

2 AKAR PERSAMAAN NON LINIER

2.1 PENDAHULUAN DAN METODE BISEKSI

Dalam komputasi berbagai Masalah, sering sekali muncul di dalamnya masalah perhitungan Akar dari Fungsi. Di sini apabila diketahui fungsi $y=f(x)$, kita hendak mencari suatu nilai $x = p$, sedemikian sehingga $f(p) = 0$. Secara Analitik berarti mencari absis titik potong grafik fungsi tersebut dengan Sumbu X.

Untuk $f(x)$ yang merupakan polinom berderajat satu (garis lurus), ataupun Fungsi Kuadrat, Masalah ini dengan mudah dapat dipecahkan. Untuk menghitung Akar Persamaan Kuadrat, kita dapat menggunakan rumus pq, ataupun rumus abc.

Namun bagaimana halnya dengan Fungsi yang lain?

Perhitungan Akar Fungsi ini sebenarnya merupakan Masalah yang klasik dalam Matematika.

Untuk itu, berbagai Metode secara numerik telah dikembangkan orang. Secara garis besarnya, Metode yang digunakan dapat dikelompokkan menjadi dua bagian. Yang pertama adalah Metode perhitungan tanpa menggunakan turunan (derivatif). Sedangkan Metode yang kedua merupakan Metode yang memanfaatkan derivatif. kelompok pertama, kita kenal di antaranya Metode Biseksi, Metode Regula-falsi, Metode Sekan, dan Metode Atkinson. Mereka akan kita bahas terlebih dahulu. Kali pertama ini kita bahas Metode Biseksi.

Metode dengan menggunakan derivatif, terkenal sebagai Metode Newton. Metode ini masih cukup efektif dan paling sering digunakan orang.

LATIHAN

Buatlah Program kecil untuk menghitung Akar suatu Fungsi Kuadrat $f(x)=ax^2+bx+c$, memanfaatkan Rumus abc, serta memperhatikan kemungkinan dari Akar, apakah Real Berbeda, Real Kembar atau Kompleks.

2.1.1 PEMANFAATAN KEMAMPUAN KOMPUTER

Sebelum kita membahas Metode Numerik yang lazim digunakan tersebut, seperti Metode Biseksi, kita perkenalkan lebih dahulu suatu cara yang murni memanfaatkan ketangguhan Komputer, yakni dengan melakukan Pencarian berulang-ulang terhadap Akar. Sebenarnya, prinsip adalah sama dengan Metode Biseksi. Di sini kita lakukan pembagian Interval bukan menjadi 2 Subinterval seperti pada Metode Biseksi, tetapi pembagian menjadi sekaligus banyak Subinterval.

Untuk mendapatkan gambaran yang cukup jelas, misalkan kita hendak mencari Akar dari $f(x)=2x^3-5x^2-7$

Kita menghitung (sebaiknya dengan Komputer) nilai fungsi pada suatu Interval tertentu, untuk beberapa x .

Misalnya kita ambil Interval $[-10,10]$, yang kita bagi menjadi 10 Subinterval, dan kita cari nilai fungsi pada ujung-ujung Subinterval tersebut, yakni pada $x = -10, -8, \dots, 8, 10$

Programnya dapat berupa seperti ini :

Program 2.1

```
5 PRINT " X", "F(X)":PRINT
10 DEF FN F(X)=2*X^3-5*X^2-7
20 K = -10
30 PRINT K, FN F(X)
40 IF K<10 THEN K+K+2:GO TO 30
50 END
```

Kalau Program 2.1 tersebut kita jalankan diperoleh output seperti Tabel 2.1, berikut ini :
Tabel 2.1.

X	F(X)
10	-2507
8	-1351
6	-619
4	-215
2	-43
0	-7
2	-11
4	41
6	245
8	697
10	1493

Dari hasil di atas kita lihat bahwa $F(2) = -11$, dan $F(4) = 41$, fungsi berubah tanda dalam Interval $[2,4]$ tersebut. Ini berarti Akar terletak di dalam Interval tersebut, atau artinya Akar adalah antara $x = 2$ dan $x = 4$.

Sekarang Interval {2,4} tersebut kita bagi-bagi lagi, katakanlah menjadi 10 Subinterval, dengan lebar masing-masing = 0.2

Dengan sedikit modifikasi Program 2.1 menjadi

```
5 PRINT " X","F(X)":PRINT
10 DEF FN F(X)=2*X^3-5*X^2-7
20 K = 2
30 PRINT K, FN F(X)
40 IF K<4 THEN K+K+0.2: GO TO 30
50 END
```

diperoleh hasil seperti pada Tabel 5.2.

Tabel 2.2

X	F(X)
2.0	-11.0000
2.2	-9.9040
2.4	-8.1520
2.6	-5.6480
2.8	-2.2960
3.0	2.0000
3.2	7.3360
3.4	13.8080
3.6	21.5120
3.8	30.5440
4.0	41.0000

Dari hasil di atas kita lihat bahwa $F(2.8)=-2.2960$, dan $F(3.0)=2.0000$, fungsi berubah tanda dalam Interval [2.8,3] tersebut. Ini berarti Akar terletak di dalam Interval tersebut, atau artinya Akar adalah antara $x = 2.8$ dan $x = 3$.

Sekarang Interval {2.8,3} tersebut kita bagi-bagi lagi menjadi 10 Subinterval, dengan lebar masing-masing = 0.02

Dengan sedikit modifikasi lagi pada Program yang lalu, diperoleh hasil pada Tabel 5.3.

Tabel 2.3.

X	F(X)
2.80	-2.2960
2.82	-1.9105
2.84	-1.5154
2.86	-1.1107
2.88	-0.6963
2.90	-0.2720
2.92	0.1622
2.94	0.6064
2.96	1.0607
2.98	1.5252
3.00	2.0000

Dari hasil di atas kita lihat bahwa fungsi berubah tanda dalam Interval $[2.90, 2.92]$, berarti Akar terletak di dalam Interval tersebut.

Interval tersebut kita bagi lagi menjadi 10 Subinterval, dengan lebar masing-masing = 0.002

Tabel 2.4

X	F(X)
2.900	-0.2719956
2.902	-0.2290268
2.906	-0.1859589
2.908	-0.0995140
2.910	-0.0561447
2.912	-0.0126762
2.914	0.0308914
2.916	0.0745620
2.918	0.1183319
2.920	0.1622009

Terlihat bahwa Akar terletak antara 2.192 dan 2.194;. Kalau sekali lagi kita lakukan Proses, terlihat :

Tabel 2.5

X	F(X)
2.9120	-0.0126915
2.9122	-8.338928E-03
2.9124	-3.986359E-03
2.9126	3.700256E-04
2.9128	4.72641E-03
2.9130	9.082794E-02
2.9132	1.343918E-02
2.9134	1.779938E-02
2.9136	2.215958E-02
2.9138	2.651978E-02
2.9140	3.087998E-02

Dengan demikian, Akar berada antara 2.9124 dan 2.9126.

Demikian seterusnya. Tinggalsekarang kita inginkan ketelitian sampai berapa desimal. Untuk ketelitian hingga 3 desimal, diperoleh Akar 2.193, serta untuk 4 desimal diperoleh 2.1926, setelah kita melihat Tabel 2.6 di bawah ini :

Tabel 2.6

X	F(X)
2.91240	-3.986359E-03
2.91242	-3.547669E-03
2.91244	-3.112793E-03
2.91246	-2.677918E-03
2.91246	-2.243042E-03
2.91248	-2.243042E-03
2.91250	-1.804352E-03
2.91252	-1.369476E-03
2.91254	-9.346008E-04
2.91256	-4.997254E-04
2.91258	-6.103516E-05
2.91260	3.77655E-04

LATIHAN

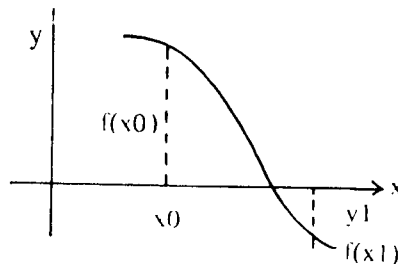
Sebagai latihan, silakan Anda menyusun kembali Program-program di atas sehingga diperoleh sebuah Program yang terpadu.

2.1.2 METODE BISEKSI

Sekarang kita lihat bagaimana menggunakan Metode Biseksi. Dalam menggunakan Metode kelompok pertama, pertama kita selalu mengasumsikan bahwa Fungsi $f(x)$ adalah kontinu pada interval yang kita libatkan. Kita juga harus telah mengetahui bahwa Akar yang kita cari berada pada interval yang kita libatkan tersebut.

Untuk dapat menentukan interval tersebut di atas kita dapat membuat sketsa grafik Fungsi tersebut. Dapat juga dengan membuat tabel nilai Fungsi pada suatu interval yang cukup besar, seperti terlihat pada Tabel 5.1 yang lalu. Seperti kita ketahui tentunya, nilai Akar adalah nilai x pada titik potong grafik Fungsi $f(x)$ dengan Sumbu X. Pada interval di sekitar titik potong tersebut, kurva akan berubah tanda, naik dari negatif menjadi positif atau turun dari positif ke negatif.

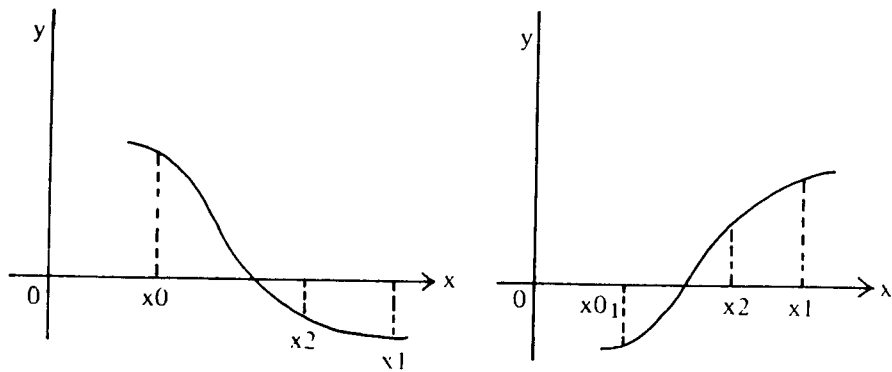
Pada Metode Biseksi, kita setiap kali iterasi membagi dua interval yang memuat Akar Fungsi, sampai lebar interval mencapai suatu bilangan yang berada dalam toleransi kita. Diasumsikan bahwa hanya satu Akar terdapat dalam interval $[x_0, x_1]$ seperti pada Gambar (2.1). Maka berlaku $f(x_0) \cdot f(x_1) \leq 0$. Tanda = berbaku bila x_0 atau x_1 merupakan Akar.



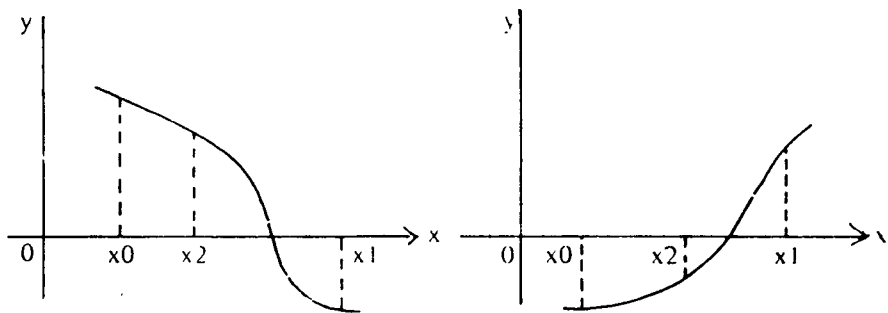
Gambar 2.1

Titik tengah dari interval, yang kita sebut x_2 , adalah $x_2 = (x_0 + x_1)/2$. Harga Fungsi di x_2 dapat dihitung. Apabila $f(x_0) \cdot f(x_2) \leq 0$, maka Akar akan berada pada interval $[x_0, x_2]$. Tanda = berbaku bila $f(x_2) = 0$, yakni bila x_2 ternyata merupakan Akar. Dalam hal $f(x_0) \cdot f(x_2) > 0$, Akar akan berada dalam interval $[x_2, x_1]$. Perhatikan Gambar 2.2a, dan Gambar 2.2b yang dapat menjelaskan dua kemungkinan di atas.

Apabila terjadi kasus pertama, kita gantikan nilai x_1 dengan nilai dari x_2 , dan kita ulangi iterasi kita pada interval $[x_0, x_1]$ yang baru. Sedangkan pada kasus kedua, nilai x_0 yang diisi dengan nilai x_2 .



(a)



(b)

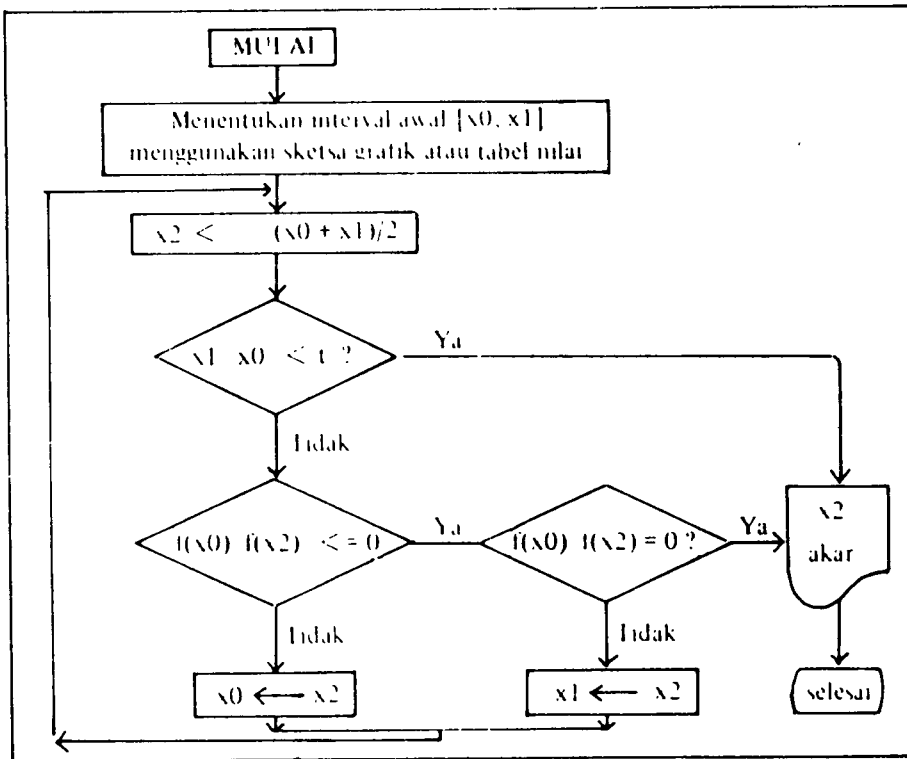
Gambar 2.2

Bagi pembaca yang terbiasa dengan penyajian diagram alur, Gambar 2.3 kiranya dapat memperjelas proses pelaksanaan Metode Biseksi kita tersebut.

Demikianlah, setelah dilakukan n kali Biseksi, diperoleh interval yang lebarnya $(x_1 - x_0)/2^n$. Jadi bila lebar interval telah menjadi lebih kecil dari t , maka Akar (yang sebenarnya) berselisih kurang dari t terhadap kedua titik ujung interval kecil tersebut. Oleh karena itu, jika kita menginginkan toleransi kesalahan lebih kecil dari t , maka diperlukan paling sedikit ${}^2\log(x_1 - x_0)$ Biseksi, kecuali bila Akar tepat pada ujung interval. Perhitungannya diperoleh dari :

$$\frac{x_1 - x_0}{2^n} < t, \text{ berarti } 2^n > \frac{x_1 - x_0}{t}$$

$$\text{atau } n > {}^2\log \frac{x_1 - x_0}{t}$$



Gambar 2.3

Berikut ini Program BASIC untuk Metode Biseksi tersebut (sebagai contoh untuk fungsi $f(x) = x - e^{1/x}$).

Program 2.2

```

10 REM Mencari akar dengan
20 REM metode biseksi
30 INPUT "Batas kiri interval ";X0
40 INPUT "Batas kanan interval ";X1
50 INPUT "toleransi";T
60 DEF FN F(X) = X - EXP(1/X)
70 IF (X1 - X0) < T THEN 150
80 F0 = FN F(X0): F1 = FN F(X1)
90 X2 = (X0 + X1)/2
100 F2 = FN F(2)
110 IF F0*F2 <= 0 THEN 130
120 X0 = X2 : GO TO 70
130 IF F0*F2 = 0 THEN 150
140 X1 = X2 : GO TO 70
150 PRINT "Akar = ";X2
160 END
  
```

Selain dengan cara menghitung banyaknya iterasi, kita dapat pula menghentikan pelaksanaan proses Biseksi dengan setiap kali memeriksa lebar interval apakah sudah lebih kecil dari t .

Berikut ini contoh pelaksanaan untuk $f(x) = x - e^{1/x}$.

Kita menggunakan tabel Fungsi, cukup untuk $x = 1, 2, 3$

Tabel 2.7

x	1	2	3
f(x)	-1,718282	.3512788	1.604388

Di sini memang kita memerlukan kejelian dalam menduga di mana kiranya letak Akar Fungsi. Pada Tabel 2.7 di atas terlihat bahwa pada interval $[1, 2]$ harga Fungsi berubah dari negatif menjadi positif. Interval awal kita adalah interval tersebut, yakni dengan $x_0 = 1$ dan $x_1 = 2$. Berikut ini tabel tiap-tiap iterasi yang kita hentikan dengan toleransi diambil $t = 10^{-4}$.

Tabel 2.8

Iterasi	x_0	x_1	x_2	$x_1 - x_0$	$f(x_0) * f(x_2)$
1	1	2	1.5	1	0.769331
2	1.5	2	1.75	0.5	9.320569E-03
3	1.75	2	1.875	0.25	-3.543344E-03
4	1.75	1.875	1.8125	0.125	-1.585734E-03
5	1.75	1.8125	1.78125	0.0625	-5.847278E-04
6	1.75	1.78125	1.765625	0.03125	-7.82354E-05
7	1.75	1.765625	1.757813	0.015625	1.765676E-04
8	1.757813	1.765625	1.761719	0.0078125	2.002125E-05
9	1.761719	1.765625	1.763672	3.90625E-03	-1.658997E-06
10	1.761719	1.763672	1.762695	1.953125E-03	1.949645E-06
11	1.762695	1.763672	1.763184	9.765625E-04	5.086031E-08
12	1.763184	1.763672	1.763428	4.882813E-04	-1.975457-08
13	1.763184	1.763428	1.763306	2.441406E-004	-7.992753E-09
14	1.763184	1.763306	1.763245	1.220703E-04	-2.097181E-09
15	1.763184	1.763245	1.763214	6.103516E-05	1.763214

Akar = 1.763214

LATIHAN

Lengkapilah Program 2.2 tersebut, sehingga diperoleh Output seperti Tabel 2.8.

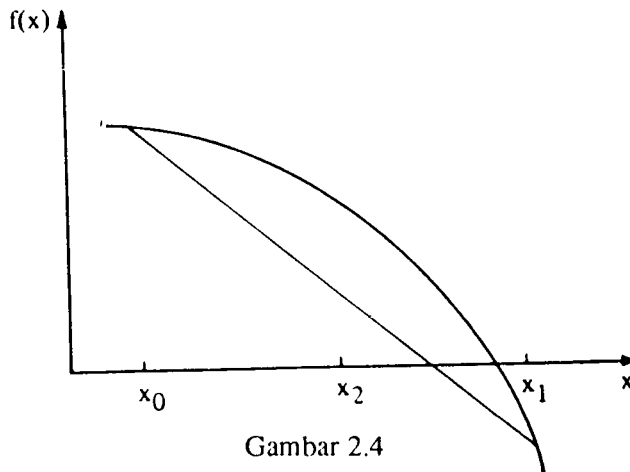
2.2 METODE REGULAFALSI

Metode Biseksi menjamin bahwa ia selalu berhasil menemukan Akar yang kita cari. Ia selalu konvergen. Namun satu kelemahan metode ini, ia bekerja dengan sangat lambat. Ia selalu mencari titik tengah X_2 sebagai titik ujung interval berikutnya. Ia tak memandang bahwa sebenarnya Akar telah berada dekat sekali dengan X_0 ataupun X_1 .

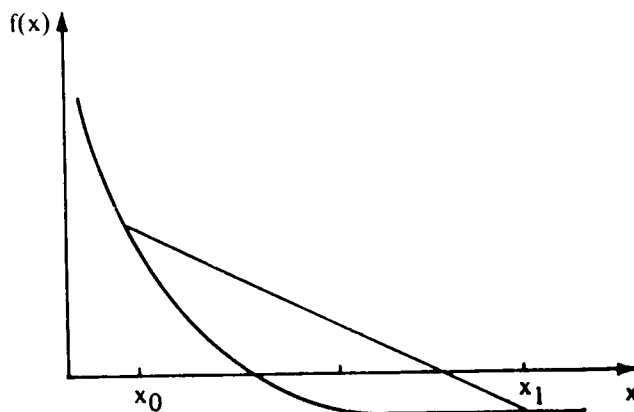
Salah satu alternatif lain untuk mempercepat perhitungan Akar ini adalah pemakaian Metode RegulaFalsi. Metode ini disebut juga Metode False Positions.

Dalam Metode RegulaFalsi ini, sama seperti Metode Biseksi, kita menentukan terlebih dahulu sebuah interval $[X_0, X_1]$ yang memuat Akar. Akar kita jepit dengan interval bagian dari $[X_0, X_1]$ tersebut secara berulang-ulang. Dalam Metode ini, penentuan titik ujung interval baru X_2 dilakukan dengan menarik garis lurus dari titik $(X_0, f(X_0))$ ke titik $(X_1, f(X_1))$. Titik potong garis dengan sumbu X adalah titik X_2 .

Perhatikan Gambar (2.4).



Gambar 2.4



Gambar 2.4 (a)

Pada Gambar 2.4, juga terlihat lebih cepatnya Metode Regula-falsi menjepit Akar dibandingkan dengan Metode Biseksi. Memang di sini Akar jauh lebih dekat ke x_1 . Namun hal ini tidak selalu demikian. Gambar 2.5 memperlihatkan contoh penurunan ukuran interval dari Metode Regula-falsi yang lebih lambat dibandingkan Metode Biseksi.

Dalam perhitungan praktisnya, garis lurus antara kedua titik $(X_0, f(X_0))$ dan $(X_1, f(X_1))$ tidak kita gambar. Persamaan garis tersebut dapat dihitung secara analitik biasa, yakni $(x-X_0)/(X_1-X_0) = (y-f(X_0))/(f(X_1)-f(X_0))$.

Perpotongan dengan sumbu X adalah titik berabsis X_2 dengan ordinat $y = 0$.

Jadi

$$(X_2 - X_0) / (X_1 - X_0) = -f(X_0) / (f(X_1) - f(X_0)),$$

atau

$$X_2 = (X_0 * f(X_1) - X_1 * f(X_0)) / (f(X_1) - f(X_0))$$

Rumus ini dapat pula kita tulis

$$X_2 = X_1 - P * f(X_1),$$

dengan $P = (X_1 - X_0) / (f(X_1) - f(X_0))$.

Perhitungan seterusnya adalah sama seperti pada Metode Biseksi. Ada baiknya bila Pembaca memperhatikan kembali tulisan bagian 2.1 yang lampau. Berikut ini prosedur dari Metode Regula-falsi secara Bahasa Pascal.

Procedure regulafalsi

```
(function f(x,real) : real ;
x0,xl,xtol : real; maxits posints ;
var
    root, fatroot : real;
var
    noofits : posints ;
var
    cunverged : boolean) ;
{Diasumsikan x0 < xl}
var
    x2, fx, f0 : real ;
    itcount : 0..maxint ;
    state : (iterating, withintol, maxitsreached) ;
begin
    itcount := 0;
    state := iterating;
    repeat
        itcount := itcount + 1;
        f0 := x0-exp(i/x0);
        f1 := xl-exp(l/xl) ;
        X2 := xl-(xl-x0)* f1/(f1-f0) ;
        f2 := x2-exp(l/x2);
```

```

    if f0*f2 < 0
    then
        x1 := x2
    else
        begin
            x0 := x2 end;
    if |(x2-x1)/x1| <= xtol
    then
        state := Withintol
    else
        if itcount = maxits
        then state := maxitsreached
    until state <> iterating or state = withintol ;
    converged := state withintol;
    root := x1-(x1-x0)*f1/(f1-f0);
    fatroot := f(root) ; noofits := itcount
end(regulafalsi).

```

Untuk menunjukkan iterasi, sebaiknya kita mengambil toleransi secara relatif, yakni:

$$|(X_2 - X_1)/X_1 \text{ atau } (X_2 - X_0)/X_0|$$

Hal ini disebabkan kemungkinan bahwa selisih $X_2 - X_1$ atau $X_2 - X_0$ yang mendekati nol.

Bagi pembaca yang mungkin terbiasa dengan bahasa komputer BASIC, berikut ini ditampilkan Program untuk menghitung Akar dengan Metode Regulafalsi kita tersebut. Kita ambil sebagai contoh adalah fungsi $f(x) = x - e^{(1/x)}$

Program 2.3:Metode Regulafalsi

```

5   CLS
10  REM METODE REGULAFALSI 20 REM UNTK MENENTUKAN AKAR 30
    REMXO DAN XI ADALAH
40  REM BATAS KIRI DANKANAN INTERVAL
50  INPUT "BATAS KIRI"; XO 60 INPUT "BATAS KANAN"; XI
70  REM T Adalah TOLERANSI
80  REM M = MAKSIMUM ITERASI
90  INPUT "TOLERANSI"; T 100 INPUT "MAKSIMUM ITERASI";M 110 REM
    I=ITERASI KE 120 I = 0
122 FOR K = 1 TO 80
124 PRINT "-"; NEXT K
125 PRINT TAB(6); "ITERASI"; TAB(19); "XO"; TAB(27); "X1"; TAB(40); "X2";
    TAB(53); "X1-X0";TAB(70);F0*F2
127 FOR U= I TO 80
128 PRINT "-": NEXT U 130 I = I + 1
132 FO = XO - EXP(1/XO)
134 FI = XI - EXP(1/XI)

```

```

140 X2 = X1 - (X1-X0) * F1/(F1-FO) 142 F2=X2-EXP(1/X2)
150 PRINT TAB(8); I; TAB(18); XO ; TAB(26);X1;TAB(39);X2;TAB (52); X1-XO;
    TAB(66) ; FO * F2
160 IF ABS((X2 - X1)/X1) < T THEN 240
165 IF C = M THEN 240
170 IF FO * F2 > O THEN 210 180 IF FO * F2 = 0 THEN 230 190 X1 = X2
200 GOTO 130
210 XO = X2
220 GOTO 130
230 REM X2 = AKAR
240 PRINT "AKAR=" ; X2 250 FOR T = 1 TO 80
260 PRINT "-";NEXT T
300 END

```

Bila Program di atas kita RUN, hasilnya adalah
 BATAS KIRI? 1
 BATAS KANAN? 2
 TOLERANSI? 1E-04
 MAKSIMUM ITERASI? 15

ITERASI	XO	XI	X2	XI-XO	FO*F2
1	1	2	1.830264	1	-.177486
2	1	1.830264	1.783184	.8302641	-5.347201E-02
3	1	1.783184	1.769252	.7831838	-1.620983E-02
4	1	1.769252	1.765052	.7692521	-4.922598E-03
5	1	1.765052	1.763778	.7650518	-1.495501E-03
6	1	1.763778	1.763392	.7637782	-4.543244E-04
7	1	1.763392	1.763274	.7633915	-1.376492E-04

AKAR = 1.763274

2.3 METODE SEKAN

Pada bagian yang baru lalu, dapat dilihat bagaimana Metode Regula-falsi bekerja. Mula-mula ditentukan sebuah interval $[X_0, X_1]$ yang mengandung Akar fungsi $f(x)$ yang kita cari. Pada setiap kali iterasi, salah satu titik ujung interval, X_0 atau X_1 , bergerak memperkecil panjang interval. Nampak bahwa Akar fungsi seolah-olah semakin dijepit oleh titik ujung interval tersebut. Iterasi dihentikan setelah dicapai jumlah maksimum iterasi tertentu ataupun setelah toleransi tertentu.

Untuk menentukan titik ujung baru interval, kita tarik garis lurus melalui titik $(X_0, f(X_0))$ dan $(X_1, f(X_1))$. Titik ujung baru, yakni X_2 , adalah perpotongan garis lurus di atas dengan sumbu x. Setelah diperoleh titik X_2 , dengan kriteria sama seperti pada Metode Biseksi, kita tentukan interval $[X_0, X_1]$ untuk iterasi berikutnya.

Metode Regula falsi mengandung suatu kelemahan. Di sini, lain dengan Metode Biseksi, panjang interval penjepit $[X_0, X_1]$ yakni $X_1 - X_0$ kebanyakan tidak dapat mendekati nol. Hal ini disebabkan kebanyakan fungsi mempunyai grafik yang cekung penuh atau cembung penuh di sekitar Akar. Karenannya hanya salah satu titik ujung interval, X_0 atau X_1 , yang bergerak menjepit sementara titik ujung yang lain selalu tetap untuk seluruh iterasi. Perhatikan tabel yang merupakan tabel iterasi Metode Regula falsi untuk fungsi $f(x) = x - e^{1/x}$, yang lalu. Titik X_0 tidak bergerak sampai seluruh iterasi selesai dijalankan.

Dengan demikian, kita sebaiknya tidak menggunakan toleransi $X_1 - X_0 \leq xtol$, karena mungkin sekali panjang interval penjepit $[X_0, X_1]$ tersebut masih cukup besar, padahal titik ujung interval telah bergerak sampai dekat sekali ke Akar. Kalau hal terakhir di atas terjadi, dan iterasi kita lanjutkan, pergeseran titik ujung tersebut akan sangat kecil sekali. Jadi, yang kita amati adalah pergeseran relatif titik ujung interval, yakni apakah $(X_1 - X_2) / X_1$ atau $(X_1 - X_2) / X_0 \leq xtol$. Dapat dicatat bahwa dalam Metode numerik orang lebih suka menggunakan pendekatan dengan error relatif dibandingkan pendekatan dengan error mutlak. Di sinipun kita tidak menggunakan pengamatan terhadap nilai $X_1 - X_2$ ataupun $X_0 - X_2$.

Berdasarkan keadaan yang terdapat pada Metode Regula falsi, dilakukan modifikasi terhadapnya. Metode modifikasi Regula falsi ini disebut *Metode Sekan*, yang kerap kali disebut juga *Metode Interpolasi Linier*.

Perbedaan utama antara Metode Sekan dengan Metode Regula falsi serta Biseksi adalah bahwa pada Metode Sekan tidak dilakukan penjepitan Akar. Pada Metode Sekan mula-mula kita mengambil dua titik sembarang $(X_0, f(X_0))$ dan $(X_1, f(X_1))$ pada fungsi dengan interval $[X_0, X_1]$ tidak harus menjepit Akar. Jadi dapat saja $f(X_0)$ dan $f(X_1)$ bertanda sama. Sama seperti pada Metode Regula falsi, kita tarik garis lurus melalui kedua titik $(X_0, f(X_0))$, dan $(X_1, f(X_1))$ yang memotong sumbu x di titik $(X_2, 0)$. Jadi sama seperti pada Metode Regula falsi, nilai X_2 adalah sama dengan

$$X_2 = (X_1 - X_0) * f(X_1) / (f(X_1) - f(X_0))$$

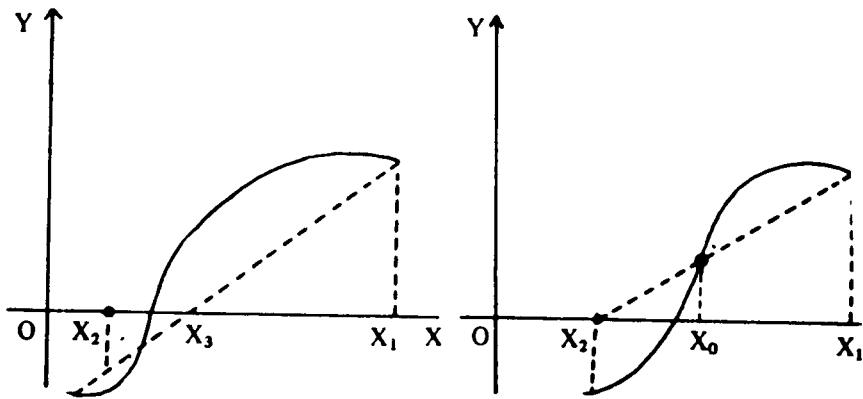
Untuk melaksanakan iterasi berikutnya, kita lakukan pergeseran, yakni $X_0 \leftarrow X_1$, dan $X_1 \leftarrow X_2$, yakni harga X_0 baru diisi (diganti) dengan harga X_1 lama, sementara harga X_1 baru diisi dengan harga X_2 .

Demikianlah dilakukan iterasi seperti di atas, sampai tercapai batas maksimum maxits, ataupun sampai diperoleh toleransi $(X_1 - X_2) / X_1 \leq X_{tol}$.

Jadi kalau kita tinjau ke proses pelaksanaan Metode Sekan terhadap sebuah fungsi tertentu, maka akan diperoleh sebarisan x_i , $i = 0, 1, \dots, k, k+1, \dots$, yang mempunyai hubungan rekurensi:

$$x_{j+1} = x_j - (x_j - x_{j-1}) * f(x_j) / (f(x_j) - f(x_{j-1}))$$

Bila iterasi kita hentikan setelah iterasi ke k maka x_{k+1} merupakan Akar (pendekatan) dari fungsi. Untuk lebih jelasnya proses Metode Sekan di atas dapat dilihat pada Gambar (2.5), untuk iterasi pertama dan kedua.



Gambar 2.5

Selanjutnya *Procedure Secant* berikut ini, secara Pascal, kiranya dapat lebih menjelaskan lagi. Di sini perlu diperhatikan adanya kemungkinan terjadinya overflow. Hal ini disebabkan oleh pembagian oleh nol bila $f(X_1) - f(X_0)$ kecil sekali mendekati nol. Hal serupa juga terjadi bila x_1 dekat sekali ke nol.

Procedure Secant

```

(function f(c:real): real;
x0,x1,xtol :real; maxits : points;
var
root,fatroot : real; var noofits: points;
var
converged : secountcomes);
const
assumedzero = 1E-20;
var
f0,f1,newf :real;
itcount : 0..maxint;
x0iscurrent : boolean;
state : secstates;
begin
f0 :=f(x0); f1:= f(x1); x0iscurrent := false;
itcount := 0; state := iterating;
repeat
if abs(f1-f0) <= assumedzero then state := tooflat else
begin
itcount := itcount+1;

```

```

if x0iscurrent then
begin
x1 := x1-(x1-x0)*f1/(f1-f0)
f1 := f(x1); newf := f1;
end
else
begin
x0 := x0-(x0-x1)*f0/(f0-f1)
f0 := f(x0); newf := f0
end;
x0iscurrent := not x0iscurrent;
if abs(x0) <= assumedzero then state := toonearzero else
if abs(x1-x0)/x0 <= xtoll then state := withinxtoll else
if itcount = maxits then state := maxitsreached
end
until state <> iterating;
outcome := state;
if x0iscurrent then
begin
root := x0; fatroot := f0
end
else
begin
root := x1; fatroot := f1
end;
nofits := itcount
end{secant};

```

Untuk melihat lebih efisiennya Metode Sekan dibandingkan dengan Metode Biseksi, kita dapat memperhatikan Tabel 2.9 berikut ini. Untuk menghitung Akar fungsi $f(x) = x e^{1/x}$, Metode Biseksi membutuhkan 21 iterasi, sedangkan Metode Sekan cukup 5 iterasi.

Tabel 2.9. Perbandingan Metode Biseksi dan Sekan

Iterasi	Biseksi	Sekan
1	1.500000	1.830264
2	1.750000	1.759565
3	1.875000	1.763286
4	1.812500	1.763223
5	1.781250	1.763223
Konvergen	21	5

LATIHAN

Silakan Anda membuat sebuah Program Metode Sekan ini.

2.4 METODE ITERASI TITIK-TETAP

Kali ini akan kita bicarakan, suatu Metode yang berbeda sekali dengan Metode Biseksi, Regula-falsi ataupun Aekan. Metode tersebut adalah Metode iterasi Titik-tetap. Implementasi iterasi Titik-tetap ini sebenarnya cukup mudah dan sederhana. Namun ada beberapa syarat yang harus dipenuhi oleh fungsi, agar Metode ini dapat dilaksanakan serta dapat menghasilkan solusi.

Syarat pertama, bahwa $f(x) = 0$ harus dapat diubah ke bentuk $x = g(x)$. Kalau hal ini dapat kita lakukan, maka barulah kita menggunakan persamaan rekurensi :

$$x_{k+1} = g(x_k),$$

untuk $k = 0, 1, 2, \dots$, untuk mencari Akar tersebut.

Di sini x_0 adalah asumsi awal kita.

Bentuk Fungsi $g(x)$ dapat kita tentukan secara berbeda-beda. Pandang, misalnya fungsi $f(x) = x - e^{1/x}$, yang akan kita hitung akarnya.

Ia dapat kita tulis menjadi

$$f(x) = x - e^{1/x} = 0,$$

atau

$$x = e^{1/x}.$$

Jadi, $g(x) = e^{1/x}$.

Dapat pula, dengan melakukan operasi logaritma natural \ln terhadap kedua ruas persamaan, diperoleh

$$\ln(x) = 1/x,$$

atau

$$x = 1/\ln x.$$

Selain itu dapat pula kita kerjakan sehingga menjadi

$$2x = x + e^{1/x},$$

atau

$$x = (x + e^{1/x})/2.$$

Berarti $g(x) = (x + e^{1/x})/2$.

Dengan melakukan iterasi menggunakan persamaan rekurensi $x_{k+1} = g(x_k)$ di atas, yakni menghitung

$$x_1 = g(x_0),$$

$$x_2 = g(x_1),$$

$$x_3 = g(x_2),$$

dan seterusnya,

diperoleh barisan nilai x_0, x_1, x_2, \dots yang kita harapkan konvergen ke Akar dari fungsi $f(x)$.

Kalau kita perhatikan bentuk $x = g(x)$, maka Akar dari $f(x)$ tak lain adalah perpotongan antara garis lurus $y = x$ dan kurva $y = g(x)$. Gambar 2.6 berikut menggambarkan Jalannya iterasi yang mengarah ke titik potong tersebut di atas ternyata tidak selalu tercapai. Iterasi

kita acapkali divergen. Untuk melihat hal tersebut, perlu dilakukan beberapa analisis matematika terhadap apa yang disebut titik-tetap.

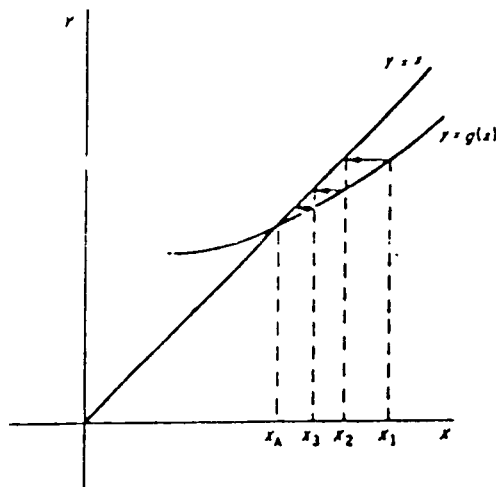
Pandang fungsi $g(x)$ yang terdefinisi pada interval $[a,b]$. Pada interval $[a,b]$ tersebut terdapat beberapa w , sedemikian sehingga $g(w) = w$. Maka kita katakan bahwa fungsi $g(x)$ mempunyai titik-tetap w dalam $[a,b]$.

Sebuah teorema telah tersedia untuk menanyakan apakah sebuah fungsi mempunyai titik-tetap pada suatu interval. Teorema tersebut mengatakan bahwa *jika $g(x)$ kontinu pada interval $[a,b]$ dan $g(x)$ berada dalam $[a,b]$ untuk semua x dalam $[a,b]$, maka $g(x)$ mempunyai titik-tetap di dalam $[a,b]$* . Selanjutnya, jika turunan pertama $g'(x)$ ada di dalam (a,b) dan $g'(x) \leq k < 1$, untuk setiap x dalam (a,b) , maka titik-tetap tersebut tunggal.

Dapat dicatat bahwa teorema di atas hanya memberi syarat cukup. Ia tidak memberi syarat perlu. Jadi kalau kondisi di atas tidak terpenuhi, masih mungkin bahwa $g(x)$ mempunyai titik-tetap yang tunggal.

Sebagai contoh adalah fungsi $x = e^{-2x}$.

Fungsi $g(x) = e^{-2x}$ mempunyai turunan $g'(x) = -2e^{-2x}$ yang pada interval $[0,1]$ misalnya di titik $0,1$ mempunyai harga $-1,6375$. Kalau kita amati fungsi e^{-2x} pada interval $[0,1]$ adalah turun langsung, sedangkan fungsi x adalah naik langsung, sehingga secara grafik perpotongan kedua fungsi tadi adalah titik-tetap yang tunggal.

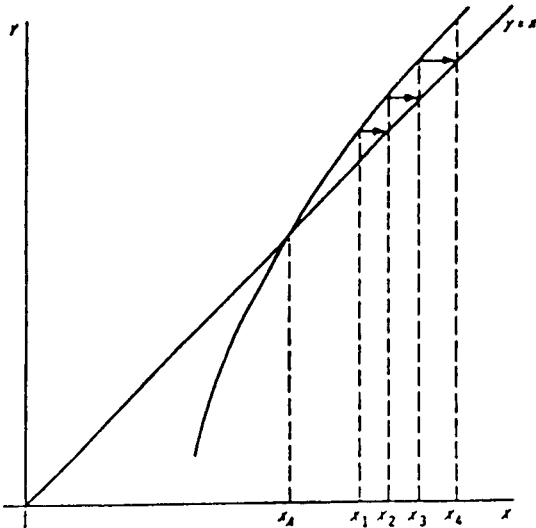


Gambar 26

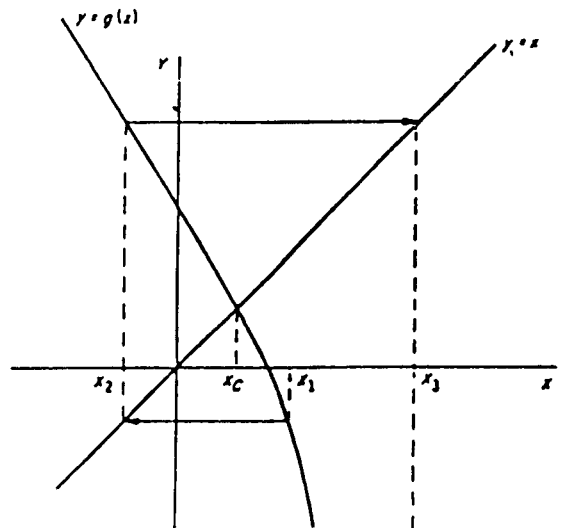
Bagaimanapun juga, teorema di atas merupakan teorema penting yang dapat kita pakai sebelum kita menggunakan Metode iterasi titik-tetap ini. Gambar 2.7(a) & 2.7(b) memperlihatkan divergen-nya iterasi karena $g'(x) < 1$.

Sekarang kita lihat bagaimana kita menggunakan teorema di atas terhadap ketiga $g(x)$ yang berbeda sewaktu kita akan menghitung Akar dari $f(x) = x - e^{1/x}$ tadi. Misalnya kita ambil

$x_0 = 1,5$. Untuk $g(x) = e^{1/x}$ kita dapatkan $g'(x) = (-e^{1/x})/x^2$ dan $g'(x_0) = 0,8657 < 1$. Iniberarti kita dapat melaksanakan iterasi titik-tetap dengan $g(x)$ di atas yang akan konvergen ke akar fungsi.



Gambar 2.7(a)



Gambar 2.7(b)

Untuk $g(x) = 1/\ln x$ didapat $g'(x) = -1/x(\ln x)^2$ dan $g'(x_0) = 4,0551 > 1$. Tentunya kita tidak menggunakan $g(x)$ ini untuk iterasi. Memang ternyata kalau kita paksakan iterasi, ia akan divergen.

Terakhir untuk $g(x) = (x + e^{1/x}) / 2$. Turunan pertama $g'(x) = 0,5 - e^{1/x} / (2x^2)$ dan $g'(x_0) = 0,0672 < 1$.

Tabel 5.10 menunjukkan pelaksanaan iterasi Titik-tetap untuk fungsi $f(x) = x - e^{1/x}$ dengan tiga buah $g(x)$ yang berbeda seperti di atas. Asumsi awal adalah $x_0 = 1,5$ dan iterasi berhenti setelah toleransi relatif $|(x - \text{old}x) / \text{old}x| \leq \text{xtol}$, dengan $\text{xtol} = 5E-07$.

Tabel 5.10

Iterasi	$g(x) = e^{1/x}$	$g(x) = 1/\ln x$	$g(x) = (x + e^{1/x})/2$
1	1.947734	2.466303	1.723867
2	1.670991	1.107763	1.755034
3	1.819291	9.771126	1.761464
4	1.732672	0.438706	1.762843
5	1.780944	-1.213701	1.763205

Berikut ini disajikan Procedure FixedPoint dalam bahasa Pascal, agar kiranya dapat lebih menjelaskan bekerjanya Metode iterasi Titik-tetap ini.

Procedure FixedPoint

```
(function g(x:real): real;
x0,xtol : real; maxits : posints;
var
    root: real; var noofits: posints;
var
    outcom :fpoutcomes);
const
    assumedzero = 1E-20
var
    x, oldx : real;
    itcount : 0..maxint
    state : fpstates;

begin
    x:=x0; itcount:=0; state:=iterating;
    repeat
    if abs(x)<= assumedzero then state := toonenearzero else
    begin
        itcount :=itcount+1; oldx:=x;
        x:= g(x);
        if abs((x-oldx)/oldx) <= xtol then state withinol else
        if it count = maxits then state := maxitssreached
    end
    until state <> iterating;
    outcome := state
    root := x; noofits := itcount
end {fixedpoint};
```

LATIHAN

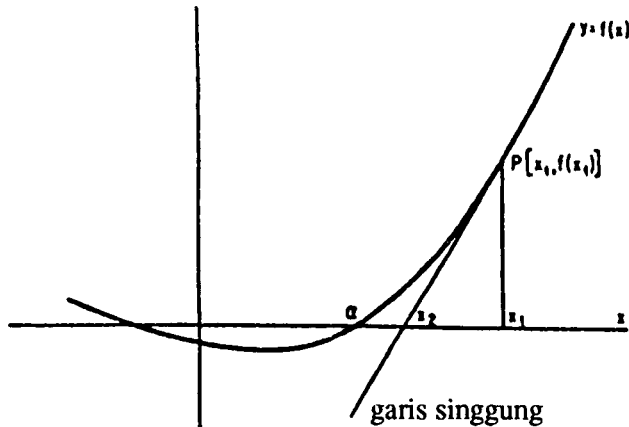
Silakan Anda membuat Program untuk Metode Iterasi Titik-Tetap ini.

2.5 METODE NEWTON-RAPHSON

Di antara Metode lain yang ada, Metode Newton ini ternyata cepat menjadi populer, dan disukai pemakai. Memang, ia relatif cepat waktu pelaksanaannya dibandingkan Metode lain. Metode ini dikenal pula sebagai Metode Newton-Raphson.

Pandang bahwa a adalah Akar dari fungsi $f(x)$ yang kita inginkan. Misalkan x_1 titik pendekatan awal kita. Kita usahakan x_1 tersebut sedekat mungkin dengan garis singgung

pada kurva $f(x)$ di titik $P(x_1, f(x_1))$. Garis singgung ini memotong sumbu X pada sebuah titik yang lebih dekat lagi ke a dibandingkan x_1 . Kita sebut titik potong tersebut x_2 . Perhatikan gambar 2.8



Gambar 2.8

Dari kalkulus kita ingat bahwa persamaan garis singgung melalui titik P tersebut adalah :

$$y - f(x_1) = f'(x_1)(x - x_1).$$

Di sini $f'(x_1)$ adalah gradien garis singgung.

Kalau garis singgung tersebut kita potongkan dengan sumbu X, akan kita peroleh titik x_2 yang memenuhi

$$x_2 = x_1 - f(x_1)/f'(x_1).$$

Kalau proses ini kita ulang terus menerus maka akan kita peroleh nilai x_2 yang semakin dekat kepada Akar. Konvergensi tercapai dengan sangat cepat.

Secara rekurensi persamaan di atas menjadi

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

di sini $i = 1, 2, 3, \dots$ dan $f'(x_i)$ adalah turunan pertama dari $f(x)$ pada titik x_i .

Dalam menggunakan Metode ini, jelas turunan fungsi $f'(x_i)$ harus dengan mudah ditentukan.

Sekarang kita perhatikan beberapa contoh. Misalkan kita diminta menghitung Akar terkecil yang positif dari $x^3 - 5x + 3$. Dengan menggunakan tabel perhitungan nilai fungsi didapat bahwa Akar-Akar fungsi ketiganya real, yang pertama bernilai antara -2 dan -3, yang kedua antara 1 dan 2, serta Akar yang ditanyakan berada antara 0 dan 1.

Di sini $f'(x) = 3x^2 - 5$. Bila kita ambil $x_1 = 1$, maka $x_2 = 1 - 1/2 = 0,5$. Kemudian berturut-turut kita hitung x_3, x_4 dan seterusnya, diperoleh :

$$x_3 = 0,5 + 5/34 = 0,64$$

$$x_4 = 0,64 + 0,062144/3,7712 = 0,6565$$

$$x_5 = 0,6565 + 0,00044/4121125/3,70702325 = 0,656620$$

$$x_6 = 0,656620 + 0,0000001597528/3,7065503268$$

$$= 0,656620431047$$

Terlihat bahwa konvergensi tercapai dengan cepat sekali.

Contoh berikut adalah fungsi transcenden $f(x) = e^{-0,4x} - 9$. Menurut hasil yang bisa kita peroleh dari grafik fungsi tersebut, maka akan menghasilkan Akar negatif -22,5 dan Akar positif sekitar 6. Sekarang, kita mencoba akan mencarinya dengan Metode Newton berikut ini :

$$f(t) = e^{0,4t} - 0,4t - 9$$

$$f'(t) = 0,4(e^{0,4t} - 1)$$

maka :

$$t_1 = 6$$

$$t_2 = 6 - 0,37682/4,009272 = 6,1$$

$$t_3 = 6,1 - 0,03304/4,189216$$

$$= 6,1 - 0,0079 = 6,0921$$

Ternyata hasil tersebut merupakan hasil pendekatan yang baik untuk pendekatan sampai 5 tempat desimal.

Berikut ini merupakan prosedur Newton dalam bahasa Pascal.

Di sini kita asumsikan perlengkapan Pascal berikut :

```

type
  posints=1..maxint;
  newstates=(iterating,withintol,maxitsreached,toonearzero,tooflat);
  newtoutcomes=withintol..tooflat;

```

Procedure Newton

```

(function f(x: real):real;
function f dashed(x:read): real;
x().xtol :real; maxits : posints;
var root,fatroot: real; var noofits : posints;
var outcome : newtoutcomes);
const
  assumedzero = 1E-20;
var
  state :newstates;
  x.oldx,fx,fdx : real;
  itcount : ()..maxint;
begin
  x:=x();fx:=f(x);
  itcount := (): state iterating,
  repeat
    if abs(x)<= assumedzero then state :=toonearzero else
    begin
      itcount := itcount +1
      fdx := fdashed(x);
      if abs(fdx) <= assumedzero then state := tooflat else
      begin
        oldx:= x;x:= x-fx/fdx;fx =f(x)

```

```

    if abs ((x-oldx)/oldx) <= xtol then state withinol else
    if itcount = maxits then state := maxitsreached
end;
until state <> iterating;
outcome := state;
noofits := itcount;
root := x; fatroot := fx
end(newton).

```

Sebagai perbandingan, betapa cepatnya Metode Newton ini dibandingkan dengan Metode-Metode sebelumnya, kita dapat melihatnya pada Tabel 2.11, yakni menghitung Akar fungsi $f(x) = x - e^{(1/x)}$

Tabel 2.11.

Iterasi	Newton
1	1,739987
2	1,763078
3	1,763223
4	1,763223

Konvergen dalam 4 Iterasi

=====

Berikut ini diberikan Contoh Program yang cukup lengkap mengenai Perhitungan Akar. Program 2.4 mengetengahkan Metode Newton-Raphson, khusus untuk menghitung semua Akar suatu Polinom berderajat n. Dalam Program tersebut juga diberikan sebuah Aplikasi pada bidang Kimia, berupa perhitungan Persamaan Beattie-Bridgeman. Pembaca yang berminat dapat mencoba Program tersebut.

Di sini kita memanfaatkan teknik Pembagian Sintetik Polonom (Synthetic Division).

Program 2.5 adalah Program yang lebih lengkap dari Metode Sekan serta Metode Newton-Raphson.

Program 2.4

```

0  SCREEN 0:WIDTH 80:PRINT CHR$(12):CLS:KEY OFF
5  PRINT"*****"
10 PRINT"*"
15 PRINT"*      CONTOH PROGRAM"
20 PRINT"*"
25 PRINT"*      METODE NEWTON-RAPHSON DENGAN SYNTHETIC
    DIVISION"
30 PRINT"*"

```

```

35 PRINT"*          UNTUK POLINOM DERAJAT N          *"
40 PRINT"*          *"
45 PRINT"*          DAN PERSAMAAN BEATTIE-BRIDGEMAN    *"
50 PRINT"*          *"
60 PRINT"*          *"
95 PRINT"*****"
97 INPUT "TEKAN ENTER";ZX$
99 CLS
100 "***** Program Utama *****"
110 DIM A(100,4),AA(20),B(20),XW(20)
120 GOTO 250
130 '
140 'Evaluasi polinom dan derivatifnya
150 '
160 F = 0:FP=0
170 FOR KK = N TO 0 STEP - 1
180 F = F + AA(KK) * X ^ KK
190 IF KK=0 GOTO 220
200 FP= FP+ KK*AA(KK) * X ^ (KK-1)
210 NEXT KK
220 RETURN
225 '
230 'Memilih problem untuk diselesaikan
240 '
250 CLS:PRINT"PROBLEMA UNTUK DISELESAIKAN:"
260 PRINT:PRINT"  1. POLINOM DERAJAT N"
270 PRINT:PRINT"  2. PERSAMAAN BEATTIE-BRIDGEMAN"
280 PRINT:PRINT"PILIHAN ANDA";:INPUT METH
285 IF METH<1 OR METH>2 GOTO 250
290 ON METH GOSUB 3000,4000
295 RST$="Y"
300 TIT$="AKAR"+STR$(NW-N+1)
305 IF RST$="T" OR RST$="t" THEN SCREEN 1: GOTO 430
310 '
320 'Input variabel pencarian
330 '
340 CLS:PRINT:PRINT"TENTUKAN RANGE DARI PENCARIAN"
350 PRINT "LIMIT BAWAH";: INPUT MINX
360 PRINT "LIMIT ATAS";: INPUT MAXX
400 PRINT:PRINT"BERIKAN DUGAAN TERBAIK UNTUK AKAR";:INPUT X1
410 PRINT:PRINT "BERIKAN KRITERIA KONVERGENSI (EPSILON) DARI
F";: INPUT EPS
414 IF ALLROOT$="T" OR ALLROOT$="t" GOTO 430
415 PRINT:PRINT"ANDA INGIN MENGUBAH LAGIHARGA PARAMETER DI

```

```

        ATAS UNTUK SETIAP AKAR(Y/T)":INPUT RST$
420 '
430 'Kalkulasi fungsi dalam range dari pencarian
440 '
470 DX = (MAXX -MINX) / 100
480 FOR K = 0 TO 100
490 X = MINX + K * DX
500 A(K,1)=X
510 GOSUB 140:A(K,2)=F
520 NEXT K
530 '
540 GOSUB 2000 'Rutin Mencari Akar
550 '
560 PRINT:PRINT"TEKANENTERUNTUKMELANJUTKAN":HK$=INPUT$(1)
570 IF GRAPH$="T" OR GRAPH$="t" GOTO 590
580 GOSUB 10000 'Rutin Plot
590 IF ALLROOT$="T" OR ALLROOT$="t" GOTO 1010
600 XW(N) = X1
610 '
620 'Synthetic division
630 '
640 IF N = 1 GOTO 960
650 B(N-1)=AA(N)
660 FOR R = 1 TO N - 1
670 B(N - 1 - R) = AA(N - R) + B(N - R) * X1
680 NEXT
690 N = N - 1
700 FOR JJ = N TO 0 STEP - 1
710 AA(JJ) = B(JJ)
720 NEXT
730 '
740 'cek apakah akar kompleks
750 '
760 IF N<>2 GOTO 910
770 WW=AA(1)^2-4*AA(2)*AA(0)
780 IF WW>=0 GOTO 910
790 XW(2)=-AA(1)/(2*AA(2))
800 XW(1)=SQR(-WW)/(2*AA(2))
810 PRINT " KE";NW;" AKAR ADALAH:"
820 FOR JK = NW TO 3 STEP - 1
830 PRINT XW(JK)
840 NEXT
850 '
860 'Cetak akar Kompleks

```

```

870
880 PRINT XW(2);"+";XW(1);"i"
890 PRINT XW(2);"-";XW(1);"i"
900 GOTO 1030
910 FOR PAUSE=1 TO 4000:NEXT PAUSE
920 IF N > 0 THEN SCREEN 0:WIDTH 80:GOTO 300
930 '
940 'Cetak akar
950 '
960 PRINT "KE ";NW;" AKAR ADALAH:"
970 FOR JK = NW TO 1 STEP - 1
980 PRINT XW(JK);" ";
990 NEXT
1000 '
1010 'Option untuk reset kondisi
1020 '
1030 IF METH=1 THEN GOTO 1070
1040 PRINT "ANDA INGIN KONDISI YANG BARU(Y/T)";: INPUT V$
1050 IF V$ = "T" OR V$="t" THEN GOTO 1070
1060 SCREEN 0:WIDTH 80:PRINT CHR$(12):GOTO 290
1070 PRINT "ANDA INGIN RUN KEMBALI PROGRAM(Y/T)";: INPUT V$
1080 IF V$ = "T" OR V$="t" THEN GOTO 1100
1090 SCREEN 0:WIDTH 80:PRINT CHR$(12):GOTO 250
1100 CHAIN "numer1
2000 '***** Subroutine 1: Metode Newton-Raphson *****
2010 '
2020 PRINT CHR$(12):PRINT"KONVERGEN KE AKAR";(NW-N+1):PRINT
2030 FOR ITER=0 TO 40 STEP 2
2035 X=X1:GOSUB 140
2040 PRINT TAB(3) "x=";X1 TAB(20) "f(x)=";F
2050 A(ITER,3)=X1:A(ITER,4)=0
2060 A(ITER+1,3)=X1:A(ITER+1,4)=F
2070 IF ABS (F)< EPS GOTO 2140
2110 X1=X1-F/FP
2120 NEXT ITER
2130 PRINT"MELIWATI BATAS ITERASI:TIDAK KONVERGEN":
PRINT"ITER=";ITER:PRINT"EPS=";EPS:PRINT"F=";F: PRINT"KUNCI
FUNGSI F5 AKAN MELANJUTKAN PROGRAM":STOP
2140 RETURN
2150 '
3000 '***** Subroutine 2: Definisikan polinom *****
3010 '
3020 PRINT: PRINT "DERAJAT POLINOM";: INPUT N:NW = N
3030 PRINT:PRINT "KOEFSIEN POLINOM"

```

```

3040 FOR K = N TO 0 STEP - 1
3050 PRINT "A";K;"="";: INPUT AA(K)
3060 NEXT
3070 '
3080 PRINT:PRINT"ANDA INGIN MELIHAT SEMUA AKAR(Y/T)";:INPUT ALL
ROOT$
3090 PRINT:PRINT"ANDA INGIN GRAFIKNYA(Y/T)";:INPUT GRAPH$
3100 RETURN
3110 '
4000 '**** Subroutine 3: Mendefinisikan persamaan BEATTIE-BRIDGEMAN****
4010 '
4020 N=4:NW=N
4030 PRINT:PRINT"ANDA SUDAH MEMASUKKAN KONSTANTA BEATTIE-
BRIDGEMAN(Y/T)";
4040 INPUT VK$
4050 IF VK$="Y" OR VK$="y" GOTO 4090
4060 PRINT:PRINT"MASUKKAN KONSTANTA BEATTIE-BRIDGEMAN:"
4070 INPUT "AO=";AO:INPUT "BO=";BO:INPUT "ALPHA=";ALPHA
4080 INPUT "B=";B:INPUT "C=";C
4090 PRINT:PRINT"TEMPERATUR(KELVIN)="";:INPUT T
4100 PRINT "TEKANAN/PRESSURE(ATM)";: INPUT P
4110 R = .08206
4120 BETA = R * T * BO - AO - R * C / (T ^ 2)
4130 V=R*T/P
4140 GAMMA = - R * T * BO * B + ALPHA * AO - R * BO * C / (T ^ 2)
4150 DLTA = R * BO * B * C / (T ^ 2)
4160 AA(4)=P:AA(3)=-R*T:AA(2)=-BETA:AA(1)=-GAMMA:AA(0)=-DLTA
4170 PRINT : PRINT "VOLUME GAS IDEAL=";V
4180 PRINT: PRINT"BETA=";BETA:PRINT"GAMMA=";GAMMA
:PRINT"DLTA=";DLTA:PRINT
4190 PRINT:PRINT"INGIN MELIHAT SEMUA AKAR (Y/T)";:INPUT ALLROOT$
4200 PRINT:PRINT"SOLUSI SECARA GRAFIKAL (Y/T)";:INPUT GRAPH$
4210 RETURN
4220 '
10000 '***** Subroutine 4: Gambar Grafik*****
10010 '
10020 CLS:SCREEN 1
10030 'Menempatkan maksima dan minima untuk penskalaan Sumbu
10040 MINX=A(0,1):MAXX=A(0,1):MINY =A(0,2):MAXY =A(0,2)
10050 FOR K = 0 TO 100
10060 IF A(K ,1) < MINX THEN MINX = A(K ,1)
10070 IF A(K ,1) > MAXX THEN MAXX = A(K ,1)
10080 IF A(K ,2) < MINY THEN MINY = A(K ,2)
10090 IF A(K ,2) > MAXY THEN MAXY = A(K ,2)

```

```

10100 NEXT K
10110 'Sumbu Y
10115 IF MINY>0 THEN MAXY=1.1*MAXY: MINY=.9*MINY
10120 IF MAXY<0 THEN MAXY=.9*MAXY: MINY=1.1*MINY
10130 VIEW (40,5)-(315,145)
10140 WINDOW (MINX,MINY)-(MAXX,MAXY)
10150 'Menggambar Sumbu
10160 LINE(MINX,0)-(MAXX,0)
10170 LINE(0,MINY)-(0,MAXY)
10180 'Memberi Label Sumbu
10190 VP=1+(5+140*(MAXY/((MAXY-MINY))))/8
10200 HP=-1+(40+275*(ABS(MINX)/((MAXX-MINX))))/8
10210 IF (MINX/MAXX)>0 THEN LINE(MINX,MINY)-(MINX,MAXY):HP=4
10215 IF (MINY/MAXY)>0 THEN LINE(MINX,MINY)-(MAXX,MINY)
10220 LOCATE VP,40:PRINT"x"
10230 LOCATE 1,HP:PRINT"f(x)"
10240 'Mengatur nama Sumbu Vertikal
10250 IF LEN(NAM$)>16 GOTO 10310
10260 V1=3+(14-LEN(NAM$))/2
10270 FOR I=1 TO LEN(NAM$)
10280 LOCATE V1+I,2:PRINT MID$(NAM$,I,1)
10290 NEXT I
10300 GOTO 10340
10310 FOR I=1 TO 16
10320 LOCATE 3+I,2:PRINT MID$(NAM$,I,1)
10330 NEXT I
10340 'Mengatur nama Sumbu horisontal
10350 IF LEN(TIT$)>35 GOTO 10390
10360 LOCATE 22,5+(36-LEN(TIT$))/2
10370 PRINT TIT$
10380 GOTO 10400
10390 LOCATE 22,5:PRINT MID$(TIT$,1,36)
10400 'Menggambar Fungsi
10410 X=A(0,1):Y=A(0,2)
10420 PSET(X,Y)
10430 FOR I=1 TO 100
10440 X=A(I,1):Y=A(I,2)
10450 LINE -(X,Y)
10460 NEXT I
10470 'Menempatkan Akar
10480 FOR I=0 TO ITER
10490 LINE(A(I,3),A(I,4))-(A(I+1,3),A(I+1,4))
10500 FOR PAUSE=1 TO 500:NEXT PAUSE
10510 NEXT I

```

```
10520 RETURN
10530 '
```

Program 25

```
0 SCREEN 0:WIDTH 80:CLS
5 PRINT "*****"
10 PRINT"*"
15 PRINT"* CONTOHPROGRAM"
20 PRINT"*"
25 PRINT"* SEKAN/INTERPOLASI LINIER dan NEWTON-RAPHSON *"
30 PRINT"*"
35 PRINT"* UNTUK SOLUSI PERSAMAAN NONLINIER"
40 PRINT"*"
50 PRINT"*"
85 PRINT"*****"
99 INPUT"TEKAN ENTER ";ZZ$
100 "***** Program Utama *****"
110 DIM A(100,4)
120 'Berikut ini adalah fungsi,
130 'turunannya dan namanya,
140 'ia dapat didefinisikan oleh anda
150 '
160 DEF FNF(X)=SIN(X)
170 DEF FNFP(X)=COS(X)
180 NAM$="SINUS"
190 '
200 '
210 '
220 '
230 'Memilih metode yang digunakan
240 '
250 CLS:PRINT"FUNGSI ANDA ADALAH FUNGSI
";NAM$:PRINT:PRINT"METODE YANG DIGUNAKAN:"
260 PRINT:PRINT" 1. INTERPOLASI LINIER / REGULAFALSI"
270 PRINT:PRINT" 2. NEWTON-RAPHSON"
280 PRINT:PRINT"PILIHAN ANDA";:INPUT METH
290 IF METH<1 OR METH>2 GOTO 280
300 IF METH=1 THEN TIT$="METODE INTERPOLASI LINIER"
305 IF METH=2 THEN TIT$="METODE NEWTON-RAPHSON "
310 '
320 'Masukkan Variabel program
330 '
340 PRINT: PRINT "DEFINISIKAN RANGE DARI SEARCH:"
350 PRINT "LIMIT BAWAH";: INPUT MINX
```

```

360 PRINT "LIMIT ATAS"; INPUT MAXX
370 IF METH=2 GOTO 400
380 PRINT:PRINT "BERIKAN HARGA AWAL DARI SEARCH.":PRINT"YANG
SATU LEBIH KECIL DAN YANG LAIN LEBIH BESAR DARI AKAR.":
INPUT X1:INPUT X2
390 GOTO 410
400 PRINT:PRINT"BERIKAN DUGAAN TERBAIK UNTUK AKAR";:INPUT X1
410 PRINT:PRINT "BERIKAN KRITERIA KONVERGENSI(EPSILON) UNTUK
F ";:INPUT EPS
420 '
430 'Menghitung fungsi dalam range dari search
440 '
470  $DX = (MAXX - MINX) / 100$ 
480 FOR K = 0 TO 100
490  $X = MINX + K * DX$ 
500  $A(K,1) = X$ 
510  $A(K,2) = FNF(X)$ 
520 NEXT K
530 '
540 ON METH GOSUB 1000,2000 'Rutin Mencari Akar
550 '
560 PRINT:PRINT"TEKAN SEBARANG KUNCI UNTUK
MELANJUTKAN":HK$=INPUT$(1)
570 GOSUB 10000 'routine gambar grafik
650 '
660 'Pilihan untuk proses lagi
670 '
680 PRINT: PRINT "ANDA INGIN PROSES LAGI (Y/T)";: INPUT V$
690 IF V$ = "T" OR V$="t" THEN 710
700 IF V$="Y" OR V$="y" THEN PRINT CHR$(12):SCREEN 0:WIDTH 80:COLOR
15,0,5:GOTO 250
703 GOTO 680
710 END
1000 ***** Subroutine 1: Metode Interpolasi Linier *****
1010 '
1020 PRINT:PRINT"KONVERGEN KE AKAR.":PRINT
1030 PRINT TAB(3) "x=";X1 TAB(20) "f(x)=";FNF(X1)
1040 FOR ITER=0 TO 40 STEP 2
1050 PRINT TAB(3) "x=";X2 TAB(20) "f(x)=";FNF(X2)
1060  $A(ITER,3)=X1:A(ITER,4)=FNF(X1)$ 
1070  $A(ITER+1,3)=X2:A(ITER+1,4)=FNF(X2)$ 
1080 IF ABS (FNF(X2))< EPS GOTO 1140
1090  $F1=FNF(X1)$ 
1100  $F2=FNF(X2)$ 

```

```

1110 X2=X1-(F1/((F2-F1)/(X2-X1)))
1120 NEXT ITER
1130 PRINT"MELAMPAUI BATAS ITERASI:TIDAK KONVERGEN
    ":PRINT"ITERASI=";ITER: PRINT"EPSILON=";EPS:PRINT"F=";F2:
    PRINT"TEKAN KUNCI F5 UNTUK MELANJUTKAN ":STOP
1140 RETURN
1150 '
2000 '***** Subroutine 2: Metode Newton-Raphson *****
2010 '
2020 PRINT:PRINT"KONVERGEN KE AKAR:":PRINT
2030 FOR ITER=0 TO 40 STEP 2
2040 PRINT TAB(3) "x=";X1 TAB(20) "f(x)=";FNF(X1)
2050 A(ITER,3)=X1:A(ITER,4)=0
2060 A(ITER+1,3)=X1:A(ITER+1,4)=FNF(X1)
2070 IF ABS (FNF(X1))< EPS GOTO 2140
2090 F1=FNF(X1)
2100 FP=FNFP(X1)
2110 X1=X1-F1/FP
2120 NEXT ITER
2130 PRINT"MELEBIHI BATAS ITERASI:TIDAK KONVERGEN":
    PRINT"ITERASI=";ITER:PRINT"EPSILON=";EPS:PRINT"F=";F1:
    PRINT"TEKAN KUNCI F5 UNTUK MELANJUTKAN PROGRAM":STOP
2140 RETURN
2150 '
10000 '***** Subroutine 3: Menggambar grafik *****
10010 '
10020 CLS:SCREEN 1
10030 'maxima and minima untuk menskala sumbu
10040 MINX=A(0,1):MAXX=A(0,1):MINY =A(0,2):MAXY =A(0,2)
10050 FOR K = 0 TO 100
10060 IF A(K ,1) < MINX THEN MINX = A(K ,1)
10070 IF A(K ,1) > MAXX THEN MAXX = A(K ,1)
10080 IF A(K ,2) < MINY THEN MINY = A(K ,2)
10090 IF A(K ,2) > MAXY THEN MAXY = A(K ,2)
10100 NEXT K
10110 'Expand the y-axis
10115 IF MINY>0 THEN MAXY=1.1*MAXY: MINY=.9*MINY
10120 IF MAXY<0 THEN MAXY=.9*MAXY: MINY=1.1*MINY
10130 VIEW (40,5)-(315,145)
10140 WINDOW (MINX,MINY)-(MAXX,MAXY)
10150 'Menggambar sumbu
10160 LINE(MINX,0)-(MAXX,0)
10170 LINE(0,MINY)-(0,MAXY)

```

```

10180 'Memberi Label Sumbu
10190 VP=1+(5+140*(MAXY/((MAXY-MINY))))/8
10200 HP=-1+(40+275*(ABS(MINX)/((MAXX-MINX))))/8
10210 IF (MINX/MAXX)>0 THEN LINE(MINX,MINY)-(MINX,MAXY):HP=4
10215 IF (MINY/MAXY)>0 THEN LINE(MINX,MINY)-(MAXX,MINY)
10220 LOCATE VP,40:PRINT"x"
10230 LOCATE 1,HP:PRINT"f(x)"
10240 'Center vertical axis title
10250 IF LEN(NAM$)>16 GOTO 10310
10260 V1=3+(14-LEN(NAM$))/2
10270 FOR I=1 TO LEN(NAM$)
10280 LOCATE V1+I,2:PRINT MID$(NAM$,I,1)
10290 NEXT I
10300 GOTO 10340
10310 FOR I=1 TO 16
10320 LOCATE 3+I,2:PRINT MID$(NAM$,I,1)
10330 NEXT I
10340 'Center horizontal title
10350 IF LEN(TIT$)>35 GOTO 10390
10360 LOCATE 22,5+(36-LEN(TIT$))/2
10370 PRINT TIT$
10380 GOTO 10400
10390 LOCATE 22,5:PRINT MID$(TIT$,1,36)
10400 'Menggambar fungsi
10410 X= A(0,1):Y=A(0,2)
10420 PSET(X,Y)
10430 FOR I=1 TO 100
10440 X=A(I,1) :Y=A(I,2)
10450 LINE -(X,Y)
10460 NEXT I
10470 'Menempatkan akar
10480 FOR I=0 TO ITER
10490 LINE(A(I,3),A(I,4))-(A(I+1,3),A(I+1,4))
10500 FOR PAUSE=1 TO 500:NEXT PAUSE
10510 NEXT I
10520 RETURN
10530 '

```