

# Bab 3

## Solusi Persamaan Nirlanjar

---

Saya tidak tahu bagaimana saya tampak pada dunia; tetapi bagi saya sendiri saya nampaknya hanyalah seperti seorang anak laki-laki yang bermain-main di pantai, dan mengalihkan diri sendiri sekarang dan kemudian menemukan koral yang lebih halus atau kerang yang lebih indah daripada yang biasa, sementara samudera besar kebenaran semuanya terbentang di hadapan saya tak terungkap.  
(Isaac Newton)

Dalam bidang sains dan rekayasa, para ahli ilmu alam dan rekayasawan sering berhadapan dengan persoalan mencari solusi persamaan –lazim disebut **akar persamaan** (*roots of equation*) atau **nilai-nilai nol**– yang berbentuk  $f(x) = 0$ . Beberapa persamaan sederhana mudah ditemukan akarnya. Misalnya  $2x - 3 = 0$ , pemecahannya adalah dengan memindahkan  $-3$  ke ruas kanan sehingga menjadi  $2x = 3$ , dengan demikian solusi atau akarnya adalah  $x = 3/2$ . Begitu juga persamaan kuadrat seperti  $x^2 - 4x - 5 = 0$ , akar-akarnya mudah ditemukan dengan cara pemfaktoran menjadi  $(x - 5)(x + 1) = 0$  sehingga  $x_1 = 5$  dan  $x_2 = -1$ .

Umumnya persamaan yang akan dipecahkan muncul dalam bentuk nirlanjar (*non linear*) yang melibatkan bentuk sinus, cosinus, eksponensial, logaritma, dan fungsi transenden lainnya. Misalnya,

1. Tentukan akar riil terkecil dari

$$9.34 - 21.97x + 16.3x^3 - 3.704x^5 = 0$$

2. Kecepatan ke atas sebuah roket dapat dihitung dengan memakai rumus berikut:

$$v = u \ln \left| \frac{m_0}{m_0 - qt} \right| - gt$$

yang dalam hal ini  $v$  adalah kecepatan ke atas,  $u$  adalah kecepatan pada saat bahan bakar dikeluarkan relatif terhadap roket,  $m_0$  massa awal roket pada saat  $t = 0$ ,  $q$  laju pemakaian bahan bakar, dan  $g$  percepatan gravitasi ( $= 9.8 \text{ m/det}^2$ ). Jika  $u = 2200 \text{ m/det}$ ,  $m_0 = 160000 \text{ kg}$ , dan  $q = 2680 \text{ kg/det}$ , hitunglah waktu saat  $v = 1000 \text{ m/det}$ . (Nyatakan persamaan dengan ruas kanan sama dengan 0:

$$v - u \ln[m_0/(m_0 - qt)] + gt = 0 \quad )$$

3. Dalam teknik kelautan, persamaan gelombang berdiri yang dipantulkan oleh dermaga pelabuhan diberikan oleh

$$h = h_0 \{ \sin(2px/I) \cos(2ptv/I) + e^{-x} \}$$

Tentukan  $x$  jika  $h = 0.5h_0$ ,  $I = 20$ ,  $t = 10$  dan  $v = 50$ !

(Nyatakan persamaan dengan ruas kanan sama dengan 0:

$$h - h_0 \{ \sin(2px/I) \cos(2ptv/I) + e^{-x} \} = 0 \quad )$$

4. Suatu arus osilasi dalam rangkaian listrik diberikan oleh

$$I = 10e^{-t} \sin(2pt)$$

yang dalam hal ini  $t$  dalam detik. Tentukan semua nilai  $t$  sedemikian sehingga  $I = 2$  ampere.

(Nyatakan persamaan dengan ruas kanan sama dengan 0:

$$I - 10e^{-t} \sin(2pt) = 0 \quad )$$

5. Dalam bidang teknik lingkungan, persamaan berikut ini dapat digunakan untuk menghitung tingkat oksigen pada hilir sungai dari tempat pembuangan limbah:

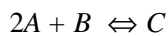
$$c = 10 - 15(e^{-0.1x} - e^{-0.5x})$$

yang dalam hal ini  $x$  adalah jarak hilir sungai ke tempat pembuangan limbah. Tentukan jarak hilir sungai tersebut bila pembacaan pertama pada alat pengukur tingkat oksigen adalah 4 bila pengukur berada 5 mil dari pembuangan.

(Nyatakan persamaan dengan ruas kanan sama dengan 0:

$$c - 10 - 15(e^{-0.1x} - e^{-0.5x}) = 0 \quad )$$

## 6. Reaksi kesetimbangan



dapat dicirikan oleh hubungan setimbang

$$K = \frac{[C]}{[A]^2 [B]}$$

yang dalam hal ini  $[.]$  menyatakan konsentrasi zat kimia. Andaikan bahwa kita mendefinisikan peubah  $x$  sebagai jumlah mol  $C$  yang dihasilkan. Hukum kekekalan massa dapat dipakai untuk merumuskan ulang hubungan keseimbangan itu sebagai

$$K = \frac{[C_0] + x}{([A_0] - 2x)([B_0] - x)}$$

yang dalam hal ini indeks 0 menunjukkan konsentrasi awal tiap unsur. Jika diketahui tetapan kesetimbangan  $K = 1.25 \times 10^{-2}$ , dan konsentrasi larutan  $[A_0] = 50$ ,  $[B_0] = 40$ , dan  $[C_0] = 5$ , hitunglah  $x$ .

(Nyatakan persamaan dengan ruas kanan sama dengan 0:

$$K - \frac{[C_0] + x}{([A_0] - 2x)([B_0] - x)} = 0 \quad )$$

Keenam contoh di atas memperlihatkan bentuk persamaan yang rumit/kompleks yang tidak dapat dipecahkan secara analitik (seperti persamaan kuadrat pada paragraf awal). Bila metode analitik tidak dapat menyelesaikan persamaan, maka kita masih bisa mencari solusinya dengan menggunakan metode numerik.

## 3.1 Rumusan Masalah

Persoalan mencari solusi persamaan nirlanjar dapat dirumuskan secara singkat sebagai berikut: tentukan nilai  $x$  yang memenuhi persamaan

$$f(x) = 0 \tag{P.3.1}$$

yaitu nilai  $x = s$  sedemikian sehingga  $f(s)$  sama dengan nol.

## 3.2 Metode Pencarian Akar

Dalam metode numerik, pencarian akar  $f(x) = 0$  dilakukan secara lelaran (iteratif). Sampai saat ini sudah banyak ditemukan metode pencarian akar. Secara umum, semua metode pencarian akar tersebut dapat dikelompokkan menjadi dua golongan besar:

1. **Metode tertutup** atau metode pengurung (*bracketing method*)

Metode yang termasuk ke dalam golongan ini mencari akar di dalam selang  $[a, b]$ . Selang  $[a, b]$  sudah dipastikan berisi minimal satu buah akar, karena itu metode jenis ini selalu berhasil menemukan akar. Dengan kata lain, lelarannya selalu konvergen (menuju) ke akar, karena itu metode tertutup kadang-kadang dinamakan juga **metode konvergen**.

2. **Metode terbuka**

Berbeda dengan metode tertutup, metode terbuka tidak memerlukan selang  $[a, b]$  yang mengandung akar. Yang diperlukan adalah tebakan (*guess*) awal akar, lalu, dengan prosedur lelaran, kita menggunakannya untuk menghitung hampiran akar yang baru. Pada setiap kali lelaran, hampiran akar yang lama dipakai untuk menghitung hampiran akar yang baru. Mungkin saja hampiran akar yang baru mendekati akar sejati (konvergen), atau mungkin juga menjauhinya (divergen). Karena itu, metode terbuka tidak selalu berhasil menemukan akar, kadang-kadang konvergen, kadangkala ia divergen.

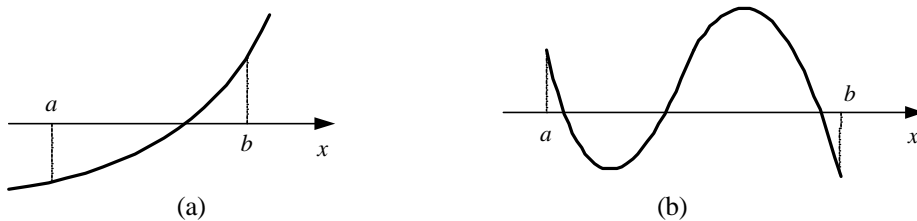
## 3.3 Metode Tertutup

Seperti yang telah dijelaskan, metode tertutup memerlukan selang  $[a, b]$  yang mengandung akar. Sebagaimana namanya, selang tersebut “mengurung” akar sejati. Tata-ancang (*strategy*) yang dipakai adalah mengurangi lebar selang secara sistematis sehingga lebar selang tersebut semakin sempit, dan karenanya menuju akar yang benar.

Dalam sebuah selang mungkin terdapat lebih dari satu buah akar atau tidak ada akar sama sekali. Secara grafik dapat ditunjukkan bahwa jika:

$$(1) f(a)f(b) < 0$$

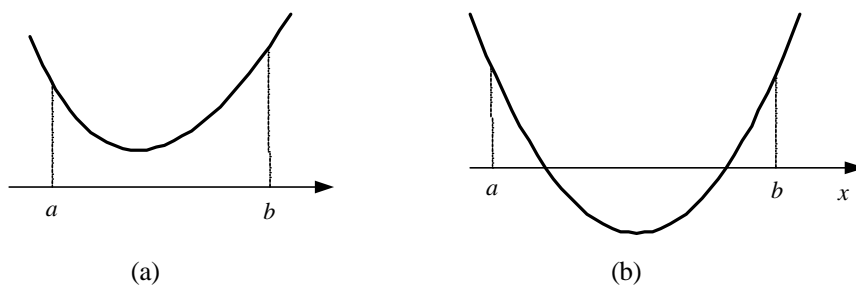
maka terdapat akar sebanyak bilangan ganjil (Gambar 3.1).



**Gambar 3.1** Banyaknya akar ganjil

(2)  $f(a)f(b) > 0$

maka terdapat akar sebanyak bilangan genap atau tidak ada akar sama sekali (Gambar 3.2).



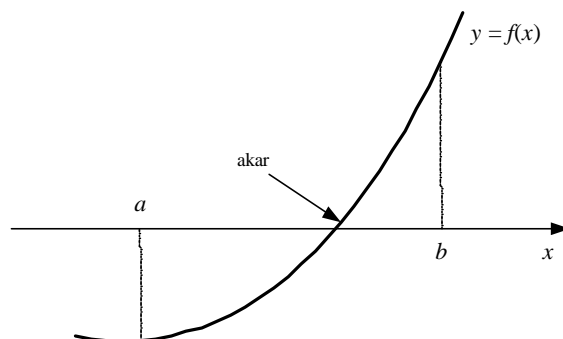
**Gambar 3.2** Banyaknya akar genap

### Syarat Cukup Keberadaan Akar

Gambar 3.1 memperlihatkan bahwa selalu ada akar di dalam selang  $[a, b]$  jika nilai fungsi berbeda tanda (+/-) di  $x = a$  dan  $x = b$ . Tidak demikian halnya jika nilai fungsi di ujung-ujung selang sama tandanya, yang mengisyaratkan mungkin ada akar atau tidak ada sama sekali. Jadi, jika nilai fungsi berbeda tanda di ujung-ujung selang, pastilah terdapat paling sedikit satu buah akar di dalam selang tersebut. Dengan kata lain, syarat cukup keberadaan akar persamaan kita tulis sebagai berikut:

Jika  $f(a)f(b) < 0$  dan  $f(x)$  menerus di dalam selang  $[a, b]$ , maka paling sedikit terdapat satu buah akar persamaan  $f(x) = 0$  di dalam selang  $[a, b]$ .

Syarat ini disebut syarat cukup<sup>1</sup> -bukan syarat perlu- sebab meskipun nilai-nilai di ujung selang tidak berbeda tanda, mungkin saja terdapat akar di dalam selang tersebut (seperti ditunjukkan pada Gambar 3.2). Syarat cukup keberadaan akar ini ditunjukkan pada Gambar 3.3.



**Gambar 3.3** Lokasi akar

Ada dua masalah yang terjadi karena ketidaktepatan mengambil selang  $[a, b]$ . Masalah pertama adalah bila di dalam selang  $[a, b]$  terdapat lebih dari satu buah akar. Sekali suatu metode tertutup digunakan untuk mencari akar di dalam selang  $[a, b]$ , ia hanya menemukan sebuah akar saja. Karena itu, bila kita mengambil selang  $[a, b]$  yang mengandung lebih dari satu akar, hanya satu buah akar saja yang berhasil ditemukan (lihat kembali Gambar 3.1(b)).

Masalah kedua adalah bila mengambil selang  $[a, b]$  yang tidak memenuhi syarat cukup. Adakalanya kita dapat “kehilangan” akar karena selang  $[a, b]$  yang diambil ternyata tidak memenuhi syarat cukup  $f(a)f(b) < 0$ . Sehingga, kita mungkin sampai pada kesimpulan tidak terdapat akar di dalam selang  $[a, b]$  tersebut, padahal seharusnya ada (lihat kembali Gambar 3.2 (b)).

Untuk mengatasi kedua masalah di atas, pengguna metode tertutup disarankan mengambil selang yang berukuran cukup kecil yang memuat hanya satu akar. Ada dua pendekatan yang dapat kita gunakan dalam memilih selang tersebut.

<sup>1</sup> Bentuk implikasi “jika  $p$  maka  $q$ ” bisa dibaca sebagai “ $p$  adalah syarat cukup untuk  $q$ ”. Di dalam kalkulus proposisi, pernyataan “jika  $p$  maka  $q$ ” (dilambangkan dengan  $p \rightarrow q$ ) adalah benar kecuali jika  $p$  benar dan  $q$  salah. Jadi, pernyataan tersebut tetap benar meskipun  $f(a)f(b) > 0$  dan di dalam selang  $[a, b]$  terdapat paling sedikit satu buah akar atau tidak terdapat akar sama sekali. Pernyataan tersebut jelas salah bila  $f(a)f(b) > 0$  dan di dalam selang  $[a, b]$  terdapat paling sedikit satu buah akar (tidak mungkin).

Pendekatan pertama adalah membuat grafik fungsi di bidang  $X$ - $Y$ , lalu melihat di mana perpotongannya dengan sumbu- $X$ . Dari sini kita dapat mengira-ngira selang yang memuat titik potong tersebut. Grafik fungsi dapat dibuat dengan program yang ditulis sendiri, atau lebih praktis menggunakan paket program yang dapat membuat grafik fungsi.

Pendekatan yang kedua adalah dengan mencetak nilai fungsi pada titik-titik absis yang berjarak tetap. Jarak titik ini dapat diatur cukup kecil. Jika tanda fungsi berubah pada sebuah selang, pasti terdapat minimal satu akar di dalamnya. Program 3.1 berisi prosedur untuk menemukan selang yang cukup kecil yang mengandung akar. Program ini mencetak tabel titik-titik sepanjang selang  $[a, b]$ . Dari tabel tersebut kita dapat menentukan upaselang yang nilai fungsi di ujung-ujungnya berbeda tanda. Keberhasilan dari pendekatan ini bergantung pada jarak antara titik-titik absis. Semakin kecil jarak titik absis, semakin besar peluang menemukan selang yang mengandung hanya sebuah akar.

**Program 3.1** Menemukan selang kecil yang mengandung akar

```

procedure Cari_SelangKecilYangMengandungAkar(a, b, h: real);
{ Menentukan dan mencetak nilai-nilai fungsi untuk absis x di dalam
selang [a, b]. Jarak antara tiap absis adalah h.
K.Awal: a dan b adalah ujung-ujung selang, nilainya sudah terdefenisil;
      h adalah jarak antara tiap absis x
K.Akhir: tabel yang berisi x dan f(x) dicetak ke layar
}
var
  x : real;
begin
  x:=a;
  writeln('-----');
  writeln('      x          f(x)  ');
  writeln('-----');
  while x <= b do begin
    writeln(x:5:2, f(x):10:6);
    x:=x+h;
  end;
  { x > b }
  writeln('-----');
end;

```

Bila Program 3.1 digunakan untuk mencari selang kecil yang mengandung akar pada fungsi  $f(x) = e^x - 5x^2$  mulai dari  $a = -0.5$  sampai  $b = 1.4$  dengan kenaikan absis sebesar  $h = 0.1$ , maka hasilnya tampak pada tabel berikut:

x	f(x)
-0.50	-0.643469
-0.40	-0.129680
-0.30	0.290818
-0.20	0.618731
-0.10	0.854837
0.00	1.000000
0.10	1.055171
0.20	1.021403
0.30	0.899859
0.40	0.691825
0.50	0.398721
0.60	0.022119
0.70	-0.436247
0.80	-0.974459
0.90	-1.590397
1.00	-2.281718
1.10	-3.045834
1.20	-3.879883
1.30	-4.780703
1.40	-5.744800

Berdasarkan tabel di atas, selang yang cukup kecil yang mengandung akar adalah

$$[-0.40, -0.30] \text{ dan } [0.60, 0.70]$$

karena nilai fungsi berubah tanda di ujung-ujung selangnya. Selang  $[0.00, 1.00]$  juga dapat kita ambil tetapi cukup lebar, demikian juga  $[-0.50, 1.40]$ ,  $[-0.30, 0.80]$ , dan seterusnya.

Ada dua metode klasik yang termasuk ke dalam metode tertutup, yaitu **metode bagidua** dan **metode regula-falsi**. Masing-masing metode kita bahas lebih rinci di bawah ini.

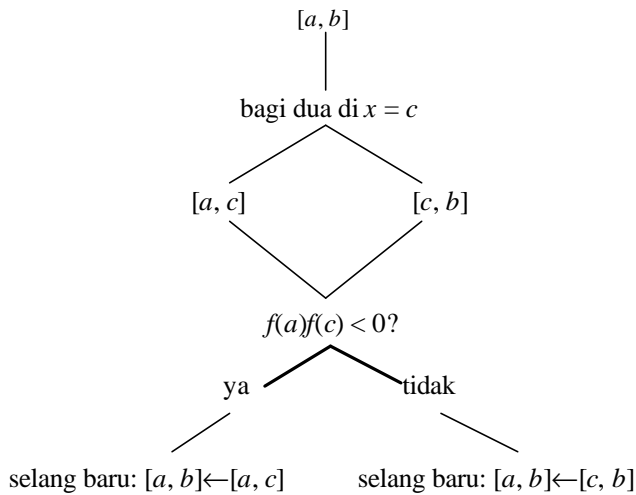
### 3.3.1 Metode Bagidua<sup>2</sup>

Misalkan kita telah menentukan selang  $[a, b]$  sehingga  $f(a)f(b) < 0$ . Pada setiap kali lelaran, selang  $[a, b]$  kita bagi dua di  $x = c$ , sehingga terdapat dua buah upaselang yang berukuran sama, yaitu selang  $[a, c]$  dan  $[c, b]$ . Selang yang diambil untuk lelaran berikutnya adalah upaselang yang memuat akar, bergantung pada apakah  $f(a)f(c) < 0$  atau  $f(c)f(b) < 0$ .

---

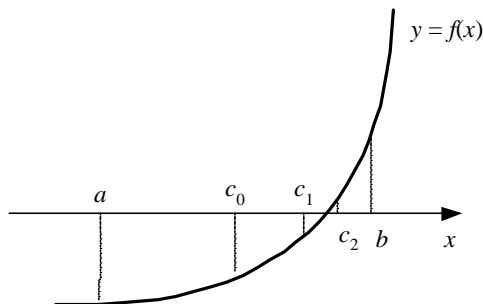
<sup>2</sup>Nama lainnya adalah *metode Bolzano*





Selang yang baru dibagi dua lagi dengan cara yang sama. Begitu seterusnya sampai ukuran selang yang baru sudah sangat kecil (lihat Gambar 3.4). Kondisi berhenti lelaran dapat dipilih salah satu dari tiga kriteria berikut:

1. Lebar selang baru:  $|a - b| < \epsilon$ , yang dalam hal ini  $\epsilon$  adalah nilai toleransi lebar selang yang mengurung akar.
2. Nilai fungsi di hampiran akar:  $f(c) = 0$ . Beberapa bahasa pemrograman membolehkan perbandingan dua buah bilangan riil, sehingga perbandingan  $f(c) = 0$  dibenarkan. Namun kalau kita kembali ke konsep awal bahwa dua buah bilangan riil tidak dapat dibandingkan kesamaannya karena representasinya di dalam mesin tidak tepat, maka kita dapat menggunakan bilangan yang sangat kecil (misalnya epsilon mesin) sebagai pengganti nilai 0. Dengan demikian, menguji kesamaan  $f(c) = 0$  dapat kita hampiri dengan  $f(c) < \text{epsilon\_mesin}$ .
3. Galat relatif hampiran akar:  $|(c_{\text{baru}} - c_{\text{lama}})/c_{\text{baru}}| < d$ , yang dalam hal ini  $d$  adalah galat relatif hampiran yang diinginkan.



**Gambar 3.4** Proses pembagian selang  $[a, b]$  dengan metode bagidua

Program 3.2 berisi algoritma metode bagidua. Di dalam algoritma tersebut, format penulisan keluaran tidak dituliskan untuk menghindari kerumitan algoritma dari hal-hal yang tidak esensial.

**Program 3.2** Metode bagidua

```

procedure BagiDua(a,b: real);
{ Mencari akar  $f(x)=0$  di dalam selang  $[a,b]$  dengan metode bagidua
  K.Awal : a dan b adalah ujung-ujung selang sehingga  $f(a)*f(b) < 0$ ,
           nilai a dan b sudah terdefinisi.
  K.Akhir : Hampiran akar tercetak di layar.
}
const
  epsilon1 = 0.000001;      {batas lebar selang akhir lelaran}
  epsilon2 = 0.00000001;   {bilangan yang sangat kecil, mendekati nol}
begin
  repeat
    c:=(a+b)/2;      { titik tengah  $[a,b]$ }
    if f(a)*f(c) < 0 then
      b:=c           {selang baru  $[a,b]=[a,c]$ }
    else
      a:=c;          {selang baru  $[a,b]=[c,b]$ }
  until (ABS(a-b)< epsilon1) or (f(c)) < epsilon2);
  { c adalah akar persamaan }
  writeln('Hampiran kar = ', x:10:6);
end;

```

### Kasus yang Mungkin Terjadi pada Penggunaan Metode Bagidua

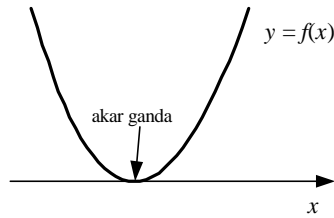
#### 1. Jumlah akar lebih dari satu

Bila dalam selang  $[a, b]$  terdapat lebih dari satu akar (banyaknya akar ganjil), hanya satu buah akar yang dapat ditemukan (lihat kembali Gambar 3.1(b)). Cara mengatasinya: gunakan selang  $[a,b]$  yang cukup kecil yang memuat hanya satu buah akar.

#### 2. Akar ganda.

Metode bagidua tidak berhasil menemukan akar ganda. Hal ini disebabkan karena tidak terdapat perbedaan tanda di ujung-ujung selang yang baru (Gambar 3.5).

Contoh:  $f(x) = (x - 3)^2 = (x - 3)(x - 3)$ , mempunyai dua akar yang sama, yaitu  $x = 3$ .

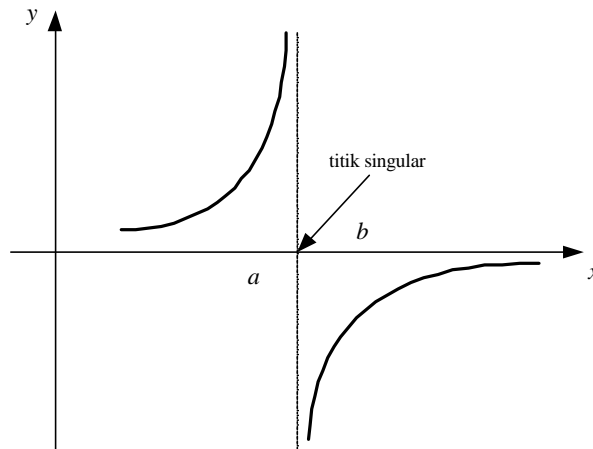


**Gambar 3.4** Akar ganda

Cara mengatasinya: akan dibahas pada upabab 3.5.

### 3. Singularitas.

Pada titik singular, nilai fungsinya tidak terdefinisi. Bila selang  $[a, b]$  mengandung titik singular, lelaran metode bagidua tidak pernah berhenti. Penyebabnya, metode bagidua menganggap titik singular sebagai akar karena lelaran cenderung konvergen. Yang sebenarnya, titik singular bukanlah akar, melainkan *akar semu* (Gambar 3.6)



**Gambar 3.6** Fungsi singular

Cara mengatasinya: periksa nilai  $|f(b) - f(a)|$ . Jika  $|f(b) - f(a)|$  konvergen ke nol, akar yang dicari pasti akar sejati, tetapi jika  $|f(b) - f(a)|$  divergen, akar yang dicari merupakan titik singular (akar semu).

Pada setiap lelaran pada metode bagidua, kita mencatat bahwa selisih antara akar sejati dengan akar hampiran tidak pernah melebihi setengah panjang selang saat itu. Pernyataan ini dinyatakan dengan teorema berikut.

**TEOREMA 3.1.** Jika  $f(x)$  menerus di dalam selang  $[a, b]$  dengan  $f(a)f(b) < 0$  dan  $s \in [a, b]$  sehingga  $f(s) = 0$  dan  $c_r = (a_r + b_r)/2$ , maka selalu berlaku dua ketidaksamaan berikut:

$$(i) \quad |s - c_r| \leq |b_r - a_r|/2$$

dan

$$(ii) \quad |s - c_r| \leq \frac{|b - a|}{2^{r+1}}, \quad r = 0, 1, 2, \dots$$

**Bukti:**

Misalkan pada lelaran ke- $r$  kita mendapatkan selang  $[a_r, b_r]$  yang panjangnya setengah panjang selang sebelumnya,  $[a_{r-1}, b_{r-1}]$ .

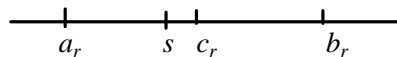
Jadi,

$$|b_r - a_r| = |b_{r-1} - a_{r-1}|/2$$

Jelaslah bahwa

$$\begin{aligned} |b_1 - a_1| &= |b_0 - a_0|/2 = |b - a|/2 \\ |b_2 - a_2| &= |b_1 - a_1|/2 = |b - a|/2^2 \\ |b_3 - a_3| &= |b_2 - a_2|/2 = |b - a|/2^3 \\ &\dots \\ |b_r - a_r| &= |b_{r-1} - a_{r-1}|/2 = |b - a|/2^r \end{aligned}$$

Pada lelaran ke- $r$ , posisi  $c_r$  (akar hampiran) dan  $s$  (akar sejati) adalah seperti diagram berikut:



Berdasarkan diagram di atas jelaslah bahwa

$$|s - c_r| \leq \frac{|b_r - a_r|}{2}$$

Selanjutnya,

$$|s - c_r| \leq \frac{|b_r - a_r|}{2} = \frac{1}{2} \frac{|b - a|}{2^r} = \frac{|b - a|}{2^{r+1}} \quad \blacksquare$$

Jadi, selisih antara akar sejati dengan akar hampiran tidak pernah lebih dari setengah epsilon.

Dengan mengingat kriteria berhenti adalah  $|b_r - a_r| < \epsilon$ , maka dari (i) terlihat bahwa

$$|s - c_r| < a / 2$$

sehingga

$$\frac{|b - a|}{2^{r+1}} < \frac{\epsilon}{2}$$

$$\Leftrightarrow 2^r > |b - a| / \epsilon$$

$$\Leftrightarrow r \ln(2) > \ln(|b - a|) - \ln(\epsilon) \quad \text{ket: } \ln \text{ adalah logaritma natural}$$

$$\Leftrightarrow r > \frac{\ln(|b - a|) - \ln(\epsilon)}{\ln(2)}$$

$$\Leftrightarrow R > \frac{\ln(|b - a|) - \ln(\epsilon)}{\ln(2)}$$

yang dalam hal ini  $R$  adalah jumlah lelaran (jumlah pembagian selang) yang dibutuhkan untuk menjamin bahwa  $c$  adalah hampiran akar yang memiliki galat kurang dari  $\epsilon$ .

### Contoh 3.1

Temukan akar  $f(x) = e^x - 5x^2$  di dalam selang  $[0, 1]$  dan  $\epsilon = 0.00001$ .

**Penyelesaian:** Tabel lelaran menggunakan metode bagidua:

$r$	$a$	$c$	$b$	$f(a)$	$f(c)$	$f(b)$	Selang baru	Lebar nya
0	0.000000	0.500000	1.000000	1.000000	0.398721	-2.281718	[c, b]	0.500000
1	0.500000	0.750000	1.000000	0.398721	-0.695500	-2.281718	[a, c]	0.250000
2	0.500000	0.625000	0.750000	0.398721	-0.084879	-0.695500	[a, c]	0.125000
3	0.500000	0.562500	0.625000	0.398721	0.173023	-0.084879	[c, b]	0.062500
4	0.562500	0.593750	0.625000	0.173023	0.048071	-0.084879	[c, b]	0.031250
5	0.593750	0.609375	0.625000	0.048071	-0.017408	-0.084879	[a, c]	0.015625
6	0.593750	0.601563	0.609375	0.048071	0.015581	-0.017408	[c, b]	0.007813
7	0.601563	0.605469	0.609375	0.015581	-0.000851	-0.017408	[a, c]	0.003906
8	0.601563	0.603516	0.605469	0.015581	0.007380	-0.000851	[c, b]	0.001953
9	0.603516	0.604492	0.605469	0.007380	0.003268	-0.000851	[c, b]	0.000977

10	0.604492	0.604980	0.605469	0.003268	0.001210	-0.000851	[c, b]	0.000488
11	0.604980	0.605225	0.605469	0.001210	0.000179	-0.000851	[c, b]	0.000244
12	0.605225	0.605347	0.605469	0.000179	-0.000336	-0.000851	[a, c]	0.000122
13	0.605225	0.605286	0.605347	0.000179	-0.000078	-0.000336	[a, c]	0.000061
14	0.605225	0.605255	0.605286	0.000179	0.000051	-0.000078	[c, b]	0.000031
15	0.605255	0.605270	0.605286	0.000051	-0.000014	-0.000078	[a, c]	0.000015
16	0.605255	0.605263	0.605270	0.000051	0.000018	-0.000014	[c, b]	0.000008

Jadi, hampiran akarnya adalah  $x = 0.605263$

Jumlah lelaran yang dibutuhkan

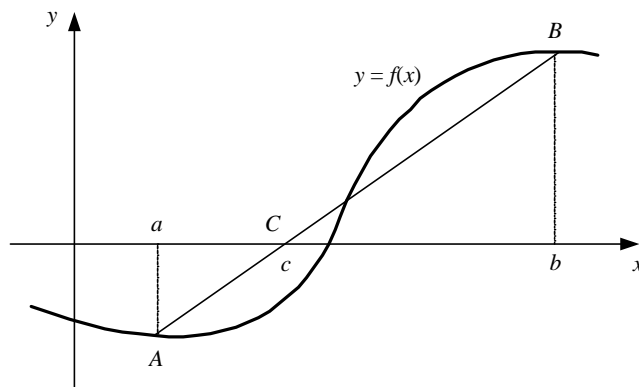
$$R > \frac{\ln(|1 - 0|) - \ln(0.00001)}{\ln(2)}$$

$$> 16.60964$$

Jadi, dibutuhkan minimal 17 kali lelaran ( $r=0$  sampai dengan  $r=16$ ), sesuai dengan jumlah lelaran pada tabel, agar galat akar hampiran kurang dari  $e$ . ■

### 3.3.2 Metode Regula-Falsi

Meskipun metode bagidua selalu berhasil menemukan akar, tetapi kecepatan konvergensinya sangat lambat. Kecepatan konvergensi dapat ditingkatkan bila nilai  $f(a)$  dan  $f(b)$  juga turut diperhitungkan. Logikanya, bila  $f(a)$  lebih dekat ke nol daripada  $f(b)$  tentu akar lebih dekat ke  $x = a$  daripada ke  $x = b$ . Metode yang memanfaatkan nilai  $f(a)$  dan  $f(b)$  ini adalah **metode regula-falsi** (bahasa Latin) atau **metode posisi palsu**. (*false position method*). Dengan metode regula-falsi, dibuat garis lurus yang menghubungkan titik  $(a, f(a))$  dan  $(b, f(b))$ . Perpotongan garis tersebut dengan sumbu- $x$  merupakan taksiran akar yang diperbaiki. Garis lurus tadi seolah-olah berlaku menggantikan kurva  $f(x)$  dan memberikan posisi palsu dari akar.



Gambar 3.7 Metode regula-falsi

Perhatikan Gambar 3.7:

gradien garis AB = gradien garis BC

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

yang dapat disederhanakan menjadi

$$c = b - \frac{f(b)(b-a)}{f(b)-f(a)} \quad (\text{P.3.2})$$

Algoritma regula-falsi (lihat Program 3.3) hampir sama dengan algoritma bagidua kecuali pada perhitungan nilai  $c$ .

**Program 3.3** Metode regula-falsi

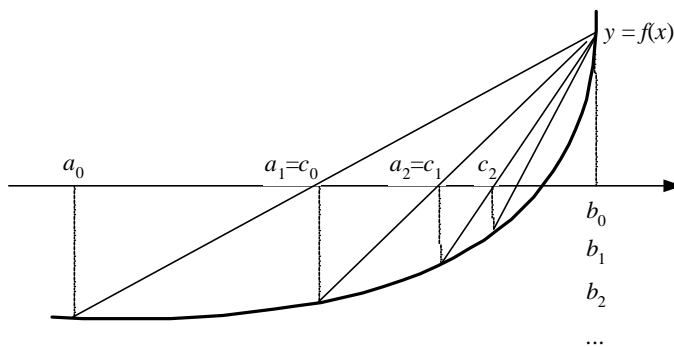
```
procedure regula_falsi(a, b: real);
{ Mencari akar f(x)=0 di dalam selang [a,b] dengan metode regulafalsi
  K.Awal : a dan b adalah ujung-ujung selang sehingga f(a)*f(b) < 0,
           harga a dan b sudah terdefenisi
  K.Akhir : Hampiran akar tercetak di layar
}
const
  epsilon1 = 0.00001;           {batas lebar selang akhir lelaran}
  epsilon2 = 0.000001;        {bilangan yang sangat kecil, bisa diganti }
begin
  repeat
    c:=b-(f(b)*(b-a)/(f(b)-f(a)));
    if abs(f(c))< epsilon2 then   {f(c) = 0, c adalah akar}
      begin
        a:=c;
        b:=c;
      end
    else
      if f(a)*f(c) < 0 then
        b:=c; {selang baru [a,b]=[a,c]}
      else
        a:=c; {selang baru [a,b]=[c,b]}
      until ABS(a-b)< epsilon1;
      { c adalah hampiran akar }
      writeln('Hampiran akar : ', c:10:6);
    end;
```

Secara umum, metode regula-falsi lebih cepat konvergensinya dibandingkan dengan metode bagidua. Namun, pada beberapa kasus kecepatan konvergensinya justru lebih lambat. Bila kita memakai Program 3.4 untuk menghitung akar  $f(x) = e^x - 5x^2$  di dalam selang  $[0, 1]$  dan  $\varepsilon = 0.00001$ , maka tabel lelarannya yang dihasilkan adalah sebagai berikut:

$r$	$a$	$c$	$b$	$f(a)$	$f(c)$	$f(b)$	Selang baru	Lebar nya
0	0.000000	0.304718	1.000000	1.000000	0.891976	-2.281718	[c,b]	0.695282
1	0.304718	0.500129	1.000000	0.891976	0.398287	-2.281718	[c,b]	0.499871
2	0.500129	0.574417	1.000000	0.398287	0.126319	-2.281718	[c,b]	0.425583
3	0.574417	0.596742	1.000000	0.126319	0.035686	-2.281718	[c,b]	0.403258
4	0.596742	0.602952	1.000000	0.035686	0.009750	-2.281718	[c,b]	0.397048
5	0.602952	0.604641	1.000000	0.009750	0.002639	-2.281718	[c,b]	0.395359
6	0.604641	0.605098	1.000000	0.002639	0.000713	-2.281718	[c,b]	0.394902
7	0.605098	0.605222	1.000000	0.000713	0.000192	-2.281718	[c,b]	0.394778
8	0.605222	0.605255	1.000000	0.000192	0.000052	-2.281718	[c,b]	0.394745
9	0.605255	0.605264	1.000000	0.000052	0.000014	-2.281718	[c,b]	0.394736
10	0.605264	0.605266	1.000000	0.000014	0.000004	-2.281718	[c,b]	0.394734
11	0.605266	0.605267	1.000000	0.000004	0.000001	-2.281718	[c,b]	0.394733
12	0.605267	0.605267	1.000000	0.000001	0.000000	-2.281718	[c,b]	0.394733
13	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
14	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
15	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
16	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
17	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
18	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
19	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
20	0.605267	0.605267	1.000000	0.000000	0.000000	-2.281718	[c,b]	0.394733
21	0.605267	0.605267	1.000000	0.000000	-0.000000	-2.281718	[a,c]	0.000000

Hampiran akar  $x = 0.605267$

Jumlah lelaran tabel di atas = 22, lebih banyak daripada jumlah lelaran metode bagidua. Bila diperhatikan, dari lelaran 12 sampai lelaran 21, nilai  $a$ ,  $b$ ,  $c$  tidak pernah berubah, padahal  $f(c)$  sudah sangat kecil ( $\approx 0$ ). Kasus seperti ini akan terjadi bila kurva fungsinya cekung (konkaf) di dalam selang  $[a, b]$ . Akibatnya, garis potongnya selalu terletak di atas kurva (bila kurvanya cekung ke atas) atau selalu terletak di bawah kurva (bila kurvanya cekung ke bawah). Perhatikan Gambar 3.8.



Gambar 3.8 Garis potong selalu terletak di atas kurva  $y = f(x)$



Pada kondisi yang paling ekstrim,  $|b - a_r|$  tidak pernah lebih kecil dari  $\epsilon$ , sebab salah satu titik ujung selang, dalam hal ini  $b$ , selalu tetap untuk setiap lelaran  $r = 0, 1, 2, \dots$ . Titik ujung selang yang tidak pernah berubah itu dinamakan **titik mandek** (*stagnant point*). Pada titik mandek,

$$|b_r - a_r| = |b - a_r| \quad r = 0, 1, 2, \dots$$

yang dapat mengakibatkan program mengalami *looping*. Untuk mengatasi hal ini, kondisi berhenti pada algoritma regula-falsi harus kita tambah dengan memeriksa apakah nilai  $f(c)$  sudah sangat kecil sehingga mendekati nol. Jadi, kondisi pada `repeat-until` menjadi

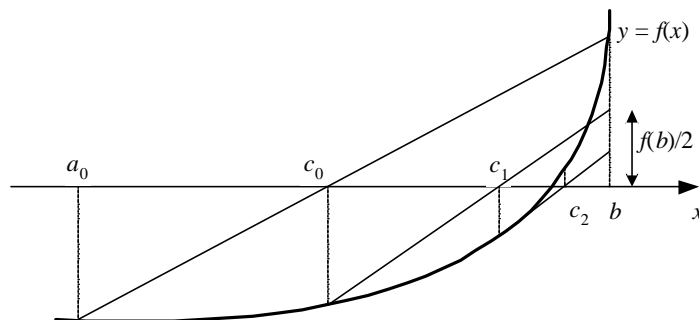
```
until (ABS(a-b) < epsilon1) or (ABS(f(c)) < epsilon2)
```

Bila perubahan ini diterapkan pada soal pencarian akar di atas dengan `epsilon2 = 0.000001`, lelarannya akan berhenti pada  $r = 12$  dengan akar  $x = 0.605267$ .

### Perbaikan Metode Regula-Falsi

Untuk mengatasi kemungkinan kasus titik mandek, metode regula-falsi kemudian diperbaiki (*modified false position method*). Caranya, pada akhir lelaran  $r = 0$ , kita sudah memperoleh selang baru akan dipakai pada lelaran  $r = 1$ . Berdasarkan selang baru tersebut, tentukan titik ujung selang yang tidak berubah (jumlah perulangan  $> 1$ ) - yang kemudian menjadi titik mandek. Nilai  $f$  pada titik mandek itu diganti menjadi setengah kalinya, yang akan dipakai pada lelaran  $r = 1$ .

Misalkan fungsi  $f(x)$  cekung ke atas di dalam selang  $[a, b]$  seperti yang ditunjukkan pada Gambar 3.9.



**Gambar 3.9** Perbaikan metode regula-falsi

Setelah menghitung nilai  $c_0$  pada lelaran  $r = 0$ , ujung selang  $b$  untuk lelaran  $r = 1$  tidak berubah. Titik  $b$  menjadi titik mandek. Karena itu, untuk lelaran  $r = 1$ , nilai  $f(b)$  yang dipakai adalah  $f(b)/2$ . Begitu juga untuk lelaran  $r = 2$ , nilai  $f(b)$  yang dipakai adalah setengah dari nilai  $f(b)$  sebelumnya. Pada akhir lelaran  $r = 2$ ,  $c_2$  sudah terletak di bawah kurva  $y = f(x)$ . Selang yang dipakai selanjutnya adalah  $[c_1, c_2]$ . Dengan cara ini kita dapat menghilangkan titik mandek yang berkepanjangan. Program 3.3 kita modifikasi menjadi Program 3.4.

**Program 3.4** Metode regula-falsi yang diperbaiki

```

procedure perbaikan_regula_falsi(a, b: real);
{ Mencari akar  $f(x)=0$  di dalam selang  $[a,b]$  dengan metode regula-falsi
yang diperbaiki
  K.Awal :  $a$  dan  $b$  adalah ujung-ujung selang sehingga  $f(a)*f(b) < 0$ ,
          harga  $a$  dan  $b$  sudah terdefinisi
  K.Akhir : akar persamaan tercetak di layar
}
const
  epsilon1 = 0.00001; {batas lebar selang akhir lelaran}
  epsilon2 = 0.000001; {batas galat nilai fungsi di hampiran akar}
var
  FA, FB, simpan : real;
  mandek_kiri, mandek_kanan : integer; {jumlah perulangan titik
                                         ujung selang}
begin
  FA:=f(a); FB:=f(b);
  mandek_kiri:=1; mandek_kanan:=1;
  repeat
    c:=b-(FB*(b-a)/(FB-FA));
    if abs(f(c)) < epsilon2 then {f(c) = 0, c adalah akar}
      begin
        a:=c;
        b:=c;
      end
    else
      begin
        if f(a)*f(c) < 0 then
          begin
            b:=c {selang baru [a,b]=[a,c]}
            FB:=f(c);
            mandek_kiri:=mandek_kiri + 1;
            mandek_kanan:=0;
            if mandek_kiri > 1 then
              FA:=FA/2; {a menjadi titik mandek }
            end
          else
            begin
              a:=c; {selang baru [a,b]=[c,b]}
              FA:=f(c);
              mandek_kanan:=mandek_kanan + 1;
              mandek_kiri:=0;
              if mandek_kanan > 1 then
                FB:=FB/2; {b menjadi titik mandek}
              end
            end
          end
        end
      end
  end;

```

```

end;
until (ABS(a-b)< epsilon1) OR (ABS(f(c)) < epsilon2);
{ c adalah taksiran akar }
writeln('Hampiran akar : ', c:10:6);
end;

```

Tabel lelaran dari Program 3.4 untuk menghitung akar  $f(x) = e^x - 5x^2$  di dalam selang  $[0, 1]$  dengan  $\epsilon = 0.00001$  dan  $\delta = 0.000001$  adalah sebagai berikut:

$r$	$a$	$c$	$b$	$f(a)$	$f(c)$	$f(b)$	Selang baru	Lebar nya
0	0.000000	0.304718	1.000000	1.000000	0.891976	-2.281718	[c,b]	0.695282
						(*2)		
1	0.304718	0.609797	1.000000	0.891976	-0.019205	-1.140859	[a,c]	0.305079
2	0.304718	0.603367	0.609797	0.891976	0.008005	-0.019205	[c,b]	0.006430
3	0.603367	0.605259	0.609797	0.008005	0.000035	-0.019205	[c,b]	0.004538
						(*2)		
4	0.605259	0.605275	0.609797	0.000035	-0.000035	-0.009602	[a,c]	0.000017
5	0.605259	0.605267	0.605275	0.000035	0.000000	-0.000035	[c,b]	0.000008

Hampiran akar  $x = 0.605267$

Terlihat bahwa jumlah lelarannya berkurang menjadi sepertiga semula. Harus dicatat bahwa metode regula-falsi yang diperbaiki tetap berlaku untuk fungsi yang tidak cekung sekalipun. Jadi, jika anda memprogram dengan metode regula-falsi, pakailah Program 3.4 ini untuk semua kemungkinan kasus fungsi.

### 3.4 Metode Terbuka

Tidak seperti pada metode tertutup, metode terbuka tidak memerlukan selang yang mengurung akar. Yang diperlukan hanya sebuah tebakan awal akar atau dua buah tebakan yang tidak perlu mengurung akar. Inilah alasan mengapa metodenya dinamakan metode terbuka. Hampiran akar sekarang didasarkan pada hampiran akar sebelumnya melalui prosedur lelaran. Kadangkala lelaran konvergen ke akar sejati, kadangkala ia divergen. Namun, apabila lelarannya konvergen, konvergensinya itu berlangsung sangat cepat dibandingkan dengan metode tertutup.

Yang termasuk ke dalam metode terbuka:

1. Metode lelaran titik-tetap (*fixed-point iteration*)
2. Metode Newton-Raphson
3. Metode *secant*

### 3.4.1. Metode Lelaran Titik-Tetap

Metode ini kadang-kadang dinamakan juga metode lelaran sederhana, metode langsung, atau metode sulih beruntun. Kesederhanaan metode ini karena pembentukan prosedur lelarannya mudah dibentuk sebagai berikut:

Susunlah persamaan  $f(x) = 0$  menjadi bentuk  $x = g(x)$ . Lalu, bentuklah menjadi prosedur lelaran

$$x_{r+1} = g(x_r) \quad (\text{P.3.3})$$

dan terkalah sebuah nilai awal  $x_0$ , lalu hitung nilai  $x_1, x_2, x_3, \dots$ , yang mudah-mudahan konvergen ke akar sejati  $s$  sedemikian sehingga

$$f(s) = 0 \text{ dan } s = g(s).$$

Kondisi berhenti lelaran dinyatakan bila

$$|x_{r+1} - x_r| < \epsilon$$

atau bila menggunakan galat relatif hampiran

$$\left| \frac{x_{r+1} - x_r}{x_{r+1}} \right| < d$$

dengan  $\epsilon$  dan  $d$  telah ditetapkan sebelumnya. Program lelaran titik-tetap ditunjukkan oleh Program 3.5.

**Program 3.5** Metode lelaran titik-tetap

```
procedure lelaran_titik_tetap(x:real);
{ mencari akar f(x) = 0 dengan metode lelaran titik-tetap
  K.Awal : x adalah tebakan awal akar, nilainya sudah terdefinisi
  K.Akhir: akar persamaan tercetak di layar
}
const
  epsilon = 0.000001;
var
  x_sebelumnya: real;
  function g(x:real): real;
  {mengembalikan nilai g(x). Definisikan g(x), , lihat Contoh 3.2 }
begin
  repeat
    x_sebelumnya:=x;
```

```

x:=g(x);
until ABS(x-x_sebelumnya) < epsilon;
{ x adalah hampiran akar }

write('Hampiran akar x = ', x:10:6);
end;

```

Program 3.5 hanya menangani lelaran yang konvergen. Program harus dimodifikasi menjadi Program 3.6 untuk menangani lelaran yang divergen. Salah satu cara penanganannya adalah dengan membatasi jumlah maksimum lelaran (*Nmaks*). Jika jumlah lelaran lebih besar dari *Nmaks*, maka diasumsikan lelarannya divergen.

**Program 3.6** Metode lelaran titik-tetap (dengan penanganan kasus divergen)

```

procedure lelaran_titik_tetap(x:real);
{ mencari akar  $f(x) = 0$  dengan metode lelaran titik-tetap
  K.Awal : x adalah tebakan awal akar, nilainya sudah terdefinisi
  K.Akhir: akar persamaan tercetak di layar
}
const
  epsilon = 0.000001;
  Nmaks = 30;
var
  x_sebelumnya: real; { hampiran nilai akar pada lelaran sebelumnya }
  i : integer;       { pencacah jumlah lelaran }

  function g(x:real): real;
  {mengembalikan nilai  $g(x)$ . Definisikan  $g(x)$  di sini, lihat Contoh 3.2 }
begin
  i:=0;
  repeat
    x_sebelumnya:=x;
    x:=g(x);
    i:=i+1;
  until (ABS(x-x_sebelumnya) < epsilon) or (i > Nmaks);
  { x adalah hampiran akar }

  if i > Nmaks then
    write('Divergen!')
  else
    write('Hampiran akar x = ', x:10:6);
end;

```

### Contoh 3.2

Carilah akar persamaan  $f(x) = x^2 - 2x - 3 = 0$  dengan metode lelaran titik-tetap. Gunakan  $\epsilon = 0.000001$ .

#### Penyelesaian:

Terdapat beberapa kemungkinan prosedur lelaran yang dapat dibentuk.

(a)  $x^2 - 2x - 3 = 0$

$$x^2 = 2x + 3$$

$$x = \sqrt{2x + 3}$$

Dalam hal ini,  $g(x) = \sqrt{2x + 3}$ . Prosedur lelarannya adalah  $x_{r+1} = \sqrt{2x_r + 3}$ .  
Ambil terkaan awal  $x_0=4$

Tabel lelarannya:

$r$	$x_r$	$ x_{r+1} - x_r $
0	4.000000	-
1	3.316625	0.683375
2	3.103748	0.212877
3	3.034385	0.069362
4	3.011440	0.022945
5	3.003811	0.007629
6	3.001270	0.002541
7	3.000423	0.000847
8	3.000141	0.000282
9	3.000047	0.000094
10	3.000016	0.000031
11	3.000005	0.000010
12	3.000002	0.000003
13	3.000001	0.000001
14	3.000000	0.000000

Hampiran akar  $x = 3.000000$

(konvergen monoton)

(b)  $x^2 - 2x - 3 = 0$   
 $x(x-2) = 3$   
 $x = 3/(x - 2)$

Dalam hal ini,  $g(x) = 3/(x - 2)$ . Prosedur lelarannya adalah  $x_{r+1} = 3/(x_r - 2)$ .  
Ambil terkaan awal  $x_0 = 4$

Tabel lelarannya:

$i$	$x_r$	$ x_{r+1} - x_r $
0	4.000000	-
1	1.500000	2.500000
2	-6.000000	7.500000
3	-0.375000	5.625000
4	-1.263158	0.888158
5	-0.919355	0.343803
6	-1.027624	0.108269

7	-0.990876	0.036748
8	-1.003051	0.012175
9	-0.998984	0.004066
10	-1.000339	0.001355
11	-0.999887	0.000452
12	-1.000038	0.000151
13	-0.999987	0.000050
14	-1.000004	0.000017
15	-0.999999	0.000006
16	-1.000000	0.000002
17	-1.000000	0.000001

Hampiran akar  $x = -1.000000$

(konvergen beresilasi)

(c)  $x^2 - 2x - 3 = 0$   
 $x = (x^2 - 3)/2$

Prosedur lelarannya adalah  $x_{r+1} = (x_r^2 - 3)/2$ . Ambil terkaan awal  $x_0=4$

Tabel lelarannya:

$i$	$x_r$	$ x_{r+1} - x_r $
0	4.000000	-
1	6.500000	2.500000
2	19.625000	13.125000
3	191.070313	171.445312
4	18252.432159	18061.361847
	...	

Ternyata lelarannya divergen! ■

### Contoh 3.3

Apa yang terjadi dengan pemilihan beragam nilai  $x_0$  pada pencarian akar persamaan

$$x^3 + 6x - 3 = 0$$

dengan prosedur lelaran

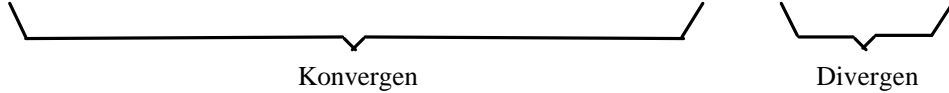
$$x_{r+1} = \frac{-x_r^3 + 3}{6} \quad \text{[PUR84]}$$

Cobakan dengan:  $x_0 = 0.5$ ,  
 $x_0 = 1.5$ ,  
 $x_0 = 2.2$ ,  
 $x_0 = 2.7$

**Penyelesaian:**

Tabel lelarannya adalah sebagai berikut:

$r$	$x_r$	$r$	$x_r$	$r$	$x_r$	$r$	$x_r$
0	0.5	0	1.5	0	2.2	0	2.7
1	0.4791667	1	-0.0625	1	-1.2744667	1	-2.7805
2	0.4816638	2	0.5000407	2	0.8451745	2	4.0827578
3	0.4813757	3	0.4791616	3	0.3993792	3	-10.842521
...	...	4	0.4816644	4	0.4893829	4	212.9416
7	0.4814056	...	...	...	...	5	-16909274.5
8	0.4814056	9	0.4814056	9	0.4814054		
		10	0.4814056	10	0.4814056		
				11	0.4814056		



Terlihat dengan pengambilan  $x_0$  yang cukup dekat ke akar sejati, proses akan konvergen, tetapi jika kita mengambil  $x_0$  terlalu jauh dari akar sejati, ia akan divergen. ■

Kadang-kadang lelaran konvergen, kadang-kadang ia divergen. Adakah suatu “tanda” bagi kita untuk mengetahui kapan suatu lelaran konvergen dan kapan divergen?

**Kriteria konvergensi**

Diberikan prosedur lelaran

$$x_{r+1} = g(x_r) \tag{P.3.4}$$

Misalkan  $x = s$  adalah solusi  $f(x) = 0$  sehingga  $f(s) = 0$  dan  $s = g(s)$ . Selisih antara  $x_{r+1}$  dan  $s$  adalah

$$\begin{aligned} x_{r+1} - s &= g(x_r) - s \\ &= \frac{g(x_r) - s}{(x_r - s)} (x_r - s) \end{aligned} \tag{P.3.5}$$

Terapkan teorema nilai rata-rata pada persamaan (P.3.5) sehingga

$$x_{r+1} - s = g'(t)(x_r - s) \tag{P.3.6}$$



yang dalam hal ini  $x_{r+1} < t < s$ . Misalkan galat pada lelaran ke- $r$  dan lelaran ke- $(r+1)$  adalah

$$\mathbf{e}_r = x_r - s \text{ dan } \mathbf{e}_{r+1} = x_{r+1} - s$$

Persamaan (P.4.6) dapat kita tulis menjadi

$$\mathbf{e}_{r+1} = g'(t) \mathbf{e}_r \tag{P.3.7}$$

atau dalam tanda mutlak

$$|\mathbf{e}_{r+1}| = |g'(t)| |\mathbf{e}_r| \leq K |\mathbf{e}_r|$$

Berapakah batas-batas nilai  $K$  itu?

Misalkan  $x_0$  dan  $x$  berada di dalam selang sejauh  $2h$  dari  $s$ , yaitu  $s - h < x < s + h$ . Jika lelaran konvergen di dalam selang tersebut, yaitu  $x_0, x_1, x_2, x_3, \dots$  menuju  $s$ , maka galat setiap lelaran berkurang. Jadi, haruslah dipenuhi kondisi

$$|\mathbf{e}_{r+1}| \leq K |\mathbf{e}_r| \leq K^2 |\mathbf{e}_{r-1}| \leq K^3 |\mathbf{e}_{r-2}| \leq \dots \leq K^{r+1} |\mathbf{e}_0|$$

Kondisi tersebut hanya berlaku jika

$$g'(x) \leq K < 1$$

Karena  $K < 1$ , maka  $K^{r+1} \rightarrow 0$  untuk  $r \rightarrow \infty$ ; di sini  $|x_{r+1} - s| \rightarrow 0$ .

**TEOREMA 3.2.** Misalkan  $g(x)$  dan  $g'(x)$  menerus di dalam selang  $[a, b] = [s-h, s+h]$  yang mengandung titik tetap  $s$  dan nilai awal  $x_0$  dipilih dalam selang tersebut. Jika  $|g'(x)| < 1$  untuk semua  $x \in [a, b]$  maka lelaran  $x_{r+1} = g(x_r)$  akan konvergen ke  $s$ . Pada kasus ini  $s$  disebut juga *titik atraktif*. Jika  $|g'(x)| > 1$  untuk semua  $x \in [a, b]$  maka lelaran  $x_{r+1} = g(x_r)$  akan divergen dari  $s$ .

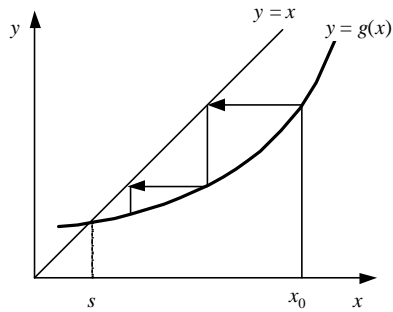
Teorema 3.2 dapat kita sarikan sebagai berikut:

Di dalam selang  $I = [s-h, s+h]$ , dengan  $s$  titik tetap,

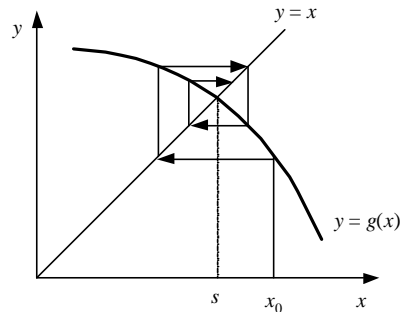
- jika  $0 < g'(x) < 1$  untuk setiap  $x \in I$ , maka lelaran *konvergen monoton*;
- jika  $-1 < g'(x) < 0$  untuk setiap  $x \in I$ , maka lelaran *konvergen bersosilasi*;
- jika  $g'(x) > 1$  untuk setiap  $x \in I$ , maka lelaran *divergen monoton*;
- jika  $g'(x) < -1$  untuk setiap  $x \in I$ , maka lelaran *divergrn bersosilasi*.

Semuanya dirangkum seperti pada Gambar 3.10..

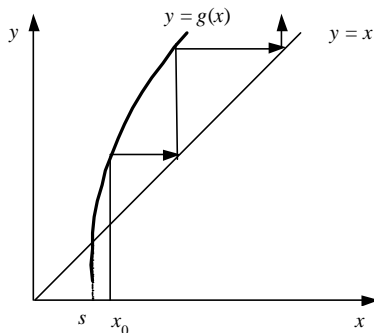
Sebagai catatan, keadaan  $|g'(x)| = 1$  tidak didefinisikan. Catat juga bahwa semakin dekat nilai  $|g'(x)|$  ke nol di dekat akar, semakin cepat kekonvergenan metode lelaran titik-tetap ini [PUR84].



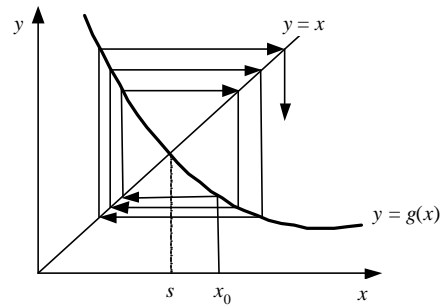
(a) Konvergen monoton:  $0 < g'(x) < 1$



(b) Konvergen berosilasi:  $-1 < g'(x) < 0$



(c) Divergen monoton:  $g'(x) > 1$



(d) Divergen berosilasi:  $g'(x) < -1$

**Gambar 3.10** Jenis-jenis kekonvergenan

Sekarang, mari kita analisis mengapa pencarian akar persamaan  $x^2 - 2x - 3 = 0$  pada Contoh 3.2 dan pencarian akar persamaan  $x^3 + 6x - 3 = 0$  pada Contoh 3.3 dengan bermacam-macam prosedur lelaran dan tebakan awal kadang-kadang konvergen dan kadang-kadang divergen.

(i) Prosedur lelaran pertama:  $x_{r+1} = \sqrt{2x_r + 3}$

$$g(x) = \sqrt{2x + 3}$$

$$g'(x) = \frac{1}{2\sqrt{2x+3}}$$

Terlihat bahwa  $|g'(x)| < 1$  untuk  $x$  di sekitar titik-tetap  $s = 3$ . Karena itu, pengambilan tebakan awal  $x_0 = 4$  akan menghasilkan lelaran yang konvergen sebab  $|g'(4)| = |1/[2\sqrt{(8+3)}]| = 0.1508 < 1$ .

(ii) Prosedur lelaran kedua:  $x_{r+1} = 3/(x_r - 2)$

$$g(x) = 3/(x-2)$$

$$g'(x) = -3/(x-2)^2$$

Terlihat bahwa  $|g'(x)| < 1$  untuk  $x$  di sekitar titik-tetap  $s = 3$ . Karena itu, pengambilan tebakan awal  $x_0 = 4$  akan menghasilkan lelaran yang konvergen sebab

$$|g'(4)| = |-3/(4-2)^2| = 0.75 < 1.$$

(iii) Prosedur lelaran ketiga  $x_{r+1} = (x_r^2 - 3)/2$

$$g(x) = (x^2 - 3)/2$$

$$g'(x) = x$$

Terlihat bahwa  $|g'(x)| > 1$  untuk  $x$  di sekitar titik-tetap  $s = 3$ . Karena itu, pengambilan tebakan awal  $x_0 = 4$  akan menghasilkan lelaran yang divergen sebab

$$|g'(4)| = |4| = 4 > 1.$$

(iv) Prosedur lelaran pada Contoh 3.3:  $x_{r+1} = (-x_r^3 + 3)/6$

$$g(x) = (-x^3 + 3)/6$$

$$g'(x) = -x^2/2$$

Terlihat bahwa  $|g'(x)| < 1$  untuk  $x$  di sekitar titik-tetap  $s = 0.48$ . Pemilihan  $x_0 = 0.5$  akan menjamin lelaran konvergen sebab  $|g'(x_0)| < 1$ . Untuk  $x_0 = 1.5$  dan  $x_0 = 2.2$  memang nilai  $|g'(x_0)| > 1$  tetapi lelarannya masih tetap konvergen, namun  $x_0 = 2.7$  terlalu jauh dari titik-tetap sehingga lelarannya divergen. Dapatkah kita menentukan batas-batas selang yang menjamin prosedur lelaran akan konvergen di dalamnya? Temukan jawabannya pada Contoh 3.4 di bawah ini.

### Contoh 3.4

Pada Contoh 3.3 di atas, tentukan selang sehingga prosedur lelaran

$$x_{r+1} = (-x_r^3 + 3)/6$$

konvergen?

#### Penyelesaian:

$$g(x) = (-x^3 + 3)/6$$

$$g'(x) = -x^2/2$$

Syarat konvergen adalah  $|g'(x)| < 1$ . Jadi,

$$\Leftrightarrow |-x^2/2| < 1$$

$$\Leftrightarrow -1 < -x^2/2 < 1$$

$$\Leftrightarrow 2 > x^2 > -2$$

$$\Leftrightarrow -2 < x^2 < 2$$

Urai satu per satu:

(i)  $x^2 > -2$  { tidak ada  $x$  yang memenuhi}

(ii)  $x^2 < 2$ , dipenuhi oleh

$$\Leftrightarrow x^2 - 2 < 0$$

$$\Leftrightarrow -\sqrt{2} < x < \sqrt{2}$$

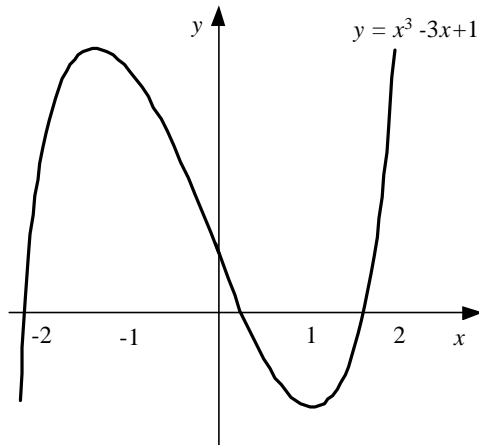
Jadi, prosedur lelaran  $x_{r+1} = (-x_r^3 + 3)/6$  konvergen di dalam selang  $-\sqrt{2} < x < \sqrt{2}$ . Kita dapat memilih  $x_0$  dalam selang tersebut yang menjamin lelaran akan konvergen. ■

### Contoh 3.5

Gunakan metode lelaran titik-tetap untuk mencari akar persamaan

$$x^3 - 3x + 1 \text{ dalam selang } [1, 2] \quad [\text{PUR84}]$$

**Catatan** : selang  $[1, 2]$  ini sebenarnya tidak digunakan dalam proses lelaran sebagaimana halnya pada metode bagidua. Selang ini diberikan untuk memastikan bahwa suatu prosedur lelaran titik-tetap konvergen di dalamnya. Kurva fungsi  $y = x^3 - 3x + 1$  diperlihatkan pada Gambar 3.11.



Gambar 3.11 Kurva  $y = x^3 - 3x + 1$

**Penyelesaian:**

(i)  $x_{r+1} = (x_r^3 + 1)/3$

Tetapi, karena  $|g'(x)| = |x^2| > 1$  dalam selang  $[1, 2]$ , maka prosedur lelaran ini tidak digunakan.

(ii)  $x_{r+1} = -1/(x_r^2 - 3)$

Tetapi, karena  $|g'(x)| = |2x/(x^2 - 3)^3| > 1$  dalam selang  $[1, 2]$ , maka prosedur lelaran ini tidak digunakan.

(iii)  $x_{r+1} = 3/x_r - 1/x_r^2$

Ternyata  $|g'(x)| = |(-3x + 2)/x^3| \leq 1$  di dalam selang  $[1, 2]$ , yaitu,  $g'(x)$  naik dari  $g'(1) = -1$  ke  $g'(2) = -1/2$ . Jadi,  $|g'(x)|$  lebih kecil dari 1 dalam selang  $[1, 2]$ . Dengan mengambil  $x = 1.5$ , prosedur lelarannya konvergen ke akar  $x = 1.5320889$  seperti pada tabel berikut ini.

$r$	$x$
0	1.5
1	1.5555556
2	1.5153061
...	...
43	1.5320888
44	1.5320889
45	1.5320889



Contoh 3.5 menunjukkan bahwa ada dua hal yang mempengaruhi kekonvergenan prosedur lelaran:

1. Bentuk formula  $x_{r+1} = g(x_r)$
2. Pemilihan tebakan awal  $x$

**Catatan:** Meskipun  $|g'(x)| > 1$  menyatakan lelaran divergen, tetapi kita harus hati-hati dengan pernyataan ini. Sebabnya, walaupun  $x_r$  divergen dari suatu akar, runtunan lelarannya mungkin konvergen ke akar yang lain. Kasus seperti ini ditunjukkan pada Contoh 3.6 di bawah ini.

### Contoh 3.6

Tentukan akar persamaan  $f(x) = x^2 - 4x + 3 = 0$  dengan prosedur lelaran

$$x_{r+1} = (x_r^2 + 3)/4$$

#### Penyelesaian:

Jika prosedur lelaran  $x_{r+1} = (x_r^2 + 3)/4$  konvergen ke titik-tetap  $s$ , maka

$$\begin{aligned} \text{limit } x_r &= s \\ r &\rightarrow \infty \end{aligned}$$

sehingga

$$\begin{aligned} s &= (s^2 + 3)/4 \\ s^2 - 4s + 3 &= 0 \\ (s - 3)(s - 1) &= 0 \end{aligned}$$

yang memberikan  $s_1 = 1$  atau  $s_2 = 3$ . Jadi, lelaran konvergen ke akar  $x = 1$  atau akar  $x = 3$ . Dari

$$g(x) = (x^2 + 3)/4$$

diperoleh

$$g'(x) = x/2$$

Gambarkan kurva  $y = x$  dan  $y = (x^2 + 3)/4$  seperti pada Gambar 3.12.

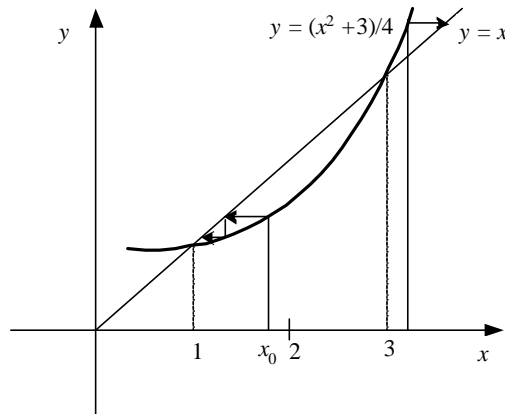
Prosedur lelaran akan konvergen bila

$$\begin{aligned} \Leftrightarrow |g'(x)| &> 1 \\ \Leftrightarrow -1 < x/2 < 1 \end{aligned}$$

atau

$$-2 < x < 2$$

Sehingga pemilihan  $x_0$  dalam selang  $-2 < x < 2$  menjamin lelaran konvergen ke akar  $x = 1$ . Dari Gambar 3.12 terlihat bahwa lelaran juga konvergen ke akar  $x = 1$  untuk pemilihan  $x_0$  dalam selang  $2 < x < 3$ . Padahal, kalau dihitung, dalam selang  $2 < x < 3$ ,  $|g'(x)| > 1$  yang menyatakan bahwa lelarannya divergen. Lelaran divergen dari akar  $x = 3$  tetapi konvergen ke akar  $x = 1$ . ■



**Gambar 3.12** Kurva  $y=x$  dan  $y=(x^2 + 3)/4$

Sebagai contoh terakhir metode lelaran titik-tetap, mari kita hitung akar fungsi pada Contoh 3.1, yaitu  $f(x) = e^x - 5x^2$ .

**Contoh 3.7**

Hitunglah akar  $f(x) = e^x - 5x^2$  dengan metode lelaran titik-tetap. Gunakan  $\epsilon = 0.00001$ . Tebak awal akar  $x_0=1$ .

**Penyelesaian:**

Salah satu prosedur lelaran yang dapat dibuat adalah

$$\begin{aligned}
 e^x - 5x^2 &= 0 \\
 e^x &= 5x^2 \\
 x &= \sqrt{e^x/5} \\
 x_{r+1} &= \text{SQRT}(\text{EXP}(x_r)/5)
 \end{aligned}$$

Tabel lelarannya:

$i$	$x_r$	$ x_{r+1} - x_r $
0	0.500000	-
1	0.574234	0.074234

2	0.595948	0.021714
3	0.602453	0.006506
4	0.604416	0.001963
5	0.605010	0.000593
6	0.605189	0.000180
7	0.605244	0.000054
8	0.605260	0.000016
9	0.605265	0.000005
10	0.605266	0.000002
11	0.605267	0.000000

-----  
 Hampiran akar  $x = 0.605267$  ■

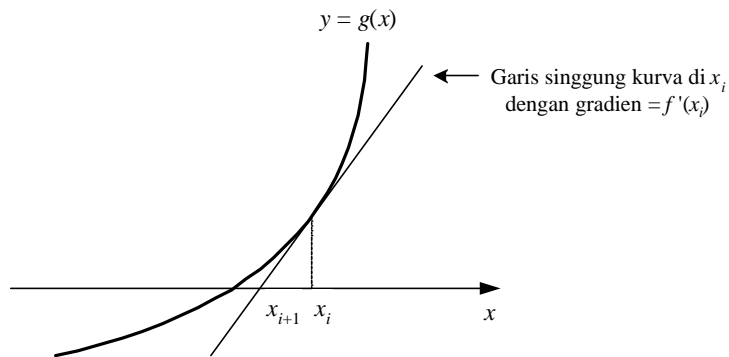
### 3.4.2 Metode Newton-Raphson<sup>3</sup>

Di antara semua metode pencarian akar, metode Newton-Raphsonlah yang paling terkenal dan paling banyak dipakai dalam terapan sains dan rekayasa. Metode ini paling disukai karena konvergensinya paling cepat diantara metode lainnya.

Ada dua pendekatan dalam menurunkan rumus metode Newton-Raphson, yaitu:

- penurunan rumus Newton-Raphson secara geometri,
- penurunan rumus Newton-Raphson dengan bantuan deret Taylor.

#### (a) Penurunan rumus Newton-Raphson secara geometri



Gambar 3.13 Tafsiran geometri metode Newton-Raphson

<sup>3</sup>Beberapa buku menyebutnya *metode Newton* saja. Joseph Raphson (1648 - 1715) adalah matematikawan Inggris yang mempublikasikan metode Newton.



Dari Gambar 3.13, gradien garis singgung di  $x_r$  adalah

$$m = f'(x_r) = \frac{\Delta y}{\Delta x} = \frac{f(x_r) - 0}{x_r - x_{r+1}} \quad (\text{P.3.8})$$

atau

$$f'(x_r) = \frac{f(x_r)}{x_r - x_{r+1}} \quad (\text{P.3.9})$$

sehingga prosedur lelaran metode Newton-Raphson adalah

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}, \quad f'(x_r) \neq 0. \quad (\text{P.3.10})$$

**(b) Penurunan rumus Newton-Raphson dengan bantuan deret Taylor**

Uraikan  $f(x_{r+1})$  di sekitar  $x_r$  ke dalam deret Taylor:

$$f(x_{r+1}) \approx f(x_r) + (x_{r+1} - x_r)f'(x_r) + \frac{(x_{r+1} - x_r)^2}{2} f''(t), \quad x_r < t < x_{r+1} \quad (\text{P.3.11})$$

yang bila dipotong sampai suku orde-2 saja menjadi

$$f(x_{r+1}) \approx f(x_r) + (x_{r+1} - x_r)f'(x_r) \quad (\text{P.3.12})$$

dan karena persoalan mencari akar, maka  $f(x_{r+1}) = 0$ , sehingga

$$0 = f(x_r) + (x_{r+1} - x_r)f'(x_r) \quad (\text{P.3.13})$$

atau

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}, \quad f'(x_r) \neq 0 \quad (\text{P.3.14})$$

yang merupakan rumus metode Newton-Raphson.

Kondisi berhenti lelaran Newton-Raphsin adalah bila

$$|x_{r+1} - x_r| < \mathbf{e}$$

atau bila menggunakan galat relatif hampiran

$$\left| \frac{x_{r+1} - x_r}{x_{r+1}} \right| < \mathbf{d}$$

dengan  $\mathbf{e}$  dan  $\mathbf{d}$  adalah toleransi galat yang diinginkan.

**Catatan:**

1. Jika terjadi  $f'(x_r) = 0$ , ulang kembali perhitungan lelaran dengan  $x_0$  yang lain.
2. Jika persamaan  $f(x) = 0$  memiliki lebih dari satu akar, pemilihan  $x_0$  yang berbeda-beda dapat menemukan akar yang lain.
3. Dapat pula terjadi lelaran konvergen ke akar yang berbeda dari yang diharapkan (seperti halnya pada metode lelaran titik-tetap).

**Program 3.7** Metode Newton-Raphson

```
procedure Newton_Raphson(x:real);
{ Mencari akar persamaan f(x) = 0 dengan metode Newton-Raphson
  K.Awal : x adalah tebakan awal akar, nilainya sudah terdefinisi
  K.Akhir: akar persamaan tercetak di layar
}
const
  epsilon = 0.000001;
var
  x_sebelumnya: real;

  function f(x:real):real;
  { mengembalikan nilai f(x). Definisi f(x) bergantung pada persoalan }

  function f_aksen(x:real):real;
  { mengembalikan nilai f'(x). Definisi f'(x) bergantung
    pada persoalan }

begin
  repeat
    x_sebelumnya:=x;
    x:=x - f(x)/f_aksen(x);
  until (ABS(x-x_sebelumnya) < epsilon)

  { x adalah hampiran akar persamaan }
  write('Hampiran akar x = ', x:10:6);

end;
```

**Catatan:** Program 3.7 ini belum menangani kasus pembagian dengan 0 atau  $\approx 0$  dan kasus divergen. Program 3.8 di bawah ini merupakan modifikasi dari Program 3.7 untuk menangani pembagian dengan 0 dan kasus divergen.

**Program 3.8** Metode Newton-Raphson (dengan penanganan kasus divergen dan pembagian dengan 0)

```

procedure Newton_Raphson(x:real);
{ Mencari akar persamaan  $f(x) = 0$  dengan metode Newton-Raphson
  K.Awal : x adalah tebakan awal akar, nilainya sudah terdefinisi
  K.Akhir: akar persamaan tercetak di layar
}
const
  epsilon1 = 0.000001;      { toleransi galat akar hampiran }
  epsilon2 = 0.000000001;  { toleransi nilai yang hampir 0 }
  Nmaks = 30;              { jumlah maksimum lelaran }
var
  x_sebelumnya: real;
  i : integer;
  berhenti : boolean;      { jika  $f'(x) \lll 0$ , stop ! }

  function f(x:real):real;
  { mengembalikan nilai  $f(x)$ . Definisi  $f(x)$  bergantung pada persoalan }

  function f_aksen(x:real):real;
  { mengembalikan nilai  $f'(x)$ . Definisi  $f'(x)$  bergantung
    pada persoalan }

begin
  i:=0;
  berhenti:=false;
  repeat
    if ABS(f_aksen(x)) < epsilon2 then
      berhenti:=true; { menghindari pembagian bilangan yang » 0}
    else
      begin
        x_sebelumnya:=x;
        x:=x - f(x)/f_aksen(x);
        i:=i+1;
      end;
  until (ABS(x-x_sebelumnya) < epsilon1) or (berhenti) or (i > Nmaks)

  if berhenti then
    writeln('Pembagian dengan bilangan yang hampir 0')
  else
    if i > Nmaks then
      writeln('Divergen')
    else
      { x adalah hampiran akar persamaan }
      write('Hampiran akar x = ', x:10:6);
    endif
  endif
end;

```

**Contoh 3.8**

Hitunglah akar  $f(x) = e^x - 5x^2$  dengan metode Newton-Raphson. Gunakan  $\epsilon = 0.00001$ .  
Tebakan awal akar  $x_0 = 1$ .

**Penyelesaian:**

$$f(x) = e^x - 5x^2$$

$$f'(x) = e^x - 10x$$

Prosedur lelaran Newton-Raphson:

$$x_{r+1} = x_r - \frac{e^x - 5x^2}{e^x - 10x}$$

Tebakan awal  $x_0 = 1$

Tabel lelarannya:

$i$	$x_r$	$ x_{r+1} - x_r $
0	0.500000	-
1	0.618976	0.118976
2	0.605444	0.013532
3	0.605267	0.000177
4	0.605267	0.000000

Hampiran akar  $x = 0.605267$  ■

Contoh 3.8 di atas memperlihatkan bahwa metode Newton-Raphson memerlukan sedikit lelaran, dibandingkan dengan metode bagidua, metode regula falsi, dan metode lelaran titik-tetap. Metode Newton-Raphson sangat berguna untuk menghitung fungsi-fungsi dasar, seperti akar bilangan, nilai  $e$ , arcsin ( $x$ ), dan sebagainya. Contoh 3.9 dan Contoh 3.10 memperlihatkan penggunaan metode Newton-Raphson untuk menghitung akar bilangan dan nilai pecahan.

**Contoh 3.9**

Tentukan bagaimana cara menentukan  $\sqrt{c}$  dengan metode Newton-Raphson.

**Penyelesaian:**

Misalkan  $\sqrt{c} = x$ . Kuadratkan kedua ruas sehingga  $c = x^2 \Leftrightarrow x^2 - c = 0$ .

Di sini  $f(x) = x^2 - c$  dan  $f'(x) = 2x$ . Prosedur lelaran Newton-Raphsonnya adalah

$$x_{r+1} = x_r - \frac{x_r^2 - c}{2x_r} = 0.5(x_r + c/x_r)$$

Untuk  $c = 2$ , dengan memilih  $x_0 = 1$  dan  $\varepsilon = 0.000001$ , kita peroleh

$$\begin{aligned} x_1 &= 1.500000 \\ x_2 &= 1.416667 \\ x_3 &= 1.414216 \\ x_4 &= 1.414214 \end{aligned}$$

Jadi,  $\sqrt{2} \approx 1.414214$  ■

### **Contoh 3.10**

Bagaimana menghitung nilai  $1/c$  dengan metode Newton-Raphson?

#### **Penyelesaian:**

Misalkan  $1/c = x \Leftrightarrow 1/x = c \Leftrightarrow 1/x - c = 0$ . Di sini  $f(x) = 1/x - c$  dan  $f'(x) = -1/x^2$ .

Prosedur Newton-Raphsonnya adalah

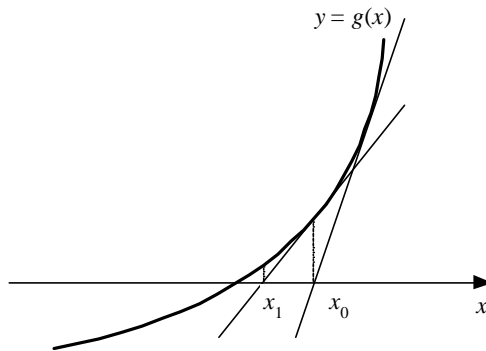
$$x_{r+1} = x_r - \frac{(1/x_r - c)}{-1/x_r^2} = x_r(2 - cx_r)$$

Untuk  $c = 7$ , dengan memilih  $x_0 = 0.2$  dan  $\varepsilon = 0.0000001$ , kita peroleh

$$\begin{aligned} x_1 &= 0.1200000 \\ x_2 &= 0.1392000 \\ x_3 &= 0.1427635 \\ x_4 &= 0.1428570 \\ x_5 &= 0.1428751 \\ x_6 &= 0.1428571 \end{aligned}$$

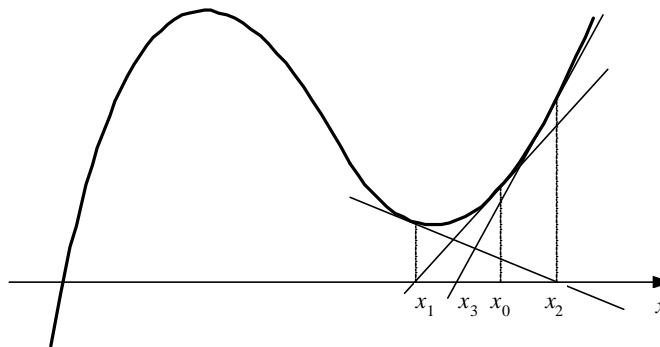
Jadi,  $1/7 \approx 0.1428571$  ■

Secara umum, bila metode Newton-Raphson konvergen, kekonvergenannya itu berlangsung sangat cepat, seperti yang dilukiskan pada Gambar 3.14. Titik potong garis singgung fungsi dengan sumbu- $x$  semakin cepat bergerak mendekati akar sejati.



**Gambar 3.14** Kecepatan konvergensi metode Newton-Raphson sangat cepat

Karena metode Newton-Raphson tergolong metode terbuka, maka dalam beberapa kasus lelarannya mungkin divergen. Bahkan, kalau kurvanya seperti pada Gambar 3.15 serta pemilihan  $x_0$  yang jauh dari akar sejati, lelarannya akan beresilasi di sekitar cekungan **kin**.



**Gambar 3.15** Lelaran metode Newton-Raphson yang divergen

Membuat grafik fungsi sangat membantu dalam pencarian akar. Grafik fungsi dapat memperlihatkan secara visual lokasi akar sejati. Dengan demikian tebakan awal yang bagus untuk akar dapat diturunkan. Pemilihan tebakan awal sebaiknya cukup dekat dengan akar. Selain itu, kita juga dapat mengetahui apakah fungsi tersebut mempunyai akar tidak. Pada kasus tidak ada akar, lelarannya akan divergen beresilasi.

### Kriteria konvergensi metode Newton-Raphson

Apakah persyaratan agar metode Newton-Raphson konvergen? Tinjau kembali bentuk umum prosedur lelaran metode terbuka,

$$x_{r+1} = g(x_r)$$

Karena metode Newton-Raphson termasuk metode terbuka, maka dalam hal ini,

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

Dengan mengingat syarat perlu agar lelaran konvergen adalah  $|g'(x)| < 1$ , maka

$$\begin{aligned} g'(x) &= 1 - \frac{[f'(x)f'(x) - f(x)f''(x)]}{[f'(x)]^2} \\ &= \frac{f(x)f''(x)}{[f'(x)]^2} \end{aligned} \quad (\text{P.3.18})$$

Karena itu, metode Newton-Raphson akan konvergen bila

$$\left| \frac{f(x)f''(x)}{[f'(x)]^2} \right| < 1$$

dengan syarat  $f'(x) \neq 0$ .

### 3.4.3 Orde Konvergensi Metode Terbuka

Prosedur lelaran pada setiap metode terbuka dapat ditulis dalam bentuk

$$x_{r+1} = g(x_r) \quad (\text{P.3.19})$$

misalnya pada metode Newton-Raphson  $g(x_r) = x_r - f(x_r)/f'(x_r)$ . Misalkan  $x_r$  adalah hampiran terhadap akar sejati  $s$  sehingga  $s = g(s)$ . Maka, berdasarkan konsep galat yang sudah dijelaskan di dalam Bab 2,  $s = x_r + \mathbf{e}_r$  dengan  $\mathbf{e}_r$  adalah galat dari  $x_r$ . Uraikan  $g(s)$  di sekitar  $x_r$ :

$$\begin{aligned} g(s) &= g(x_r) + g'(x_r)(s - x_r) + \frac{1}{2} g''(x_r)(s - x_r)^2 + \dots \\ &= g(x_r) + g'(x_r)\mathbf{e}_r + \frac{1}{2} g''(x_r)\mathbf{e}_r^2 + \dots \end{aligned} \quad (\text{P.3.20})$$

Kurangi persamaan (P.3.20) dengan persamaan (P.3.19):

$$\begin{aligned}
g(s) &= g(x_r) + g'(x_r)\mathbf{e}_r + \frac{1}{2} g''(x_r)\mathbf{e}_r^2 + \dots \\
-x_{r+1} &= g(x_r) \\
\hline
g(s) - x_{r+1} &= g'(x_r)\mathbf{e}_r + \frac{1}{2} g''(x_r)\mathbf{e}_r^2 + \dots
\end{aligned}$$

Karena  $g(s) = s$ , maka

$$s - x_{r+1} = g'(x_r)\mathbf{e}_r + \frac{1}{2} g''(x_r)\mathbf{e}_r^2 + \dots$$

Misalkan  $s - x_{r+1} = \mathbf{e}_{r+1}$ , sehingga

$$\mathbf{e}_{r+1} = g'(x_r)\mathbf{e}_r + \frac{1}{2} g''(x_r)\mathbf{e}_r^2 + \dots \quad (\text{P.3.21})$$

Bilangan pangkat dari  $\mathbf{e}_r$  menunjukkan orde (atau laju) konvergensi prosedur lelaran:

(a)  $\mathbf{e}_{r+1} \approx g'(t)\mathbf{e}_r$ ,  $x_r < t < x_{r+1}$  : prosedur lelaran berorde satu  
(P.3.22)

(b)  $\mathbf{e}_{r+1} \approx \frac{1}{2} g''(t)\mathbf{e}_r^2$ ,  $x_r < t < x_{r+1}$  : prosedur lelaran berorde dua (P.3.23)

Metode Newton-Raphson termasuk ke dalam metode terbuka berorde dua. Pernyataan ini kita buktikan di bawah ini.

### Orde konvergensi metode Newton-Raphson

Pada metode Newton-Raphson,  $g(x_r) = x_r - f(x_r) / f'(x_r)$ . Turunan pertama dari  $g(x_r)$  adalah (dari persamaan P.3.18):

$$g'(x_r) = \frac{f(x_r)f''(x_r)}{[f'(x_r)]^2} \quad (\text{P.3.24})$$

Jika  $x_r$  adalah akar persamaan  $f(x) = 0$ , maka  $f(x_r) = 0$ , sehingga

$$g'(x_r) = 0$$

Ini berarti metode Newton-Raphson paling sedikit berorde dua. Turunan kedua dari  $g(x_r)$  adalah

$$g''(x_r) = f''(x_r) / f'(x_r) \quad (\text{P.3.25})$$

Sulihkan (P.3.25) ke dalam (P.3.23):



$$\mathbf{e}_{r+1} = \frac{f''(x_r)\mathbf{e}_r^2}{2f'(x_r)} \quad (\text{P.3.26})$$

Persamaan (P.3.26) ini mempunyai tiga arti:

1. Galat lelaran sekarang sebanding dengan kuadrat galat lelaran sebelumnya. Jika galat lelaran sekarang misalnya 0.001, maka pada lelaran berikutnya galatnya sebanding dengan 0.000001. Hal inilah yang menyebabkan metode Newton-Raphson sangat cepat menemukan akar (jika lelarannya konvergen).
2. Jumlah angka bena akan berlipat dua pada tiap lelaran. Ini merupakan konsekuensi dari hal nomor 1 di atas.
3. Orde konvergensi metode Newton-Raphson adalah kuadratik. sehingga ia dinamakan juga *metode kuadratik*.

Cara lain untuk menemukan orde konvergensi metode Newton-Raphson adalah dengan meneruskan penurunan rumus Newton-Raphson dari deret Taylornya sebagai berikut. Perhatikan kembali persamaan (P.3.11) di atas. Bila  $x_{r+1} = s$  sehingga  $f(x_{r+1}) = f(s) = 0$ , dalam hal ini  $s$  adalah akar sejati, sulihkan  $s$  ke dalam persamaan (P.3.11) di atas:

$$0 = f(x_r) + (s - x_r)f'(x_r) + \frac{(s - x_r)^2 f''(t)}{2} \quad (\text{P.3.27})$$

Kurangi (P.3.27) dengan (P.3.13):

$$\begin{aligned} 0 &= f(x_r) + (s - x_r)f'(x_r) + \frac{(s - x_r)^2 f''(t)}{2} \\ 0 &= f(x_r) + (x_{r+1} - x_r)f'(x_r) && - \\ \hline 0 &= (s - x_{r+1})f'(x_r) + \frac{(s - x_r)^2 f''(t)}{2} \end{aligned} \quad (\text{P.3.28})$$

Misalkan  $s - x_{r+1} = \mathbf{e}_{r+1}$  dan  $s - x_r = \mathbf{e}_r$ , maka persamaan (P.3.28) dapat ditulis menjadi

$$\mathbf{e}_{r+1}f'(x_r) + \frac{\mathbf{e}_r^2 f''(t)}{2} = 0$$

atau

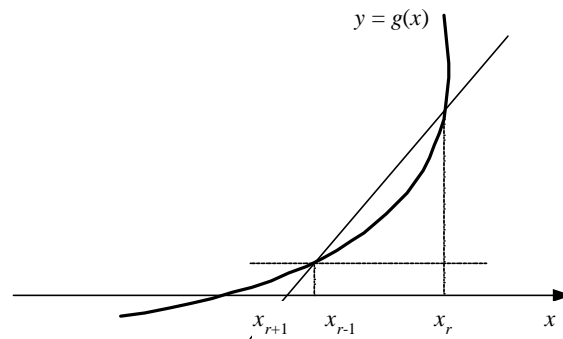
$$e_{r+1} = \frac{f''(t)e_r^2}{2f'(x_r)} \quad (\text{P.3.29})$$

yang sama dengan (P.3.26), kecuali pada  $f''(x_r)$  dan  $f''(t)$ , tetapi perbedaan ini tidak begitu penting, sebab yang dicari adalah pangkat dari  $e_r$ .

Pada proses pencarian akar dengan metode Newton-Raphson, muncul kesulitan jika  $|f'(x)|$  terlalu dekat ke nol, dan kita harus menggunakan bilangan berketelitian ganda untuk memperoleh  $f(x)$  dan  $f'(x)$  cukup teliti [KRE88]. Persamaan nirlanjat  $f(x) = 0$  yang mempunyai kasus seperti ini disebut berkondisi buruk (lihat pembahasan kondisi buruk di dalam Bab 2).

### 3.4.4 Metode Secant

Prosedur lelaran metode Newton-Raphson memerlukan perhitungan turunan fungsi,  $f'(x)$ . Sayangnya, tidak semua fungsi mudah dicari turunannya, terutama fungsi yang bentuknya rumit. Turunan fungsi dapat dihilangkan dengan cara menggantinya dengan bentuk lain yang ekuivalen. Modifikasi metode Newton-Raphson ini dinamakan *metode secant*.



**Gambar 3.16** Metode Secant

Berdasarkan Gambar 3.16, dapat kita hitung gradien

$$f'(x_r) = \frac{\Delta y}{\Delta x} = \frac{AC}{BC} = \frac{f(x_r) - f(x_{r-1})}{x_r - x_{r-1}} \quad (\text{P.3.30})$$

Sulihkan (P.3.30) ke dalam rumus Newton-Raphson:

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}$$

sehingga diperoleh

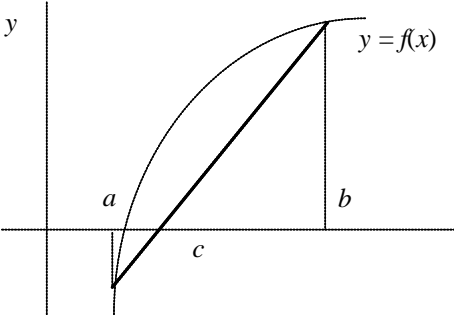
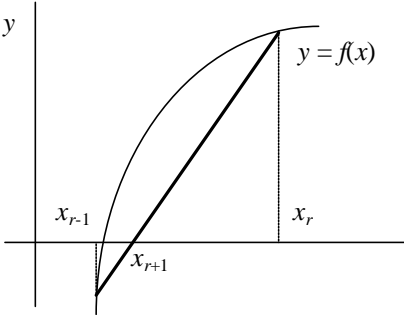
$$x_{r+1} = x_r - \frac{f(x_r)(x_r - x_{r-1})}{f(x_r) - f(x_{r-1})} \quad (\text{P.3.31})$$

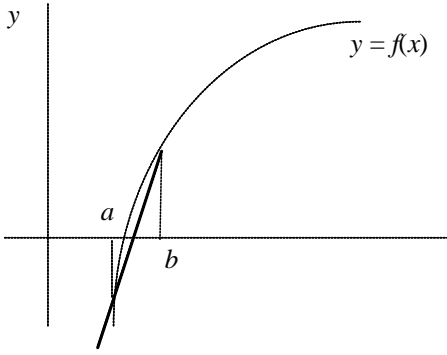
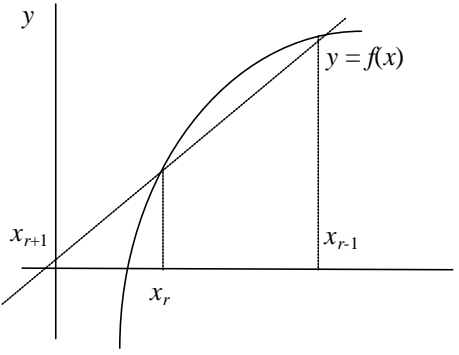
yang merupakan prosedur lelaran metode secant. Dalam hal ini, diperlukan dua buah tebakan awal akar, yaitu  $x_0$  dan  $x_1$ . Kondisi berhenti lelaran adalah bila

$$|x_{r+1} - x_r| < \mathbf{e} \text{ (galat mutlak) atau } \left| \frac{x_{r+1} - x_r}{x_{r+1}} \right| < \mathbf{d} \text{ (galat hampiran)}$$

dengan  $\mathbf{e}$  dan  $\mathbf{d}$  adalah toleransi galat.

Sepintas metode secant mirip dengan metode regula-falsi, namun sesungguhnya prinsip dasar keduanya berbeda, seperti yang dirangkum pada tabel di bawah ini:

Metode Regula Falsi	Metode Secant
1. Diperlukan dua buah nilai awal $a$ dan $b$ (ujung-ujung selang) sedemikian sehingga $f(a)f(b) < 0$ .	1. Diperlukan dua buah nilai awal $x_0$ dan $x_1$ (tebakan awal akar), tetapi <u>tidak harus</u> $f(x_0)f(x_1) < 0$ .
<p>2. <u>Lelaran pertama:</u></p>  <p>Pada lelaran pertama, tidak ada perbedaan antara regula-falsi dan secant. Perbedaan baru muncul pada lelaran kedua.</p>	<p>2. <u>Lelaran pertama:</u></p>  <p>Pada lelaran pertama tidak ada perbedaan antara secant dan regula falsi. Perbedaan baru muncul pada lelaran kedua.</p>

<p><u>Lelaran kedua:</u></p>  <p>Perpotongan garis lurus dengan sumbu-<math>x</math> tetap berada di dalam selang yang mengandung akar.</p>	<p><u>Lelaran kedua:</u></p>  <p>Perpotongan garis lurus dengan sumbu-<math>x</math> mungkin menjauhi akar.</p>
<p>3. Berdasarkan nomor 2 di atas, lelarannya <i>selalu</i> konvergen</p>	<p>3. Berdasarkan nomor 2 di atas, lelarannya <i>mungkin</i> divergen.</p>

Program 3.9 berikut berisi algoritma metode *secant*.

**Program 3.9** Metode *Secant*

```

procedure Secant(x0, x1:real);
{ Mencari akar persamaan  $f(x) = 0$  dengan metode secant
  K.Awal :  $x_0$  dan  $x_1$  adalah tebakan awal akar, terdefinisi nilainya
  K.Akhir: akar persamaan tercetak di layar
}
const
  epsilon = 0.000001;      { toleransi galat akar hampiran }
var
  x_sebelumnya: real;

  function f(x:real):real;
  { mengembalikan nilai  $f(x)$ . Definisi  $f(x)$  bergantung pada persoalan }

begin
  repeat
    x_sebelumnya:=x1;
    x:=x-(f(x1)*(x1 - x0)/(f(x1)-f(x0)));
    x0:=x1;
    x1:=x;
  until (ABS(x-x_sebelumnya) < epsilon);
  { x adalah hampiran akar persamaan }
  write('Hampiran akar x = ', x:10:6);
end;

```

**Catatan:** Program 3.9 belum menangani kasus pembagian dengan 0 atau  $\approx 0$  dan kasus divergen. Program harus dimodifikasi untuk menangani pembagian dengan 0 atau  $\approx 0$  dan kasus divergen menjadi Program 3.10 berikut.

**Program 3.10** Perbaikan metode *Secant*

```

procedure Secant(x0, x1:real);
{ Mencari akar persamaan  $f(x) = 0$  dengan metode secant
  K.Awal :  $x_0$  dan  $x_1$  adalah tebakan awal akar, terdefinisi nilainya
  K.Akhir: Hampiran akar tercetak di layar
}
const
  epsilon1 = 0.000001;    { toleransi galat akar hampiran }
  epsilon2 = 0.000000001; { toleransi nilai yang hampir 0 }
  Nmaks = 30;             { jumlah maksimum lelaran }
var
  x_sebelumnya: real;
  berhenti: boolean;
  i : integer;

  function f(x:real):real;
  { mengembalikan nilai  $f(x)$ . Definisi  $f(x)$  bergantung pada persoalan }

begin
  i:=0;
  repeat
    if ABS(f(x1)- f(x0)) < epsilon2 then
      berhenti:=true; { menghindari pembagian bilangan yang » 0 }
    else
      begin
        x_sebelumnya:=x1;
        x:=x-(f(x1)*(x1 - x0)/(f(x1)-f(x0)));
        x0:=x1;
        x1:=x;
        i:=i+1;
      end;
  until (ABS(x-x_sebelumnya) < epsilon1) or (berhenti) or (i > Nmaks);

  if berhenti then
    writeln('Pembagian dengan bilangan yang hampir 0')
  else
    if i > Nmaks then
      writeln('Divergen')
    else
      { x adalah hampiran akar persamaan }
      write('Hampiran akar x = ', x:10:6);
    {endif}
  {endif}
end;

```

**Contoh 3.11**

Hitunglah akar  $f(x) = e^x - 5x^2$  dengan metode secant. Gunakan  $\epsilon = 0.00001$ . Tebakan awal akar  $x_0 = 0.5$  dan  $x_1 = 1$ .

**Penyelesaian:**

Tabel lelarannya:

$i$	$x_r$	$ x_{r+1} - x_r $
0	0.500000	-
1	1.000000	0.500000
3	-0.797042	1.797042
4	10.235035	11.032077
5	-0.795942	11.030977
6	-0.794846	0.001096
7	-0.472759	0.322087
8	-0.400829	0.071930
9	-0.374194	0.026635
10	-0.371501	0.002692
11	-0.371418	0.000083
12	-0.371418	0.000000

Akar  $x = -0.371418$

Ternyata lelarannya mengarah ke akar yang lain, yaitu  $x = -0.371418$  ■

### 3.5 Akar Ganda

Akar ganda (*multiple roots*) terjadi bila kurva fungsi menyinggung sumbu- $x$ , misalnya:

- (i)  $f(x) = x^3 - 5x^2 + 7x - 3 = (x-3)(x-1)(x-1)$  memiliki akar ganda dua di  $x = 1$
- (ii)  $f(x) = x^4 - 6x^3 + 12x^2 - 10x + 3 = (x-3)(x-1)(x-1)(x-1)$  memiliki akar ganda tiga di  $x = 1$ .

Pada pembahasan terdahulu kita telah menyinggung bahwa metode bagidua dan metode tertutup lainnya tidak dapat digunakan untuk mencari akar ganda, sebab fungsi tidak berubah tanda di sekeliling akar. Metode terbuka, seperti metode Newton-Raphson, sebenarnya dapat diterapkan di sini. Tetapi, bila digunakan metode Newton-Raphson untuk mencari akar ganda, kecepatan konvergensinya berjalan secara lanjar, tidak lagi kuadratis sebagaimana aslinya. Agar konvergensi metode Newton-Raphson tetap kuadratik untuk akar ganda, maka Ralston dan Rabinowitz mengusulkan alternatif metode Newton-Raphson [CHA91] sebagai berikut:

$$x_{r+1} = x_r - m \frac{f(x_r)}{f'(x_r)} \quad (\text{P.3.32})$$

dengan  $m$  adalah **bilangan multiplisitas** akar, misalnya .

- akar tunggal,  $m = 1$ ,
- akar ganda dua,  $m = 2$ ,
- akar ganda tiga,  $m = 3$ , dan seterusnya.

Namun alternatif ini tidak memuaskan karena kita perlu tahu terlebih dahulu bilangan multiplisitas akar. Disamping itu, untuk  $x$  dekat akar ganda, nilai  $f(x) \approx 0$  dan juga nilai  $f'(x) \approx 0$ , yang dapat mengakibatkan pembagian dengan nol. Pembagian dengan nol ini dapat dihindari dengan melihat fakta bahwa  $f(x)$  lebih dulu nol sebelum  $f'(x)$ . Jadi,

**if**  $f(x) \approx 0$  **then** hentikan lelaran

Ralston dan Rabinowitz mengusulkan alternatif lain [CHA91]. Didefinisikan

$$u(x) = f(x) / f'(x) \quad (\text{P.3.33})$$

(Perhatikan, bentuk  $u(x)$  ini memiliki akar yang sama dengan  $f(x)$ , sebab, jika  $u(x) = 0$  maka  $f(x) = 0$ ).

Selanjutnya,

$$x_{r+1} = x_r - \frac{u(x_r)}{u'(x_r)} \quad (\text{P.3.34})$$

yang dalam hal ini,

$$u'(x) = [f(x)/f'(x)]' = \frac{f'(x)f'(x) - f''(x)f(x)}{[f'(x)]^2} = \frac{[f'(x)]^2 - f''(x)f(x)}{[f'(x)]^2} \quad (\text{P.3.35})$$

sehingga

$$x_{r+1} = x_r - \frac{f(x_r)/f'(x_r)}{\frac{[f'(x_r)]^2 - f''(x_r)f(x_r)}{[f'(x_r)]^2}}$$

atau

$$x_{r+1} = x_r - \frac{f(x_r)f'(x_r)}{[f'(x_r)]^2 - f''(x_r)f(x_r)} \quad (\text{P.3.36})$$

Meskipun rumus (P.3.36) ini lebih disukai untuk akar ganda, namun ia kurang mangkus sebab memerlukan lebih banyak komputasi daripada metode Newton-Raphson yang baku. Rumus (P.3.36) berlaku secara umum, yaitu ia tetap dapat dipakai untuk pencarian akar tidak ganda sekalipun.

Metode secant juga dapat dimodifikasi dengan menyulihkan  $u(x) = f(x) / f'(x)$  ke dalam rumusnya. Rumus yang dihasilkan adalah [CHA91]:

$$x_{r+1} = x_r - \frac{u(x_r)(x_{r-1} - x_r)}{u(x_{r-1}) - u(x_r)} \quad (\text{P.3.37})$$

### **Contoh 3.12**

Hitung akar  $f(x) = x^3 - 5x^2 + 7x - 3$  dengan metode Newton-Raphson baku dan metode Newton-Raphson yang diperbaiki. Tebakan awal  $x_0 = 0$ . [CHA91]

#### **Penyelesaian:**

$$f(x) = x^3 - 5x^2 + 7x - 3$$

$$f'(x) = 3x^2 - 10x + 7$$

$$f''(x) = 6x - 10$$

Dengan metode Newton-Raphson baku:

$$x_{r+1} = \frac{x_r - \frac{x_r^3 - 5x_r^2 + 7x_r - 3}{3x_r^2 - 10x_r + 7}}{1}$$

Dengan metode Newton-Raphson yang dimodifikasi:

$$x_{r+1} = x_r - \frac{(x_r^3 - 5x_r^2 + 7x_r - 3)(3x_r^2 - 10x_r + 7)}{(3x_r^2 - 10x_r + 7)^2 - (6x_r - 10)(x_r^3 - 5x_r^2 + 7x_r - 3)}$$

Tabel lelarannya adalah:



Metode Newton Raphson baku		Metode Newton Raphson yang dimodifikasi	
$r$	$x_r$	$r$	$x_r$
0	0.000000000	0	0.000000000
1	0.428571429	1	1.105263158
2	0.685714286	2	1.003081664
3	0.832865400	3	1.000002382
4	0.913328983		
5	0.955783293		
6	0.977655101		

Lelaran konvergen ke akar  $x = 1$ . Terlihat dari tabel di atas bahwa metode Newton yang dimodifikasi memiliki jumlah lelaran lebih sedikit. ■

### 3.6 Akar-Akar Polinom

Bentuk baku polinom derajat  $\leq n$  adalah

$$p(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n \quad (\text{P.3.38})$$

dengan  $a_i$  adalah konstanta riil,  $i = 0, 1, 2, \dots, n$ , dan  $a_n \neq 0$ . Polinom  $p(x)$  memiliki  $n$  buah akar, baik akar nyata maupun akar kompleks. Akar kompleks muncul dalam pasangan konjugasi,  $w = u + vi$  dan  $w = u - vi$ , dengan  $i = \sqrt{-1}$ . Contohnya, polinom  $p(x) = 5 - 4x + x^2$  mempunyai akar  $2 + i$  dan  $2 - i$ .

Semua metode pencarian akar dapat diterapkan pada polinom. Misalnya dengan metode Newton-Raphson,

$$x_{r+1} = x_r - \frac{p(x_r)}{p'(x_r)} \quad (\text{P.3.39})$$

Masalahnya, evaluasi polinom,  $p(x_r)$  dan  $p'(x_r)$  membutuhkan banyak operasi perkalian (termasuk perpangkatan). Semakin tinggi derajat polinomnya tentu semakin banyak operasi perkalian yang diperlukan, yang berarti semakin besar rambatan galat pembulatangannya (ingat, komputer menggunakan bilangan titik-kambang). Karena itu, harus dicari suatu metode perhitungan polinom dengan sedikit operasi perkalian.

### 3.6.1 Metode Horner untuk Evaluasi Polinom

Menghitung langsung  $p(x)$  untuk  $x = 1$  tidak mangkus sebab melibatkan banyak operasi perkalian. Metode Horner, atau disebut juga metode perkalian bersarang (*nested multiplication*) menyediakan cara perhitungan polinom dengan sedikit operasi perkalian. Dalam hal ini, polinom  $p(x)$  dinyatakan sebagai perkalian bersarang

$$p(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \dots + x(a_{n-1} + a_n x)))\dots)) \quad (\text{P.3.40})$$

#### Contoh 3.12

Nyatakan  $p(x) = 8 + 6x + 2x^2 + 5x^3$  dalam bentuk perkalian bersarang.

#### Penyelesaian:

$$\begin{aligned} p(x) &= 8 + 6x + 2x^2 + 5x^3 && \text{(6 buah perkalian)} \\ &= 8 + x(6 + x(2 + 5x)) && \text{(hanya 3 buah perkalian)} \end{aligned}$$

Perhitungan  $p(x)$  untuk  $x = 2$  adalah

$$p(2) = 8 + 2(6 + 2(2 + 5 \cdot 2)) = 68 \quad \blacksquare$$

Metode perkalian bersarang untuk menghitung  $p(t)$  seringkali dinyatakan dalam bentuk tabel Horner berikut:

$t$	$a_n$	$a_{n-1}$	$a_{n-2}$	$\dots$	$a_2$	$a_1$	$a_0$
		$tb_n$	$tb_{n-1}$	$\dots$	$tb_3$	$tb_2$	$tb_1$

$$b_n = a_n \quad b_{n-1} = a_{n-1} + tb_n \quad b_{n-2} = a_{n-2} + tb_{n-1} \quad b_2 = a_2 + tb_3 \quad b_1 = a_1 + tb_2 \quad b_0 = a_0 + tb_1$$

————— polinom sisa —————

Hasil evaluasi:  $p(t) = b_0$

Jadi, untuk Contoh 3.12 di atas,

2	5	2	6	8
		10	24	60
	5	12	30	68 = $p(2)$

dan menghasilkan polinom sisa  $5x^2 + 12x + 30$ .

**Program 3.11** Menghitung  $p(x)$  untuk  $x = t$  dengan metode Horner

```
{ Dalam program utama telah didefinisikan:
  const n=...; {derajat polinom}
  var a, b, c: array[1..n] of real
}
function p(t:real):real;
{ menghitung p(t) dengan metode Horner}
var
  k: integer;
begin
  b[n]:=a[n];
  for k:=n-1 downto 0 do
    b[k]:=a[k] + b[k+1]*t;
  {end for}
  p:=b[0];
end;
```

### 3.6.2 Pencarian Akar-akar Polinom

Proses perhitungan  $p(x)$  untuk  $x = t$  dengan menggunakan metode Horner sering dinamakan *pembagian sintetis*  $p(x):(x-t)$ , menghasilkan  $q(x)$  dan sisa  $b_0$ ,

$$\left[ \frac{p(x)}{(x-t)} = q(x) \right] + \text{sisa } b_0 \quad (\text{P.3.41})$$

atau

$$p(x) = b_0 + (x-t) q(x) \quad (\text{P.3.42})$$

yang dalam hal ini,

$$q(x) = b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_3 x^2 + b_2 x + b_1 \quad (\text{P.3.43})$$

Untuk Contoh 3.12 di atas,

$$p(x) = 8 + 6x + 2x^2 + 5x^3 = 68 + (x-2)(5x^2 + 12x + 30)$$

Jika  $t$  adalah hampiran akar polinom  $p(x)$  maka

$$p(t) = b_0 + (t-t) q(t) = b_0 + 0 = b_0$$

(Perhatikan, jika  $t$  akar sejati, maka  $b_0 = 0$ )

Akar-akar lain dari  $p(x)$  dapat dicari dari polinom  $q(x)$  sebab setiap akar  $q(x)$  juga adalah akar  $p(x)$ . Proses reduksi polinom ini disebut *deflasi (deflation)*. Koefisien-koefisien  $q(x)$ , yaitu  $b_n, b_{n-1}, \dots, b_3, b_2, b_1$  dapat ditemukan langsung dari tabel Horner,

$$\begin{aligned} b_n &= a_n \\ b_{n-1} &= a_{n-1} + tb_n \\ b_{n-2} &= a_{n-2} + tb_{n-1} \\ &\dots \\ b_2 &= a_2 + tb_3 \\ b_1 &= a_1 + tb_2 \end{aligned}$$

Algoritmanya,

```
b[n]:=a[n];
for k:=n-1 downto 1 do
  b[k]:=a[k] + t*b[k+1]
{endfor}
```

Misalkan akar polinom dihitung dengan metode Newton-Raphson,

$$x_{r+1} = x_r - \frac{p(x_r)}{p'(x_r)}$$

maka proses pencarian akar secara deflasi dapat dirumuskan dalam langkah 1 sampai 4 berikut ini.

**Langkah 1:**

Menghitung  $p(x_r)$  dapat dilakukan secara mangkus dengan metode Horner.

Misalkan  $t = x_r$  adalah hampiran akar polinom  $p(x)$ ,

$$p(x) = b_0 + (x - x_r) q(x)$$

Perhitungan  $p(x_r)$  menghasilkan

$$p(x_r) = b_0 + (x_r - x_r) q(x_r) = b_0$$

Nilai  $p(x_r) = b_0$  ini dapat dihitung dengan function p

**Langkah 2:**

Menghitung  $p'(x_r)$  secara mangkus:

Misalkan  $t = x_r$  adalah hampiran akar polinom  $p(x)$ ,

$$p(x) = b_0 + (x - x_r) q(x)$$

Turunan dari  $p$  adalah

$$p'(x) = 0 + 1 \cdot q(x) + (x - x_r) q'(x) = q(x) + (x - x_r) q'(x)$$

sehingga

$$p'(x_r) = q(x_r) + (x_r - x_r) q'(x_r) = q(x_r)$$

Koefisien polinom  $q(x)$  dapat ditentukan dari langkah 1. Selanjutnya  $q(x_r)$  dapat dihitung dengan function  $q$  berikut:

**Program 3.12** Menghitung  $p'(t) = q(t)$

```
function q(t:real):real;
{ menghitung p'(t)=q(t) dengan metode Horner}
var
  k : integer;
begin
  c[n]:=b[n];
  for k:=n-1 downto 1 do
    c[k]:=b[k] + t*c[k+1]
  {endfor}
  q:=c[1];
end;
```

**Langkah 3:**

$$x_{r+1} = x_r - \frac{p(x_r)}{p'(x_r)}$$

**Langkah 4:**

Ulangi langkah 1, 2 dan 3 di atas sampai  $|x_{r+1} - x_r| < \epsilon$ .

**Program 3.13** Prosedur Newton-Raphson untuk menghitung akar polinom

```
procedure Newton_Raphson_untuk_polinom(n:integer; x:real);
{ procedure Newton-Raphson untuk menghitung akar polinom p(x) yang
  berderajat n dengan tebakan awal akar x
  K.Awal : n adalah derajat polinom; x adalah tebakan awal akar;
           kedua nilai sudah terdefinisi
  K.Akhir: Hampiran akar polinom tercetak di layar.
}
const
  epsilon = 0.0000001;
var
  x_sebelumnya: real;

  function p(t:real):real;
  {menghitung p(t) dengan metode Horner}
  var
    k: integer;
  begin
    b[n]:=a[n];
    for k:=n-1 downto 0 do
      b[k]:=a[k] + b[k+1]*t;
    {end for}
    p:=b[0];
  end {p};

  function q(t:real):real;
  { menghitung p'(t)=q(t) dengan metode Horner}
  var
    k : integer;
  begin
    c[n]:=b[n];
    for k:=n-1 downto 1 do
      c[k]:=b[k] + t*c[k+1]
    {end for}
    q:=c[1];
  end {q} ;

begin
  repeat
    x_sebelumnya:=x;
    x:=x - p(x)/q(x);
  until ABS(x - x_sebelumnya) < epsilon;

  { x adalah akar polinom }
  writeln('Hampiran akar = ', x:10:6);
end;
```

Program 3.13 ini hanya menemukan satu buah akar polinom. Untuk mencari seluruh akar nyata polinom, harus dilakukan proses deflasi. Setelah akar pertama  $x_1$  diperoleh, polinom  $p(x)$  dapat ditulis sebagai

$$p(x) = (x - x_1) q(x) + b_0$$

yang dalam hal ini  $q(x) = b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_3 x^2 + b_2 x + b_1$ . Koefisien-koefisien  $q(x)$ , yaitu  $b_n, b_{n-1}, \dots, b_3, b_2, b_1$  diperoleh di akhir Program 3.11, yang telah tersimpan pada elemen larik  $b[n-1], b[n-2], \dots, b[2], b[1]$ . Selanjutnya panggil Program 3.13 untuk mencari akar polinom  $q(x)$  yang berderajat  $n-1$  dengan tebakan awalnya dapat digunakan  $x_1$  (atau boleh bilangan lain). Setelah akar kedua  $x_2$  diperoleh, polinom  $p(x)$  dapat ditulis sebagai

$$q(x) = (x - x_2) r(x) + b_1$$

yang dalam hal ini

$$r(x) = b_{n-1} x^{n-2} + b_{n-2} x^{n-3} + \dots + b_3 x^2 + b_2 x + b_1.$$

$r(x)$  adalah polinom derajat  $n-2$  dengan koefisien  $b_{n-1}, b_{n-2}, \dots, b_3, b_2, b_1$  diperoleh di akhir Program 3.11 pada elemen larik  $b[n-1], b[n-2], \dots, b[2], b[1]$ . Selanjutnya panggil kembali Program 3.13 untuk mencari akar polinom  $r(x)$  yang berderajat  $n-2$  dengan tebakan awalnya dapat digunakan  $x_2$ . Begitu seterusnya sampai polinom sisa yang ditemukan berderajat 0. Atau, dapat juga sampai polinom sisa berderajat dua.

Algoritma selengkapnya adalah:

```

write('Tebakan awal untuk akar pertama: '); readln(x);
repeat
  Newton_Raphson_untuk_polinom(n, x);

  { salin koefisien b[n], b[n-1], ..., b[1] ke dalam
    a[n-1], a[n-2], ..., a[0] untuk pencarian akar selanjutnya }
  for i:=n downto 1 do
    a[i-1]:=b[i];
  {endfor}
  n:=n-1; { derajat polinom sisa berkurang satu }
until n=0;

```

**Contoh 3.14**

[GER85] Temukan seluruh akar nyata polinom

$$p(x) = x^5 - 12x^4 - 293x^3 + 3444x^2 + 20884x - 240240$$

dengan tebakan awal akar  $x_0=11$ .

**Penyelesaian:**

Panggil prosedur

```
Newton_Raphson_untuk_polinom(5, 11);
```

untuk mencari akar polinom  $p(x)$  berderajat 5 sengan tebakan awal akar  $x_0=11$ .

Diperoleh akar pertama, yaitu  $x_1 = 13.99990$

Deflasi  $\rightarrow p(x) = (x - x_1) q(x) + b_0$

yang dalam hal ini  $q(x) = x^4 + 1999895x^3 - 2650015x^2 - 2659927x + 17160.13$

Panggil prosedur

```
Newton_Raphson_untuk_polinom(4, 13.99990);
```

untuk mencari akar polinom  $q(x)$  berderajat 4 dengan tebakan awal akar  $x_0=13.99990$

Diperoleh akar kedua  $x_2 = 12.00016$

Deflasi  $\rightarrow q(x) = (x - x_2) r(x) + b_1$

yang dalam hal ini  $r(x) = x^3 + 1400005x^2 - 9699867x - 1429992$

Panggil prosedur

```
Newton_Raphson_untuk_polinom(3, 12.00016);
```

untuk mencari akar polinom  $r(x)$  berderajat 3 dengan tebakan awal akar  $x_0=12.00016$

Diperoleh akar  $x_3 = 9.999949$

Deflasi  $\rightarrow r(x) = (x - x_3) s(x) + b_2$

yang dalam hal ini  $s(x) = x^2 + 2396998x + 1429999$

Demikian seterusnya sampai kita temukan akar keempat dan akar kelima sebagai berikut:

$$x_4 = -12.99991$$

$$x_5 = -11.00006$$

■



### 3.6.3 Lokasi Akar Polinom

Metode Newton-Raphson memerlukan tebakan awal akar. Bagaimanakah menemukan tebakan awal akar yang bagus untuk polinom? Misalkan akar-akar diberi indeks dan diurut menaik sedemikian sehingga

$$|x_1| \leq |x_2| \leq |x_3| \leq \dots \leq |x_n|$$

Tebakan awal untuk akar terkecil  $x_1$  menggunakan hampiran

$$\begin{aligned} a_0 + a_1 x &\approx 0 \\ x &\approx -a_0/a_1 \end{aligned} \quad (\text{P.3.44})$$

yang dapat dijadikan sebagai tebakan awal untuk menemukan  $x_1$

Tebakan awal untuk akar terbesar  $x_n$  menggunakan hampiran

$$\begin{aligned} a_{n-1}x^{n-1} + a_n x^n &\approx 0 \\ x &\approx -a_{n-1}/a_n \end{aligned} \quad (\text{P.3.45})$$

yang dapat dijadikan sebagai tebakan awal untuk menemukan  $x_n$

#### **Contoh 3.15**

[NOB72] Tentukan tebakan awal untuk mencari akar polinom  $x^2 - 200x + 1 = 0$ .

#### **Penyelesaian:**

Tebakan awal untuk akar terkecil adalah

$$x_0 = -1/(-200) = 1/200$$

Tebakan awal untuk akar terbesar adalah

$$x_0 = -(-200)/1 = 200 \quad \blacksquare$$

## 3.7 Sistem Persamaan Nirlanjar

Di dalam dunia nyata, umumnya model matematika muncul dalam bentuk sistem persamaan. Persamaan yang diselesaikan tidak hanya satu, tetapi dapat lebih dari satu, sehingga membentuk sebuah sistem yang disebut sistem persamaan nirlanjar. Bentuk umum sistem persamaan nirlanjar dapat ditulis sebagai berikut:

$$\begin{aligned}
f_1(x_1, x_2, \dots, x_n) &= 0 \\
f_2(x_1, x_2, \dots, x_n) &= 0 \\
&\dots \\
f_n(x_1, x_2, \dots, x_n) &= 0
\end{aligned}
\tag{P.3.46}$$

Penyelesaian sistem ini adalah himpunan nilai  $x$  simultan,  $x_1, x_2, \dots, x_n$ , yang memenuhi seluruh persamaan. Sistem persamaan dapat diselesaikan secara berlelar dengan metode lelaran titik-tetap atau dengan metode Newton-Raphson.

### 3.7.1 Metode Lelaran Titik-Tetap

Prosedur lelarannya titik-tetap untuk sistem dengan dua persamaan nirlanjar:

$$\begin{aligned}
x_{r+1} &= g_1(x_r, y_r) \\
y_{r+1} &= g_2(x_r, y_r) \\
r &= 0, 1, 2, \dots
\end{aligned}
\tag{P.3.47}$$

Metode lelaran titik-tetap seperti ini dinamakan metode **lelaran Jacobi**. Kondisi berhenti (konvergen) adalah

$$|x_{r+1} - x_r| < \mathbf{e} \text{ dan } |y_{r+1} - y_r| < \mathbf{e}$$

Kecepatan konvergensi lelaran titik-tetap ini dapat ditingkatkan. Nilai  $x_{r+1}$  yang baru dihitung langsung dipakai untuk menghitung  $y_{r+1}$ . Jadi,

$$\begin{aligned}
x_{r+1} &= g_1(x_r, y_r) \\
y_{r+1} &= g_2(x_{r+1}, y_r) \\
r &= 0, 1, 2, \dots
\end{aligned}
\tag{P.3.48}$$

Metode lelaran titik-tetap seperti ini dinamakan metode **lelaran Seidel**. Kondisi berhenti (konvergen) adalah

$$|x_{r+1} - x_r| < \mathbf{e} \text{ dan } |y_{r+1} - y_r| < \mathbf{e}$$

Untuk fungsi dengan tiga persamaan nirlanjar, lelaran Seidel-nya adalah

$$\begin{aligned}
x_{r+1} &= g_1(x_r, y_r, z_r) \\
y_{r+1} &= g_2(x_{r+1}, y_r, z_r) \\
z_{r+1} &= g_3(x_{r+1}, y_{r+1}, z_r) \\
r &= 0, 1, 2, \dots
\end{aligned}
\tag{P.3.49}$$

Kondisi berhenti (konvergen) adalah

$$|x_{r+1} - x_r| < \mathbf{e} \text{ dan } |y_{r+1} - y_r| < \mathbf{e} \text{ dan } |z_{r+1} - z_r| < \mathbf{e}$$

**Contoh 3.16**

[CHA91] Selesaikan sistem persamaan nirlanjar berikut ini,

$$f_1(x, y) = x^2 + xy - 10 = 0$$

$$f_2(x, y) = y + 3xy^2 - 57 = 0$$

(Akar sejatinya adalah  $x = 2$  dan  $y = 3$ )

**Penyelesaian:**

Prosedur lelaran titik-tetapnya adalah

$$x_{r+1} = \frac{10 - x_r^2}{y_r}$$

$$y_{r+1} = 57 - 3x_{r+1}y_r^2$$

Berikan tebakan awal  $x_0 = 1.5$  dan  $y_0 = 3.5$  dan  $\epsilon = 0.000001$

Tabel lelarannya:

r	x	y	$ x_{r+1} - x_r $	$ y_{r+1} - y_r $
0	1.500000	3.500000	-	-
1	2.214286	-24.375000	0.714286	27.875000
2	-0.209105	429.713648	2.423391	454.088648
3	0.023170	-12778.041781	0.232275	13207.755429
...				

Ternyata lelarannya divergen!

Sekarang kita ubah persamaan prosedur lelarannya menjadi

$$x_{r+1} = \sqrt{10 - x_r y_r}$$

$$y_{r+1} = \sqrt{\frac{57 - y_r}{3x_{r+1}}}$$

Tebakan awal  $x_0 = 1.5$  dan  $y_0 = 3.5$  dan  $\epsilon = 0.000001$

Hasilnya,

r	x	y	$ x_{r+1} - x_r $	$ y_{r+1} - y_r $
0	1.500000	3.500000	-	-
1	2.179449	2.860506	0.679449	0.639494
2	1.940534	3.049551	0.238916	0.189045
3	2.020456	2.983405	0.079922	0.066146

4	1.993028	3.005704	0.027428	0.022300
5	2.002385	2.998054	0.009357	0.007650
6	1.999185	3.000666	0.003200	0.002611
7	2.000279	2.999773	0.001094	0.000893
8	1.999905	3.000078	0.000374	0.000305
9	2.000033	2.999973	0.000128	0.000104
10	1.999989	3.000009	0.000044	0.000036
11	2.000004	2.999997	0.000015	0.000012
12	1.999999	3.000001	0.000005	0.000004
13	2.000000	3.000000	0.000002	0.000001
14	2.000000	3.000000	0.000001	0.000000

Akar  $x = 2.000000$   
 $y = 3.000000$

■

Contoh 3.15 ini memperlihatkan bahwa konvergensi metode lelaran titik-tetap sangat bergantung pada bentuk persamaan prosedur lelaran dan tebakan awal. Syarat perlu kekonvergenan untuk sistem dengan dua persamaan nirlanjar adalah

$$\left| \frac{\partial g_1}{\partial x} \right| + \left| \frac{\partial g_1}{\partial y} \right| < 1$$

dan

$$\left| \frac{\partial g_2}{\partial x} \right| + \left| \frac{\partial g_2}{\partial y} \right| < 1$$

di dalam selang yang mengandung titik tetap  $(p, q)$ .

### 3.7.2 Metode Newton-Raphson

Ingatlah kembali bahwa metode Newton-Raphson dapat diturunkan dari deret Taylor,

$$f(x_{r+1}) \approx f(x_r) + (x_{r+1} - x_r) f'(x_r)$$

dan karena persoalan mencari akar, maka  $f(x_{r+1}) = 0$ , sehingga

$$0 = f(x_r) + (x_{r+1} - x_r) f'(x_r)$$

atau

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}, \quad f'(x_r) \neq 0$$

Untuk fungsi dengan dua peubah, deret Taylor orde pertama dapat dituliskan untuk masing-masing persamaan sebagai

$$u_{r+1} = u_r + (x_{r+1} - x_r) \frac{\partial u_r}{\partial x} + (y_{r+1} - y_r) \frac{\partial u_r}{\partial y} \quad (\text{P.3.50})$$

dan

$$v_{r+1} = v_r + (x_{r+1} - x_r) \frac{\partial v_r}{\partial x} + (y_{r+1} - y_r) \frac{\partial v_r}{\partial y} \quad (\text{P.3.51})$$

Karena persoalan mencari akar, maka  $u_{r+1} = 0$  dan  $v_{r+1} = 0$ , untuk memberikan

$$\frac{\partial u_r}{\partial x} x_{r+1} + \frac{\partial u_r}{\partial y} y_{r+1} = -u_r + x_r \frac{\partial u_r}{\partial x} + y_r \frac{\partial u_r}{\partial y}$$

$$\frac{\partial v_r}{\partial x} x_{r+1} + \frac{\partial v_r}{\partial y} y_{r+1} = -v_r + x_r \frac{\partial v_r}{\partial x} + y_r \frac{\partial v_r}{\partial y}$$

Dengan sedikit manipulasi aljabar, kedua persamaan terakhir ini dapat dipecahkan menjadi

$$x_{r+1} = x_r - \frac{u_r \frac{\partial v_r}{\partial y} + v_r \frac{\partial u_r}{\partial y}}{\frac{\partial u_r}{\partial x} \frac{\partial v_r}{\partial y} - \frac{\partial u_r}{\partial y} \frac{\partial v_r}{\partial x}} \quad (\text{P.3.52})$$

dan

$$y_{r+1} = y_r + \frac{u_r \frac{\partial v_r}{\partial x} - v_r \frac{\partial u_r}{\partial x}}{\frac{\partial u_r}{\partial x} \frac{\partial v_r}{\partial y} - \frac{\partial u_r}{\partial y} \frac{\partial v_r}{\partial x}} \quad (\text{P.4.53})$$

Penyebut dari masing-masing persamaan ini diacu sebagai **determinan Jacobi** dari sistem tersebut [CHA91]. Metode Newton-Raphson dapat dirampatkan (*generalization*) untuk sistem dengan  $n$  persamaan.

**Contoh 3.17**

[CHA91] Gunakan metode Newton-Raphson untuk mencari akar

$$\begin{aligned} f_1(x, y) = u &= x^2 + xy - 10 = 0 \\ f_2(x, y) = v &= y + 3xy^2 - 57 = 0 \end{aligned}$$

dengan tebakan awal  $x_0=1.5$  dan  $y_0=3.5$

**Penyelesaian:**

$$\frac{\partial u_o}{\partial x} = 2x + y = 2(1.5) + 3.5 = 6.5$$

$$\frac{\partial u_o}{\partial y} = x = 1.5$$

$$\frac{\partial v_o}{\partial x} = 3y^2 = 3(3.5)^2 = 36.75$$

$$\frac{\partial v_o}{\partial y} = 1 + 6xy = 1 + 6(1.5) = 32.5$$

Determinan Jacobi untuk lelaran pertama adalah

$$6.5(32.5) - 1.5(36.75) = 156.125$$

Nilai-nilai fungsi dapat dihitung dari tebakan awal sebagai

$$\begin{aligned} u_0 &= (1.5)^2 + 1.5(3.5) - 10 = -2.5 \\ v_0 &= (3.5)^2 + 3(1.5)(3.5)^2 - 57 = 1.625 \end{aligned}$$

Nilai  $x$  dan  $y$  pada lelaran pertama adalah

$$x_0 = \frac{1.5 - (-2.5)(32.5) - 1.625(1.5)}{156.125} = 2.03603$$

dan

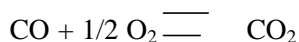
$$y_0 = \frac{3.5 - (-2.5)(36.75) - 1.625(6.5)}{156.125} = 2.84388$$

Apabila lelarannya diteruskan, ia konvergen ke akar sejati  $x = 2$  dan  $y = 3$ . ■

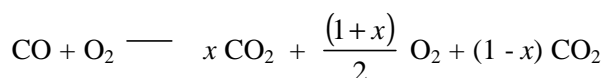
Seperti halnya metode lelaran titik-tetap, metode Newton-Raphson mungkin saja divergen jika tebakan awal tidak cukup dekat ke akar. Penggambaran kurva masing-masing persamaan secara grafik dapat membantu pemilihan tebakan awal yang bagus.

### 3.8 Contoh Soal Terapan

Dalam suatu proses Teknik Kimia, campuran karbon monoksida dan oksigen mencapai kesetimbangan pada suhu  $3000^\circ\text{K}$  dan tekanan 5 atm. Reaksi teoritisnya adalah



Reaksi kimia yang sebenarnya terjadi dapat ditulis sebagai



Persamaan kesetimbangan kimia untuk menentukan fraksi mol CO yang tersisa, yaitu  $x$ , ditulis sebagai

$$K_p = \frac{(1-x)(3+x)^{1/2}}{x(x+1)^{1/2} p^{1/2}}, \quad 0 < x < 1$$

yang dalam hal ini,  $K_p = 3.06$  adalah tetapan kesetimbangan untuk reaksi  $\text{CO} + 1/2 \text{O}_2$  pada  $3000^\circ\text{K}$  dan  $P = 5$  atm. Tentukan nilai  $x$  dengan metode regula falsi yang diperbaiki.

#### Penyelesaian:

Persoalan ini memang lebih tepat diselesaikan dengan metode tertutup karena  $x$  adalah fraksi mol yang nilainya terletak antara 0 dan 1.

Fungsi yang akan dicari akarnya dapat ditulis sebagai

$$f(x) = \frac{(1-x)(3+x)^{1/2}}{x(x+1)^{1/2} p^{1/2}} - K_p, \quad 0 < x < 1$$

dengan  $K_p = 3.06$  dan  $P = 5$  atm.

Selang yang mengandung akar adalah  $[0.1, 0.9]$ . Nilai fungsi di ujung-ujung selang adalah

$$f(0.1) = 3.696815 \text{ dan } f(0.9) = -2.988809$$

yang memenuhi  $f(0.1) f(0.9) < 0$ .

Tabel lelarannya adalah:

$r$	$a$	$c$	$b$	$f(a)$	$f(c)$	$f(b)$	Selang baru	Lebarnya
0	0.100000	0.542360	0.900000	3.696815	-2.488120	-2.988809	[a, c]	0.442360
1	0.100000	0.288552	0.542360	1.848407	-1.298490	-2.488120	[a, c]	0.188552
2	0.100000	0.178401	0.288552	0.924204	0.322490	-1.298490	[c, b]	0.110151
3	0.178401	0.200315	0.288552	0.322490	-0.144794	-1.298490	[a, c]	0.021914
4	0.178401	0.193525	0.200315	0.322490	-0.011477	-0.144794	[a, c]	0.015124
5	0.178401	0.192520	0.193525	0.161242	0.009064	-0.011477	[c, b]	0.001005
6	0.192520	0.192963	0.193525	0.009064	-0.000027	-0.011477	[a, c]	0.000443
7	0.192520	0.192962	0.192963	0.009064	-0.000000	-0.000027	[a, c]	0.000442

Hampiran akar  $x = 0.192962$

Jadi, setelah reaksi berlangsung, fraksi mol CO yang tersisa adalah 0.192962.

Hal-hal kecil membentuk kesempurnaan,  
tetapi kesempurnaan bukanlah hal yang kecil.  
(Michael Angello)

## Soal Latihan

1. Tahun 1225 Leonardo da Pisa mencari akar persamaan

$$f(x) = x^3 + 2x^2 + 10x - 20 = 0$$

dan menemukan  $x = 1.368808107$ . Tidak seorang pun yang mengetahui cara Leonardo menemukan nilai ini. Sekarang, rahasia itu dapat dipecahkan dengan metode lelaran titik-tetap. Bentuklah semua kemungkinan prosedur lelaran titik-tetap dari  $f(x) = 0$ , lalu dengan memberikaan sembarang tebakan awal (misalnya  $x_0 = 1$ ), tentukan prosedur lelaran mana yang menghasilkan akar persamaan yang ditemukan Leonardo itu.

2. Apa yang terjadi jika persamaan  $x^2 = 2$  diatur sebagai  $x_{r+1} = 2/x_r$  dan metode lelaran titik-tetap digunakan untuk menemukan akar kuadrat dari 2?
3. Tentukan titik potong kurva  $f(x) = e^{-x}$  dengan kurva  $g(x) = \sin(x)$  dengan metode Newton-Raphson.



4. Tentukan selang sehingga prosedur lelaran  $x_{r+1} = x_r/2 - \cos(2x_r)$  konvergen di dalam selang itu ( $x$  dalam radian)
5. Perlihatkan bahwa semua akar  $x^{20} - 1 = 0$  berkondisi baik.
6. Gunakan metode
  - (i) bagidua
  - (ii) regula-falsi

untuk menemukan akar persamaan Leonardo dalam selang  $[1, 1.5]$ , dan juga dengan metode

- (iii) Newton-Raphson,  $x_0 = 1$
- (iv) secant,  $x_0 = 1, x_1 = 1.5$

Untuk semua metode,  $\varepsilon = 10^{-6}$

7. Diketahui lingkaran  $x^2 + y^2 = 2$  dan hiperbola  $x^2 - y^2 = 1$ . Tentukan titik potong kedua kurva dengan metode lelaran titik-tetap (Soal ini adalah mencari solusi sistem persamaan nirlanjar).
8. Diberikan prosedur lelaran  $x_{r+1} = 0.9x_r$  dan nilai awal  $x_0 = 1$ . Menurut orang matematik, untuk  $r = 1, 2, 3, \dots$

$$x_r = (0.9)x_{r-1} = (0.9)(0.9)x_{r-2} = \dots = (0.9)^r x_0 = (0.9)^r$$

dan menyimpulkan bahwa

$$\lim_{r \rightarrow \infty} x_r = 0$$

Bagi orang numerik, harus ditanyakan terlebih dahulu berapa banyak angka bena yang digunakan.

Berapakah

$$\lim_{r \rightarrow \infty} x_r$$

menurut orang numerik bila digunakan:

- (a) satu angka bena
- (b) dua angka bena

(Petunjuk: lakukan sejumlah lelaran tanpa memprogramnya dengan komputer. Setiap perhitungan harus taat asas dengan jumlah angka bena yang digunakan)

9. Misalkan metode bagidua akan digunakan untuk menemukan akar dalam selang  $[-1,5]$ . Berapa kali selang harus dibagi dua untuk menjamin bahwa hampiran  $c_r$  mempunyai ketelitian  $0.5 \times 10^{-9}$ .
10. Dapatkah metode Newton-Raphson digunakan untuk memecahkan:
- (i)  $f(x) = 0$  jika  $f(x) = x^{1/3}$
  - (ii)  $f(x) = 0$  jika  $f(x) = (x-3)^{1/2}$  dan tebakan awal  $x_0 = 4$ ?

Mengapa?

11. Bagaimana bentuk prosedur lelaran Newton-Raphson untuk menghitung  $e$  (bilangan natural?)
12. Nilai arcsin 1 dapat dihitung dari persamaan

$$\sin(x) = 1 \quad \Leftrightarrow \quad \sin(x) - 1 = 0$$

Persamaan ini mempunyai akar ganda dua di  $x = \pi/2$  (periksa!). Tentukan akar ganda itu dengan:

- (a) metode Newton-Raphson baku
  - (b) metode Newton-Raphson dengan faktor multiplisitas akar
  - (c) metode Newton-Raphson yang dimodifikasi untuk menangani kasus akar ganda
13. Gunakan metode Newton-Raphson untuk menghitung  $(47)^{1/4}$  sampai enam angka bena.
14. Perhatikan bahwa bial metode Newton-Raphson diterapkan pada bermacam-macam  $f(x)$  di bawah ini, maka prosedur lelarannya akan mengarah pada pencarian  $\sqrt{a}$ , untuk  $a > 0$ .
- (i)  $f(x) = x^2 - a$
  - (ii)  $f(x) = 1 - a/x^2$
15. Misalkan  $f(x) = \cos(x)$
- (a) Tentukan prosedur lelaran Newton-Raphsonnya

- (b) Jika kita ingin menghitung akar  $x = 3\pi/2$ , dapatkah kita gunakan tebakan awal  $x_0 = 3$ ? Mengapa?
- (c) Seperti (b), bagaimana jika  $x_0 = 5$ ? Mengapa?