Information technology – lecture 4

Elements of computer programming. Programming languages

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

Project "The development of the didactic potential of Cracow 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

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

Project "The development of the didactic potential of Cracow 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

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.

Project "The development of the didactic potential of Cracow 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

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.

Project "The development of the didactic potential of Cracow 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

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

Project "The development of the didactic potential of Cracow 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

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)

Project "The development of the didactic potential of Cracow 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

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

Project "The development of the didactic potential of Cracow 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

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/`

Project "The development of the didactic potential of Cracow 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

Software taxonomies

Licensing:

- Open Source
- Proprietary

Scope:

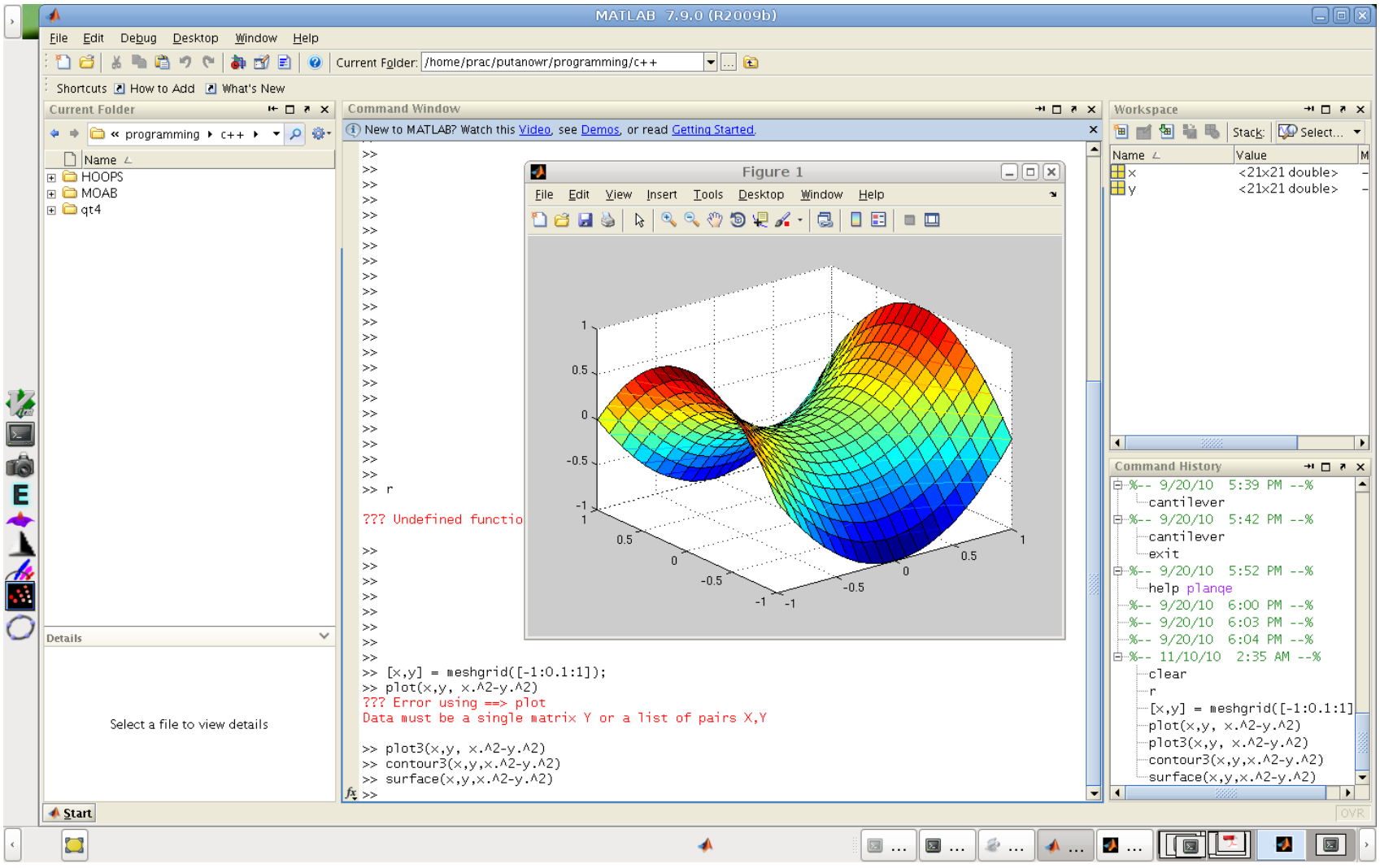
- Symbolic computations
- Numerical computations

Operating mode:

- WYSWIG, GUI
- traditional programming, CLI

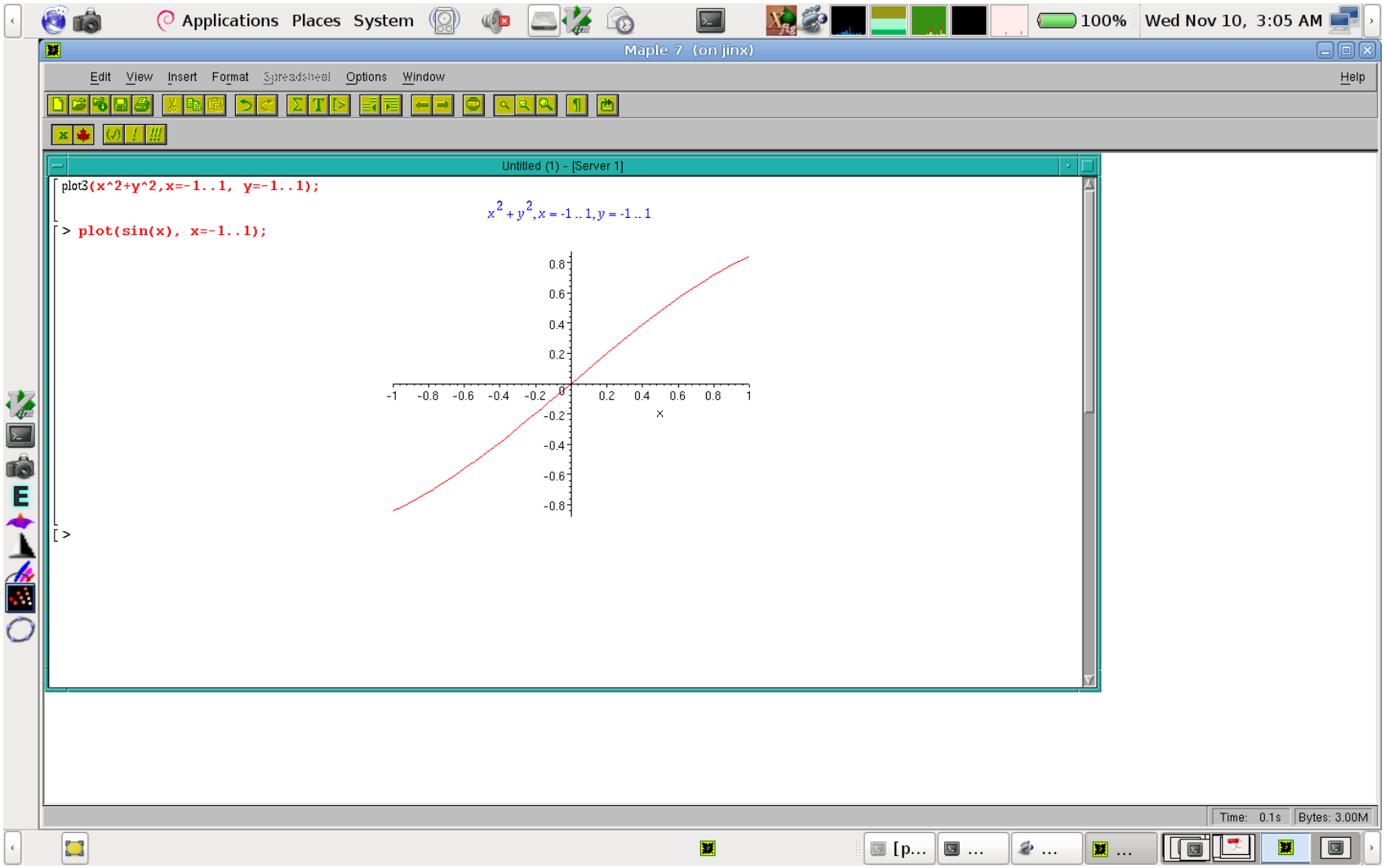
Project "The development of the didactic potential of Cracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education

Matlab



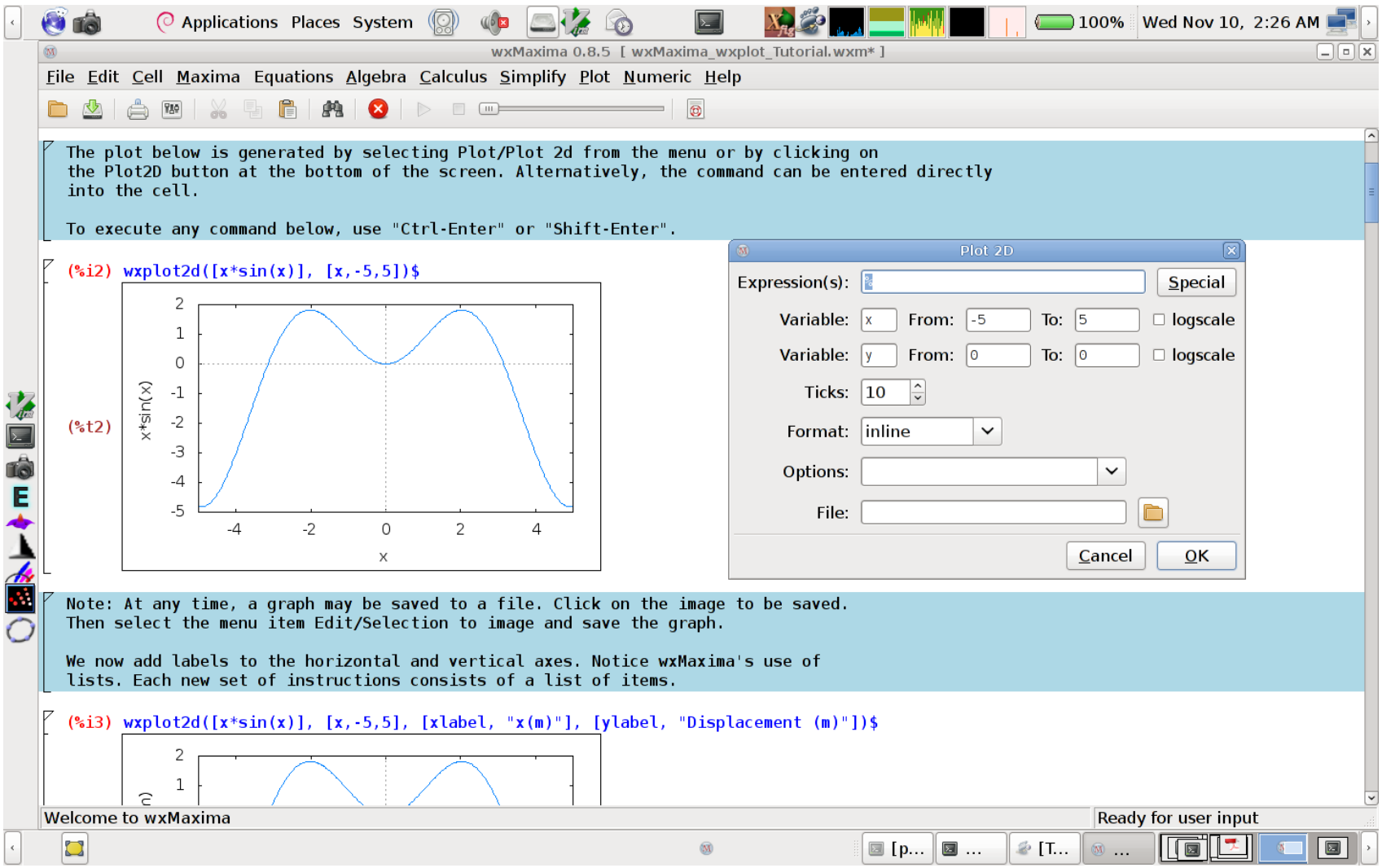
Project "The development of the didactic potential of Cracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education

Maple



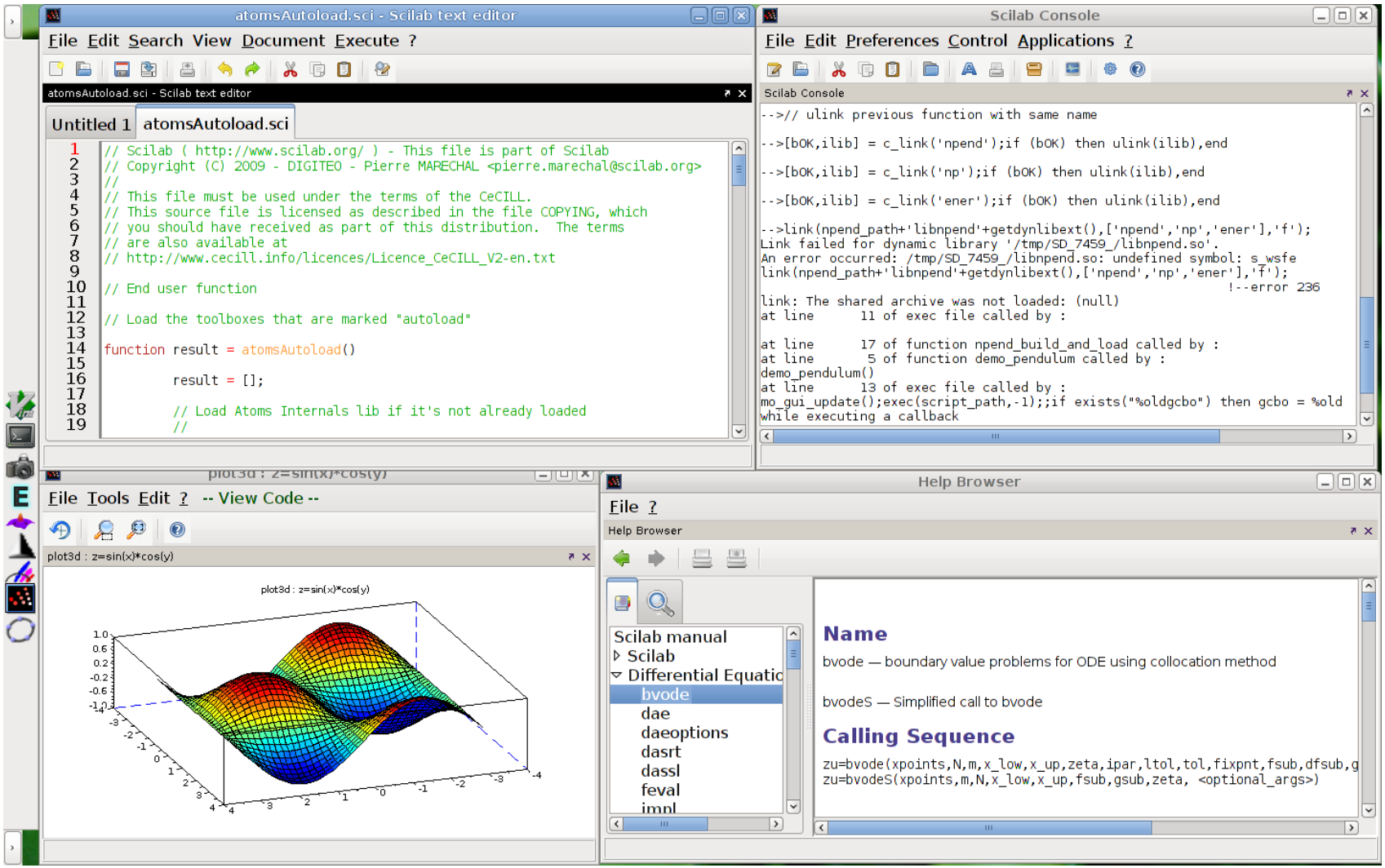
Project "The development of the didactic potential of Cracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education

Maxima + wx = wxMaxima



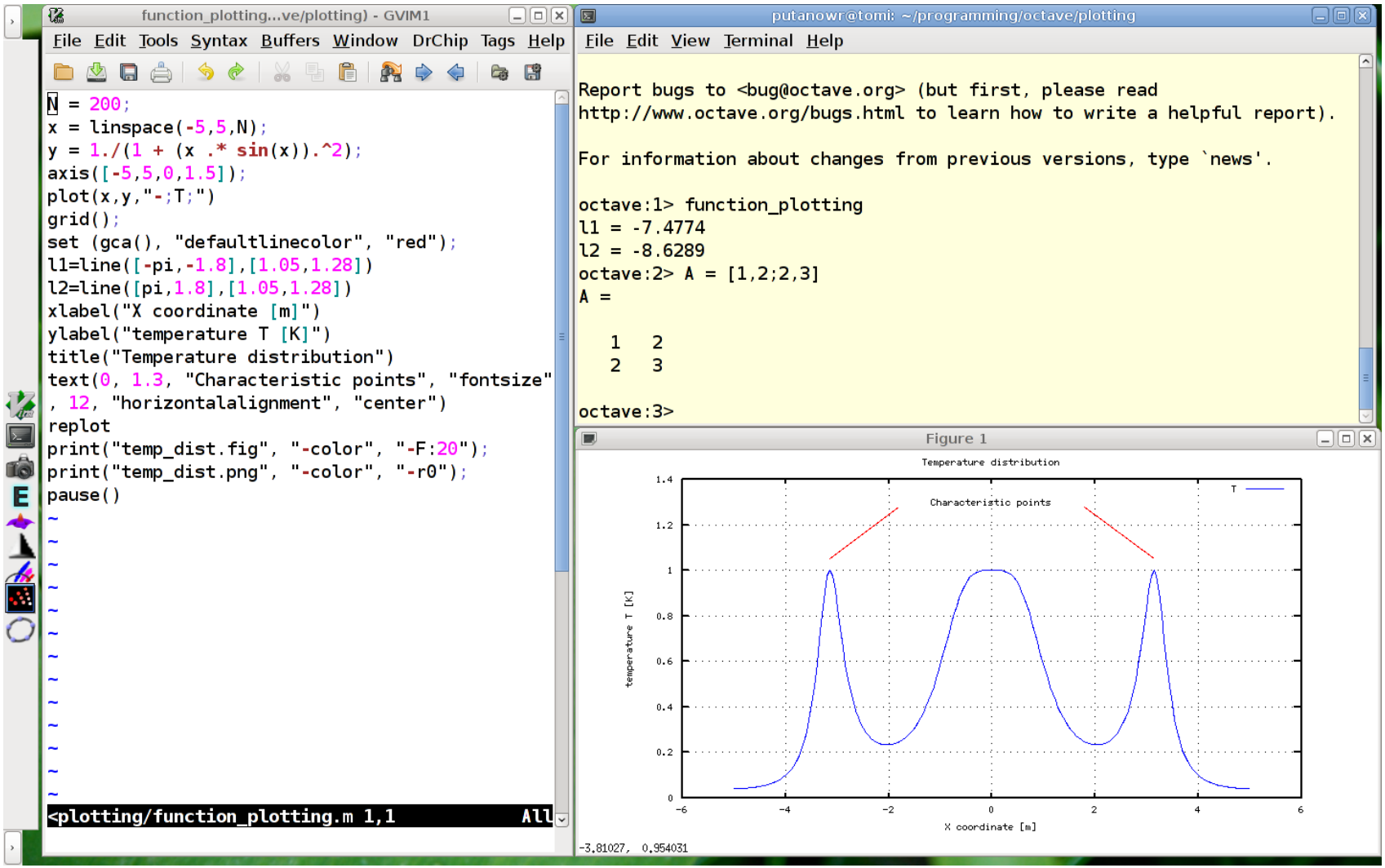
Project "The development of the didactic potential of Cracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education

Scilab



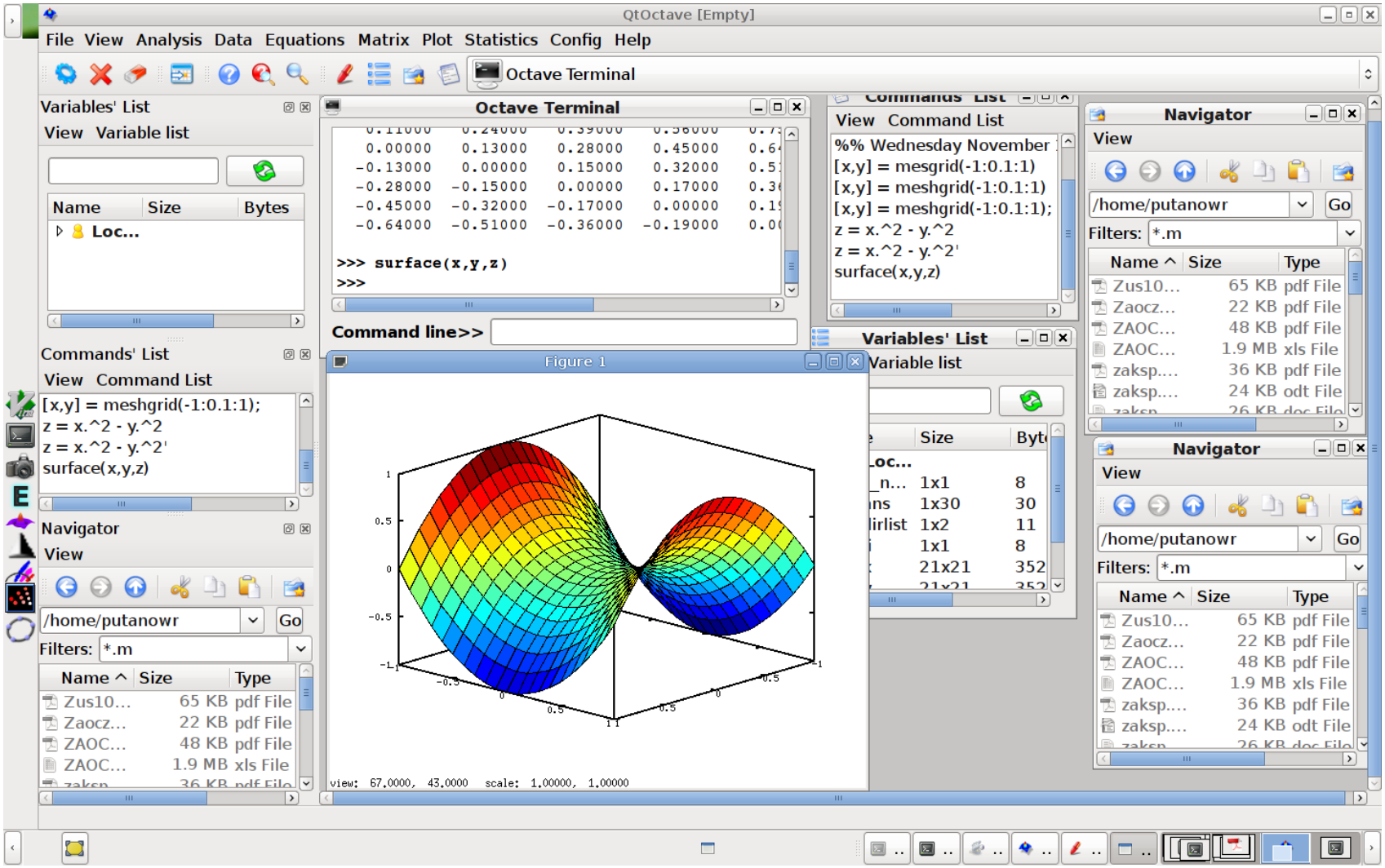
Project "The development of the didactic potential of Cracow University of Technology in the range of modern construction" is co-financed by the European Union within the confines of the European Social Fund and realized under surveillance of Ministry of Science and Higher Education

Octave



Project "The development of the didactic potential of Cracow 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

Octave + Qt = QtOctave



Project "The development of the didactic potential of Cracow 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: C versus Octave program

```c
#include <stdio.h>

#define DIM 3
int main() {
  char *fname = "vect.dat";
  FILE *fh = fopen(fname, "r");

  double u[DIM];
  double v[DIM];
  int i;

  for (i=0; i<DIM; i++) {
    fscanf(fh, "%lf", u+i);
  }
  for (i=0; i<DIM; i++) {
    fscanf(fh, "%lf", v+i);
  }
  double s=0;
  for (i=0; i<DIM; i++) {
    s += u[i]*v[i];
  }
  printf("Scalar product of u and v: %g\n", s);
  return 0;
}
```

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

Project "The development of the didactic potential of Cracow 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 versus Octave program

```
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);
```

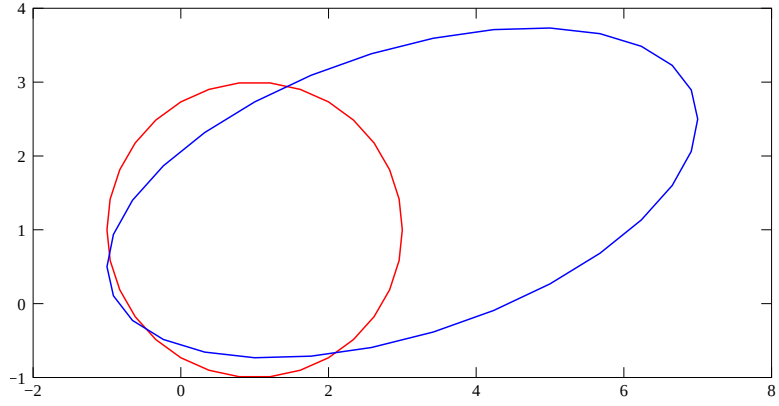
```
1  fname = 'vect.dat';
2  A = load('vect.dat');
3  u = A(1:3);
4  v = A(4:6);
5  s = dot(u,v);
6  printf('Scalar product of u and v: %g\n', s
```

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

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

$$\hat{x} = ax + by + c$$
$$\hat{y} = ex + fy + g$$



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

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

```
1  function [XY] = polygon(x,y,R,N)
2    t = linspace(0,2*pi,N+1);
3    XY = zeros(N+1,2);
4    XY(:,1) = x + R*cos(t);
5    XY(:,2) = y + R*sin(t);
6  end
```

```
7  function plot_poly(XY, color)
8    h = line(XY(:,1), XY(:,2));
9    set(h, 'color',color);
10 end
```

```
11 function [NXY] = transform_poly(XY, T)
12   NXY=zeros(size(XY));
13   for i=1:rows(XY)
14     x = XY(i,1);
15     y = XY(i,2);
16     xh = T(1,1)*x + T(1,2)*y + T(1,3);
17     yh = T(2,1)*x + T(2,2)*y + T(2,3);
18     NXY(i,:) = [xh,yh];
19   endfor
20 endfunction
```

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

Project "The development of the didactic potential of Cracow 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

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`
6. 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

Project "The development of the didactic potential of Cracow 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

Thank you for your attention

Project "The development of the didactic potential of Cracow 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

```
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);
```

Project "The development of the didactic potential of Cracow 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 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);
```

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

```
1  function [XY] = polygon(x,y,R,N)
2    t = linspace(0,2*pi,N+1);
3    XY = zeros(N+1,2);
4    XY(:,1) = x + R*cos(t);
5    XY(:,2) = y + R*sin(t);
6  end
```

back to the main code

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

```
7  function plot_poly(XY, color)
8    h = line(XY(:,1), XY(:,2));
9    set(h, 'color',color);
10 end
```

back to the main code

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

```
11  function [NXY] = transform_poly(XY, T)
12    NXY=zeros(size(XY));
13    for i=1:rows(XY)
14      x = XY(i,1);
15      y = XY(i,2);
16      xh = T(1,1)*x + T(1,2)*y + T(1,3);
17      yh = T(2,1)*x + T(2,2)*y + T(2,3);
18      NXY(i,:) = [xh,yh];
19    endfor
20  endfunction
```

back to the main code

Project "The development of the didactic potential of Cracow 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

Case study: affine transformation in 2D

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

back to the main code

Project "The development of the didactic potential of Cracow 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