisi program untuk mencari solusi persamaan differensial parsial yaitu persamaan panas. Program ini secara *default* berisi 8 masukan, yaitu fungsi nilai awal  $f(x)$ , koefisien nilai batas  $c_1$  dan  $c_2$ , nilai batas solusi  $a$  dan  $b$ , koefisien persamaan gelombang  $c$ , dan jumlah grid solusi  $x$  dan  $t$  yaitu  $m$  dan  $n$ , serta 1 luaran, yaitu solusi persamaan panas  $U$ . Berikut merupakan program untuk metode beda-maju untuk persamaan panas.

**Algoritma 14.2:** Metode beda maju untuk persamaan panas.

```

function panas(f,c1,c2,a,b,c,m,n)
h = a/(m-1);
k = b/(n-1);
r = c^2*k/h^2;
U = zeros(m,n);
U[1,:] .= c1;
U[m,:] .= c2;
U[2:m-1,1] = f.(h:h:(m-2)*h)';
for j = 2:n
    for i = 2:m-1
        U[i,j]=(1-2*r)*U[i,j-1]+ r*(U[i-1,j-1]+U[i+1,j-1]);
    end
end
U=U';
return U,r
end

```

**Metode Dirichlet untuk Persamaan Laplace** berisi program untuk mencari solusi persamaan differensial parsial yaitu persamaan Laplace dengan kondisi batas Dirichlet. Program ini secara *default* berisi 8 masukan, yaitu fungsi nilai batas  $f_1(x)$ ,  $f_2(x)$ ,  $f_3(x)$ , dan  $f_4(x)$ , nilai batas solusi  $a$  dan  $b$ , dan ukuran langkah  $h$ , serta 1 luaran, yaitu solusi persamaan Laplace dengan kondisi batas Dirichlet  $U$ . Berikut merupakan program metode Dirichlet untuk persamaan Laplace.

**Algoritma 14.3:** Metode Dirichlet untuk persamaan Laplace

```

function dirichlet(f1,f2,f3,f4,a,b,h)
maxi = 100;
tol = 1e-7;
n = Int(a/h)+1;
m = Int(b/h)+1;
U = (a.* (f1.(0).+f2.(0)).+b.* (f3.(0).+f4.(0)))./(2*a.+2*b).+ones(n,m);
# Masalah nilai batas
U[1:n,1]=f1.(0:h:(n-1)*h);
U[1:n,m]=f2.(0:h:(n-1)*h);
U[1,1:m]=f3.(0:h:(m-1)*h);
U[n,1:m]=f4.(0:h:(m-1)*h);
U[1,1]=(U[1,2]+U[2,1])/2;
U[1,m]=(U[1,m-1]+U[2,m])/2;
U[n,1]=(U[n-1,1]+U[n,2])/2;

```

```

U[n,m]=(U[n-1,m]+U[n,m-1])/2;
w = 4/(2+sqrt(4-(cos(pi/(n-1))+cos(pi/(m-1)))^2));
err=1; iter=0;
while (err>tol)&&(iter<=maxi)
    err = 0;
    for j = 2:m-1
        for i = 2:n-1
            relx = w*(U[i,j+1]+U[i,j-1]+U[i+1,j]+U[i-1,j]-4*U[i,j])/4;
            U[i,j]=U[i,j]+relx;
            if err<=abs(relx)
                err=abs(relx);
            end
        end
    end
    iter=iter+1;
end
U = U';
end

```

#### 14.4.1 Persamaan Hiperbolik

Persamaan pertama yang dipelajari dari persamaan differensial parsial adalah persamaan hiperbolik. Salah satu contoh persamaan hiperbolik adalah persamaan gelombang (*wave equation*).

**Contoh 14.5 :** Diberikan suatu persamaan gelombang dari senar yang bergetar seperti berikut.

$$u_{tt}(x,t) = 4u_{xx}(x,t) \quad \text{dengan } 0 < x < 1 \text{ dan } 0 < t < 0.5$$

dengan batas

$$\begin{aligned} u(0,t) &= 0 \text{ dan } u(1,t) = 0 && \text{untuk } 0 \leq t \leq 0.5 \\ u(x,0) &= f(x) = \sin(\pi x) + \sin(2\pi x) && \text{untuk } 0 \leq x \leq 1 \\ u_t(x,0) &= g(x) = 0 && \text{untuk } 0 \leq x \leq 1 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari masalah persamaan gelombang di atas serta gambar solusi numerik dalam grafik 3 dimensi.

**Langkah 1:** Penyelesaian persamaan gelombang menggunakan metode beda-hingga pada Program 14.1, jika diketahui  $h = 0.1$  dan  $k = 0.05$ .

```

Inp: f(x) = sin(pi*x)+sin(2*pi*x);
      g(x) = 0;
      a = 1;
      b = 0.5;
      c = 2;
      m = 11;
      t = 0:0.05:b;
      n = length(t);
      U = gelombang(f,g,a,b,c,m,n)

```

```
Out: 11x11 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
 0.0   0.896802   1.53884   ...  -0.363271  -0.278768  0.0
 0.0   0.769421   1.32844   ...  -0.210404  -0.181636  0.0
 0.0   0.431636   0.769421   ...  0.181636   0.0683644  0.0
 0.0  -2.22045e-16  0.0515989   ...  0.639384   0.363271  0.0
 0.0  -0.380037  -0.587785   ...  0.951057   0.57102   0.0
 0.0  -0.587785  -0.951057   ...  0.951057   0.587785  0.0
 0.0  -0.57102   -0.951057   ...  0.587785   0.380037  0.0
 0.0  -0.363271  -0.639384   ...  -0.0515989  0.0   0.0
 0.0  -0.0683644  -0.181636   ...  -0.769421  -0.431636  0.0
 0.0   0.181636   0.210404   ...  -1.32844   -0.769421  0.0
 0.0   0.278768   0.363271   ...  -1.53884  -0.896802  0.0
```

**Langkah 2:** Cek kestabilan solusi yang dihasilkan.

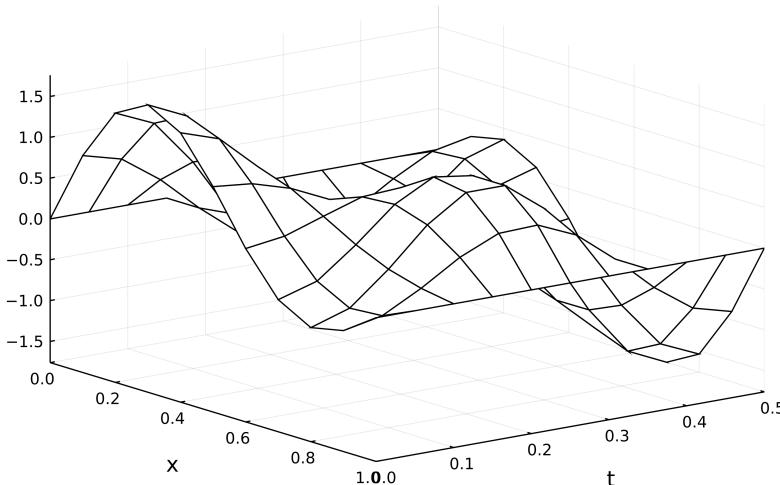
Dengan  $h = 0.1$  dan  $k = 0.05$  serta  $c = 2$ , nilai rasio  $r$  untuk persamaan gelombang adalah

$$r = c \frac{k}{h} = 2 \frac{0.05}{0.1} = 1$$

Karena  $r \leq 1$ , maka solusi persamaan gelombang pada **Langkah 1** stabil.

**Langkah 3:** Gambar grafik solusi persamaan gelombang dalam grafik 3 dimensi menggunakan wireframe.

```
Inp: using Plots
      x = 0:0.1:a;
      t = 0:0.05:b;
      plot(x,t,U,st=:wireframe,camera=(50,45))
```



**Gambar 14.12:** Solusi persamaan gelombang Contoh 14.5

**Soal Latihan:** Diberikan suatu persamaan gelombang dari senar yang bergetar seperti berikut.

$$u_{tt}(x,t) = 4u_{xx}(x,t) \quad \text{dengan } 0 < x < 2 \text{ dan } 0 < t < 1$$

dengan batas

$$\begin{aligned} u(0,t) &= 0 \text{ dan } u(2,t) = 0 && \text{untuk } 0 \leq t \leq 1 \\ u(x,0) &= f(x) = -\sin(\pi x)/(x+1) && \text{untuk } 0 \leq x \leq 2 \\ u_t(x,0) &= g(x) = 0 && \text{untuk } 0 \leq x \leq 2 \end{aligned}$$

Hitung solusi numerik dari masalah persamaan gelombang di atas menggunakan ukuran langkah  $h = 0.1$  dan  $k = 0.05$ , kemudian gambarkan solusi numerik dalam suatu grafik 3 dimensi dengan langkah-langkah seperti pada Contoh 14.5.

#### 14.4.2 Persamaan Parabolik

Persamaan selanjutnya yang akan dipelajari dari persamaan differensial parsial adalah persamaan parabolik. Salah satu contoh persamaan hiperbolik adalah persamaan panas (*heat equation*).

**Contoh 14.6 :** Diberikan suatu persamaan panas seperti berikut.

$$u_t(x,t) = u_{xx}(x,t) \quad \text{untuk } 0 < x < 1 \text{ dan } 0 < t < 0.2$$

dengan kondisi awal dan kondisi batas

$$\begin{aligned} u(x,0) &= f(x) = 4x - 4x^2 && \text{untuk } t = 0 \text{ dan } 0 \leq x \leq 1. \\ u(0,t) &= c_1 = 0 && \text{untuk } x = 0 \text{ dan } 0 \leq t \leq 0.2 \\ u(1,t) &= c_2 = 0 && \text{untuk } x = 1 \text{ dan } 0 \leq t \leq 0.2 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari masalah persamaan panas di atas serta gambar solusi numerik dalam grafik 3 dimensi.

**Langkah 1:** Penyelesaian persamaan panas menggunakan metode beda-maju (*forward-difference*) pada Program 14.2, jika diketahui  $h = 0.2$  dan  $k = 0.02$ .

```
Inp: f(x) = 4*x - 4*x.^2;
c1 = 0;
c2 = 0;
a = 1;
b = 0.2;
c = 1 ;
m = 6;
n = 11;
Inp: U,r = panas(f,c1,c2,a,b,c,m,n)
@show r
U

Out: r = 0.4999999999999999
Out: 11x6 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
 0.0  0.64      0.96      0.96      0.64      0.0
 0.0  0.48      0.8       0.8       0.48      0.0
 0.0  0.4       0.64      0.64      0.4       0.0
 0.0  0.32      0.52       0.52      0.32      0.0
 0.0  0.26      0.42       0.42      0.26      0.0
 0.0  0.21      0.34       0.34      0.21      0.0
 0.0  0.17      0.275      0.275      0.17      0.0
 0.0  0.1375    0.2225     0.2225     0.1375     0.0
 0.0  0.11125   0.18       0.18       0.11125   0.0
 0.0  0.09       0.145625   0.145625   0.09       0.0
 0.0  0.0728125 0.117813   0.117813   0.0728125  0.0
```

**Langkah 2:** Cek kestabilan solusi yang dihasilkan.

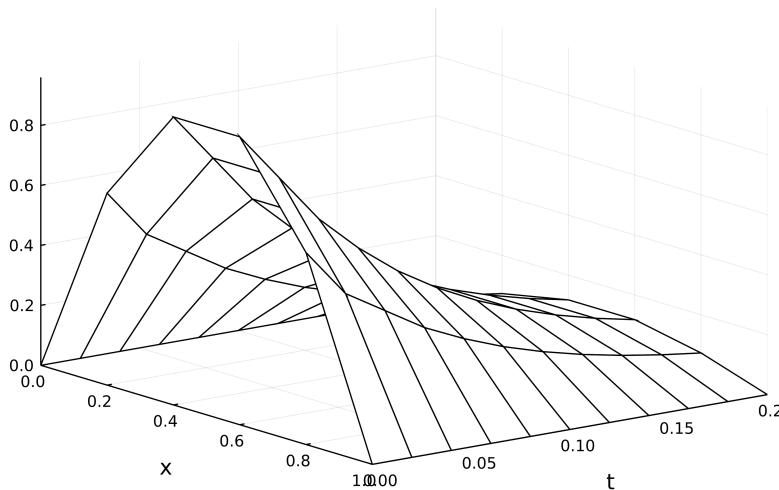
Dengan  $h = 0.2$  dan  $k = 0.02$  serta  $c = 1$ , nilai rasio  $r$  untuk persamaan panas adalah

$$r = c^2 \frac{k}{h^2} = 1 \frac{0.02}{0.2^2} = 0.5$$

Karena  $r = 0.5$ , maka solusi persamaan panas pada **Langkah 1** adalah stabil.

**Langkah 3:** Gambar grafik solusi persamaan gelombang dengan plot 3 dimensi menggunakan wireframe.

```
Inp: x = 0:0.2:a
      t = 0:0.02:b
      plot(x,t,U,st=:wireframe,camera=(50,45))
      xlabel!("x")
      ylabel!("t")
```



**Gambar 14.13:** Solusi persamaan panas pada Contoh 14.6.

**Soal Latihan:** Diberikan suatu persamaan panas seperti berikut.

$$u_t(x,t) = u_{xx}(x,t) \quad \text{untuk } 0 < x < 1 \text{ dan } 0 < t < 0.1$$

dengan kondisi awal dan kondisi batas

$$\begin{aligned} u(x,0) &= f(x) = -\sin(2\pi x) && \text{untuk } t = 0 \text{ dan } 0 \leq x \leq 1. \\ u(0,t) &= c_1 = 0 && \text{untuk } x = 0 \text{ dan } 0 \leq t \leq 0.1 \\ u(1,t) &= c_2 = -1 && \text{untuk } x = 1 \text{ dan } 0 \leq t \leq 0.1 \end{aligned}$$

Hitung solusi numerik dari masalah persamaan panas di atas menggunakan ukuran langkah  $h = 0.1$  dan  $k = 0.005$ , kemudian gambarkan solusi numerik dalam suatu grafik 3 dimensi dengan langkah-langkah seperti pada Contoh 14.6.

### 14.4.3 Persamaan Eliptik

Persamaan selanjutnya yang akan dipelajari dari persamaan differensial parsial adalah persamaan eliptik. Salah satu contoh persamaan hiperbolik adalah persamaan Laplace.

**Contoh 14.7 Kondisi Batas Dirichlet:** Diketahui suatu persamaan Laplace seperti berikut.

$$u_{xx} + u_{yy} = 0$$

dimana  $R = \{(x,y) \mid 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan masalah nilai batas

$$\begin{aligned} u(x, 0) &= 20 & \text{dan} & \quad u(x, 4) = 180 & \text{untuk } 0 < x < 4 \\ u(0, y) &= 80 & \text{dan} & \quad u(4, y) = 0 & \text{untuk } 0 < y < 4 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari masalah persamaan Laplace di atas menggunakan

1. sistem linear yang terbentuk dari stensil pada Persamaan 14.23 dengan  $h = 1$
  2. metode dirichlet pada Program 14.3 dengan  $h = 1$  dan  $h = 0.1$
- serta gambar solusi numerik dalam grafik 3 dimensi.

**Langkah 1:** Pendefinisian SPL berdasarkan stensil yang diperoleh dengan  $h = 1$ .

Karena digunakan nilai  $h = 1$ , jumlah grid yang akan terbentuk adalah  $5 \times 5$  dengan pembagian grid seperti pada Gambar 14.9. Berdasarkan Persamaan 14.23, diperoleh sistem persamaan linear yaitu

$$\left[ \begin{array}{ccccccccc} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{array} \right] \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \end{bmatrix} = \begin{bmatrix} -100 \\ -20 \\ -20 \\ -80 \\ 0 \\ -0 \\ -260 \\ -180 \\ -180 \end{bmatrix}$$

**Langkah 2:** Selesaikan sistem linear di atas untuk mendapatkan solusi interior dari masalah persamaan Laplace yang diberikan.

```
Inp: A = [ -4 1 0 1 0 0 0 0 0
           1 -4 1 0 1 0 0 0 0
           0 1 -4 0 0 1 0 0 0
           1 0 0 -4 1 0 1 0 0
           0 1 0 1 -4 1 0 1 0
           0 0 1 0 1 -4 0 0 1
           0 0 0 1 0 0 -4 1 0
           0 0 0 0 1 0 1 -4 1
           0 0 0 0 0 1 0 1 -4]
B = [-100;-20;-20;-80;0;0;-260;-180;-180]
```

```
Inp: P = A\B
```

```
Out: 9-element Array{Float64,1}:
      55.71428571428571
      43.21428571428571
      27.142857142857146
      79.64285714285714
      70.0
      45.35714285714286
      112.85714285714285
      111.78571428571428
      84.28571428571429
```

**Langkah 3:** Ubah solusi interior tersebut ke bentuk matriks dan sisipkan kondisi batas di setiap sisi.

```
Inp: reshape(P,3,3)'

Out: 3x3 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
      55.7143   43.2143   27.1429
      79.6429   70.0      45.3571
     112.857   111.786   84.2857

Inp: U = zeros(5,5)
      # interior
      U[2:end-1,2:end-1] = reshape(P,3,3)'

Inp: # sepanjang tepi
      U[1,2:end-1] .= 20
      U[end,2:end-1] .= 180
      U[2:end-1,1] .= 80
      U[2:end-1,end] .= 0

Inp: # titik pojok
      U[1,1] = (U[1,2]+U[2,1])/2
      U[1,end] = (U[1,end-1]+U[2,end])/2
      U[end,1] = (U[end-1,1]+U[end,2])/2
      U[end,end] = (U[end-1,end]+U[end,end-1])/2
      U

Out: 5x5 Array{Float64,2}:
      50.0    20.0    20.0    20.0    10.0
      80.0    55.7143  43.2143  27.1429  0.0
      80.0    79.6429  70.0      45.3571  0.0
      80.0    112.857  111.786   84.2857  0.0
     130.0    180.0    180.0    180.0    90.0
```

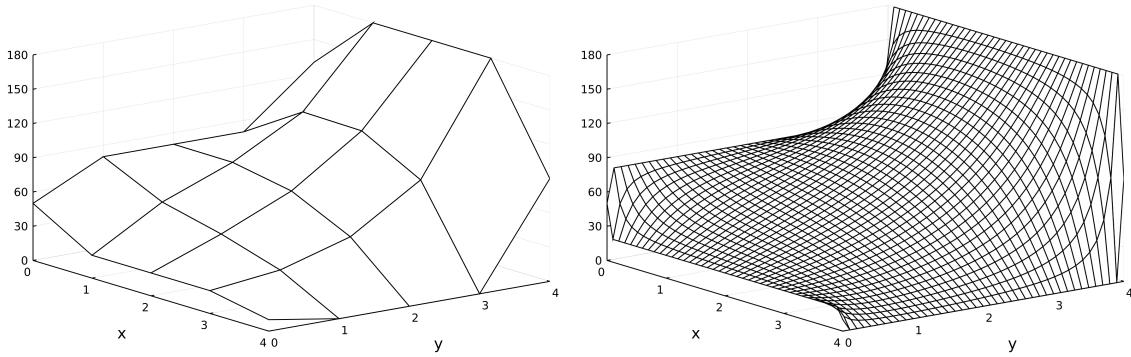
**Langkah 4:** Penyelesaian persamaan Laplace di atas menggunakan metode Dirichlet pada Program 14.3 dengan ukuran langkah  $h = 1$ .

```
Inp: f1(x)= 20+0*x;
      f2(x)= 180+0*x;
      f3(y)= 80+0*y;
      f4(y)= 0*y;
      a = 4; b = 4;
      h = 1;
      U = dirichlet(f1,f2,f3,f4,a,b,h)

Out: 5x5 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
      50.0    20.0    20.0    20.0    10.0
      80.0    55.7143  43.2143  27.1429  0.0
      80.0    79.6429  70.0      45.3571  0.0
      80.0    112.857  111.786   84.2857  0.0
     130.0    180.0    180.0    180.0    90.0
```

**Langkah 5:** Penyelesaian persamaan Laplace di atas menggunakan metode Dirichlet pada Program 14.3 dengan ukuran langkah  $h = 0.1$ .

```
Inp: f1(x)= 20+0*x;
      f2(x)= 180+0*x;
      f3(y)= 80+0*y;
      f4(y)= 0*y;
      a = 4; b = 4;
      h = 0.1;
      U = dirichlet(f1,f2,f3,f4,a,b,h)
```

(a) Solusi dengan  $h = 1$ (b) Solusi dengan  $h = 0.1$ **Gambar 14.14:** Solusi persamaan Laplace dengan kondisi batas Dirichlet.

**Soal Latihan:** Diketahui suatu persamaan Laplace seperti berikut.

$$u_{xx} + u_{yy} = 0$$

dimana  $R = \{(x,y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}$  dengan masalah nilai batas

$$\begin{aligned} u(x, 0) &= 10 & \text{dan} & \quad u(x, 3) = 90 & \text{untuk } 0 < x < 3 \\ u(0, y) &= 70 & \text{dan} & \quad u(3, y) = 0 & \text{untuk } 0 < y < 3 \end{aligned}$$

Hitung solusi numerik dari masalah persamaan Laplace di atas menggunakan

1. sistem linear yang terbentuk dari stensil pada Persamaan 14.23 dengan  $h = 1$
2. metode dirichlet pada Program 14.3 dengan  $h = 1$  dan  $h = 0.1$

serta gambar solusi numerik dalam grafik 3 dimensi dengan langkah-langkah seperti pada Contoh 14.7.

**Contoh 14.8 Kondisi Batas Neumann:** Diberikan persamaan Laplace  $\nabla^2 u = 0$  pada daerah persegi  $R = \{(x,y) \mid 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan  $u(x,y)$  menyatakan suhu pada titik  $(x,y)$  dan nilai-nilai batas Neumann berikut:

$$\begin{aligned} u_y(x, 0) &= 0 & \text{dan} & \quad u(x, 4) = 180 & \text{untuk } 0 < x < 4 \\ u(0, y) &= 80 & \text{dan} & \quad u(4, y) = 0 & \text{untuk } 0 < y < 4 \end{aligned}$$

Berikut merupakan langkah-langkah untuk menghitung solusi numerik dari masalah persamaan Laplace di atas menggunakan sistem linear yang terbentuk dari stensil persamaan Laplace dan Neumann dengan  $h = 1$  serta gambar solusi numerik dalam grafik 3 dimensi.

**Langkah 1:** Pendefinisian SPL berdasarkan stensil yang diperoleh dengan  $h = 1$ .

Karena digunakan nilai  $h = 1$ , jumlah grid yang akan terbentuk adalah  $5 \times 5$  dengan pembagian grid seperti pada Gambar 14.11. Perhatikan titik  $q_1$ . Pada titik tersebut berlaku batas Neumann yaitu  $u_y(x, 0) = 0$ . Dengan formula beda-pusat, misalkan terdapat variable dummy  $q_{-1}$  dibawah  $q_1$ , sehingga akan diperoleh

$$u_y(x, 0) \approx \frac{q_4 - q_{-1}}{2\Delta y} = 0 \Leftrightarrow q_{-1} = q_4$$

Dengan demikian, persamaan dengan  $q_1$  sebagai pusat adalah  $80 - 4q_1 + q_2 + 2q_4 = 0$  dan hal serupa berlaku untuk  $q_2$  dan  $q_3$ . Jadi, sistem persamaan linear yang berkorespondensi dengan Gambar 14.11 adalah

$$\left[ \begin{array}{cccccccccccc} -4 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{array} \right] = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \\ q_7 \\ q_8 \\ q_9 \\ q_{10} \\ q_{11} \\ q_{12} \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 0 \\ -80 \\ 0 \\ 0 \\ -80 \\ 0 \\ 0 \\ -260 \\ -180 \\ -180 \end{bmatrix}$$

**Langkah 2:** Selesaikan sistem linear di atas untuk mendapatkan solusi interior dan tepi dengan batas Neumann dari masalah persamaan Laplace yang diberikan.

```
Inp: A = [ -4 1 0 2 0 0 0 0 0 0 0 0
           1 -4 1 0 2 0 0 0 0 0 0 0
           0 1 -4 0 0 2 0 0 0 0 0 0
           1 0 0 -4 1 0 1 0 0 0 0 0
           0 1 0 1 -4 1 0 1 0 0 0 0
           0 0 1 0 1 -4 0 0 1 0 0 0
           0 0 0 1 0 0 -4 1 0 1 0 0
           0 0 0 0 1 0 1 -4 1 0 1 0
           0 0 0 0 0 1 0 1 -4 0 0 1
           0 0 0 0 0 0 1 0 0 -4 1 0
           0 0 0 0 0 0 0 1 0 1 -4 1
           0 0 0 0 0 0 0 0 1 0 1 -4]
B = [-80;0;0;-80;0;0;-80;0;0;-260;-180;-180]
q = A\B
```

```
Out: 12-element Array{Float64,1}:
    71.82182412865356
    56.854295717748435
    32.234195262674184
    75.21650039843291
    61.68058173983301
    36.04124266647415
    87.36359572524509
    78.61028817667653
    50.250193663389425
    115.62759432587092
    115.1467815782386
    86.34924381040702
```

**Langkah 3:** Ubah solusi interior tersebut ke bentuk matriks dan sisipkan kondisi batas di setiap sisi.

```
Inp: reshape(q, 3, 4)'
```

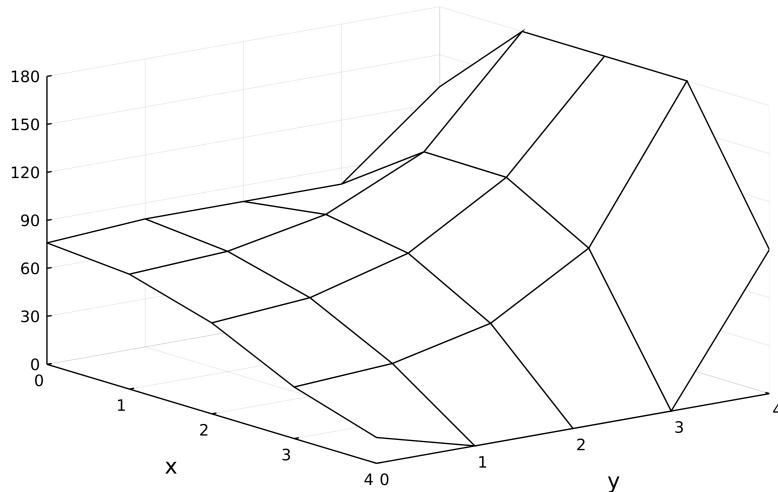
```
Out: 4x3 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
    71.8218   56.8543   32.2342
    75.2165   61.6806   36.0412
    87.3636   78.6103   50.2502
    115.628   115.147   86.3492
```

```
Inp: U = zeros(5,5)
    # interior
    U[1:end-1,2:end-1] = reshape(q,3,4)'

Inp: # sepanjang tepi
    U[end,2:end-1] .= 180
    U[2:end-1,1] .= 80
    U[2:end-1,end] .= 0

Inp: # titik pojok
    U[1,1] = (U[1,2]+U[2,1])/2
    U[1,end] = (U[1,end-1]+U[2,end])/2
    U[end,1] = (U[end-1,1]+U[end,2])/2
    U[end,end] = (U[end-1,end]+U[end,end-1])/2
U

Out: 5x5 Array{Float64,2}:
 75.9109  71.8218  56.8543  32.2342  16.1171
 80.0      75.2165  61.6806  36.0412  0.0
 80.0      87.3636  78.6103  50.2502  0.0
 80.0      115.628   115.147   86.3492  0.0
130.0     180.0     180.0     180.0     90.0
```



**Gambar 14.15:** Solusi persamaan Laplace dengan batas Neumann pada Contoh 14.8.

---

**Soal Latihan:** Diketahui suatu persamaan Laplace seperti berikut.

$$u_{xx} + u_{yy} = 0$$

dimana  $R = \{(x,y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}$  dengan masalah nilai batas

$$\begin{aligned} u(x, 0) &= 10 & \text{dan} & \quad u_y(x, 3) = 0 & \text{untuk } 0 < x < 3 \\ u(0, y) &= 70 & \text{dan} & \quad u_x(3, y) = 0 & \text{untuk } 0 < y < 3 \end{aligned}$$

Hitung solusi numerik dari masalah persamaan Laplace di atas menggunakan sistem persamaan linear yang terbentuk dari stensil persamaan Laplace dan Neumann dengan  $h = 1$  serta gambar solusi numerik dalam grafik 3 dimensi dengan langkah-langkah seperti pada Contoh 14.8.

---

## 14.5 Latihan-Latihan

### 14.5.1 Ulasan Materi

1. Jelaskan apa yang dimaksud dengan persamaan diferensial parsial! Apa perbedaan-nya dengan persamaan diferensial biasa?
2. Sebutkan dan jelaskan tiga kategori dari persamaan *quasi-linear*! Berikan contoh untuk masing-masing kategori!
3. Sebutkan dan jelaskan minimal tiga metode yang dapat digunakan untuk menyelesaikan suatu persamaan diferensial parsial!
4. Jelaskan apa yang dimaksud dengan persamaan gelombang dan berikan contohnya!
5. Jelaskan bagaimana cara untuk menyelesaikan persamaan gelombang (*wave equation*) menggunakan metode beda-hingga!
6. Jelaskan cara untuk memeriksa kestabilan solusi metode beda-hingga untuk menyelesaikan persamaan gelombang (*wave equation*)!
7. Jelaskan apa yang dimaksud dengan persamaan panas dan berikan contohnya!
8. Jelaskan bagaimana cara untuk menyelesaikan persamaan panas (*heat equation*) menggunakan metode beda-hingga!
9. Jelaskan cara untuk memeriksa kestabilan solusi metode beda-hingga untuk menyelesaikan persamaan panas (*heat equation*)!
10. Sebutkan dan jelaskan minimal tiga persamaan yang tergolong ke dalam persamaan eliptik!
11. Sebutkan dan jelaskan dua jenis kondisi batas pada persamaan Laplace!
12. Jelaskan bagaimana cara untuk menyelesaikan persamaan Laplace dengan kondisi batas Dirichlet menggunakan metode beda-hingga!
13. Jelaskan bagaimana cara untuk menyelesaikan persamaan Laplace dengan kondisi batas Neumann menggunakan metode beda-hingga!

### 14.5.2 Soal Pemrograman

#### Persamaan Gelombang

Untuk nomor 1 sampai 5, gunakan metode beda-hingga untuk menyelesaikan persamaan gelombang  $u_{tt} = c^2 u_{xx}$  untuk  $0 \leq x \leq a$  dan  $0 \leq t \leq b$  dengan nilai batas

$$\begin{aligned} u(0,t) &= 0 \text{ dan } u(a,t) = 0 & \text{untuk } 0 \leq t \leq b \\ u(x,0) &= f(x) & \text{untuk } 0 \leq x \leq a \\ u'(x,0) &= g(x) & \text{untuk } 0 < x < a \end{aligned}$$

menggunakan langkah-langkah seperti berikut.

1. Cek nilai rasio  $r$  dari persamaan gelombang, apakah solusi yang dihasilkan akan stabil?
2. Jika stabil, hitunglah solusi numerik persamaan gelombang berikut.
3. Jika tidak stabil, tentukan terlebih dahulu ukuran langkah  $h$  dan  $k$  sehingga solusi yang dihasilkan stabil.
4. Buatlah plot dari solusi tersebut dengan fungsi `surf`.

1.  $a = 1, b = 1, c = 1$ , dengan fungsi

$$\begin{aligned}f(x) &= \sin(\pi x) \\g(x) &= 0\end{aligned}$$

dengan ukuran langkah  $h = 0.1, k = 0.1$

2.  $a = 1, b = 1, c = 1$ , dengan fungsi

$$\begin{aligned}f(x) &= x - x^2 \\g(x) &= 0\end{aligned}$$

dengan ukuran langkah  $h = 0.1, k = 0.1$

3.  $a = 1, b = 1, c = 1$ , dengan fungsi

$$\begin{aligned}f(x) &= \begin{cases} 2x & , \text{untuk } 0 \leq x \leq \frac{1}{2} \\ 2 - 2x & , \text{untuk } \frac{1}{2} \leq x \leq 1 \end{cases} \\g(x) &= 0\end{aligned}$$

dengan ukuran langkah  $h = 0.1, k = 0.1$

4.  $a = 1, b = 1, c = 2$ , dengan fungsi

$$\begin{aligned}f(x) &= \sin(\pi x) \\g(x) &= 0\end{aligned}$$

dengan ukuran langkah  $h = 0.1, k = 0.1$

5.  $a = 1, b = 1, c = 2$ , dengan fungsi

$$\begin{aligned}f(x) &= \sin(2\pi x) + \sin(4\pi x) \\g(x) &= 0\end{aligned}$$

dengan ukuran langkah  $h = 0.1, k = 0.05$

### Persamaan Panas

Untuk nomor 6 sampai 10, gunakan metode beda-maju untuk menyelesaikan persamaan panas  $u_t = c^2 u_{xx}$  untuk  $0 \leq x \leq a$  dan  $0 \leq t \leq b$  dengan masalah nilai awal dan batas

$$\begin{aligned}u(x, 0) &= f(x) && \text{untuk } 0 \leq x \leq a \\u(0, t) &= c_1 \text{ dan } u(a, t) = c_2 && \text{untuk } 0 \leq t \leq b\end{aligned}$$

menggunakan langkah-langkah seperti berikut.

(Petunjuk: gunakan  $c_1 = c_2 = 0$  dan  $a = 1, b = 0.1$ )

1. Tentukan ukuran langkah  $k$ , sehingga solusi yang dihasilkan stabil.
2. Hitung solusi numerik persamaan panas.
3. Buatlah plot dari solusi tersebut.

6.  $c = 2$  dan ukuran langkah  $h = 0.1$  dengan fungsi  $f(x) = x^2$ .

7.  $c = 1$  dan ukuran langkah  $h = 0.1$  dengan fungsi  $f(x) = x^2 + \sin(x)$ .

8.  $c = 2$  dan ukuran langkah  $h = 0.1$  dengan fungsi  $f(x) = \sin(\pi x) + \sin(2\pi x)$ .
9.  $c = 1$  dan ukuran langkah  $h = 0.05$  dengan fungsi  $f(x) = 1 - |2x - 1|$ .
10.  $c = 2$  dan ukuran langkah  $h = 0.1$  dengan fungsi  $f(x) = \sin(\pi x)$ .

### Persamaan Laplace

Untuk nomor 11 sampai 15, gunakan stensil untuk kondisi batas Dirichlet dan/atau Neumann untuk menyelesaikan persamaan Laplace berikut, kemudian gambarkan solusi numerik dalam suatu plot 3 dimensi.

11. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}$  dengan batas  $u(x,0) = 10$  dan  $u(x,3) = 90$   
 $u(0,y) = 70$  dan  $u(3,y) = 0$
12. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}$  dengan batas  $u(x,0) = 10$  dan  $u(x,3) = 90$   
 $u(0,y) = 70$  dan  $u_x(3,y) = 0$
13. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}$  dengan batas  $u_y(x,0) = x$  dan  $u(x,3) = 90$   
 $u(0,y) = 70$  dan  $u_x(3,y) = 0$
14. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan batas  $u(x,0) = 10$  dan  $u(x,4) = 120$   
 $u(0,y) = 90$  dan  $u(4,y) = 40$
15. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan batas  $u(x,0) = 10$  dan  $u(x,4) = 120$   
 $u_x(0,y) = 0$  dan  $u(4,y) = 40$
16. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 4, 0 \leq y \leq 4\}$  dengan batas  $u(x,0) = 10$  dan  $u(x,4) = 120$   
 $u_x(0,y) = 10$  dan  $u_x(4,y) = 0$
17. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 1.5, 0 \leq y \leq 1.5\}$  dengan batas  $u(x,0) = x^3$  dan  $u(x,1.5) = x^3$   
 $u(0,y) = 0$  dan  $u(1.5,y) = 1$
18. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 1.5, 0 \leq y \leq 1.5\}$  dengan batas  $u(x,0) = x^3$  dan  $u(x,1.5) = x^3$   
 $u_x(0,y) = 0$  dan  $u(1.5,y) = 1$
19. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 1.5, 0 \leq y \leq 1.5\}$  dengan batas  $u(x,0) = x^3$  dan  $u(x,1.5) = x^3$   
 $u_x(0,y) = 0$  dan  $u_x(1.5,y) = 1$
20. Diketahui persamaan  $\nabla^2 u = 0$ ,  $R = \{(x,y) \mid 0 \leq x \leq 1, 0 \leq y \leq 1\}$  dengan batas  $u(x,0) = x^4$  dan  $u(x,1) = x^4 - 13.5x^2 + 5.0625$   
 $u(0,y) = y^4$  dan  $u(1,y) = y^4 - 13.5y^2 + 5.0625$

---

## DAFTAR PUSTAKA

- Ayatullah, F., Julianto, M. T., Garnadi, A. D., dan Nurdiani, S. (2012). Metode *Conjugate Gradient* Paralel untuk Menyelesaikan Sistem Persamaan Linear dalam SCILAB. *Journal of Mathematics and Its Applications*, 11(2):19–36.
- Erliana, W., Garnadi, A. D., Nurdiani, S., dan Julianto, M. T. (2015). Penyelesaian Masalah Syarat Batas Persamaan Diferensial Biasa dalam Software R dengan menggunakan bvpsolve. *Journal of Mathematics and Its Applications*, 14(2):9–18.
- Garnadi, A. D., Ayatullah, F., Ekastrya, D., Nurdiani, S., Julianto, M. T., dan Erliana, W. (2015). Penyelesaian Masalah Syarat Batas Persamaan Diferensial Biasa dalam Scilab dengan menggunakan bvode. *Journal of Mathematics and Its Applications*, 14(1):55–68.
- Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill Higher Education, 2nd edition.
- Ilyas, M., Garnadi, A. D., Julianto, M. T., dan Nurdiani, S. (2020). Row Action Algebraic Reconstruction Techniques Implementation in Graphic Cards using SCILAB (*Benchmark in Medical Imaging, Seismic Imaging and Walnut Imaging*). *Preprints*.
- Matthews, J. H. dan Fink, K. D. (1999). *Numerical Methods Using MATLAB*. Prentice Hall, 3rd edition.
- Munir, R. (2013). *Metode Numerik*. Informatika.
- Ruhiyat, Ilyas, M., Garnadi, A. D., dan Nurdiani, S. (2012). Teknik Rekonstruksi Aljabar untuk Menyelesaikan Sistem Persamaan Linear dengan SCILAB. *Journal of Mathematics and Its Applications*, 11(2):11–18.
- Shampine, L. F., Kierzenka, J., dan Reichelt, M. W. (2000). Solving Boundary Value Problems for Ordinary Differential Equations in Matlab with bvp4c. [https://classes.engineering.wustl.edu/che512/bvp\\_paper.pdf](https://classes.engineering.wustl.edu/che512/bvp_paper.pdf). [Diakses pada 24 Juni 2020].
- Strong, D. M. (2005a). Iterative Methods for Solving  $Ax = b$  - gauss-seidel method. <https://www.maa.org/book/export/html/115849>. [Diakses pada 22 Februari 2020].
- Strong, D. M. (2005b). Iterative Methods for Solving  $Ax = b$  - jacobi's method. <https://www.maa.org/book/export/html/115848>. [Diakses pada 22 Februari 2020].

