G-Tech
Jurnal Teknologi Terapan

FTIKA Unira Malang | Vol. 1 | No. 1
Oktober, 2017
P-ISSN 2580-8737

SOLUSI NUMERIK PERSAMAAN NON-LINIER DENGAN METODE *BISECTION* DAN *REGULA FALSI*

Jatining Wigati
Pendidik Bidang Matematika, SMA Negeri 1 Sumberpucung, Kab. Malang
jatiwigati@gmail.com

ABSTRAK

Model matematika sangat dibutuhkan untuk memecahkan masalah dalam berbagai disiplin ilmu pengetahuan; bidang fisika, kimia, ekonomi, bidang rekayasa atau teknik. Model matematika yang disedehanakan dan linier sering tidak representatif untuk diaplikasikan dalam desain. Oleh karena itu dibutuhkan solusi numerik untuk persamaan non-linier agar dapat diaplikasikan secara nyata pada bidang desain. Tujuan percobaan makalah ini adalah untuk membandingkan akurasi dan kecepatan iterasi metode *bisection* dan *regula falsi* menggunakan Scilab v.6.6.0, dan untuk mendeskripsikan secara objektif aplikasi penyelesaian secara numerik pada desain. Persamaan non-linier yang digunakan sebagai simulasi adalah fungsi transedental, f(x), yang mengandung trigonometri. Domain x ditentukan -4 < x < 4. Hasil percobaan menunjukkan bahwa: 1) pada nilai error $1x10^{-5}$, didapatkan akar-akar persamaan yang sama. Meskipun demikian, terdapat perbedaan jumlah iterasi yang dibutuhkan masing-masing metode; metode bisection memer-lukan 19 kali iterasi, sedangkan metode regula falsi hanya memer-lukan 8 dan 12 kali iterasi untuk menemukan akar persamaan positif dan negative secara berturut-turut, dan 2) pemrograman ini dapat digunakan untuk menyelesaikan persamaan non-linier lain, baik aljabar maupun transedental, dengan mengganti definisi f(x) di awal program.

Kata Kunci: *bisection,* non-linier, numerik, *regulasi falsi*, solusi

PENDAHULUAN

Model matematika sangat dibutuhkan untuk memecahkan masalah dalam berbagai disiplin ilmu pengetahuan; bidang fisika, kimia, ekonomi, bidang rekayasa atau teknik (Mulyono, 2007). Model matematika tersebut hampir selalu disederhanakan agar dapat diselesaikan secara analitik. Untuk mendapatkan hasil yang mendekati sebenarnya, penyerdehanaan persamaan bukan solusi yang terbaik.

Metode analitik merupakan metode penyelesaian dengan rumus-rumus aljabar yang sudah baku atau lazim digunakan. Metode analitik memiliki keterbatasan dalam hal derajat persamaan yang tinggi, di mana justru persamaan dengan derajat tinggi sering dijumpai dalam bidang rekayasa atau teknik untuk menyelesaikan permasalahan pabrik, lapangan, pemodelan, dan sebagainya. Sebagai contoh, persamaan non-linier adalah persamaan gas riil, yang signifikan untuk bidang rekayasa atau teknik berhubungan dengan termodinamika. Persamaan gas riil Beattie-Bridgeman ini menyatakan hubungan antara suhu-tekanan-volume (Sasongko, 2010):

$$PV = RT + \frac{\beta}{V} + \frac{\gamma}{V^2} + \frac{\delta}{V^3}$$

Penyelesaian persamaan gas riil hampir selalu dilakukan dengan metode pendekatan atau koreksi dari persamaan gas ideal karena mustahil mengetahui keadaan sebenarnya dari fluida gas yang mengalir di suatu wadah. Namun pendekatan tersebut tidak akurat untuk memberikan kalkulasi *variabel volume* yang tepat. Keakuratan perhitungan ini sangat penting karena kesalahan perhitungan *variabel volume* sangat berpengaruh pada perencanaan/desain; kapasitas pompa, kapasitas pipa, ukuran kontainer, mesin, dan akhirnya kecelakaan kerja.

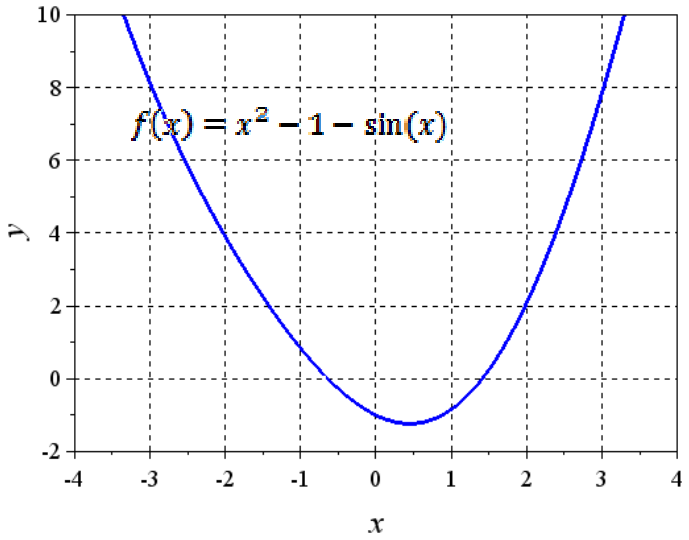
Penjelasan di atas merupakan dasar mengapa penyelesaian persamaan non-linier yang akurat sangat penting untuk implikasi berbagai bidang yang menggunakan persamaan linier sebagai alat penyelesaian masalah maupun pemodelan.

METODE

Metode yang digunakan pada karya tulis ini adalah simulasi numerikmenggunakan perangkat lunak **Scilab v.6.0.0**. Metode dan perangkat lunak ini dipilih karena penyelesaian analitik tidak memberikan nilai praktis dan terbatas. Hasil penyelesaian dan grafik, akan disajikan pula dengan bahasa pemrograman perangkat lunak ini. Analisis yang dilakukan adalah memberikan perbandingan akurasi antara metode *bisection* dan *regula falsi.*

Penyelesaian secara numerik membutuhkan proses iterasi (perhitungan berulang) dari data numerik yang ada. Pengunaan perangkat lunak Scilab v.6.0.0 akan sangat mengurangi waktu iterasi. Namun demikian, karena penyelesaian secara numeric juga merupakan *pendekatan*, kesalahan (*error*) tak terhindarkan. *Error* berhubungan erat dengan akurasi atau menyatakan seberapa jauh solusi tersebut dari nilai eksak.

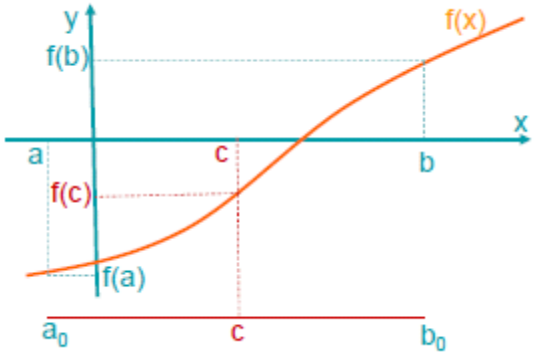
Solusi yang dimaksud adalah suatu nilai variable x sedemikian rupa, sehingga terpenuhi persamaan f(x) = 0 yang digunakan dalam model. Secara geometri, ini berarti mencari suatu titik f(x) tepat memotong sumbu x. Untuk makalah ini, ditentukan f(x) sebagai fungsi transedental yang mengandung trigonometri.



Gambar 1. Plot fungsi persamaan -4 < x < 4 dengan interval 0,001

Solusi persamaan non-linier metode *bisection*

Metode *bisection* pada prinsipnya adalah mencari nilai solusi yang memiliki *error* paling kecil di antara 2 (dua) nilai a dan b, seperti diilustrasikan pada Gambar 2.



Gambar 2. Ilustrasi penyelesaian akar persamaan dengan metode *bisection*

Penyelesaian menggunakan metode *bisection* perlu memperhatikan hal-hal berikut ini:

1. Fungsi harus kontinu pada interval x_n dan x_{n+1}
2. Nilai x_n dan x_{n+1} dapat diperoleh dengan membuat grafik fungsinya

3. Nilai toleransi (*error*) dapat ditentukan oleh pengguna ataupun didasarkan pada bidang ilmu dan permasalahan yang diselesaikan

Algoritma penyelesaian menggunakan metode *bisection* dibagi menjadi 3 (tiga) bagian utama sebagai berikut:

a. Menentukan dua nilai x awal, yaitu a dan b, dimana a < b, serta f(a) dan f(b), dimana f(a) x f(b) < 0
b. Menentukan nilai c, yaitu (a+b)/2, dan nilai f(c)
c. Melakukan iterasi dengan parameter: Jika f(a) x f(c) < 0 maka c = b, tetapi sebaliknya Jika f(a) x f (b) > 0 maka c = a, dan seterusnya hingga nilai (b-a) < e yang ditentukan pengguna.

Skrip pada perangkat lunak Scilab v.6.0.0 disusun menggunakan algoritma tersebut. Hasil skrip terlampir di bawah ini.

```
clear;

deff('[y]=f(x)','y = x.^2-1-sin(x)');   //definition of y=f(x)

//GRAPH
x_data = -4:0.001:4;    //data from -4 to 4 with interval of 0.001
clf;                    //clear existing graphs
plot(x_data,f(x_data),'linewidth',3);   //draw y =f(x) with a
xgrid;
xlabel('x');
ylabel('y');
   //GRAPH decoration
   graph=gca();
   graph.font_style=2;
   graph.font_size=4;
   graph.x_label.font_style=3;
   graph.x_label.font_size=5;
   graph.y_label.font_style=3;
   graph.y_label.font_size=5;
   graph.data_bounds=[-4 -2; 4 10];       //[xmin ymin;xmax ymax]

printf("\n")
printf("Input a and b (a<b and f(a)*f(b)<0.\n")
printf("a=?\n");a=scanf("%f")                 //step a
printf("b=?\n");b=scanf("%f")

c=(a+b)/2;                                    //step b
e=1D-05;

k=1;

if a>b then
   flag=1;
elseif f(a)*f(b) >= 0
   flag=2;
else
   flag=0;
end

while flag==0 & (b-a)>e                       //step c
   plot (c, f(c), 'go');
   printf("k=%3d, a=%f, b=%f, c=%f, b-a=%f\n", k, a, b, c, b-a)

   if f(a)*f(c)<0
      b=c;
   else //f(a)*f(c)>=0
```

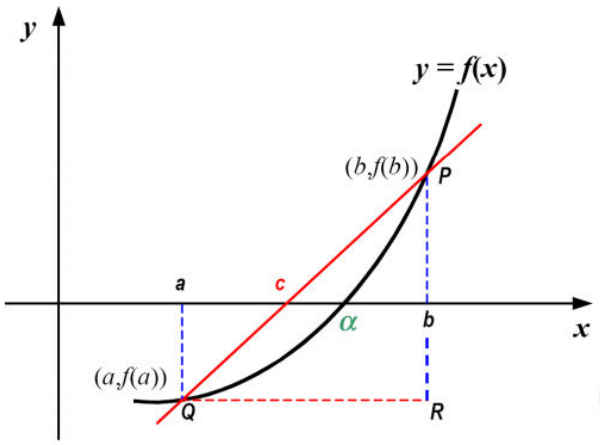
```
        a=c;
    end

    k= k+1;
    c=(a+b)/2;                          //step b
end

if flag==0 then
    printf("k=%3d, a=%f, b=%f, c=%f, b-a=%f\n\n", k, a, b, c, b-a)
    printf("Solution: x=%f\n", c)
elseif flag==1
    printf("ERROR! The value of a>b!\n")
elseif flag==2
    printf("ERROR! The value of f(a)*f(b) is positive!\n")
end
```

Solusi persamaan non-linier metode *regula falsi*

Metode *regula falsi* adalah metode pencarian akar persamaan dengan memanfaatkan kemiringan dan selisih tinggi dari 2 (dua) titik batas range (Amang, 2006), seperti diilustrasikan pada **Gambar 3**.



Gambar 3. Ilustrasi penyelesaian akar persamaan dengan metode *regula falsi*

Algoritma penyelesaian menggunakan metode *regula falsi* menyerupai algorima *bisection*, dibagi menjadi 3 (tiga) bagian utama sebagai berikut:

a. Menentukan dua nilai x awal, yaitu a dan b, dimana a < b, serta f(a) dan f(b), dimana f(a) x f(b) < 0. Menentukan nilai c, yaitu $\frac{\{a \times f(b)\} - \{b \times f(a)\}}{f(b) - f(a)}$, dan nilai f(c)
b. Melakukan iterasi dengan parameter : Jika f(a) x f(c) < 0 maka c = b, tetapi sebaliknya Jika f(a) x f (b) > 0 maka c = a, dan seterusnya hingga nilai (b-a) < e yang ditentukan pengguna.

Skrip pada perangkat lunak Scilab v.6.0.0 disusun menggunakan algoritma tersebut. Hasil skrip terlampir di bawah ini.

```
clear;

deff('[y]=f(x)','y = x.^2-1-sin(x)');       //definition of y=f(x)

  //GRAPH
  x_data = -4:0.001:4;            //data from -4 to 4 with interval of 0.001
  clf;                            //clear existing graphs
  plot(x_data,f(x_data),'linewidth',3);     //draw y =f(x) with a
  xgrid;
  xlabel('x');
  ylabel('y');
```

```
    //GRAPH decoration
    graph=gca();
    graph.font_style=2;
    graph.font_size=4;
    graph.x_label.font_style=3;
    graph.x_label.font_size=5;
    graph.y_label.font_style=3;
    graph.y_label.font_size=5;
    graph.data_bounds=[-4 -2; 4 10];       //[xmin ymin;xmax ymax]

printf("\n")
printf("Input a and b (a<b and f(a)*f(b)<0.\n")
printf("a=?\n");a=scanf("%f")                  //step a
printf("b=?\n");b=scanf("%f")

c=(a*f(b)-b*f(a))/(f(b)-f(a));                 //step b
e=1D-05;

k=1;

if a>b then
    flag=1;
elseif f(a)*f(b) >= 0
    flag=2;
else
    flag=0;
end

while flag==0 & abs(f(c))>e                    //step c
    plot(c,f(c),'ro')
    printf("k=%3d, a=%f, b=%f, c=%f, b-a=%f\n", k, a, b, c, abs(f(c)))

    if f(a)*f(c)<0
        b=c;
    else //f(a)*f(c)>=0
        a=c;
    end

    k= k+1;
    c=(a*f(b)-b*f(a))/(f(b)-f(a));              //step b
end

if flag==0 then
    printf("k=%3d, a=%f, b=%f, c=%f, b-a=%f\n\n", k, a, b, c, b-a)
    printf("Solution: x=%f\n", c)
elseif flag==1
    printf("ERROR! The value of a>b!\n")
elseif flag==2
    printf("ERROR! The value of f(a)*f(b) is positive!\n")
end
```

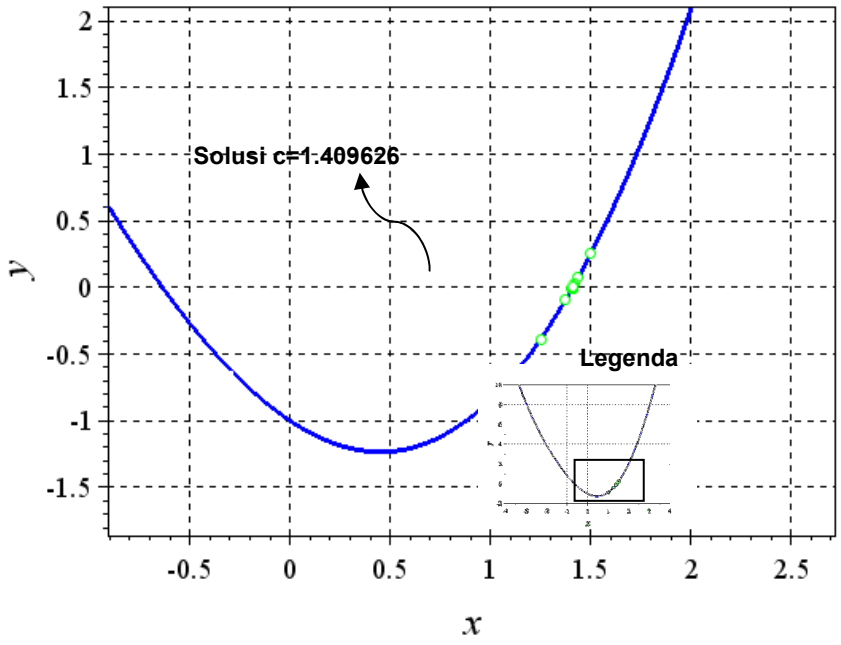
HASIL DAN PEMBAHASAN

Metode Bisection

Gambar 1 menunjukkan bahwa terdapat akar-akar persamaan yang bernilai positif (Tabel 1a) dan negatif (Tabel 1b). Tabel 1a dan 1b menunjukkan jumlah iterasi (k), nilai a, b, c, dan *error* (b-a). Nilai maksimum *error* yang ditentukan pada skrip adalah $1x10^{-5}$. Maka, ketika (b-a) < $1x10^{-5}$, iterasi dihentikan dan didapatkan solusi persamaan (c) pada nomor iterasi yang sama.

Tabel 1a. Tabulasi Iterasi Metode *Bisection* (akar positif)

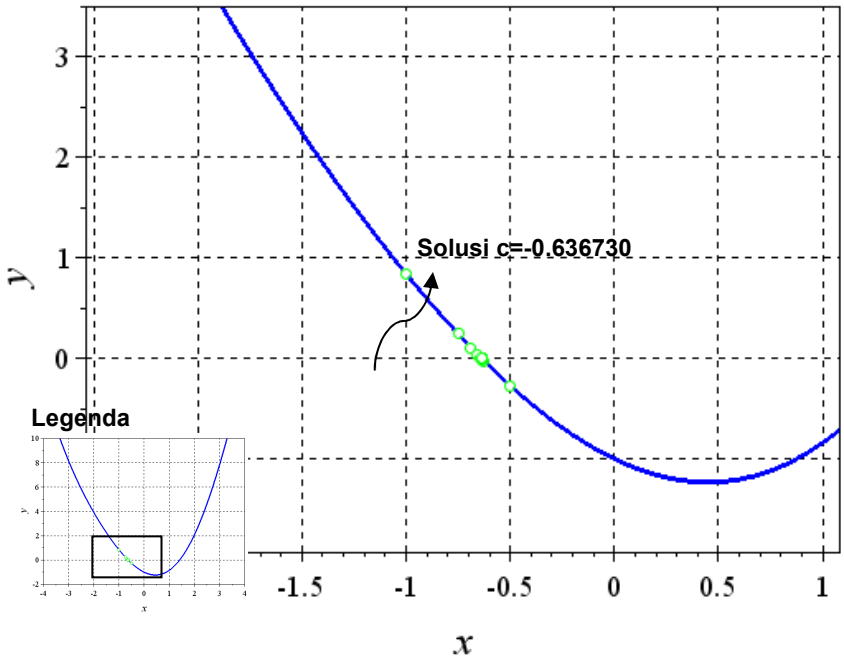
k	a	b	c	b-a
1	0.000000	2.000000	1.000000	2.000000
2	1.000000	2.000000	1.500000	1.000000
3	1.000000	1.500000	1.250000	0.500000
4	1.250000	1.500000	1.375000	0.250000
5	1.375000	1.500000	1.437500	0.125000
6	1.375000	1.437500	1.406250	0.062500
7	1.406250	1.437500	1.421875	0.031250
8	1.406250	1.421875	1.414063	0.015625
9	1.406250	1.414063	1.410156	0.007813
10	1.406250	1.410156	1.408203	0.003906
11	1.408203	1.410156	1.409180	0.001953
12	1.409180	1.410156	1.409668	0.000977
13	1.409180	1.409668	1.409424	0.000488
14	1.409424	1.409668	1.409546	0.000244
15	1.409546	1.409668	1.409607	0.000122
16	1.409607	1.409668	1.409637	0.000061
17	1.409607	1.409637	1.409622	0.000031
18	1.409622	1.409637	1.409630	0.000015
19	1.409622	1.409630	1.409626	0.000008



Gambar 4a. Penyelesaian akar positif metode *bisection*

Tabel 1b. Tabulasi Iterasi Metode *Bisection* (akar negatif)

k	a	b	c	b-a
1	-2.000000	0.000000	-1.000000	2.000000
2	-1.000000	0.000000	-0.500000	1.000000
3	-1.000000	-0.500000	-0.750000	0.500000
4	-0.750000	-0.500000	-0.625000	0.250000
5	-0.750000	-0.625000	-0.687500	0.125000
6	-0.687500	-0.625000	-0.656250	0.062500
7	-0.656250	-0.625000	-0.640625	0.031250
8	-0.640625	-0.625000	-0.632813	0.015625
9	-0.640625	-0.632813	-0.636719	0.007813
10	-0.640625	-0.636719	-0.638672	0.003906
11	-0.638672	-0.636719	-0.637695	0.001953
12	-0.637695	-0.636719	-0.637207	0.000977
13	-0.637207	-0.636719	-0.636963	0.000488
14	-0.636963	-0.636719	-0.636841	0.000244
15	-0.636841	-0.636719	-0.636780	0.000122
16	-0.636780	-0.636719	-0.636749	0.000061
17	-0.636749	-0.636719	-0.636734	0.000031
18	-0.636734	-0.636719	-0.636726	0.000015
19	-0.636734	-0.636726	-0.636730	0.000008



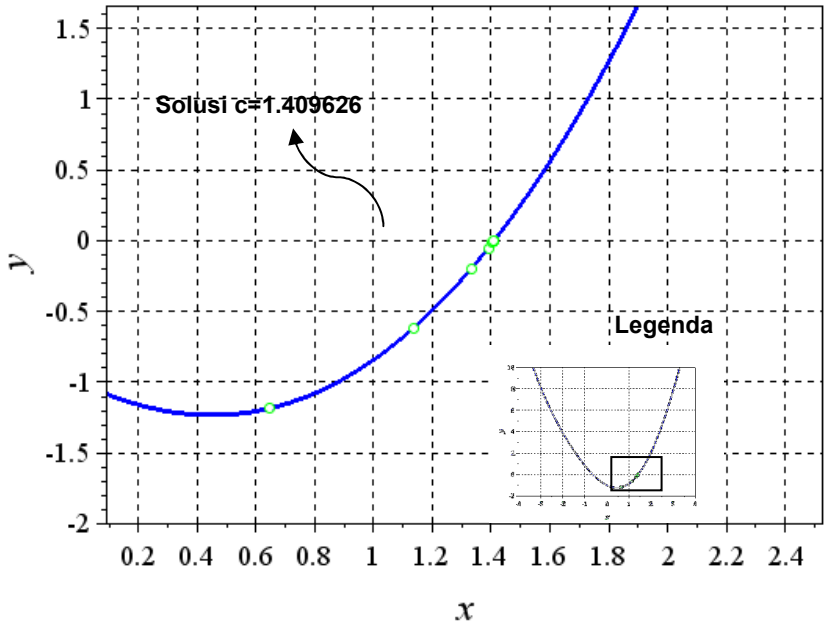
Gambar 4b. Penyelesaian akar negatif metode *bisection*

Metode Regula Falsi

Tabulasi iterasi penyelesaian dengan metode *regula falsi* disajikan pada Tabel 2a dan Tabel 2b. Tabel 2a dan 2b menunjukkan jumlah iterasi (k), nilai a, b, c, dan *error* (b-a). Nilai maksimum *error* yang ditentukan pada skrip sama dengan pada metode *bisection* yaitu $1x10^{-5}$. Maka, ketika (b-a) < $1x10^{-5}$, iterasi dihentikan dan didapatkan solusi persamaan (c) pada nomor iterasi yang sama.

Tabel 2a. Tabulasi Iterasi Metode *Regula Falsi* (akar positif)

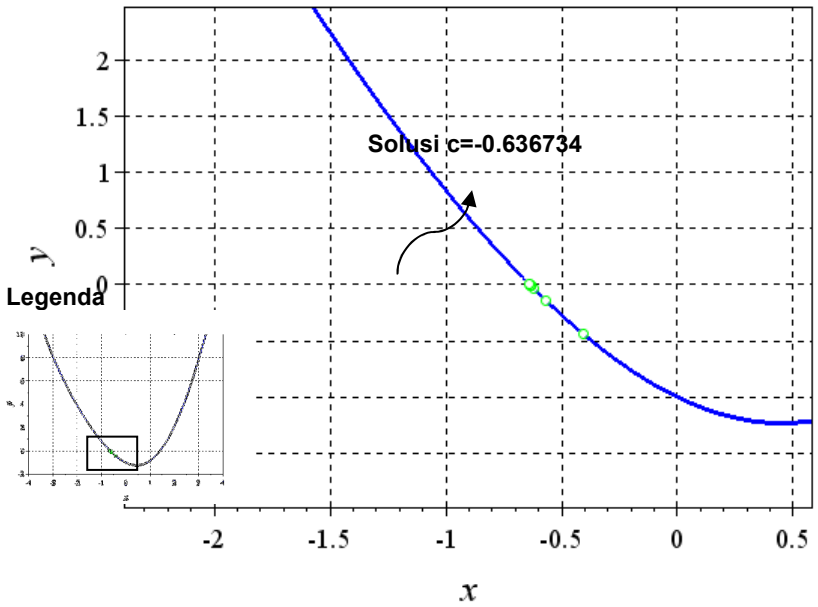
k	a	b	c	|f(c)|
1	1.000000	4.000000	1.152089	0.586306
2	1.000000	1.152089	1.501552	0.257056
3	1.501552	1.152089	1.395036	0.038468
4	1.395036	1.152089	1.412095	0.006580
5	1.412095	1.152089	1.409210	0.001101
6	1.409210	1.152089	1.409694	0.000185
7	1.409694	1.152089	1.409612	0.000031
8	1.409612	1.152089	1.409626	0.000005



Gambar 5a. Penyelesaian akar positif metode *regula falsi*

Tabel 2b. Tabulasi Iterasi Metode *Regula Falsi* (akar negatif)

k	a	b	c	|f(c)|
1	-4.000000	0.000000	-0.262412	0.671729
2	-0.262412	0.000000	-0.799377	0.355924
3	-0.799377	0.000000	-0.589544	0.096457
4	-0.589544	0.000000	-0.652480	0.032888
5	-0.652480	0.000000	-0.631704	0.010429
6	-0.631704	0.000000	-0.638362	0.003386
7	-0.638362	0.000000	-0.636207	0.001091
8	-0.636207	0.000000	-0.636902	0.000352
9	-0.636902	0.000000	-0.636678	0.000114
10	-0.636678	0.000000	-0.636750	0.000037
11	-0.636750	0.000000	-0.636727	0.000012
12	-0.636727	0.000000	-0.636734	0.000004



Gambar 5b. Penyelesaian akar negatif metode *regula falsi*

Perbandingan Metode Bisection dan Metode Regula Falsi

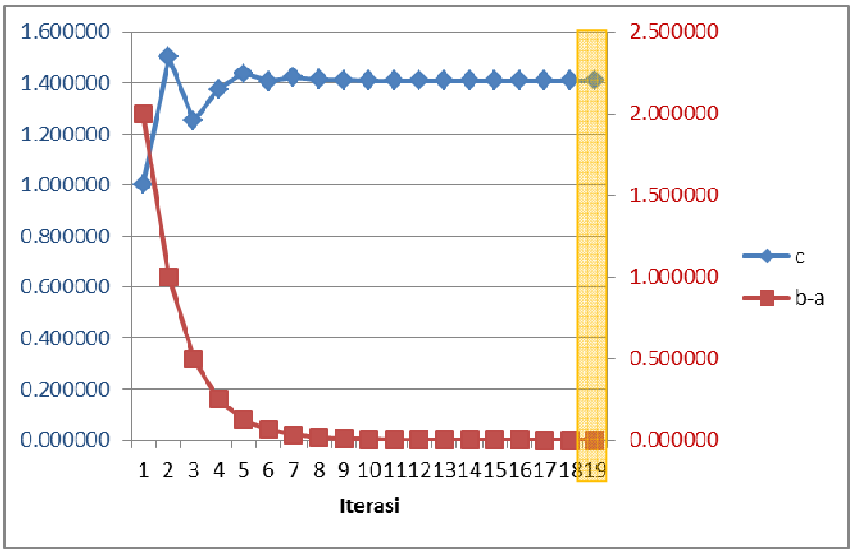
Akar-akar dari persamaan non-linier pada makalah ini dicari dengan dua metode, yaitu metode *bisection* dan metode *regula falsi*. Untuk menentukan akurasi antara dua metode sangat bergantung pada pengguna. Penentuan akurasi itu sendiri bergantung pada aplikasi fisik yang diterapkan setelah perhitungan. Sebagai contoh, satuan volume yang mencapai puluh ribuan meter kubik tidak memerlukan sampai 6 desimal karena 0,000001 m^3 tidak akan menjadi signifikan. Pertimbangan lain penentuan akurasi adalah ketelitian alat ukur yang digunakan; misalkan alat ukur penggaris memiliki ketelitian sampai 0.01 cm maka *error* yang dipasang maksimal $1x10^{-2}$.

Akar-akar persamaan yang memiliki nilai y=f(x) lebih mendekati 0 adalah nilai akar persamaan yang lebih mendekati nilai eksak. Rekapitulasi akar-akar persamaan pada masing-masing metode disajikan pada Tabel 3.

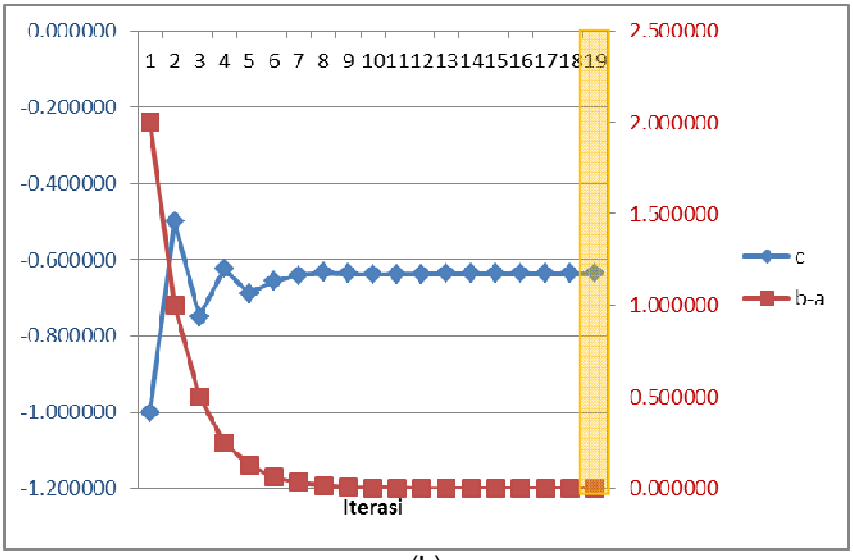
Tabel 3. Rakapitulasi Perhitungan

Metode	*Akar Negatif*	*Akar Positif*
Bisection	-0.636730	1.409626
Regula Falsi	-0.636734	1.409626

Error yang dipasang pada setiap metode adalah $1x10^{-5}$. Dengan *error* yang bernilai demikian, akar persamaan ditentukan dengan iterasi. Hasil menunjukkan bahwa terdapat perbedaan jumlah iterasi pada masing-masing metode untuk mendapatkan akar persamaan kurang dari *error* $1x10^{-5}$. Grafik *error* dan akar persamaan terhadap n-iterasi masing-masing metode ditunjukkan pada Gambar 6(a-b) dan Gambar 7(a-b).



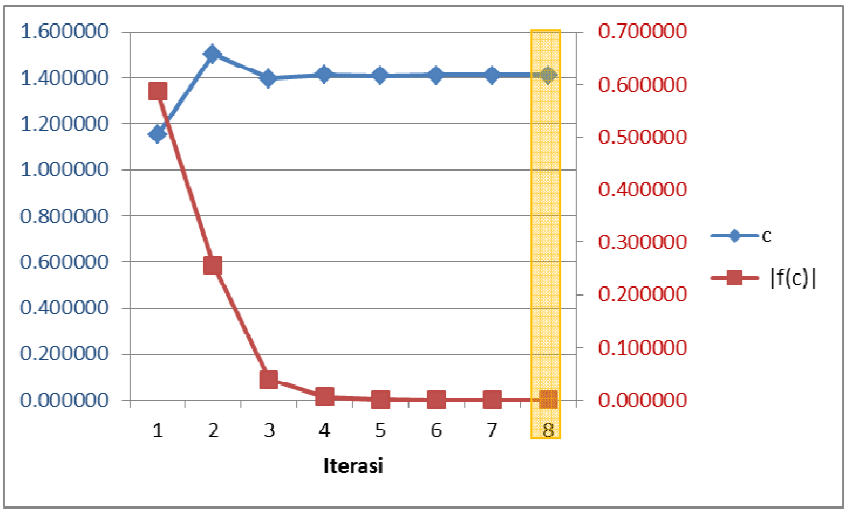
(a)



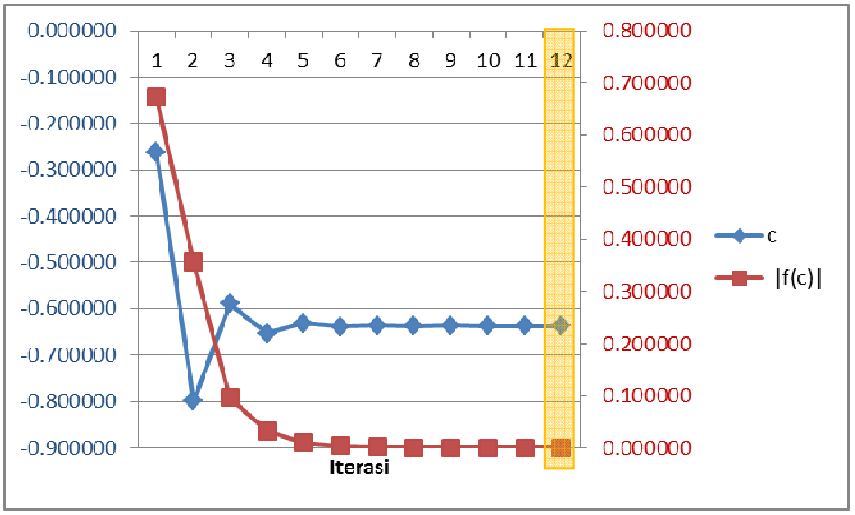
(b)

Gambar 6. Analisis *error* metode *bisection:* (a) akar positif dan (b) akar negatif

Pada penyelesaian akar-akar persamaan dengan menggunakan metode *bisection*, akar positif dan akar negatif ditemukan pada iterasi ke-19. Sedangkan pada metode *regula falsi*, akar positif ditemukan pada iterasi ke-8 dan akar negatif ditemukan pada iterasi ke-12. Membandingkan jumlah alterasi dengan nilai yang direkapitulasi pada Tabel 3 menunjukkan bahwa dengan hasil yang relatif sama dan akurasi hampir sama, metode *regula falsi* terbukti lebih efisien untuk menyelesaian persamaan non-linier.



(a)



(b)

Gambar 7. Analisis *error* metode *regula falsi:* (a) akar positif dan (b) akar negatif

Terlepas dari itu, masing-masing metode memiliki kelebihan dan kekurangan yang diuraikan pada Tabel 4. Masalah konvergensi juga menjadi isu penting dalam penyelesaian persamaan non-linier. Pada metode *bisection*, konvergensi dinyatakan :

1. a < b
2. f(a)*f(b) > 0
3. (b-a) < 0

Sedangkan pada metode *regula falsi*, kovergensi dinyatakan:

1. a < b
2. f(a)*f(b) > 0
3. |f(c)| > 0

Tabel 4. Analisis Kelebihan dan Kekurangan Metoda *Bisection*

	Bisection	Regula Falsi
Kelebihan	Selalu berhasil menemukan akas (solusi) yang dicari. Dengan kata lain, metode ini selalu konvergen → *step c*	Selalu berhasil menemukan akas (solusi) yang dicari. Dengan kata lain, metode ini selalu konvergen → *step c*
Kekurangan	• Metode biseksi hanya dapat dilakukan apabila ada akar persamaan pada interval yang diberikan. • Jika ada beberapa akar pada interval yang diberikan maka hanya satu akar saja yang dapat ditemukan. • Memiliki proses iterasi yang banyak sehingga memperlama • Proses penyelesaian. Tidak memandang bahwa sebenarnya akar atau solusi yang dicari dekat sekali dengan batas interval yang digunakan.	• Kecepatan atau laju konvergensi dari metode *regula-falsi* sama dengan metode *bisection*, yaitu konvergensi linier, namun dengan faktor pengali (konstanta) yang lebih besar dari 1 2 (factor pengali berkisar antara 1/ 2 … 1). • Oleh karena itu, proses *running* program lebih lambat untuk mencapai konvergensi.

Aplikasi

Algoritma dan skrip yang tertera pada makalah ini aplikatif untuk diterapkan pada persamaan non-linier lainnya, misalnya seperti persamaan non-linier gas riil, atau persamaan transfer panas, dengan mengubah definisi f(x) di awal skrip Scilab v.6.0.0. Sedangkan nilai maksimum *error* yang diperbolehkan bergantung pada ketelitian alat ukur, atau angka penting (*significant number*), atau justifikasi professional pengguna.

KESIMPULAN DAN SARAN

Kesimpulan

1. Program komputer pada perangkat luank Scilab v.6.0.0 untuk mencari akar-akar persamaan transedental mengeluarkan hasil berupa :
   - ✓ Nilai akar-akar persamaan non-linier melalui beberapa alterasi sampai lebih kecil dari nilai *error* yang ditentukan
   - ✓ Nilai *error* yang didapat pada masing-masing iterasi
   - ✓ Nilai fungsi, f(c), dari salah satu akar pendekatan yang diperoleh
2. Nilai akurasi yang ditentukan pada setiap metode adalah $1x10^{-5}$. Pada nilai *error* yang demikian, didapatkan akar-akar persamaan yang sama. Meskipun demikian, terdapat perbedaan jumlah iterasi yang dibutuhkan masing-masing metode; metode *bisection* memerlukan 19 kali iterasi, sedangkan metode *regula falsi* hanya memerlukan 8 dan 12 kali iterasi untuk menemukan akar persamaan positif dan negative secara berturut-turut.
3. Pemrograman ini dapat digunakan untuk menyelesaikan persamaan non-linier lain, baik aljabar maupun transedental, dengan mengganti definisi f(x) di awal program.

Saran

Penulisan kode program sebaiknya dilakukan dalam bahasa pemrograman yang lebih bagus dalam hal grafik dan akurasi. Selain itu, sebaiknya dibuat program interaktif dengan menyediakan slot untuk memasukkan persamaan yang diinginkan, *error*, dan keluaran grafik yang indah.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih pada Kepala Sekolah, rekan sejawat dan segenap staf Tata Usaha SMA Negeri 1 Sumberpucung yang membantu terselesaikanya jurnal tersebut.

REFERENSI

Amang. 2006. *BAB III: Penyelesaian Persamaan Non Linier.* Diakses dari http://lecturer.eepis-its.edu/~amang/pdf/, tanggal 29 Oktober 2017.

Higashikawa, K. 2017. *Applied IT-II*. ISEE Department of Engineering, Kyushu University.

Mulyono. 2007: *Penyelesaian Persamaan Non-linier*. Fakultas Keguruan dan Ilmu Pendidikan, Universitas Muhammadiyah Puwokerto.

Sasongko. S.B. 2010. *Metode Numerik dengan Scilab, Dasar Metode Numerik Persamaan Linear SImultan, Persamaan Non Linear, Persamaan Diferensial, dan Pengolahan Data.* Yogyakarta: Penerbit Andi, hal. 121.