computer programming, engineering problems, MATLAB language

Łukasz SOBASZEK[*]

COMPUTER PROGRAMMING AS A TOOL FOR THE ENGINEERING PROBLEMS ANALYSIS

Abstract

The paper presents the use of computer programming to solving typical engineering problems. First of all, the basic information about the computer programming ideology and types of computer programming languages were outlined. Secondly, the Matlab environment that combines computing, visualization and programming was described. In the final part of the paper the author presented solution of the beam with a uniformly distributed load problem by use of special computer program prepared in Matlab language.

1. INTRODUCTION

Computer is a very useful tool in the engineer's work. Thanks to the computer technology, many of calculations and analyses can be done faster and more efficiently. But sometimes computer software has some limitations. The solution could be an appropriate use of computer programming. Knowledge of programming allows to create unique and original software. The programmer can develop program that will help analyze definite problem. Today there are many programming languages, which help to make full use of the computer.

The one of the programming languages is Matlab language. It is a part of complex environment. In this article the use of this language to solve typical engineering problem was presented.

[*] Lublin University of Technology, Institute of Technological Systems of Information, 20-618 Lublin, Nadbystrzycka 36, e-mail: l.sobaszek@pollub.pl

2. COMPUTER PROGRAMMING

2.1. Programming ideology

First computers in the world were built to perform concrete tasks. These computers could realize only limited operation. Every change in the process required modification of the machine. Therefore, the idea to communicate with computer by means of special language understood by the digital machines was raised [1].

But it is very difficult to understand machine language. The basic part of computer – CPU (*Central Processing Unit)* – is built like an integrated circuit and it reacts the incoming bit strings. Therefore, it is very uncomfortable to use machine language. To solve this problem people invented symbolic programming languages. It is method of translating words comprehensible to people to words comprehensible to computer machines [1].

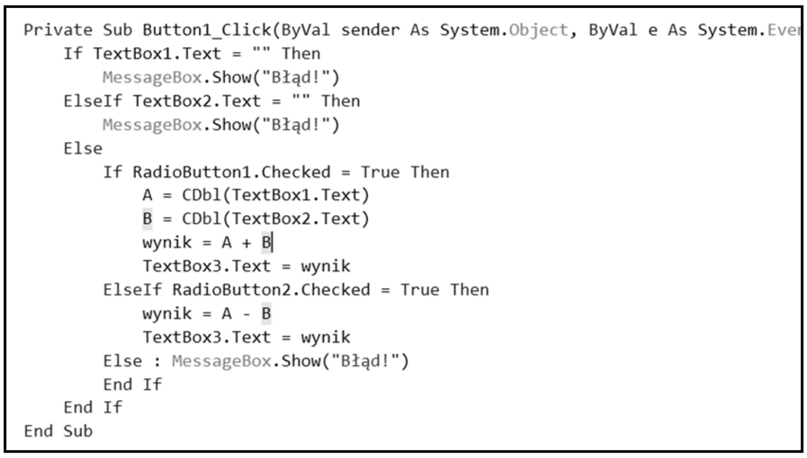
2.2. Programming languages

Programming techniques evolved in parallel with the development of computer hardware. Today term "programming language" means language which is compiled or interpreted to machine form and next executed by processor [9].

Generally, there are two kinds of programming languages – low-level programming languages and high-level programming languages.

The first group is characterized by syntax similar to the machine code. Language is build with elementary instructions understanding for CPU. One instruction is one elementary processor operation. The language of this level includes machine code and Assembler [8].

The second group – high-level programming languages – is characterized by the syntax, similar to human language. Programming code is understandable for programmer, but it requires large commitment of CPU (Fig. 1) [8]. This group include many programming languages. The most popular are: Fortran, Cobol, Logo, Basic, Pascal, Simula, C++, C#, Visual Basic, Delphi, Phyton, PHP, SQL, HTML, Java [9, 8].



```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.Eve
    If TextBox1.Text = "" Then
        MessageBox.Show("Błąd!")
    ElseIf TextBox2.Text = "" Then
        MessageBox.Show("Błąd!")
    Else
        If RadioButton1.Checked = True Then
            A = CDbl(TextBox1.Text)
            B = CDbl(TextBox2.Text)
            wynik = A + B
            TextBox3.Text = wynik
        ElseIf RadioButton2.Checked = True Then
            wynik = A - B
            TextBox3.Text = wynik
        Else : MessageBox.Show("Błąd!")
        End If
    End If
End Sub
```

Fig. 1. Visual Basic – high-level programming language [source: own study]

An example of high-level language is also Matlab language, which is a part of interactive environment. Matlab language has a lot of advantages. It helps to solve many numerical, mathematical and engineering problems faster than traditional languages – e.g. C++ or Java.

3. MATLAB ENVIRONMENT

Matlab is a software package which combines calculations, visualizations and computer programming. This software is used in many areas e.g mathematical calculation, numerical algorithms, simulation and modeling, engineering graphics and other [10].

The basic element of Matlab environment is the matrix. Therefore, the structure of the program allows to perform operation on vectors, scalars, real matrixes [6]. Matlab does not require table structures declaration. The use of two-dimensional dynamic matrix helps solving many technical problems, which are described by means of matrixes or vectors. Matlab includes special libraries (named “toolboxes”) to solve many specific problems (e.g. Optimization Toolbox, Neural Network Toolbox [3]. It also allows to create own scripts and functions. The name “MATLAB” comes from the words MATrix LABoratory [6].

3.1. Basic elements of Matlab

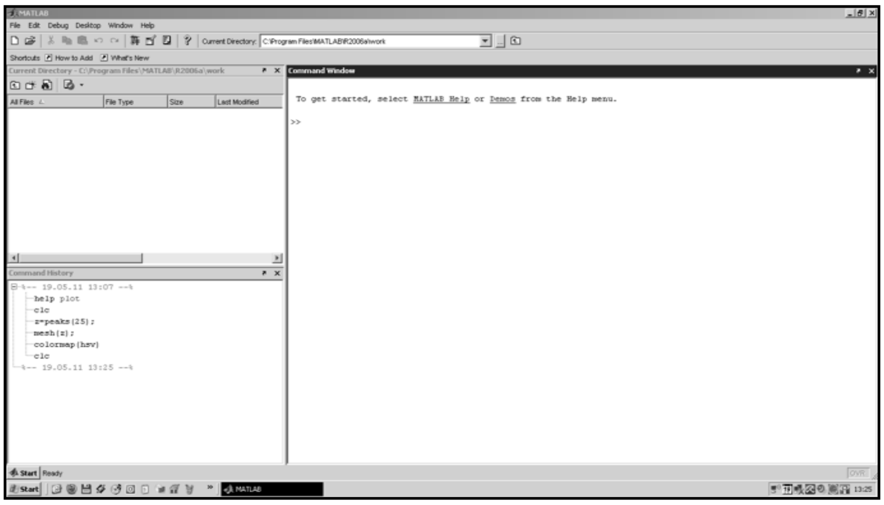
Matlab package consists of five basic elements:

- Matlab language – it is a high-level programming language, which allows to create small programs and build advanced application. This language includes function, inputs, outputs services and objective projecting elements.

- Matlab operating environment – it is a tool to manage variables in workspace, m-files, application, data import and export.
- Graphic system – includes high-level function of making two- and three dimensional graphs, image processing functions and allows to create animations. In addition, the system allows to edit graphics and build a graphical user interface.
- The library of mathematical functions – library that contains many mathematical functions:
  - basic functions (e.g. addition, trigonometric functions, complex numbers functions),
  - matrix functions (e.g. inverse matrix calculations),
  - specialized mathematical function (e.g. Fast Fourier Transform, Bassel functions).
  - API interface – it is library for creating programs in C and Fortran languages, which interact with programs written in Matlab language [3].

Work in Matlab may be twofold:

- direct – user work in typical working mode (in the form of "question – answer") (Fig. 2),
- indirect – activities are implemented by means of programs and scripts written in Matlab language. That helps to perform quick and efficient calculations and results presentation.


MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\Program Files\MATLAB\R2006a\work
Shortcuts How to Add What's New
Current Directory - C:\Program Files\MATLAB\R2006a\work
All Files
File Type
Size
Last Modified
Command Window
Command History
Start Ready

```
To get started, select MATLAB Help or Demos from the Help menu.

>>
```

```
%-- 19.05.11 13:07 --%
help plot
clc
z=peaks(25);
mesh(z);
colormap(hsv)
clc
%-- 19.05.11 13:25 --%
```

Fig. 2. Direct mode in Matlab [soure: own study]

3.2. Programming in Matlab

Matlab environment allows to create own programs and applications. Computer programming in Matlab helps to use many functions and create programs for solving many engineering problems. Matlab programming language is very clear and simple. It is very similar to the typical programming languages – e.g. C languages.

Prepared codes can be saved in special m-file. This type of file can be prepared and modified in many ways. M-file is a ASCII file with “*.m” extension. Besides the sequence of commands, this type of file may include references to other m-files or to itself. M-file can be script or function file. Script files contain strings of commands. They operate on the variables included in the workspace. The aim of the script files is to insert the data and save the results. M-files with scripts can also include computational algorithms. The second type of files are function files. They contain user-defined functions and operate on local and global variables. Communication with the workspace is via formal parameters and global variables. Function m-file should begin with the key word „function”. Moreover, file should contain name of function and list of input parameters [4].

In m-files it is able to use all of the Matlab methods and elements: preparatory functions, arithmetic operations, vectors and matrix operations and graphic functions. In the Matlab language we can use conditional statements (“if – elseif – else – end” and “try – catch – end”), switch statement (“switch – case – otherwise – end”) and iterative statement („while – end” and „for – end”). Moreover, instructions “break” and “return” are available. It is very important that the program is interactive. To realize this purpose user can use input/output functions (Tab. 1). In Matlab language, should enter the name of the variable to display the value of the variable.

Tab. 1. Input script functions [3]

Function	**Description**
x=input(‘text’)	display the text string; waiting to enter a data and assign them the variable x
x=input(‘text’,‘s’)	display the text string; waiting to enter a string and assign it the variable x
pause	stop script execution until the user presses any key
pause(n)	stop the script for n seconds

4. BEAM WITH A UNIFORMLY DISTRIBUTED LOAD

Appropriate use of elements of Matlab language helps to analyze and solve many engineering problems. An example can be analysis of beam with a uniformly distributed load, which is a typical mechanical problem.

4.1. Analysis of the problem

The simply supported beam has a length of *l*. Value of uniformly distributed load is *q* (Fig. 3). The height of rectangular cross-section of the beam is *h* and width is *b* (Fig. 4). Draw the shear force and bending moment diagrams of this beam. Calculate the deflection *f* and the angle of deflection Θ.

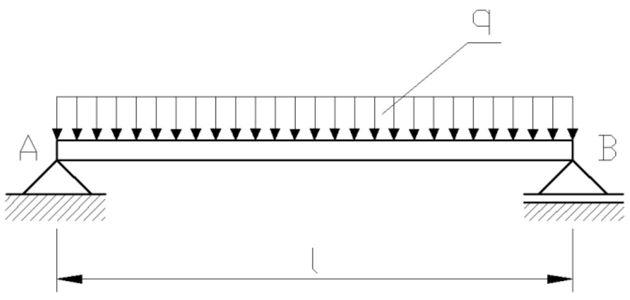


Fig 3. Considered beam [source: own study]

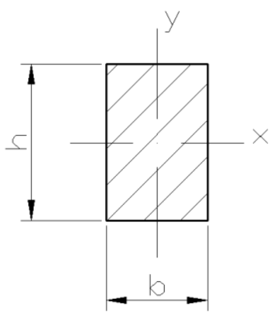


Fig. 4. Rectangular cross-section of the beam [source: own study]

The first part of this problem solve will be determinining the value of the reactions at the supports. For this purpose, uniformly distributed load replaced force *Q* = *ql*. Because this system is statically determinate, the reaction at the supports are calculated by equations:

$$\sum P_{ix} = 0 \tag{1}$$

$$\sum P_{iy} = R_A + R_B - Q = 0 \tag{2}$$

$$\sum M_{iA} = Q \cdot \frac{l}{2} - R_B \cdot l = 0 \tag{3}$$

where:

ΣP_{ix} – sum of forces in *x*-axis,
ΣP_{iy} – sum of forces in *y*-axis,
ΣM_{iA} – sum of the bending moments relative to the support A,
R_A – reaction at the support A,
R_B – reaction at the support B,
Q – force on the value of ql,
l – length of the beam.

Based on the above equations, determined reaction at the support B:

$$R_B = \frac{Q}{2} = \frac{ql}{2} \tag{4}$$

By creating a system of equations is determined reaction at the support A:

$$\begin{cases} R_B = \dfrac{ql}{2} \\ R_A + R_B = Q \end{cases} \tag{5}$$

$$\begin{cases} R_B = \dfrac{ql}{2} \\ R_A = Q - R_B = Q - \dfrac{ql}{2} = \dfrac{ql}{2} \end{cases} \tag{6}$$

where:

R_A – reaction at the support A,
R_B– reaction at the support B,
Q – force on the value of ql,
q – uniformly distributed load,
l – length of the beam.

The next step will be to determine shear force and bending moment at a distance *x* form the support A (Fig. 5).

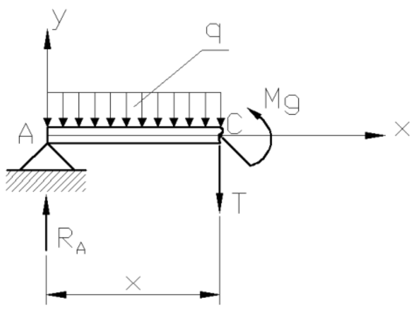


Fig. 5. Shear force *T* and bending moment *M*$_g$ [source: own study]

The value of the shear force are calculated from the following equation:

$$\sum P_{ix} = 0 \tag{7}$$

$$\sum P_{iy} = R_A - qx - T = 0 \tag{8}$$

hence:

$$T(x) = R_A - qx = \frac{ql}{2} - qx \tag{9}$$

$$T(0) = \frac{ql}{2}$$

$$T(\frac{l}{2}) = 0$$

$$T(l) = -\frac{ql}{2}$$

where:

ΣP_{ix} – sum of forces in *x*-axis,
ΣP_{iy} – sum of forces in *y*-axis,
x – distance from the supports A,
q – uniformly distributed load,
l – length of the beam,
T – shear force.

Graph of shear forces in the beam will be as follows:

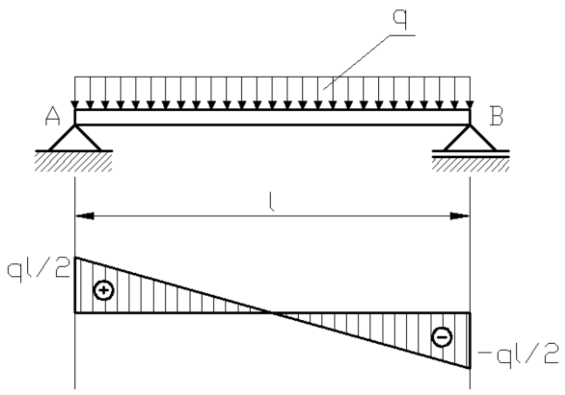


Fig. 5. Graph of shear forces [source: own study]

Value of bending moment is determined on the basis of sum of bending moments relative to the point C:

$$\sum M_{iC} = R_A \cdot x - q \cdot x \cdot \frac{x}{2} - M_g = 0 \tag{10}$$

hence:

$$M_g(x) = R_A \cdot x - q \cdot x \cdot \frac{x}{2} = \frac{ql}{2} \cdot x - \frac{qx}{2} \cdot x \tag{11}$$

$$M_g(0) = M_g(l) = 0$$

$$M_g(\frac{l}{2}) = M_{g_{\max}} = \frac{ql^2}{4} - \frac{ql^2}{8} = \frac{ql^2}{8}$$

where:

$\Sigma\, M_{iC}$ – sum of bending moments relative to the point C,
R_A – reaction at the support A,
x – distance from the supports A,
q – uniformly distributed load,
l – length of the beam,
M_g – bending moment.

Graph of bending moments in the beam will be as follows:

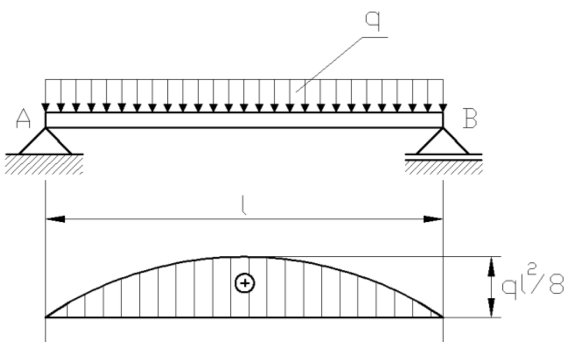


Fig. 6. Graph of bending moments [source: own study]

The deflection *f* and the angle of deflection Θ should also be calculated through analysing considered problem (Fig. 7).

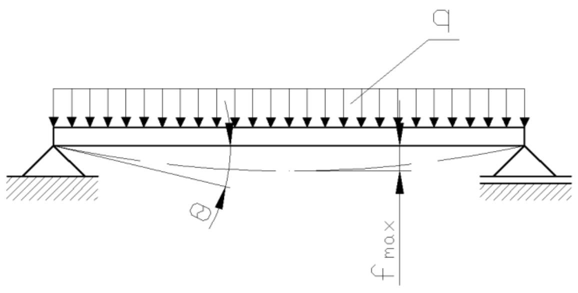


Fig. 7. Deflection *f* and the angle of deflection [source: own study]

These values are determined based on the differential equation:

$$EI\,\frac{d^2y}{dx^2} = M_g(x) \tag{12}$$

where:

E – Young modulus,
I – moment of inertia,
$\frac{d^2y}{dx^2}$ – the Second Derivatives,
$M_g(x)$ – bending moment.

After performing the necessary calculations (integrating the equation) the deflection and the angle of deflection are determined at any point in beam:

$$EI\,\frac{d^2y}{dx^2} = M_g(x) = \frac{ql}{2}\cdot x - \frac{q}{2}\cdot x^2 \tag{13}$$

$$EI\,\frac{dy}{dx} = \frac{ql}{4}\cdot x^2 - \frac{q}{6}\cdot x^3 + C$$

$$EIy = \frac{ql}{12}\cdot x^3 - \frac{q}{24}\cdot x^4 + Cx + D$$

where:

E – Young modulus,
I – moment of inertia,
$\frac{d^2y}{dx^2}$ – the Second Derivatives,
$\frac{dy}{dx}$ – the First Derivatives,
x – distance from the supports A,
y – deflection,
q – uniformly distributed load,
l – length of the beam,
C, D – constants of integration.

Use of the balance beam equations ($y_{(x=0)} = 0$, $y_{(x=l)} = 0$), determined constants of integration:

$$C = -\frac{ql^3}{24} \tag{14}$$

$$D = 0$$

Therefore, deflection and angle of deflection calculated on the basis of formulas:

$$f(x) = y = \frac{ql}{12}\cdot\frac{x^3}{EI} - \frac{q}{24}\cdot\frac{x^4}{EI} + \frac{ql^3}{24}\cdot\frac{x}{EI}\,, \tag{15}$$

$$f(0) = f(l) = 0\,,$$

$$f\left(\frac{l}{2}\right) = f_{\max} = \frac{5}{384} \cdot \frac{ql^4}{EI},$$

$$\Theta = \frac{1}{24} \cdot \frac{ql^3}{EI}, \tag{16}$$

where:

f – deflection,
Θ – angle of deflection,
E – Young modulus,
I – moment of inertia,
x – distance from the supports A,
q – uniformly distributed load,
l – length of the beam.

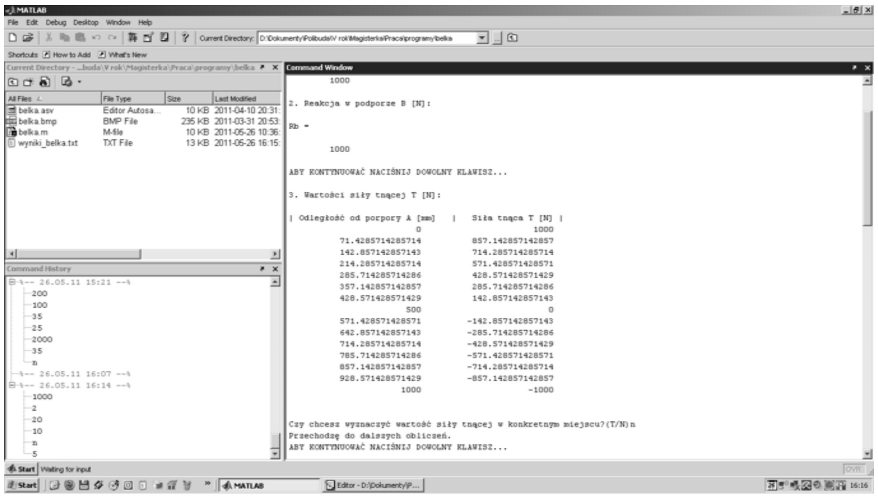
Value of moment of inertia *I* depends on the cross-section of the beam. Young modulus *E* has different value depends on the kind of material.

Presented analysis were based on publications [2, 5, 7]. The analysis above is a mathematical model of the problem. This model can be very useful during preparation of a computer program. Through using mathematical formulas in common with making programming languages, programmer can develop useful tool to analyze beam with a uniformly distributed load. The program made in such a way helps analyzing several cases of this problem.

4.2. Developed program

The use of MathWorks Company environment allows to prepare computer program which can be useful during analyze above issue. Prepared program was written by use of Matlab and helps to thoroughly analyze the problem of beam with a uniformly distributed load. Program running in direct mode – user see question and next give an answer. User entered all of necessary values and next, the computer program execute calculations.

After starting the program, the user is asked about the value of the geometrical parameters of the beam – length, height, width, and moreover value of the uniformly distributed load. Based on these variables the user gets information about values of the reactions at the supports, shear forces, bending moments, deflection and the angle of deflection.



All Files	File Type	Size	Last Modified
belka.asv	Editor Autosa...	10 KB	2011-04-10 20:31:
belka.bmp	BMP File	235 KB	2011-03-31 20:53:
belka.m	M-file	10 KB	2011-05-26 10:36:
wyniki_belka.txt	TXT File	13 KB	2011-05-26 16:15:

```
%-- 26.05.11 15:21 --%
  200
  100
  35
  25
  2000
  35
  n
%-- 26.05.11 16:07 --%
%-- 26.05.11 16:14 --%
  1000
  2
  20
  10
  n
  5
```

```
        1000

2. Reakcja w podporze B [N]:

Rb =

        1000

ABY KONTYNUOWAĆ NACIŚNIJ DOWOLNY KLAWISZ...

3. Wartości siły tnącej T [N]:

| Odległość od porpory A [mm]   |   Siła tnąca T [N] |
                          0                  1000
          71.4285714285714       857.142857142857
          142.857142857143       714.285714285714
          214.285714285714       571.428571428571
          285.714285714286       428.571428571429
          357.142857142857       285.714285714286
          428.571428571429       142.857142857143
                        500                     0
          571.428571428571      -142.857142857143
          642.857142857143      -285.714285714286
          714.285714285714      -428.571428571429
          785.714285714286      -571.428571428571
          857.142857142857      -714.285714285714
          928.571428571429      -857.142857142857
                       1000                 -1000


Czy chcesz wyznaczyć wartość siły tnącej w konkretnym miejscu?(T/N)n
Przechodzę do dalszych obliczeń.
ABY KONTYNUOWAĆ NACIŚNIJ DOWOLNY KLAWISZ...
```

MATLAB
File Edit Debug Desktop Window Help
Current Directory: D:\Dokumenty\Polibuda\V rok\Magisterka\Praca\programy\belka
Shortcuts How to Add What's New
Current Directory - ...buda\V rok\Magisterka\Praca\programy\belka
Command Window
Command History
Start Waiting for input
Start
MATLAB
Editor - D:\Dokumenty\P...
16:16

Fig. 8. Calculation of the shear forces [source: own study]

Source code begins function *clear* that deletes all of variables from memory and function *clc* that clear workspace. Function *format* helps define accuracy of the displayed results. The results of program work are exported to text file by means of special function *diary*. All of displayed information are saved in the file *results.txt* in the program directory. The input data is processed to determinate searched values. During entering geometrical parameters of considered beam user can use auxiliary schema of beam. Image is shown by means of function *imshow* and loaded by means of function *imread*. Function *figure* activates special window for auxiliary image.

Program include many conditions in its structure. This helps to avoid many problems and errors. Conditions are achieve by *if* conditional statements. Results are presented step by step. This is possible thanks to the *pause* function. Function *pause* stopped program execution until user press any key on the keyboard. Results displaying precedes function *date* that displays current date.

During the calculation function *linspace* is used. This function allows to express length of beam as vector with predefined number of elements. That helps to determine the value of shear force, bending moment and deflection at different distances from the left support of the beam. Moreover, this function is very useful while generate graphs by the program. Value of the shear force and bending moment could be determinate at a particular point of the beam.

In the developed program the user define kind of material of the beam. *Switch* conditional statements contains few option: steel, cast iron, aluminum, copper, wood or concrete.

After the calculations, program generates the graphs. Graphs of the shear force, bending moment and deflection are showed by means of function *plot*. Graphs are shown in the special graphical window (activated by the command *figure*). The developed program allows summary graphs of the deflection of the beams made of different materials (Fig. 9). This option helps to compare different variants of the calculations. Moreover, the relationship between deflection and beam material is presented on the graphs.

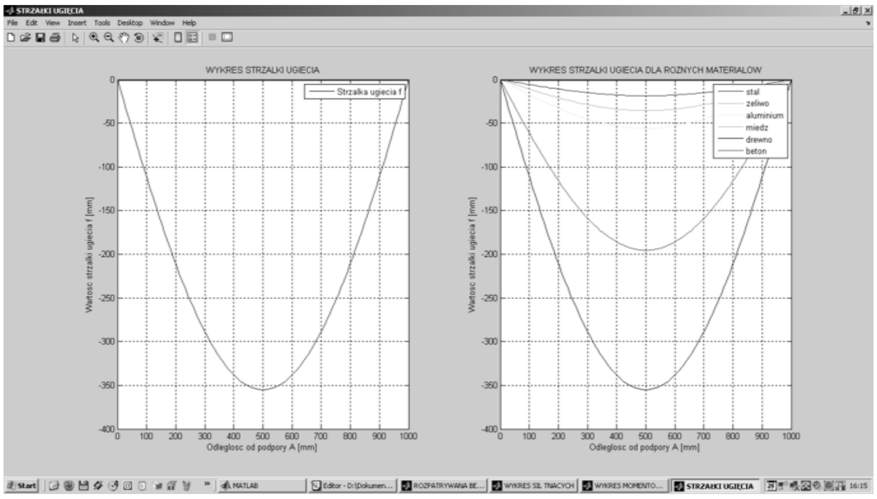


Fig. 9. Deflection of the beams made of different materials [source: own study]

In the end of the program user decide what program should do – run again and make calculation for other data or exit the program. Function *run* starts program from the beginning. Function *quit* close the program.

5. SUMMARY

Nowadays computer is used in many areas of life. It is also a necessary tool in the engineer's work. Computer programming can be very useful in engineer work too. Knowledge of programming allows to create own software and using it to solve many problems. Specially dedicated software accelerates research. There is a lot of programming languages, but Matlab language certainly stands out. Simplicity and ease of this language make that Matlab language is using in many areas. Presented problem is a one of the many, because engineer can also create programs to analyze other issues – e.g. construction problems, machining problems and many other.

REFERENCES

[1] DUCH W.: *Fascynujący świat programów komputerowych*. Nakom, Poznań, 1997.
[2] IWULSKI Z.: *Wyznaczanie sił tnących i momentów zginających w belkach. Zadania z rozwiązaniami.* Uczelniane Wydawnictwa Naukowo – Techniczne, Kraków, 2001.
[3] KAMIŃSKA A., PIŃCZYK B.: *Ćwiczenia z Matlab – przykłady i zadania*. MIKOM, Warszawa 2002.
[4] MROZEK B., MROZEK Z.: *Matlab i Simulink. Poradnik użytkownika.* Helion, Gliwice, 2004.
[5] NIEZGODZIŃSKI M. E., NIEZGODZIŃSKI T.: *Zadania z Wytrzymałości materiałów*. Wydawnictwa Naukowo – Techniczne, Warszawa.
[6] PRATAP R.: *MATLAB7 dla naukowców i inżynierów*. PWN, Warszawa, 2013.
[7] RUSSELL JOHNSTON F. B., DEWOLF JR. J.; MAZUREK D.: *Mechanics of Materials*, 2012.
[8] SAMOŁYK G.: *Podstawy programowania komputerów dla inżynierów*. Politechnika Lubelska, Lublin, 2011.
[9] SOBIESKI W.: *Języki Programowania – materiały uzupełniające do wykładów*. Olsztyn, 2006.
[10] http://www.mathworks.com