





Information technology – lecture 4 Elements of computer programming. Programming languages

Roman Putanowicz R.Putanowicz@L5.pk.edu.pl





Programming languages

There are multitude of programming languages and several ways of categorising them depending on their characteristics. Without going much into details we will distinguish the classifications:

- ► From the point of view of the complexity of instructions:
 - Low-level programming languages : assembly languages
 - ► High-level programming languages : C, C++, Java, etc.
- From the point of view of the usage patterns:
 - System programming languages: (C, C++, Fortran, Java, Ada) associated with the tags like: efficiency safety, static type control.
 - Scripting languages: (Python, Ruby, Tcl, Guile, Ch) associated with the tags like: rapid prototyping, flexibility, advanced introspection features







High-level programming languages paradigms

- Imperative programming language example Octave, C. Characterized by sequential execution of instructions, use of variables that represent memory locations, use of assignment statement to change the values of these variables.
- Functional programming language example Lisp, Haskel. Based on mathematical concept of function. Computation is expressed in term of the evaluation of functions. No variables and no assignment statements. Repetition expressed in terms of recursive function calls.
- Logic programming language example PROLOG.
 Based on principles of symbolic logic.





Programming languages for numerical simulations

Some languages are considered as better suited for writing numerical simulation codes. However picking the right language is a difficult thing often depending on non-technical issues (like available human resources in terms of programmers or local experts).

As most to the numerical algorithms utilize vector and matrix abstractions one important factor when evaluating a language is to what extend the language support direct use of these abstractions. Support for vector and matrices can be either built-in into a language (Matlab, Octave, Fortran) or can be provided by a set of libraries.

Some popular choices are: Ada, C, C++, Fortran (both 77 and 90 and above), Matlab, Octave, Python, Ch.





Python

What is Python

- Scripting, object-oriented, rapid prototyping, general purpose language
- Quite popular for writing scientific codes, especially when supported by C/C++/Fortran extension libraries
- Extensible and embeddable in applications





Selected Python features

- Portability UNIX, Windows, Mac, BeOS, VMS, Cray, …
- Compiles to interpreted byte code
- Automatic memory management through reference counting
- Many GUI libraries
- Several extension modules (NumPy for numerics)





Mathematical software

Mathematical software is software used to model, analyse, or calculate numeric, symbolic, or geometric data.(wikipedia)

Application areas:

- Symbolic mathematics computer algebra systems
- Statistics
- Geometry
- Numerical analysis

Categories of software:

- applications, e.g. GeoGebra
- interactive platforms, e.g. Scilab, Sage
- problems solving environments (PSE), e.g. Diffpack
- software libraries, e.g. GNU Scientific Library, Trilinos





Selected software packages

Alphabetical list:

- Diffpack
- Maple
- MathCad
- Mathematica
- Matlab
- Maxima http://maxima.sourceforge.net/
- Octave http://www.gnu.org/software/octave/
- R http://www.r-project.org/
- Sage http://www.r-project.org/
- Scilab http://www.scilab.org/





Software taxonomies

Licensing:

- Open Source
- Proprietary

Scope:

- Symbolic computations
- Numerical computations

Operating mode:

- WYSWIG, GUI
- traditional programming, CLI







Matlab









Maple









Maxima + wx = wxMaxima

۰	🥶 📸	○ Applications Places System (②) ④ □ 2	🔀 🌮 🌉 🔜 🥅 🔤 📊 👄 100% Wed Nov 10, 2:26 AM 📑 🕐			
	Bilo Edit	Coll Maxima Equations Algobra Calculus Simplify Plot Numeric Ho	xplot_lutorial.wxm* j			
	File Folt Cell Maxima Equations Algebra Calculus Simplify Piot Numeric Help					
	The pl the Pl into t	ot below is generated by selecting Plot/Plot 2d from the menu o lot2D button at the bottom of the screen. Alternatively, the com the cell.	r by clicking on mand can be entered directly			
	To exe	cute any command below, use "Ctrl-Enter" or "Shift-Enter".				
	F (%)2)	wmlot2d([vtsin(v)] [v .5 5])\$	Plot 2D X			
	(*****)		Expression(s): 2 Special			
			Variable: X From: -5 To: 5 🗆 logscale			
			Variable: y From: O To: O logscale			
1						
	(%t2)		Format: inline 🗸			
1Ô		4	Options: 🗸			
Е			File			
*		-4 -2 0 2 4				
+		x	<u>Cancel</u> <u>Q</u> K			
Ĩ	P Noto:	At any time, a graph may be sayed to a file. Click on the image	to be saved			
The Note: At any time, a graph may be saved to a file. Lick on the image to be saved. Then select the menu item Edit/Selection to image and save the graph. We now add labels to the horizontal and wortical area. Notice with sizes of						
	7 (Ni3)	lacement (m)"1)\$				
		2				
	Walsama		v			
0	meicome	со жамаалина				
Ù	5	8				







Scilab

,	atomsAutoload.sci - Scilab text editor		Scilab Console	x
	Eile Edit Search View Document Execute ?	Eile Edit Preferences Control Applications ?		
	🖸 🖬 📓 🖀 🥱 🏕 👗 🔃 📴 😵		2 6 X C D A A 2 9 9 0 0	
	atomsAutoload.sci - Scilab text editor	Scilab Console * 1	×	
	Untitled 1 atomsAutoload.sci	>// ulink previous function with same name	Ê	
	<pre>1 // sclue (http://www.sclue.org/,) - mis file is part of sclue</pre>	(@scilab.org> i which ms c	SEGS.ilbl = (link(rpg);f(b0) then ulink(ilb),end SEGS.ilbl = (link(rpg);f(b0) then ulink(ilb),end SEGS.ilbl = (link(rpg);f(b0) then ulink(ilb),end SEGS.ilbl = (link(rpg);f(b0)) then ulink(ilb),end SEGS.ilbl = (link(rpg);f(b0)) then ulink(ilb),end 	1
ili E	File Tools Edit 2 View Code	Eile 2	Help Browser	×
A	Noted - Second Annual Annua	A A I		î
	particular and the second seco	Scilab manual Scilab manual Differential Equ bvode dae daeoptions dasst dasst feval impl d	Name bode - boundary value problems for OCE using colocation method bodes - Simplified call to bode codes - Simplified call to bo	< 11 ×







Octave







Octave + Qt = QtOctave







Scalar product: C versus Octave program

```
#include <stdio.h>
1
                                                     1
2
                                                     2
    #define DTM 3
3
                                                     3
4
    int main() {
                                                     4
      char *fname = "vect.dat":
                                                     5
      FILE *fh = fopen(fname, "r");
                                                     6
6
7
                                                     7
      double u[DIM]:
8
                                                     8
      double v[DIM];
9
                                                     9
10
      int i;
                                                    10
11
12
      for (i=0; i<DIM; i++) {</pre>
        fscanf(fh, "%lf", u+i);
13
       3
14
      for (i=0; i<DIM; i++) {</pre>
15
        fscanf(fh, "%lf", v+i);
16
       3
      double s=0;
18
      for (i=0: i<DIM: i++) {</pre>
19
20
        s += u[i]*v[i]:
       }
21
22
      printf("Scalar product of u and v: %g\n", s);
      return 0;
23
24
```

```
1 fname = 'vect.dat';
2 fh = fopen(fname, 'r');
3 dim = 3;
4 u = fscanf(fh, '%lf', dim);
5 v = fscanf(fh, '%lf', dim);
6 s=0;
7 for i=1:dim
8 s+=u(i)*v(i);
9 end
10 printf('Scalar product of u and v: %g', s);
```







Scalar product: Octave versus Octave program

```
fname = 'vect.dat';
                                                     fname = 'vect.dat';
    fh = fopen(fname, 'r');
                                                     A = load('vect.dat');
2
                                                  2
    \dim = 3:
                                                     u = A(1:3);
3
                                                  3
    u = fscanf(fh, '%lf', dim);
                                                     v = A(4:6);
4
                                                  4
    v = fscanf(fh, '%lf', dim);
                                                     s = dot(u,v);
                                                  5
5
    s=0:
                                                     printf('Scalar product of u and v: %g\n', s
6
                                                  6
    for i=1:dim
7
      s+=u(i)*v(i):
8
9
    end
    printf('Scalar product of u and v: %g', s);
10
```





Problem: Write Octave program that illustrates affine transformation of a circle.





$$\hat{x} = T_{11}x + T_{12}y + T_{13}$$
$$\hat{y} = T_{21}x + T_{22}y + T_{23}$$







```
function [XY] = polygon(x, y, R, N)
1
      t = linspace(0, 2*pi, N+1);
2
      XY = zeros(N+1,2);
3
4
      XY(:,1) = x + R*cos(t);
5
      XY(:,2) = y + R*sin(t);
    end
6
    function plot polv(XY, color)
      h = line(XY(:,1), XY(:,2));
8
      set(h, 'color',color);
9
    end
10
    function [NXY] = transform_poly(XY, T)
      NXY=zeros(size(XY)):
      for i=1:rows(XY)
13
        x = XY(i,1);
14
        v = XY(i,2):
15
        xh = T(1,1)*x + T(1,2)*y + T(1,3);
16
        vh = T(2,1)*x + T(2,2)*v + T(2,3);
17
        NXY(i,:) = [xh,vh]:
18
      endfor
19
    endfunction
20
```

```
N=30
21
    xy = polygon(1,1,2,N);
22
    plot_poly(xy,"red");
23
24
   T = [2.0, 0.0, 1.0;
25
    0.5. 1.0. 0.0]:
    xy1 = transform_poly(xy, T);
26
    plot_poly(xy1,"blue");
    axis("equal")
28
    print("affine.fig")
29
```

```
30 pause()
```





References

- 1. Maxima http://maxima.sourceforge.net/
- 2. R http://www.r-project.org/
- 3. Sage http://www.r-project.org/
- 4. Scilab http://www.scilab.org/
- 5. GNU Octave http://www.gnu.org
- GNU Octave. A high-level interactive language for numerical computations, by John W. Eaton, David Bateman, Søren Hauberg, edition 3 for Octave version 3.0.2, Network Theory Ltd, 2008
- 7. Python http://www.python.org/
- 8. Own materials







Thank you for your attention

Project "The development of the didactic potential of Gracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education





Scalar product: Octave version 1





Scalar product: Octave version 2

1 fname = 'vect.dat';
2 A = load('vect.dat');
3 u = A(1:3);
4 v = A(4:6);
5 s = dot(u,v);
6 printf('Scalar product of u and v: %g\n', s);





back to the main code





```
7 function plot_poly(XY, color)
```

```
h = line(XY(:,1), XY(:,2));
```

```
set(h, 'color',color);
```

10 end

back to the main code





20 endfunction

back to the main code





back to the main code

;