Sekilas Tentang Scilab

Saifuddin Arief

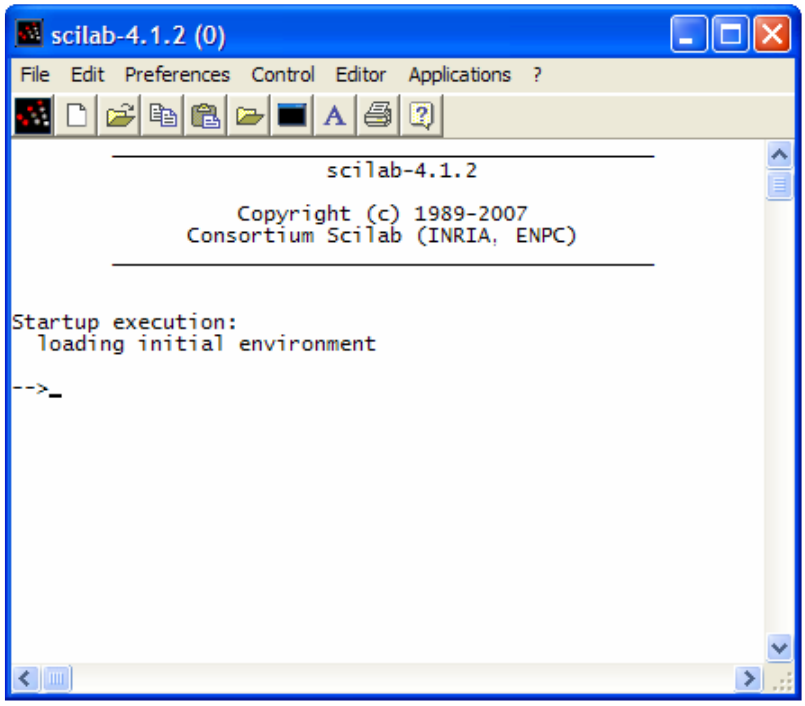
1. Sekilas Mengenai Scilab

Scilab adalah suatu perangkat lunak yang dikembangkan untuk komputasi numerik dan visualisasi data. Pada awalnya Scilab dikembangkan oleh INRIA dan ENPC, Perancis, dan sekarang pengembangan dan pemeliharaan Scilab dilakukan oleh konsorsium Scilab. Alamat website Scilab adalah http://www.scilab.org.

Kelebihan utama dari Scilab yaitu gratis (*freeware*) dan tersedia untuk berbagai sistem operasi seperti Windows, Mac OS/X, Unix dan Linux.

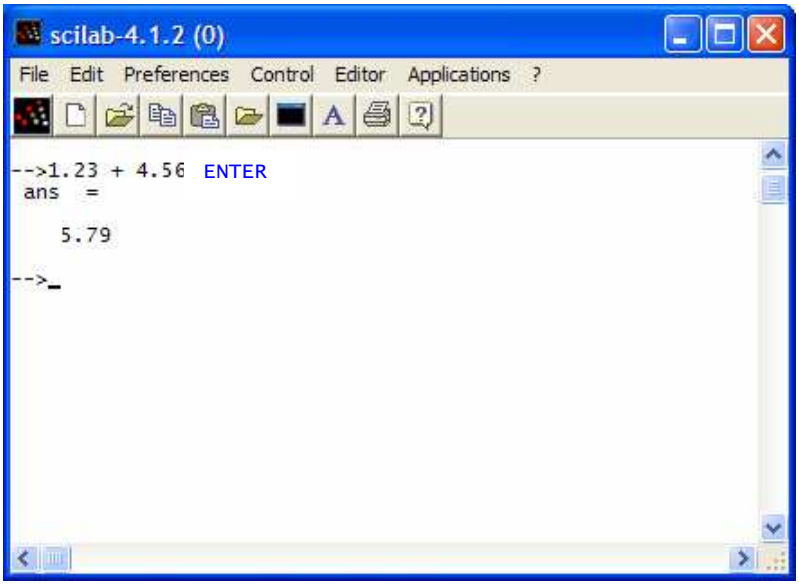
2. Dasar-Dasar Penggunaan Scilab

Scilab dapat dijalankan dari menu `Start` → `Scilab-X.Y.Z` → `Scilab-X.Y.Z`, dimana `X.Y.Z` adalah versi dari Scilab. Cara lain untuk menjalankan Scilab adalah dengan melakukan klik ganda terhadap ikon Scilab yang terdapat pada jendela Dekstop. Setelah kita jalankan perintah tersebut maka akan muncul suatu jendela Scilab, seperti yang ditunjukkan pada Gambar 1.



Gambar 1. Jendela Scilab

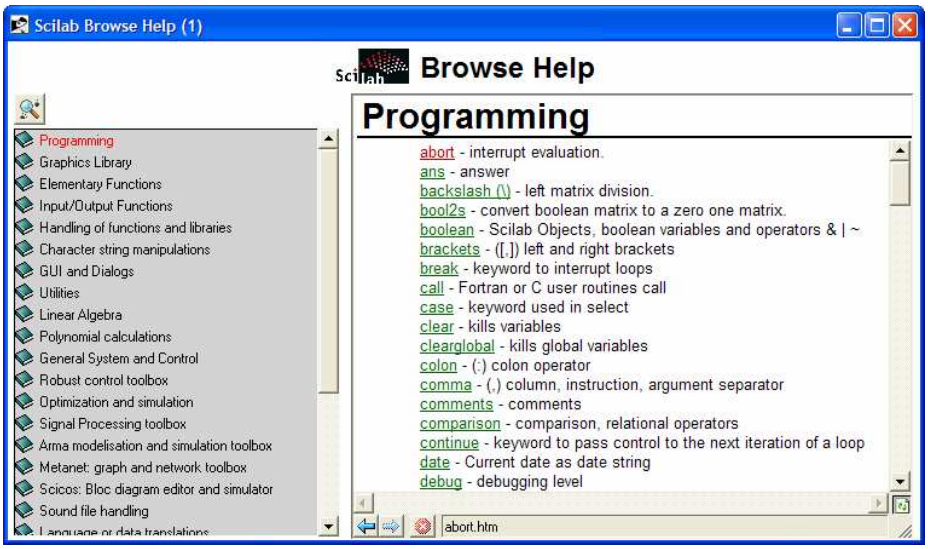
Simbol --> yang terdapat pada jendela Scilab merupakan tanda bahwa Scilab siap untuk untuk menerima suatu perintah yang akan kita berikan. Misalkan kita akan melakukan suatu perhitungan yaitu `1.23 + 4.56` maka kita harus menuliskan ekspresi matematika tersebut setelah simbol --> kemudian tekan tombol ENTER untuk melakukan eksekusi terhadap ekpresi matematika yang telah kita ketikkan. Scilab akan menampilkan hasil dari perintah yang telah kita berikan pada baris berikutnya. Gambar 2 merupakan ilustrasi mengenai contoh perhitungan yang telah kita lakukan.



Gambar 2. Contoh sebuah perintah sederhana

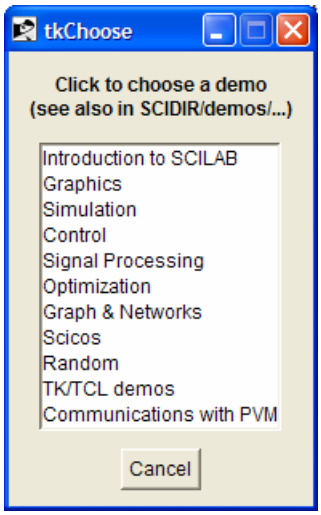
Setelah selesai melakukan suatu perhitungan atau mengerjakan suatu perintah yang kita berikan maka Scilab akan menampilkan kembali simbol `-->`. Hal ini sebagai tanda bahwa Scilab telah siap untuk mengerjakan perintah yang lain.

Scilab juga dilengkapi dengan sistem bantuan (*help*) yang cukup baik, untuk melihat sistem bantuan (help), gunakan menu `?` → `Scilab Help`. Pada jendela bantuan, kita dapat memperoleh penjelasan yang detail mengenai suatu fungsi atau operator tertentu.



Gambar 3. Jendela Bantuan

Scilab juga memberikan demonstrasi sebagai pengenalan mengenai Scilab serta beberapa contoh aplikasi penggunaannya. Untuk melihat demonstrasi Scilab, klik menu `?` → `Scilab demos`. Apabila perintah tersebut kita jalankan maka akan muncul sebuah form seperti yang terdapat pada Gambar 4.



Gambar 4

Untuk mengakhiri penggunaan Scilab, gunakan menu `File` → `Exit` atau dengan menekan tanda ☒ yang terdapat pada bagian kanan atas dari Jendela Scilab.

3. Variabel dan Ekspresi

Variabel adalah sebuah nama yang digunakan untuk menyimpan nilai suatu obyek. Notasi statemen penugasan adalah sebagai berikut:

```
x = ekspresi
```

dimana x adalah nama variabel dan ekspresi adalah suatu ekspresi matematika. Nama variabel di dalam Scilab adalah bersifat sensitif terhadap ukuran huruf, sehingga xawal dan Xawal adalah dua buah variabel yang berbeda.

Selain variabel-variabel yang dapat kita buat sendiri, di dalam Scilab telah terpasang beberapa variabel khusus yang menyatakan suatu konstanta matematika, seperti `%pi` untuk π = 3.1415927... , `%i` untuk i = $\sqrt{-1}$ serta `%e` untuk e = 2.7182818... .

Nilai dari suatu ekspresi yang kita masukkan akan ditampilkan pada baris berikutnya, kecuali jika pada akhir ekpresi tersebut kita tambahkan tanda titik koma (;).

Apabila kita mempunyai suatu ekspresi yang cukup panjang dan tidak cukup untuk ditulis pada satu baris, maka kita harus menggunakan tanda titik tiga (…) pada akhir ekspresi, sebagai tanda bahwa ekspresi bersambung pada baris berikutnya.

Untuk memperjelas perintah yang kita masukkan kita dapat menambahkan suatu baris komentar. Baris komentar dibuat dengan menggunakan tanda `//`. Segala sesuatu di belakang tanda tersebut akan diabaikan oleh Scilab. Baris komentar dapat ditulis pada suatu baris tersendiri atau di belakang suatu ekspresi.

```
-->lebar = 12.5
 lebar  =

    12.5

-->tinggi = 8;

-->luas = lebar * tinggi
 luas  =

    100.

-->r = 10;              // diameter

-->A = %pi*r^2          // Luas lingkaran
 A  =

    314.15927

-->// Berikut ini contoh penggunaan simbol tiga titik (...)

-->s = 1 + 1/2 + 1/3 + 1/4 + 1/5 + ...
-->    1/6 + 1/7 + 1/8 + 1/9 + 1/10
 s  =

    2.9289683
```

4. Ruang Kerja

Variabel-variabel yang telah kita buat akan disimpan oleh Scilab dalam ruang kerja. Untuk melihat nama-nama variabel yang telah dibuat, gunakan perintah `who`.

```
-->who
your variables are...

 s          A          r          luas       tinggi     lebar
 scicos_pal            %scicos_menu          %scicos_short
 %scicos_help          %scicos_display_mode            modelica_libs
 scicos_pal_libs       %scicos_lhb_list      %CmenuTypeOneVector
 %helps     WSCI       home       SCIHOME    CreateScilabHomeDir
 PWD        TMPDIR     MSDOS      SCI        guilib     sparselib
 xdesslib   percentlib             polylib    intlib     elemlib    utillib
 statslib   alglib     siglib     optlib     autolib    roblib     soundlib
 metalib    armalib    tkscilib   tdcslib    s2flib     mtlblib    %F
 %T         %z         %s         %nan       %inf       COMPILER   %gtk
 %gui       %pvm       %tk        $          %t         %f         %eps
 %io        %i         %e
 using       31751 elements  out of    5000000.
           and          64 variables out of       9231

your global variables are...

 LANGUAGE   %helps     demolist   %browsehelp          LCC
 %toolboxes            %toolboxes_dir
 using        1029 elements  out of      11000.
           and           7 variables out of        767
```

Terlihat bahwa perintah who, selain menampilkan variabel-variabel yang telah kita buat, juga menampilkan variabel-variabel yang telah terpasang pada Scilab.

Untuk menghapus suatu variabel, gunakan perintah `clear`.

```
-->clear luas tinggi       // menghapus variabel luas dan tinggi

-->clear                   // menghapus semua variabel yang telah kita buat
```

5. Operator-operator dan Fungsi-fungsi Matematika

Operator-operator untuk perhitungan aritmatika adalah sama dengan operator-operator yang terdapat pada kalkulator atau perangkat lunak lainnya yaitu `+`, `-`, `*`, `/` dan `^`. Dimana tanda-tanda tersebut masing-masing adalah simbol untuk operasi penjumlahan, pengurangan, perkalian, pembagian serta pemangkatan. Di dalam Scilab juga telah terpasang sejumlah fungsi-fungsi yang diperlukan dalam perhitungan matematika, seperti sqrt, abs, exp, sin, cos, tan dan lain sebagainya.

Berikut ini adalah contoh-contoh perhitungan matematika.

```
-->(1 + sqrt(5))/2
 ans  =

    1.618034

-->Tc = 372.7*(1 + 1/(1.242 + 1.067))
 Tc  =

    534.11187
```

```
-->dHv = (7.08*(1 - 0.6939)^0.354 + 10.95*0.2559*(1 - 0.6939)^0.456)* ...
-->      0.008314*504.4
 dHv  =

    26.374968

-->V = 0.773*sqrt(1.4*8314*261.6/29)
 V  =

    250.47731

-->dx = 109*cos(35/180*%pi)
 dx  =

    89.287573

-->TB = 310.9*(log(4.506e6) + (1 - 1.434)*log(1/5528))/log(4.506e6)
 TB  =

    386.79462
```

6. Bilangan Kompleks

Scilab juga dapat menangani bilangan kompleks dan operasi-operasi matematikanya. Bilangan kompleks dinyatakan dengan notasi `a + b*%i` , dimana a adalah komponen real dan b adalah komponen imajinernya.

```
-->z1 = 6 - 8*%i
 z1  =

    6. - 8.i

-->z2 = 3 + %i
 z2  =

    3. + i

-->p = z1 + z2
 p  =

    9. - 7.i

-->q = z1 - z2
 q  =

    3. - 9.i

-->abs(z1)
 ans  =

    10.

-->z1 * z2
 ans  =

    26. - 18.i

-->z1/z2
 ans  =

    1. - 3.i
```

7. Matrik dan Vektor

Salah satu kelebihan Scilab yaitu kemampuannya dalam menangani berbagai macam operasi manipulasi terhadap data yang berupa suatu matrik. Pada dasarnya semua data numerik di dalam Scilab dianggap sebagai suatu matrik. Vektor dan skalar merupakan bentuk khusus dari suatu matrik. Vektor adalah suatu matrik yang hanya mempunyai satu baris atau satu kolom saja, sementara itu skalar adalah suatu matrik yang hanya terdiri dari satu elemen saja.

Pembuatan data matrik dan vektor secara manual dilakukan dengan menggunakan operator kurung siku (`[ ... ]`). Dimana elemen-elemen matrik atau vektor dimasukkan diantara kedua kurung siku tersebut. Untuk memisahkan elemen yang satu dengan elemen yang lainnya yang terletak pada satu baris dapat digunakan tanda koma (`,`) atau tanda spasi. Kemudian untuk memisahkan antara baris yang satu dengan yang lainnya, gunakan tanda titik koma (`;`) atau tanda ENTER.

Contoh-contoh pembuatan matrik dan vektor adalah sebagai berikut:

```
-->X = [1 2 3; 4 5 6; 7 8 9]
 X  =

    1.    2.    3.
    4.    5.    6.
    7.    8.    9.

-->X = [1,2,3                    // Cara yang lain ...
--> 4,5,6
--> 7,8,9
-->]
 X  =

    1.    2.    3.
    4.    5.    6.
    7.    8.    9.

-->v = [1 2 3 4 5]               // vektor baris
v  =

    1.    2.    3.    4.    5.

-->w = [3;4;1;2]                 // vektor kolom
 w  =

    3.
    4.
    1.
    2.
```

Untuk membuat suatu vektor baris dimana nilai dari elemen-elemennya berubah secara konstan dari suatu awal nilai tertentu sampai nilai akhir tertentu kita dapat menggunakan operator tanda titik dua (`:`). Notasi pembuatan vektor indek adalah `i:j:k`. Apabila nilai j sama maka notasi tersebut dapat ditulis dengan notasi yang lebih singkat yaitu `i:k`.

```
-->i = 1:10
 i  =

    1.    2.    3.    4.    5.    6.    7.    8.    9.    10.
```

```
-->n = 0:0.25:1
 n  =

    0.    0.25    0.5    0.75    1.

-->m = 1:2:10
 m  =

    1.    3.    5.    7.    9.
```

Scilab juga menyediakan sejumlah fungsi yang dapat digunakan untuk membuat matrik-matrik khusus. Ilustrasi dari fungsi-fungsi tersebut diberikan pada contoh-contoh di bawah ini.

```
-->A = zeros(3,4)           // matrik nol
 A  =

    0.    0.    0.    0.
    0.    0.    0.    0.
    0.    0.    0.    0.

-->B = ones(2,5)                // matrik satuan
 B  =

    1.    1.    1.    1.    1.
    1.    1.    1.    1.    1.

-->Y = eye(3,3)                 // Matrik satuan
 Y  =

    1.    0.    0.
    0.    1.    0.
    0.    0.    1.

-->D = diag(1:4)                // Matrik diagonal
 D  =

    1.    0.    0.    0.
    0.    2.    0.    0.
    0.    0.    3.    0.
    0.    0.    0.    4.

-->Z = rand(4,5)                // Bilangan random
 Z  =

    0.2113249    0.6653811    0.8782165    0.7263507    0.2312237
    0.7560439    0.6283918    0.0683740    0.1985144    0.2164633
    0.0002211    0.8497452    0.5608486    0.5442573    0.8833888
    0.3303271    0.6857310    0.6623569    0.2320748    0.6525135
```

8. Operasi Berbasis Vektor

Di dalam Scilab, secara umum operasi-operasi matematika terhadap obyek matrik dan vektor dapat dilakukan dengan sangat mudah tanpa harus menggunakan suatu perulangan, sebagaimana yang diilustrasikan pada contoh-contoh di bawah ini.

```
-->x = 0:%pi/4:%pi
 x  =

    0.    0.7853982    1.5707963    2.3561945    3.1415927

-->cos(x)
 ans  =

    1.    0.7071068    6.123D-17  - 0.7071068  - 1.
```

```
-->u = [1 2 3];

-->y = exp(u)
 y  =

    2.7182818    7.3890561    20.085537

-->z = log(y)
 z  =

    1.    2.    3.

-->w = round(y)
 w  =

    3.    7.    20.
```

Operasi Aljabar Linier, seperti penjumlahan, pengurangan dan perkalian, juga dapat dilakukan dengan sangat mudah tanpa harus menggunakan suatu ekpresi perulangan secara eksplisit. Pada operasi aljabar argumen-argumennya harus mempunyai dimensi yang kompatibel, jika dimensi tidak kompatibel maka operasinya tidak dapat dieksekusi dan Scilab akan menampilkan suatu pesan kesalahan.

```
-->X = [9 8 5; 1 3 0; 0 4 6]
 X  =

    9.    8.    5.
    1.    3.    0.
    0.    4.    6.

-->Y = [3 2 1; 4 5 6; 9 8 7]
 Y  =

    3.    2.    1.
    4.    5.    6.
    9.    8.    7.

-->A = X + Y
 A  =

    12.    10.    6.
    5.     8.     6.
    9.     12.    13.

-->B = X - Y
 B  =

    6.    6.    4.
  - 3.  - 2.  - 6.
  - 9.  - 4.  - 1.

-->Z = A + U
          !--error 8
inconsistent addition

-->p = [1 2; 3 8]
 p  =

    1.    2.
    3.    8.

-->i = eye(2,2)
 i  =

    1.    0.
    0.    1.
```

```
-->j = [6; 4]
 j  =

    6.
    4.

-->k = ones(3,3)
 k  =

    1.    1.    1.
    1.    1.    1.
    1.    1.    1.

-->f = p*i
 f  =

    1.    2.
    3.    8.

-->g = p*j
 g  =

    14.
    50.

-->h = p*k
         !--error 10
inconsistent multiplication
```

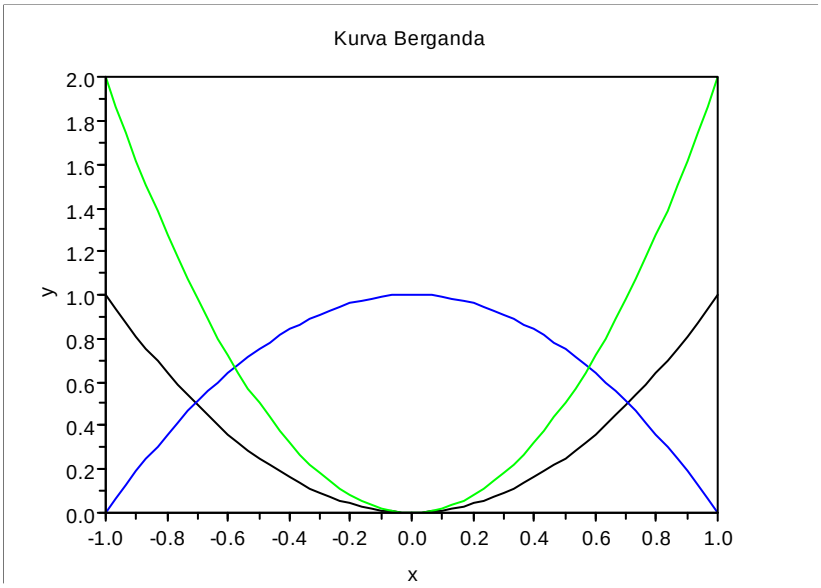
Selain operasi perkalian yang mengikuti aturan dalam Aljabar Linier, kita juga dapat melakukan operasi terhadap obyek matrik dan vektor dengan melakukan operasi secara elemen dengan elemen. Operasi elemen dengan elemen juga dapat diterapkan pada operasi pembagian. Notasi untuk perkalian dan pembagian secara elemen dengan eleman adalah `.*` dan `./`.

```
-->x = [1 2; 3 4]
 x  =

    1.    2.
    3.    4.

-->y = [5 6; 7 8]
 y  =

    5.    6.
    7.    8.

-->u = x.*y
 u  =

    5.     12.
    21.    32.

-->z = y./x
 z  =

    5.           3.
    2.3333333    2.
```

9. Visualisasi Data

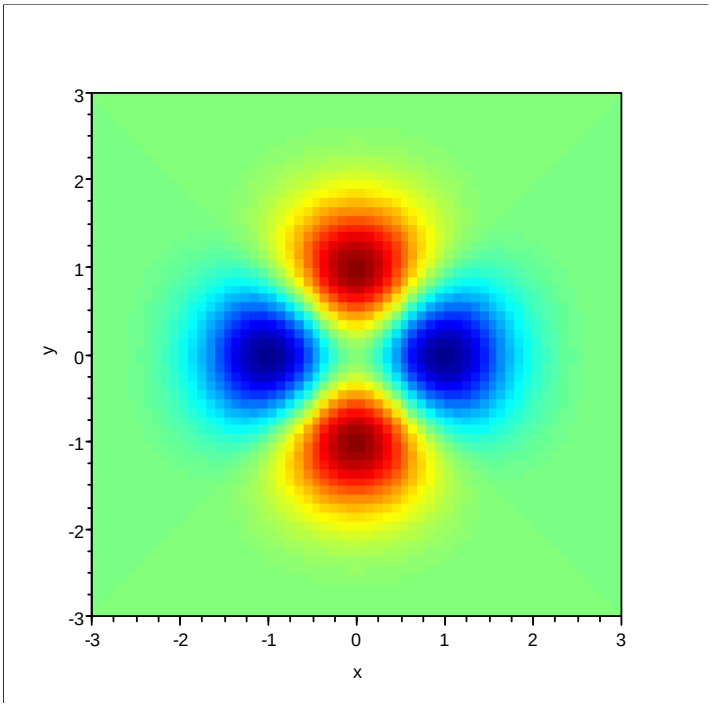
Scilab dapat digunakan untuk melakukan visualisasi data, baik secara dua dimensi maupun tiga dimensi. Untuk membuat grafik dua dimensi kita dapat menggunakan perintah plot2d, kemudian untuk grafik tiga dimensi kita dapat menggunakan perintah plot3d, seperti yang diperlihatkan pada contoh-contoh di bawah ini. Penjelasan detail mengenai pembuatan grafik dapat dilihat pada sistem bantuan yang terdapat pada program Scilab.

```
-->// Lihat Gambar 5
-->x = linspace(-1,1,61)';
-->y1 = x.^2; y2 = 1 - y1; y3 = 2*y1;
-->plot2d(x,[y1 y2 y3])
-->xtitle('Kurva Berganda','x','y')
```



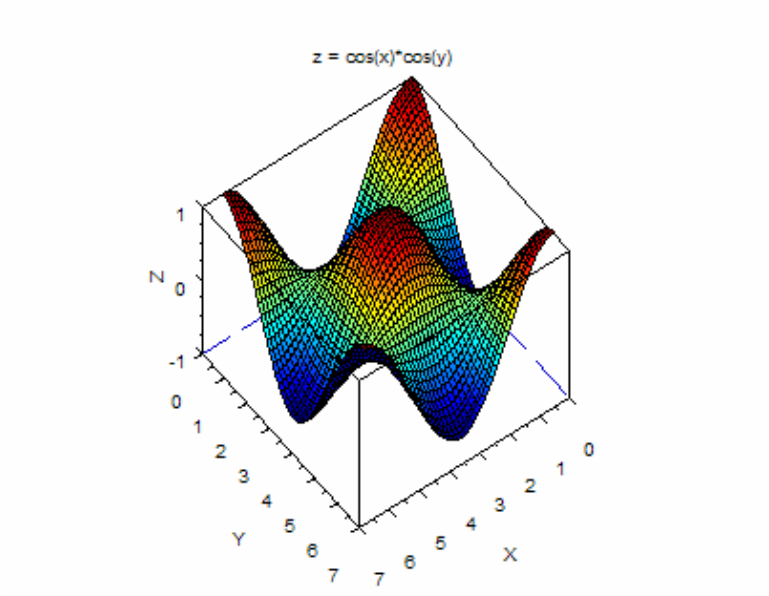
Gambar 5

```
-->// Lihat Gambar 6
-->x = -3:0.1:3; y = -3:0.1:3;
-->[X,Y] = meshgrid(x,y);
-->Z = (X.^2 - Y.^2).*exp(-X.^2 - Y.^2);
-->xset('colormap',jetcolormap(100))
-->clf, grayplot(x,y,Z), xtitle('','x','y')
```



Gambar 6

```
-->// Lihat Gambar 7
-->x = linspace(0,2*%pi,50); y = x;
-->z = cos(x')*cos(y);
-->clf, plot3d1(x,y,z)
-->xtitle('z = cos(x)*cos(y)'), xset('colormap',jetcolormap(50))
```



Gambar 7

10. Komputasi Numerik

Penyelesaian berbagai macam persoalan dalam komputasi numerik dapat dilakukan dengan mudah, seperti yang ditunjukkan pada contoh-contoh di bawah ini.

Sistem persamaan linear Ax = b dapat diselesaikan dengan mudah menggunakan operator pembagian kiri (\). Notasi `A\b` adalah ekuivalen dengan `inv(A)*b`.

```
-->A = [1 2 1 4; 2 0 4 3; 4 2 2 1; -3 1 3 2]
 A  =

    1.    2.    1.    4.
    2.    0.    4.    3.
    4.    2.    2.    1.
  - 3.    1.    3.    2.

-->b = [13; 28; 20; 6]
 b  =

    13.
    28.
    20.
    6.

-->x = A\b
 x  =

    3.
  - 1.
    4.
    2.
```

Nilai determinan dari matrik A dapat dihitung dengan menggunakan fungsi `det(A)`.

```
-->det(A)
 ans  =

  - 180.

-->rank(A)
 ans  =

    4.
```

Inversi dari suatu matrik A dapat kita hitung dengan menggunakan fungsi `inv(A)`.

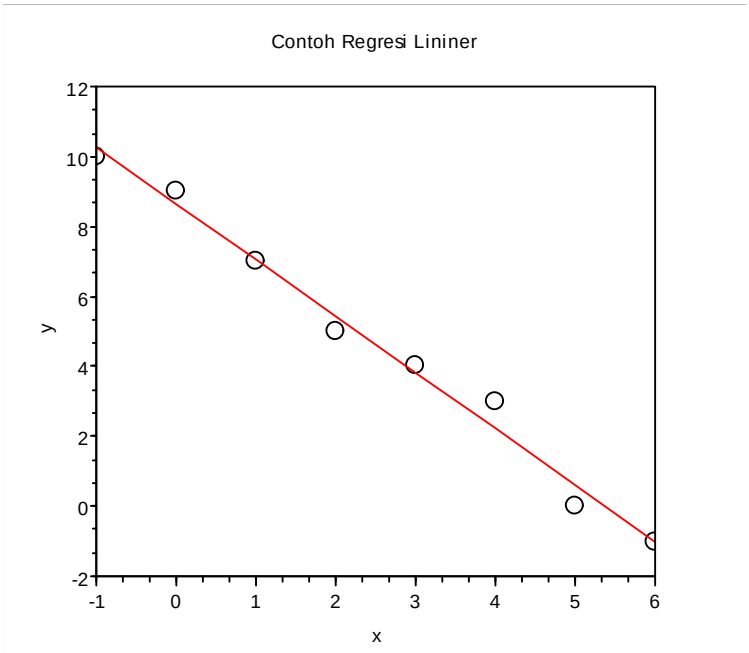
```
-->Z = inv(A)
 Z  =

    0.           0.0833333    0.0833333  - 0.1666667
    0.0666667  - 0.3444444    0.3222222    0.2222222
  - 0.2          0.1166667    0.1166667    0.1666667
    0.2666667    0.1222222  - 0.2111111  - 0.1111111

-->A*Z
 ans  =

    1.           1.110D-16    0.         - 5.551D-17
    0.           1.         - 1.110D-16    0.
  - 5.551D-17    5.551D-17    1.         - 2.776D-17
  - 1.110D-16    0.         - 5.551D-17    1.
```

Untuk mencari penyelesaian persamaan nonlinier f(x) = 0, kita dapat menggunakan fungsi `fsolve`, seperti yang digambarkan pada contoh di bawah ini.

```
-->function y = h(x)
-->  y = x^3 - x - 3
-->endfunction

-->[x0, h0, info] = fsolve(1,h)
 info  =

    1.
 h0  =

  - 8.882D-16
 x0  =

    1.6716999
```

Regresi Linier dari suatu pasangan data dapat kita lakukan dengan menggunakan fungsi `regress`.

```
-->x = [-1 0 1 2 3 4 5 6];

-->y = [10 9 7 5 4 3 0 -1];

-->koef = regress(x,y)
 koef  =

    8.6428571
  - 1.6071429

-->yr = koef(1) + koef(2)*x;

-->plot2d(x',[y' yr'],style=[-9 5])            // Lihat gambar 8

-->xtitle('Contoh Regresi Lininer','x','y')
```



Gambar 8. Contoh Regresi Linier

11. Pemrograman

Scilab menyediakan sejumlah kontrol pemrograman yang dapat kita gunakan untuk mengatur jalannya eksekusi suatu program dengan menggunakan statemen perulangan dan kondisional. Pada umumnya statemen perulangan dan kondisional digunakan dalam sebuah skrip atau fungsi.

Ilustrasi penggunaan statemen perulangan for dan kondisional if diberikan pada contoh-contoh-contoh di bawah ini.

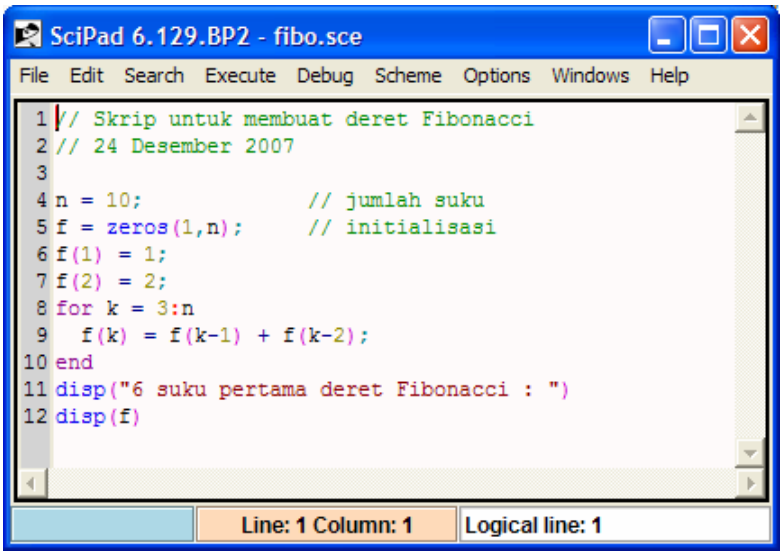
```
-->// Contoh statemen perulangan

-->H = zeros(4,4);

-->for i=1:4
-->   for j=1:4
-->      H(i,j) = 1/(i+j-1);
-->   end
-->end

-->H
 H  =

    1.          0.5         0.3333333   0.25
    0.5         0.3333333   0.25        0.2
    0.3333333   0.25        0.2         0.1666667
    0.25        0.2         0.1666667   0.1428571

-->// Contoh statemen kondisional

-->function n=nilai(kode)
-->// Fungsi untuk mengkoversi nilai dari abjad menjadi angka
-->
-->  if kode=='A'
-->    n=4;
-->  elseif kode=='B'
-->    n=3;
-->  elseif kode=='C'
-->    n=2;
-->  elseif kode=='D'
-->    n=1;
-->  else
-->    n=0;
-->  end
-->
-->endfunction

-->n1 = nilai('A')
 n1  =

    4.

-->n = nilai('E')
 n  =

    0.

-->// Contoh kombinasi dari statemen perulangan dan kondisional

-->I = zeros(3,3);

-->for i = 1:3
-->  for j = 1:3
-->    if (i==j)
-->      I(i,i) = i;
-->    end
-->  end
-->end
```

```
-->I
 I  =

    1.    0.    0.
    0.    2.    0.
    0.    0.    3.
```

12. Skrip

Skrip adalah sebuah file teks yang di dalamnya terdapat perintah-perintah Scilab. Apabila suatu skrip dijalankan maka perintah-perintah yang terdapat di dalamnya akan dieksekusi seperti seolah-olah kita mengetikkannya pada jendela perintah. Skrip dapat dibuat dengan menggunakan teks editor yang telah terpasang pada Scilab, yaitu SciPad, atau dengan menggunakan program teks editor yang lain seperti Notepad, Notepad2 dan ConText.

Berikut ini adalah contoh sebuah skrip (`fibo.sce`) yang menggambarkan perhitungan 6 suku pertama dari deret Fibonacci.



Gambar 9

Untuk menjalankan suatu skrip kita harus menggunakan perintah `exec(nama_file)`, dimana nama_file adalah nama dari file skrip. Apabila file skrip berada pada direktory kerja maka nama_file dapat dituliskan nama filenya saja, akan tetapi jika file skrip tidak berada pada direktory kerja maka nama filenya harus ditulis secara lengkap. Cara lain untuk menjalankan suatu skrip adalah dengan menggunakan menu `File → Exec`.

Berikut ini adalah output yang muncul pada jendela perintah jika skrip fibo.sce dijalankan.

```
-->exec('C:\Skrip dan Fungsi\fibo.sce');

 6 suku pertama deret Fibonacci :

    1.    2.    3.    5.    8.    13.    21.    34.    55.    89.
```

13. Fungsi

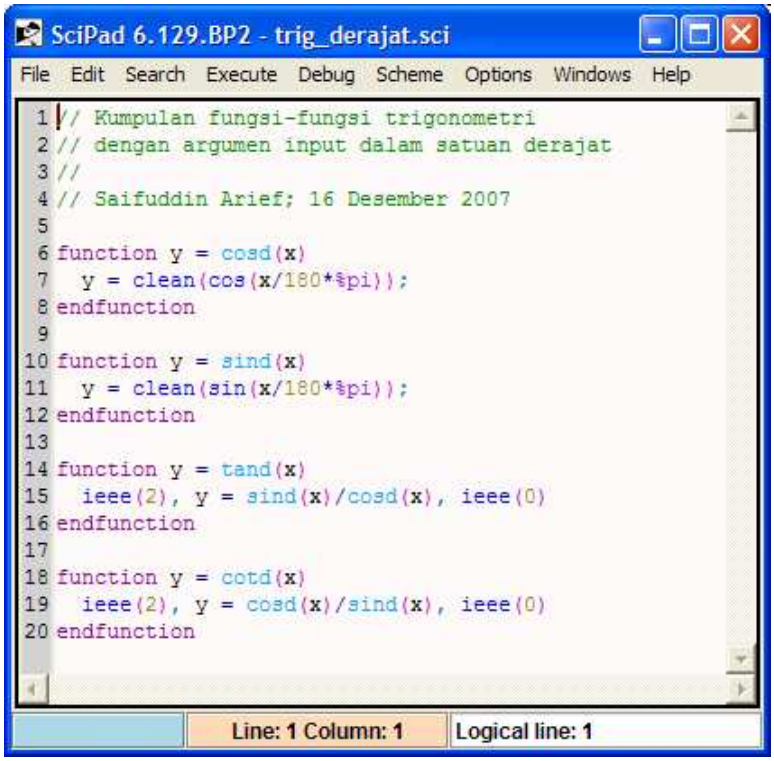
Fungsi merupakan kumpulan dari statemen-statemen Scilab yang dapat melakukan suatu komputasi atau perhitungan tertentu. Fungsi bersifat lebih fleksibel dibanding dengan skrip, karena di dalam fungsi terdapat argumen-argumen input dan output. Fungsi dapat dibuat dalam bentuk suatu file atau secara inline pada jendela perintah. Untuk suatu fungsi yang akan kita gunakan berulang kali maka biasanya fungsi tersebut dibuat dalam bentuk file fungsi, namun untuk suatu fungsi yang hanya digunakan untuk sementara maka kita dapat membuatnya secara inline pada jendela perintah.

Bentuk umum dari suatu fungsi adalah sebagai berikut:

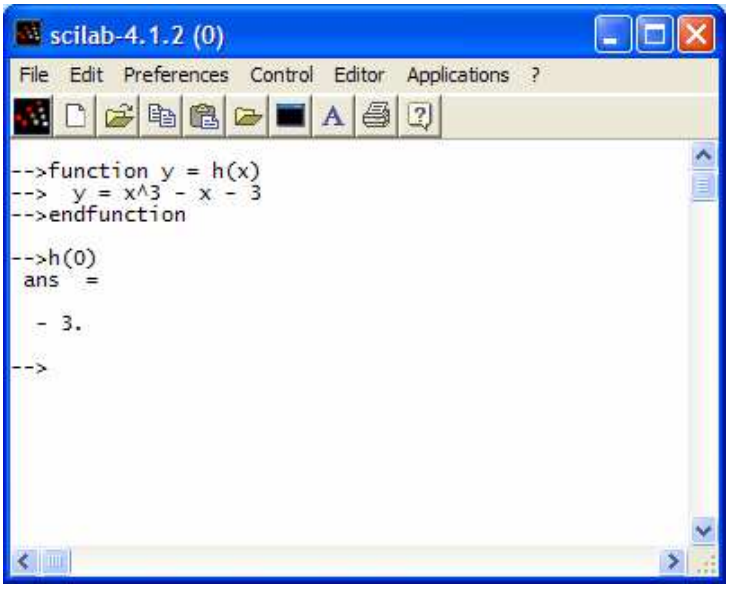
```
function [out1,out2,...] = fname(in1,in2,...)
   statemen-statemen
endfunction
```

dimana `fname` adalah nama fungsi yang kita buat, dan `in1,in2,...` adalah argumen-argumen input serta `out1,out2,...` adalah argumen-argumen output.

Gambar 10 adalah contoh suatu file fungsi, kemudian Gambar 11 adalah contoh pembuatan suatu fungsi secara inline.



Gambar 10



Gambar 11. Contoh Pembuatan Fungsi Secara Inline dan Penggunaannya.

Fungsi-fungsi yang telah kita buat dapat kita gunakan sebagaimana fungsi-fungsi yang telah terpasang pada Scilab. Untuk fungsi-fungsi yang kita buat dalam suatu file fungsi maka kita harus memanggilnya terlebih dahulu ke dalam ruang kerja dengan menggunakan perintah `exec`, seperti yang ditunjukkan pada contoh di bawah ini.

```
-->exec('C:\Skrip dan Fungsi\trig_derajat.sci');disp('exec done');

 exec done

-->cosd(180)
 ans  =

  - 1.

-->sind(90)
 ans  =

    1.

-->tand(90)
 ans  =

   Inf
```

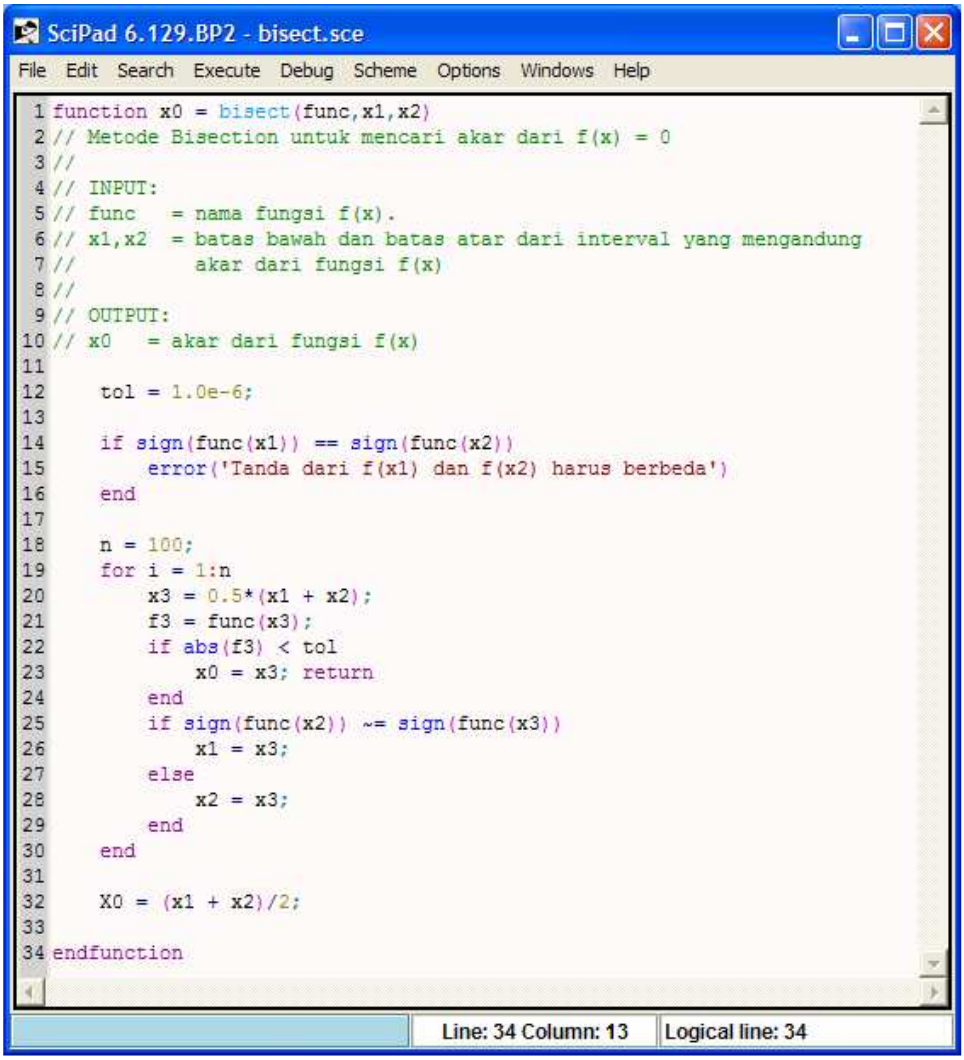
Kemudian untuk fungsi yang dibuat secara inline kita dapat langsung menggunakannya tanpa harus menggunakan perintah `exec`, seperti yang telah diperlihatkan pada Gambar 11.

Variabel-variabel yang terdapat di dalam suatu fungsi yang kita buat bersifat sebagai variabel lokal, dimana variabel-variabel hanya akan ada selama fungsi dijalankan dan tersimpan secara terpisah dari ruang kerja Scilab, kecuali jika kita menyatakannya sebagai variabel global. Penjelasan mengenai variabel lokal dan global dapat dilihat pada sistem bantuan Scilab.

Scilab juga mendukung fungsi yang bersifat rekursif, yaitu fungsi yang dapat memanggil dirinya sendiri. Berikut ini contoh sebuah fungsi rekursi.

```
-->function f = fibonnaci(n)
-->  if n <= 1 then
-->    f = 1
-->    return
-->  end
-->  f = fibonnaci(n-1) + fibonnaci(n-2)
-->endfunction

-->fibonnaci(6)
 ans  =

    13.
```

Salah satu kelebihan Scilab lainnya adalah evaluasi suatu fungsi di dalam suatu fungsi yang kita buat dapat dievaluasi secara langsung, tanpa harus menggunakan perintah eval, seperti yang terdapat pada program Matlab. Ilustrasi mengenai hal ini diperlihatkan pada Gambar 12.

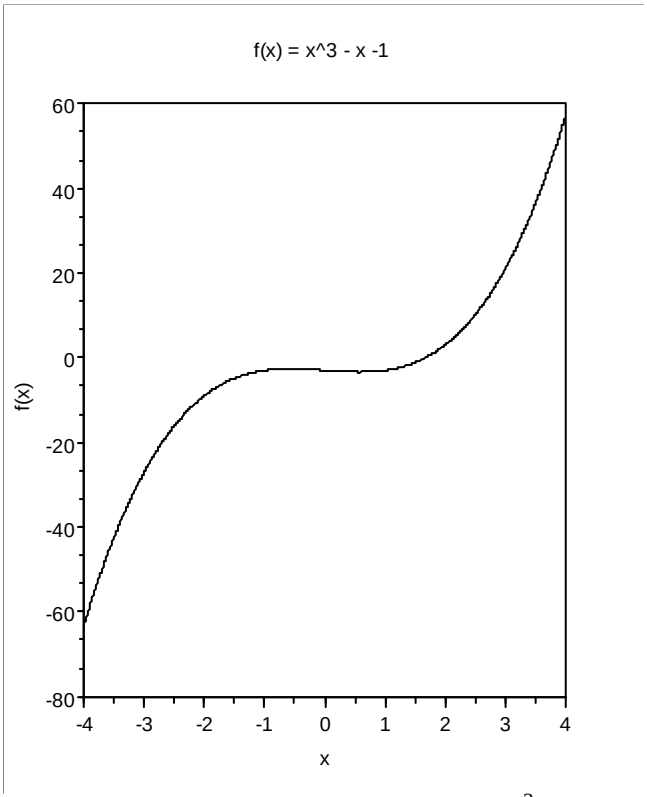



```
function x0 = bisect(func,x1,x2)
// Metode Bisection untuk mencari akar dari f(x) = 0
//
// INPUT:
// func   = nama fungsi f(x).
// x1,x2  = batas bawah dan batas atar dari interval yang mengandung
//          akar dari fungsi f(x)
//
// OUTPUT:
// x0   = akar dari fungsi f(x)

    tol = 1.0e-6;

    if sign(func(x1)) == sign(func(x2))
        error('Tanda dari f(x1) dan f(x2) harus berbeda')
    end

    n = 100;
    for i = 1:n
        x3 = 0.5*(x1 + x2);
        f3 = func(x3);
        if abs(f3) < tol
            x0 = x3; return
        end
        if sign(func(x2)) ~= sign(func(x3))
            x1 = x3;
        else
            x2 = x3;
        end
    end

    X0 = (x1 + x2)/2;

endfunction
```

Gambar 12. Metode Bisection

Berikut ini adalah contoh penggunaan fungsi bisect.

```
-->exec('C:\Skrip dan Fungsi\bisect.sci');

-->function y=f(x)
-->  y = x^3 - x - 3
-->endfunction

-->x0 = bisect(f,1,2)
 x0  =

    1.6717

-->f0 = f(x0)
 f0  =

    0.0000009
```



Gambar 13. Grafik dari fungsi $f(x) = x^3 - x - 1$

14. Direktori Kerja

Untuk melihat direktori dimana kita kerja (direktori kerja) gunakan perintah `pwd`.

```
-->pwd
 ans  =

 C:\Metode Numerik\Scilab
```

Apabila anda menjalankan perintah pwd maka tampilan yang akan muncul adalah sesuai dengan direktori dimana anda bekerja. Pada contoh ini direktori kerjanya adalah `C:\Metode Numerik\Scilab`.

Kemudian untuk pindah ke direktori yang lain gunakan perintah `chdir(nama-direktori)`.

```
-->chdir('C:\Temp')
 ans  =

    0.

-->pwd
 ans  =

 C:\Temp
```

Terlihat bahwa direktori kerja telah berubah menjadi `C:\Temp`.

15. Pengguna Scilab

Scilab telah digunakan secara meluas di seluruh dunia baik di kalangan pendidikan, penelitian maupun industri, diperkirakan Scilab telah didownload lebih dari 20 ribu dalam sebulan.

Daftar dari beberapa pengguna Scilab, baik dari kalangan universitas maupun dari industri adalah sebagai berikut: ENPC, INRIA, CEA, Ecole Polytechnique, Australian National University, University of New England, Indian Institute of Technology, Tsing Hua University, CNES, EADS, Dassault Aviation, Thales, Anagram Technologies, Klippel, Renault, Peugeot PSA.

18. Ucapan Terima kasih

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada INRIA dan ENPC (Perancis), serta konsorsium Scilab, yang telah mengembangkan dan memelihara program Scilab.

17. Daftar Pustaka

1. Arief, S., Scilab – Perangkat lunak gratis untuk komputasi numerik dan visualisasi data (draft buku).
2. Chapra, S.C., Canale, R.P., Numerical Methods for Engineers with Programming and Software Applications. WCB/McGraw-Hill, Singapore, 1998.
3. Rietsch, E., An Introdution to Scilab from a Matlab User's Point of View Version 2.6-1.0, 2002.
4. Scilab: http://www.scilab.org.

Biografi Penulis

Saifuddin Arief. Lahir di Turen, Malang, menyelesaikan S1 pada Jurusan Teknik Pertambangan, Institut Teknologi Bandung. Saat ini penulis bekerja pada sebuah perusahaan pertambangan di Sorowako, Sulawesi Selatan. Penulis dapat dihubungi dengan menggunakan alamat email: ariefs1@inco.com. Tulisan-tulisan lainnya dapat diperoleh pada alamat: http://www.scribd.com/people/view/155399-saifuddin-arief.