WARSAW UNIVERSITY OF TECHNOLOGY Index 351733
FACULTY OF CIVIL ENGINEERING
COMMITTEE FOR CIVIL AND WATER ENGINEERING
POLISH ACADEMY OF SCIENCES ISSN 1230-2945

DOI: 10.24425/ace.2022.140169

ARCHIVES OF CIVIL ENGINEERING

Vol. LXVIII ISSUE 1 2022

pp. 299 –317

Research paper

Application of Matlab software in static calculations of bridge structures

Paweł Hawryszków[1], Bronisław Czaplewski[2]

Abstract: Mathematical package Matlab is a very convenient programming language, used for calculations in the field of linear algebra for scientists and engineers. Its main advantage for civil engineers is the simplicity of the language and the wide range of application in the field of linear statics. This mathematical platform was used for programming of static calculations of multi-span, continuous, beam bridge structures. In the formulated theoretical approach, the internal forces were calculated using the method of forces. Knowing the influence matrix and load values in the unit states, the envelope of internal forces can be determined. The first step is entering the vector of loads and the second is calculating an envelope using special function. Obtaining the results from individual loads in a variety of operating conditions, it is possible to calculate the global envelope of internal forces and proceed with modifications of the model. The theoretical approach was computationally tested on the example of an alternative design concept of the MA-46 bridge along the A4 motorway. One of the biggest advantages of the discussed computational approach is the wide access to the results of intermediate calculations. Another benefits of working with mathematical packages are improving insight in the field of static calculations and getting used to working with code like in some programs for structural analysis (e.g. SOFiSTiK). The discussed computational approach is a good way to pre-design due to the little time required to compare several variants of solution, so it can be helpful in optimizing the structure.

Keywords: static calculations, bridges, envelopes of internal forces, mathematical packages, Matlab

[1]DSc., PhD., Eng., Wrocław University of Science and Technology, Faculty of Civil Engineering, ul. Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland, e-mail: pawel.hawryszkow@pwr.edu.pl, ORCID: 0000-0001-8787-1121

[2]MSc., PhD. candidate, Eng., Wrocław University of Science and Technology, Faculty of Civil Engineering, ul. Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland, e-mail: bronislaw.czaplewski@pwr.edu.pl, ORCID: 0000-0001-7775-8993

1. Introduction

One of the most important stages in the process of designing building structures is the process of determining extreme internal forces. The final result of this process is the envelope of internal forces, which is a set of minimum and maximum values of these forces in all cross-sections, for all load combinations.

The specificity of designing bridge structures lies in the movable nature of the loads, which in practice gives a very large number of load combinations. Therefore, an inherent element of the process of determining internal forces is the use of the function of the influence of static quantities, which, collected in a set for many cross-sections, form the influence matrix. This study discusses the process of determining the influence matrix for multi-span, continuous, beam bridges and then use it to determine the envelope of internal forces using Matlab.

The Matlab package [1] is the software whose programming language allows to work with matrices, vectors and structures. The matrix as the basic data type makes the computation less time consuming than is the case with commonly used programming languages. Another simplification is that there is no need to declare variables or specify the data type. Operations on entire matrices mean that one short operation often replaces several loops written in Java or C++ which is widely used in calculations in the field of linear statics.

This mathematical package is a valuable computational tool, which can be efficiently used in mechanics of structures [2]. It can significantly support the computer-aided design analysis and process of bridge superstructures optimization [3–6].

2. Formulation of the theoretical approach

2.1. Influence matrix

When starting the process of determining the influence matrix for a bridge structure the following assumptions were made:

– static scheme of a continuous beam without cantilevers,
– constant cross-section characteristics along the entire length of the beam,
– homogeneous, isotropic, linear-elastic material.

The definition of a structure model with the assumed scheme can be reduced to a vector whose elements are the lengths of successive spans of the structure. The following markings were introduced: $\boldsymbol{L} = [l_1, l_2, \ldots, l_N]$ – a vector representing the static scheme, where l_i, $i = 1, 2, \ldots, N$ denotes the length of the i-th span of a N-span continuous beam.

The influence matrix consists of vectors (in this case row vectors) representing the graphs of the searched static quantity (shear force or bending moment) from the unit force set successively at points spaced from each other by a given calculation step marked as dx. The vector $\boldsymbol{L}$ and the constant dx the only data needed to determine the influence matrix of $\boldsymbol{Im}$, therefore:

$$\boldsymbol{Im} = \boldsymbol{Im}(\boldsymbol{L}, dx) \tag{2.1}$$

The force method was used to determine the diagrams of internal forces from unit load. The first step in the method above is to create the basic scheme. In the function algorithm it is obtained by removing the first $N - 1$ supports as shown in Fig. 1.

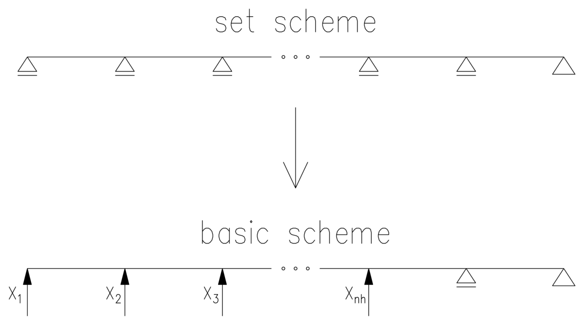

Fig. 1. Creation the basic scheme of the force method

Internal forces in the basic scheme in unit states and from forces $X_i = 1$ was determined by the elementary method using a loop *for*. The following markings were introduced:

$\overline{\boldsymbol{MP}}_i$ – line vector representing the diagram of bending moments from the unit force set at $x = (i - 1) \cdot dx$, $i = 1, 2, \ldots, n$, where n results from the quotient of the sum of the lengths of all spans and the calculation step, according to the formula: $n = \sum_{k=1}^{N} l_k \cdot \frac{1}{dx} + 1$,

$\overline{\boldsymbol{MX}}_i$ – line vector representing the diagram of bending moments induced by force $X_i = 1$, $i = 1, 2, \ldots, n_h$, where $n_h = N - 1$ is the degree of static indeterminacy of the structure model,

$\overline{\boldsymbol{SP}}_i$ – line vector representing the diagram of shear forces from the unit force set at $x = (i - 1) \cdot dx$, $i = 1, 2, \ldots, n$, where n results from the quotient of the sum of the lengths of all spans and the calculation step, according to the formula: $n = \sum_{k=1}^{N} l_k \cdot \frac{1}{dx} + 1$,

$\overline{\boldsymbol{SX}}_i$ – line vector representing the diagram of shear forces induced by force $X_i = 1$, $i = 1, 2, \ldots, n_h$, where $n_h = N - 1$ is the degree of static indeterminacy of the structure model.

The above matrices are indirectly used to solve the equation of the force method of the form:

(2.2) $$\boldsymbol{DX} + \boldsymbol{D_P} = \boldsymbol{0}$$

The solution of equation (2.2) has the form:

(2.3) $$\boldsymbol{X} = -\boldsymbol{D}^{-1}\boldsymbol{D_P}$$

where:

$\boldsymbol{D} = \{d_{ij}\}$ – flexibility matrix, $d_{ij} = \sum_{k=1}^{n} \overline{MX}_{i,k} \cdot \overline{MX}_{j,k} \cdot dx$, $i, j = 1, 2, \ldots, n_h$, where:

$\overline{MX}_{i,k}$ – k-th element of the vector $\overline{\boldsymbol{MX}}_i$,

$\overline{MX}_{j,k}$ – k-th element of the vector $\overline{\boldsymbol{MX}}_j$,

$\boldsymbol{D_P} = \left\{d_{P,ij}\right\}$ – matrix of generalized displacements in the direction of unknown X_{ij} induced by the unit force set at $x = (j-1) \cdot dx$, $d_{P,ij} = \sum_{k=1}^{n} \overline{MX}_{i,k} \cdot \overline{MP}_{j,k} \cdot dx$, $i = 1, 2, \ldots, n_h$, $j = 1, 2, \ldots, n$, where:

$\overline{MX}_{i,k}$ – k-th element of the vector $\overline{\boldsymbol{MX}}_i$,

$\overline{MP}_{j,k}$ – k-th element of the vector $\overline{\boldsymbol{MP}}_j$,

$\boldsymbol{X}$ – the result of the force method equation.

The last step to obtain the influence matrix of the searched static quantities is to solve the equations:

– for bending moments

$$\boldsymbol{Im} = \boldsymbol{X}^T \overline{\boldsymbol{MX}} + \overline{\boldsymbol{MP}} \tag{2.4}$$

where:

$$\overline{\boldsymbol{MX}} = \begin{bmatrix} \overline{\boldsymbol{MX}}_1 \\ \overline{\boldsymbol{MX}}_2 \\ \cdots \\ \overline{\boldsymbol{MX}}_{n_h} \end{bmatrix}, \qquad \overline{\boldsymbol{MP}} = \begin{bmatrix} \overline{\boldsymbol{MP}}_1 \\ \overline{\boldsymbol{MP}}_2 \\ \cdots \\ \overline{\boldsymbol{MP}}_{n} \end{bmatrix}$$

– for shear forces

$$\boldsymbol{Im} = \boldsymbol{X}^T \, \overline{\boldsymbol{SX}} + \overline{\boldsymbol{SP}} \tag{2.5}$$

where:

$$\overline{\boldsymbol{SX}} = \begin{bmatrix} \overline{\boldsymbol{SX}}_1 \\ \overline{\boldsymbol{SX}}_2 \\ \cdots \\ \overline{\boldsymbol{SX}}_{n_h} \end{bmatrix}, \qquad \overline{\boldsymbol{SP}} = \begin{bmatrix} \overline{\boldsymbol{SP}}_1 \\ \overline{\boldsymbol{SP}}_2 \\ \cdots \\ \overline{\boldsymbol{SP}}_{n} \end{bmatrix}$$

Having the appropriate function written in Matlab the influence matrix can be obtained by typing just a few numbers. The influence matrix constitutes the algebraic basis of the solution space of a given static scheme the size of which is determined by a given calculation step. In the numerical sense it is a structure model with access to all desired unit states and influence functions in any cross-section.

2.2. Internal forces envelope

On the basis of the influence matrix and the values of loads, the envelope of the searched static quantity can be determined. Due to the algorithm of determining the envelope the impacts are divided into two groups: mobile and fixed (related to the point or area of application in a given load scheme). It should be noted here that loads distributed from the vehicle uniformly distributed loading or the pedestrian crowd are classified in this case as fixed.

For the purpose of calculating road bridges a dedicated function was created for the influence of the four-axle vehicle. In the first step the function creates a combination in the form of the sum of the four unit load matrices corresponding to the axles of the vehicle. To fully reflect the passage of the standard vehicle each of the influence matrices is supplemented with zero lines corresponding to the position of the axles outside the deck. The scheme of operation is easiest to illustrate by means of rectangles symbolizing the matrices as in Fig. 2.

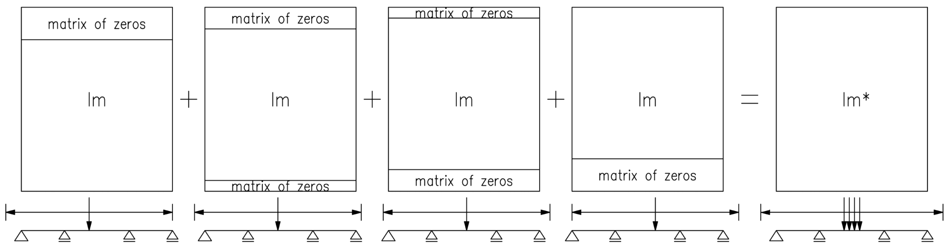

Fig. 2. Scheme of matrix operation showing the influence of four-axle vehicle driving through the bridge

In the second step the function finds the maximum and minimum element in each column and multiplies them by the load value per axle of the standard vehicle. The value returned by the function is a matrix composed of two lines representing the diagrams of minimum and maximum forces. Finally, the envelope is obtained as a function of the three variables, that is:

$$\boldsymbol{Env\,K} = \boldsymbol{Env\,K}(\boldsymbol{Im},\ P,\ dx) \tag{2.6}$$

where:

Env K – a matrix representing the extreme values of internal forces in [kN],

Im – influence matrix of the searched static quantity,

P – value of the load on the vehicle axle in [kN],

dx – calculation step in [m].

The second function to obtain the envelope of internal forces is the envelope from loads related to their point of application. Contrary to the previously described function the most unfavorable load combinations do not depend on the point of application but on whether it has a positive or negative effect on the value of the force in a given section. This is automatically resolved by checking the sign of the corresponding element of the influence matrix. Therefore, a pair of extreme values should be assigned to each analysed point in order to create vectors modelling the load state of the structure. Uniformly distributed loads should be converted into concentrated loads by multiplying their value expressed in [kN/m] by dx.

In the first step, the function transforms the ***Im*** into a minimum loads matrix ***Min*** and matrix of maximum loads ***Max***. This is done as follows:

– for positive elements

$$Max_{ij} = Im_{ij} \cdot Load_{\max,i} \tag{2.7}$$

$$Min_{ij} = Im_{ij} \cdot Load_{\min,i} \tag{2.8}$$

– for negative elements

$$Max_{ij} = Im_{ij} \cdot Load_{\min,i} \tag{2.9}$$

$$Min_{ij} = Im_{ij} \cdot Load_{\max,i} \tag{2.10}$$

where:

Max_{ij} – element of matrix ***Max*** with coordinates i, j,
Min_{ij} – element of matrix ***Min*** with coordinates i, j,
Im_{ij} – element of matrix ***Im*** with coordinates i, j,
$Im_{\max,i}$ – i-th element of vector $\boldsymbol{Load}_{\max}$,
$Im_{\min,i}$ – i-th element of vector $\boldsymbol{Load}_{\min}$,
$\boldsymbol{Load}_{\max}$ – vector of maximum loads,
$\boldsymbol{Load}_{\min}$ – vector of minimum loads.

The second step is to sum up the columns of the ***Min*** and ***Max*** matrices. As a result of summation line vectors representing the diagrams of minimum and maximum forces is obtained. Finally, the envelope is a two-line matrix returned by a three-variable function, marked as follows:

$$\boldsymbol{Env\,P} = \boldsymbol{Env\,P}(\boldsymbol{Im}, \boldsymbol{Load}_{\min}, \boldsymbol{Load}_{\max}) \tag{2.11}$$

The vectors of minimum and maximum loads can be created manually but it can also be automated by creating a special function. The following method of arranging the load parameters was proposed:

$$\boldsymbol{p} = \begin{bmatrix} p_{\min} & p_{\max} \end{bmatrix} \tag{2.12}$$

$$\boldsymbol{P} = \begin{bmatrix} x_1 & P_{1,\min} & P_{1,\max} \\ x_2 & P_{2,\min} & P_{2,\max} \\ & \cdots & \\ x_k & P_{k,\min} & P_{k,\max} \end{bmatrix} \tag{2.13}$$

where:

p – a vector representing the load uniformly distributed over the entire length in [kN/m],
P – matrix representing concentrated loads in [kN] with their points of application.

The data missing for the determination of the load matrix is the calculation step dx and target size of matrix n. As a result of some actions not quoted due to their simplicity and low importance the following function is obtained:

$$\boldsymbol{Load} = \boldsymbol{Load}(\boldsymbol{p}, \boldsymbol{P}, n, dx) = \begin{bmatrix} \boldsymbol{Load}_{\min} & \boldsymbol{Load}_{\max} \end{bmatrix} \tag{2.14}$$

3. Computational example

The analysis is based on the example of the MA-46 bridge over the Bobrzyca watercourse at km 45+958 of the A4 motorway in Poland (Fig. 3). It is located between the Bolesławiec and Krzyżowa junctions. The bridge carries the route over the watercourse by two separate superstructures for the left and right carriageway of the motorway, supported on common abutments. In a static scheme it is a three-span frame and the main girder is a concrete slab.

Fig. 3. MA-46 bridge along the A4 motorway (photo credit: Maciej Hildebrand)

For purposes of this paper and the work [7] an alternative design solution to the existing facility was proposed. In this solution the superstructure is a four-girder beam-plate system (Fig. 4). The cooperation of the girders, apart from the deck slab, is ensured by the cross-members located in the middle of the span. The general concept of the structure remains the same: a three-span concrete road bridge with not changed span lengths of 16 m, 19 m, 16 m (Fig. 5). The static scheme was simplified and defined as a continuous beam.

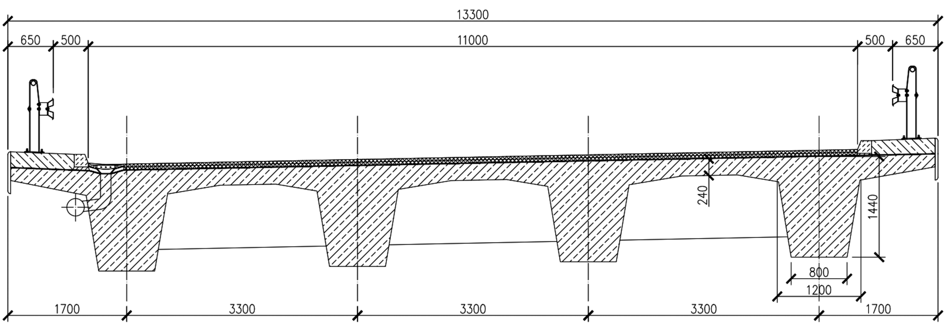

Fig. 4. Cross-section of the deck of an example bridge

With the functions described in Section 2 it is possible to quickly determine the envelopes of bending moments and shear forces for a structure model with a scheme of a continuous beam without cantilevers. The procedure for determining the individual

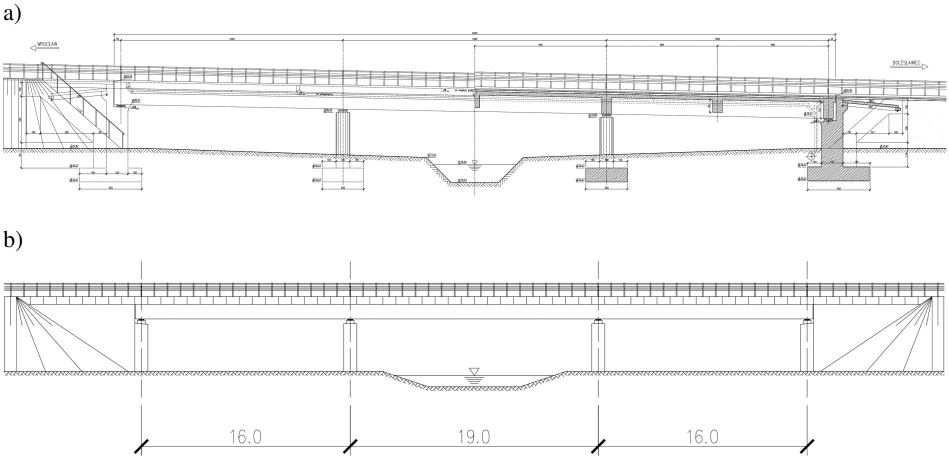

Fig. 5. Side view of an example bridge: a) technical drawing, b) ideological scheme used for static calculations

components and the total envelopes of bending moments, as well as shear forces is shown on the example of the bridge described above and presented in Fig. 4 and Fig. 5.

The static scheme was defined as $\boldsymbol{L}$ = [16 19 16] and the calculation step was assumed dx = 0.1 m. Based on these data the bending moment influence matrix was determined according to (2.1):

$$\boldsymbol{Im} = \boldsymbol{Im}(\boldsymbol{L}, dx) = \boldsymbol{Im}([16\ 19\ 16], 0.1) \tag{3.1}$$

The model was reduced to the one-dimensional space using the influence function of the transverse load distribution. Many, well known methods enabling such operation can be found in the literature, e.g. [8–10]. For purposes of this study the Guyon–Massonnet method was used [10]. After performing the calculations, the values presented in Table 1 were obtained.

Table 1. List of loads

No.	Item	Load value		
		unit	maximum	minimum
1	Deck, girders, equipment	kN/m	81.08	55.99
2	Cross-beams	kN	30.81	22.54
3	Vehicle uniformly distributed load	kN/m	16.37	–
4	Vehicle axle	kN	178.19	–

The envelope of bending moments is divided into the following three components:

$\boldsymbol{Env\,G} = \boldsymbol{Env\,P}(\boldsymbol{Im}, \boldsymbol{Load}_{\min}^{DL}, \boldsymbol{Load}_{\max}^{DL})$ – envelope obtained from dead and equipment load (DL),

$\boldsymbol{Env\,Q} = \boldsymbol{Env\,P}(\boldsymbol{Im}, \mathbf{0}, \boldsymbol{Load}^{UDL}_{\max})$ – envelope obtained from vehicle uniformly distributed load (UDL),

$\boldsymbol{Env\,V} = \boldsymbol{Env\,K}(\boldsymbol{Im}, 178.19, 0.1)$ – envelope obtained from vehicle passage.

The total envelope of bending moments is the sum of:

$$\boldsymbol{Env} = \boldsymbol{Env\,G} + \boldsymbol{Env\,Q} + \boldsymbol{Env\,V} \tag{3.2}$$

4. Discussion of results

All static calculations of the example bridge, described in Section 3, were performed in Matlab software. A block diagram of the programming code was presented in Fig. 6. The entire numerical code can be found in Table 2.

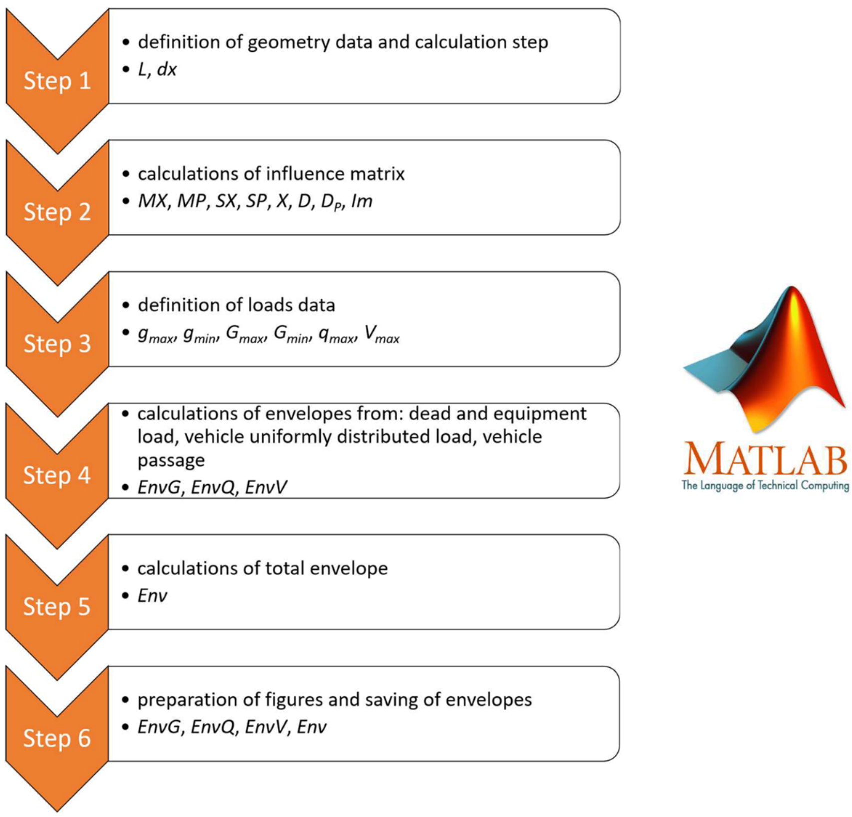

Fig. 6. Block diagram of Matlab programming code used for static calculations

The calculation results obtained from Matlab were used to generate appropriate graphs with envelopes of bending moments and shear forces (Figs. 7–11).

Table 2. Matlab numerical code

Matlab code for calculations of bending moments*

```
% definition of geometry data and calculation
step

L = [16 19 16];
dx = 0.1;

% calculations of influence matrix

nh = length(L) - 1;
n = sum(L)/dx + 1;

MX = zeros(nh,n);
MP = zeros(n,n);
X = zeros(nh,n);
D = zeros(nh,nh);
D_P = zeros(nh,n);

for i = 1:nh
  a = sum(L) - L(length(L));
  b = L(length(L));
  L_rest = L(i:length(L));
  c = sum(L) - sum(L_rest);
  R = -(sum(L_rest)/b);
  x = 0;
  for j = 1:n
  if x <= c
  MX(i,j) = 0;
  end
  if x > c && x <= a
  MX(i,j) = x - c;
  end
  if x > a
  MX(i,j) = x - c + R*(x - a);
  end
  x = x + dx;
  end
end

for i = 1:n
  a = sum(L) - L(length(L));
  b = L(length(L));
  c = (i-1)*dx;
  R = (sum(L) - c)/b;
  x = 0;
  for j = 1:n
  if x <= a
  if x <= c
  MP(i,j) = 0;
  end
  if x > c
  MP(i,j) = c - x;
  end
  end
  if x > a
  if x <= c
  MP(i,j) = R*(x - a);
  end
  if x > c
  MP(i,j) = c - x + R*(x - a);
  end
  end
  x = x + dx;
  end
end
```

Matlab code for calculations of shear forces*

```
% definition of geometry data and calculation
step

L = [16 19 16];
dx = 0.1;

% calculations of influence matrix

nh = length(L)-1;
n = sum(L)/dx + 1;

MX = zeros(nh,n);
MP = zeros(n,n);
SX = zeros(nh,n);
SP = zeros(n,n);
X = zeros(nh,n);
D = zeros(nh,nh);
D_P = zeros(nh,n);

for i = 1:nh
  a = sum(L)-L(length(L));
  b = L(length(L));
  L_rest = L(i:length(L));
  c = sum(L)-sum(L_rest);
  R = -(sum(L_rest)/b);
  x = 0;
  for j = 1:n
  if x<= c
  MX(i,j) = 0;
  end
  if x> c && x<= a
  MX(i,j) = x - c;
  end
  if x> a
  MX(i,j) = x-c + R*(x-a);
  end
  x = x + dx;
  end
end

for i = 1:n
  a = sum(L) - L(length(L));
  b = L(length(L));
  c = (i -1)*dx;
  R = (sum(L)-c)/b;
  x = 0;
  for j = 1:n
  if x <= a
  if x <= c
  MP(i,j) = 0;
  end
  if x > c
  MP(i,j) = c - x;
  end
  end
  if x > a
  if x <= c
  MP(i,j) = R*(x - a);
  end
  if x > c
  MP(i,j) = c - x + R*(x - a);
  end
  end
  x = x + dx;
  end
end
```

Table 2 [cont.]

Matlab code for calculations of bending moments*

```
for i = 1:nh
 for j = 1:nh
 x = MX(i,:).*MX(j,:);
 k = length(x);
 D(i,j)=0.5*dx*(sum(x(1:(k-1)))+sum(x(2:k)));
 end
end

for j = 1:n
 for i = 1:nh
 x = MX(i,:).*MP(j,:);
 k = length(x);
 D_P(i,j)=-0.5*dx*(sum(x(1:(k-1)))+
  sum(x(2:k)));
 end
end

for j = 1:n
 X(:,j) = D\D_P(:,j);
end

Im = X'*MX + MP;

% definition of loads data

g_max = 81.08;
g_min = 55.99;
G_max = zeros(n,1);

% index specifies location of the force

G_max(81) = 30.81;
G_max(256) = 30.81;
G_max(431) = 30.81;
G_min = zeros(n,1);
G_min(81) = 22.54;
G_min(256) = 22.54;
G_min(431) = 22.54;
q_max = 16.37;
V_max = 178.19;

% calculations of envelope from dead and
equipment load

Max = zeros(n,n);
Min = zeros(n,n);

for j = 1:n
for i = 1:n
if Im(i,j)>= 0
Max(i,j) = Im(i,j)*(G_max(i) + g_max*dx);
Min(i,j) = Im(i,j)*(G_min(i) + g_min*dx);
end
if Im(i,j)< 0
Max(i,j) = Im(i,j)*(G_min(i) + g_min*dx);
Min(i,j) = Im(i,j)*(G_max(i) + g_max*dx);
end
end
end

for i = 1:n
EnvG(i,1) = sum(Min(:,i));
EnvG(i,2) = sum(Max(:,i));
end
```

Matlab code for calculations of shear forces*

```
for i = 1:nh
 a = sum(L)-L(length(L));
 b = L(length(L));
 L_rest = L(i:length(L));
 c = sum(L)-sum(L_rest);
 R = -(sum(L_rest)/b);
 x = 0;
 for j = 1:n
 if x <= c
 SX(i,j) = 0;
 end
 if x > c && x <= a
 SX(i,j) = 1;
 end
 if x > a
 SX(i,j) = 1 + R;
 end
 x = x + dx;
 end
end

for i = 1:n
 a = sum(L)-L(length(L));
 b = L(length(L));
 c = (i-1)*dx;
 R = (sum(L)-c)/b;
 x = 0;
 for j = 1:n
 if x <= a
 if x <= c
 SP(i,j) = 0;
 end
 if x > c
 SP(i,j) = -1;
 end
 end
 if x > a
 if x <= c
 SP(i,j) = R;
 end
 if x>c
 SP(i,j) = R-1;
 end
 end
 x = x + dx;
 end
end

for i = 1:nh
for j = 1:nh
x = MX(i,:).*MX(j,:);
k = length(x);
D(i,j) = 0.5*dx*(sum(x(1:(k-1))) +
  sum(x(2:k)));
end
end

for j = 1:n
for i = 1:nh
x = MX(i,:).*MP(j,:);
k = length(x);
D_P(i,j) = -0.5*dx*(sum(x(1:(k-1))) +
  sum(x(2:k)));
end
end
```

Table 2 [cont.]

<table>
<tr><th>Matlab code for calculations of bending moments*</th><th>Matlab code for calculations of shear forces*</th></tr>
<tr><td>

```
% calculations of envelope from vehicle
uniformly distributed load

Max = zeros(n,n);
Min = zeros(n,n);

for j = 1:n
for i = 1:n
if Im(i,j)>= 0
Max(i,j) = Im(i,j)*q_max*dx;
Min(i,j) = 0;
end
if Im(i,j)<0
Max(i,j) = 0;
Min(i,j) = Im(i,j)*q_max*dx;
end
end
end

for i = 1:n
EnvQ(i,1) = sum(Min(:,i));
EnvQ(i,2) = sum(Max(:,i));
end

% calculations of envelope from vehicle
passage

temp = [zeros(36,n);Im] +
   [zeros(24,n);Im;zeros(12,n)] +
   [zeros(12,n);Im;zeros(24,n)] +
   [Im;zeros(36,n)];

for i = 1:n
EnvV(i,1) = min(temp(:,i))*V_max;
EnvV(i,2) = max(temp(:,i))*V_max;
end

% calculations of total envelope

Env = EnvG + EnvQ + EnvV;

% preparation of figures and saving of
envelopes

figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,EnvG(:,1),'b',x,EnvG(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'ydir','reverse','FontSize',14)
xlabel('x [m]')
ylabel('M [kNm]')
title('Env M(G)')
grid
saveas(gcf,'Env_M(G).fig')
saveas(gcf,'Env_M(G).bmp')

figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,EnvQ(:,1),'b',x,EnvQ(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'ydir','reverse','FontSize',14)
xlabel('x [m]')
ylabel('M [kNm]')
title('Env M(Q)')
grid
saveas(gcf,'Env_M(Q).fig')
saveas(gcf,'Env_M(Q).bmp')
```

</td><td>

```
for j = 1:n
X(:,j) = D\D_P(:,j);
end
Im = X'*SX + SP;

% definition of loads data

g_max = 81.08;
g_min = 55.99;
G_max = zeros(n,1);

% index specifies location of the force

G_max(81) = 30.81;
G_max(256) = 30.81;
G_max(431) = 30.81;
G_min = zeros(n,1);
G_min(81) = 22.54;
G_min(256) = 22.54;
G_min(431) = 22.54;
q_max = 16.37;
V_max = 178.19;

% calculations of envelope from dead and
equipment load

Max = zeros(n,n);
Min = zeros(n,n);

for j = 1:n
for i = 1:n
if Im(i,j)> = 0
Max(i,j) = Im(i,j)*(G_max(i) + g_max*dx);
Min(i,j) = Im(i,j)*(G_min(i) + g_min*dx);
end
if Im(i,j)< 0
Max(i,j) = Im(i,j)*(G_min(i) + g_min*dx);
Min(i,j) = Im(i,j)*(G_max(i) + g_max*dx);
end
end
end

for i = 1:n
EnvG(i,1) = sum(Min(:,i));
EnvG(i,2) = sum(Max(:,i));
end

% calculations of envelope from vehicle
uniformly distributed load

Max = zeros(n,n);
Min = zeros(n,n);

for j = 1:n
for i = 1:n
if Im(i,j)> = 0
Max(i,j) = Im(i,j)*q_max*dx;
Min(i,j) = 0;
end
if Im(i,j)< 0
Max(i,j) = 0;
Min(i,j) = Im(i,j)*q_max*dx;
end
end
end

for i = 1:n
EnvQ(i,1) = sum(Min(:,i));
EnvQ(i,2) = sum(Max(:,i));
end
```

</td></tr>
</table>

Table 2 [cont.]

Matlab code for calculations of bending moments*

```
figure('WindowState','maximized')
x = 0:dx:sum(l);
plot(x,EnvV(:,1),'b',x,EnvV(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'ydir','reverse','FontSize',14)
xlabel('x [m]')
ylabel('M [kNm]')
title('Env M(V)')
grid
saveas(gcf,'Env_M(V).fig')
saveas(gcf,'Env_M(V).bmp')

figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,Env(:,1),'b',x,Env(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'ydir','reverse','FontSize',14)
xlabel('x [m]')
ylabel('M [kNm]')
title('Env M(G,Q,V)')
grid
saveas(gcf,'Env_M.fig')
saveas(gcf,'Env_M.bmp')
```

MATLAB
The Language of Technical Computing

* **All rights reserved.** The data necessary to be defined to perform the calculations are marked in blue colour.

Matlab code for calculations of shear forces*

```
% calculations of envelope from vehicle
passage

temp = [zeros(36,n);Im]+[zeros(24,n);Im;
zeros(12,n)]+[zeros(12,n);Im;zeros(24,n)]+
[Im;zeros(36,n)];

for i = 1:n
 EnvV(i,1) = min(temp(:,i))*V_max;
 EnvV(i,2) = max(temp(:,i))*V_max;
end

% calculations of total envelope

Env = EnvG + EnvQ + EnvV;

% preparation of figures and saving of
envelopes

figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,EnvG(:,1),'b',x,EnvG(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'FontSize',14)
xlabel('x [m]')
ylabel('S [kN]')
title('Env S(G)')
grid
saveas(gcf,'Env_S(G).fig')
saveas(gcf,'Env_S(G).bmp')
figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,EnvQ(:,1),'b',x,EnvQ(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'FontSize',14)
xlabel('x [m]')
ylabel('S [kN]')
title('Env S(Q)')
grid
saveas(gcf,'Env_S(Q).fig')
saveas(gcf,'Env_S(Q).bmp')
figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,EnvV(:,1),'b',x,EnvV(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'FontSize',14)
xlabel('x [m]')
ylabel('S [kN]')
title('Env S(V)')
grid
saveas(gcf,'Env_S(V).fig')
saveas(gcf,'Env_S(V).bmp')
figure('WindowState','maximized')
x = 0:dx:sum(L);
plot(x,Env(:,1),'b',x,Env(:,2),'r',
'LineWidth',1.5)
xlim([0 sum(L)])
set(gca,'FontSize',14)
xlabel('x [m]')
ylabel('S [kN]')
title('Env S(G,Q,V)')
grid
saveas(gcf,'Env_S.fig')
saveas(gcf,'Env_S.bmp')
```

The envelope of bending moments caused by permanent loads (dead and equipment load) was presented in Fig. 7. The envelopes of the same internal forces caused by temporary loads were shown in Fig. 8 and Fig. 8. Figure 8 concerns the results for vehicle uniformly distributed load, whereas Figure 9 refers to the effects of static combinations for vehicle passage. The total envelope of bending moments, generated as the sum of the previously discussed envelopes, was presented in Fig. 10.

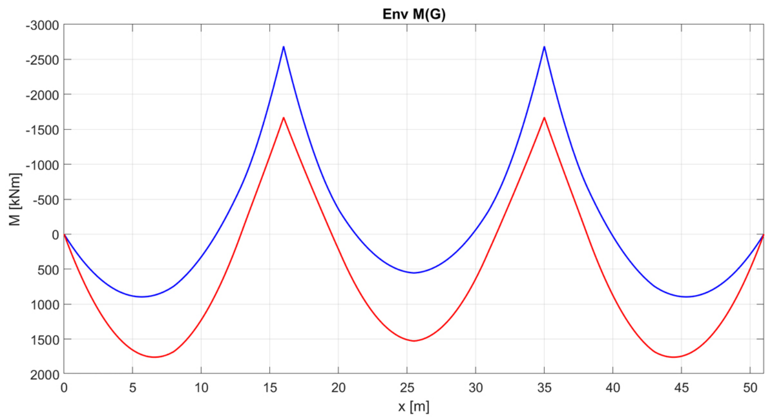

Fig. 7. Envelope (Env) of bending moments (M) due to permanent loads (G)

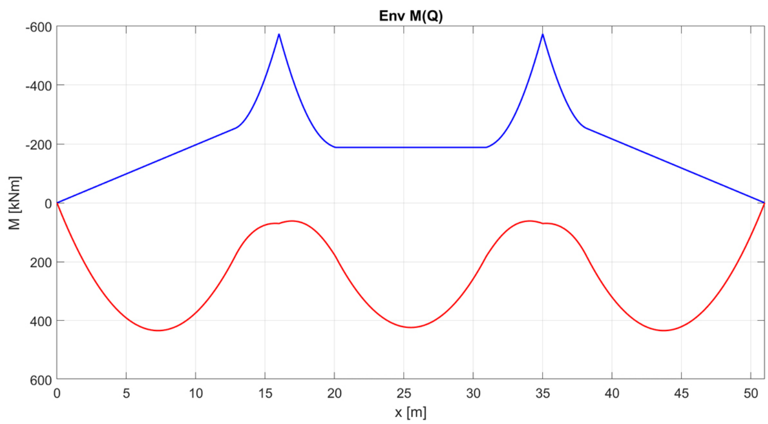

Fig. 8. Envelope (Env) of bending moments (M) due to vehicle uniformly distributed load (Q)

The process of determining the envelope of shear forces is analogous and the final result of calculations was presented in Figure 11.

The results of static calculations, efficiently supported by Matlab, were compared with extreme values obtained in FEM programs, SOFiSTiK and Autodesk Robot Structural Analysis Professional. The comparison in relation to the bending moments and shear forces is presented in Table 3 and Table 4. As it is visible, the results are very similar. Some

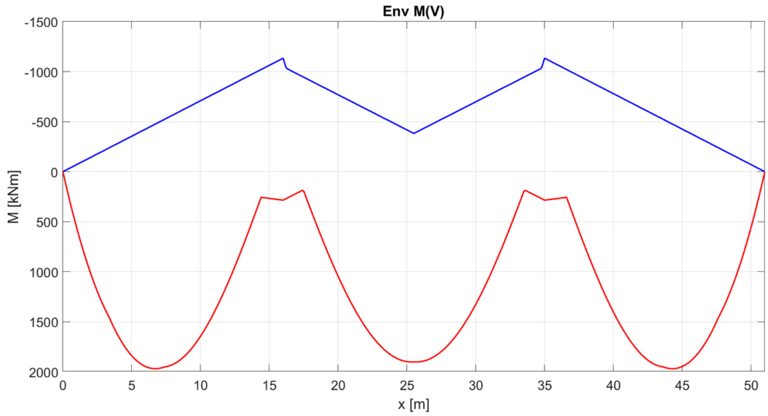

Fig. 9. Envelope (Env) of bending moments (M) due to vehicle passage (V)

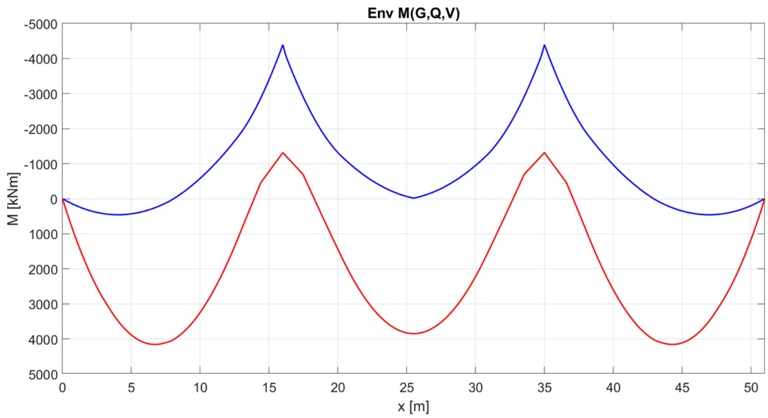

Fig. 10. Summary envelope (Env) of bending moments (M)

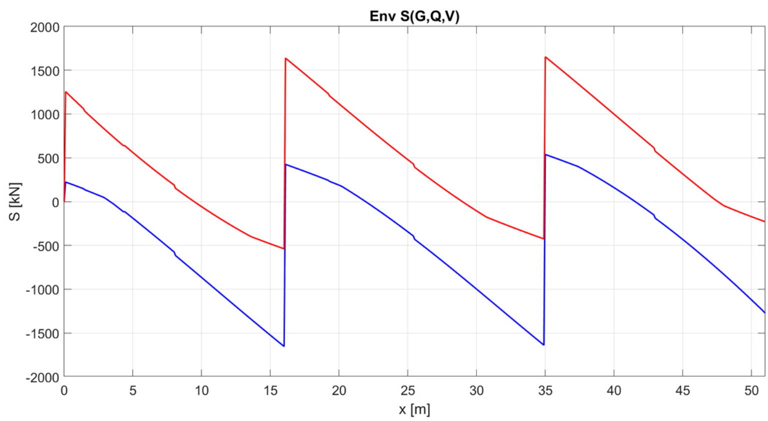

Fig. 11. Summary envelope (Env) of shear forces (S)

differences in bending moments, calculated in commercial programs, were also noticed. This inaccuracy can be caused by different theory implemented for calculations in both programs – Timoshenko and Bernoulli beam theory. The computational beam model of the structure prepared in SOFiSTiK is presented in Fig. 12.

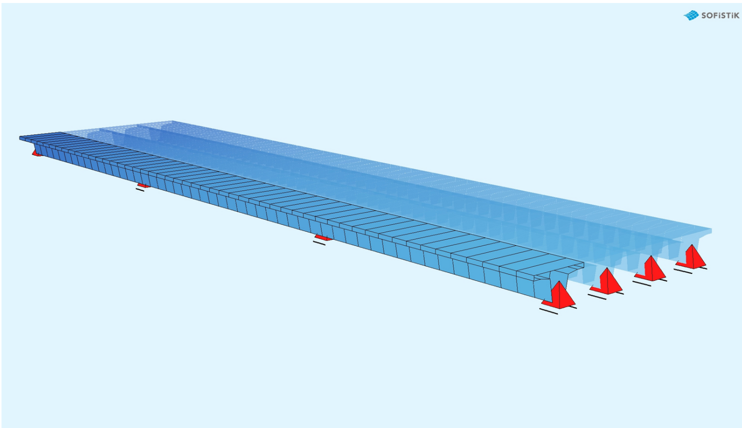

Fig. 12. Computational beam model of the structure

Table 3. Comparison of calculations' results in relation to the bending moments

Software	M(G) [kNm]	M(Q) [kNm]	M(V) [kNm]	M(G,Q,V) [kNm]	Relative difference
MATLAB	–2686.17	–573.89	–1133.86	–4393.91	–
	1759.25	433.90	1968.49	4158.37	–
ROBOT	–2686.00	–573.91	–1133.67	–4393.58	–0.01%
	1757.13	433.77	1967.82	4157.30	–0.03%
SOFiSTiK	–2672.19	–569.46	–1127.55	–4369.20	–0.56%
	1762.40	434.23	1973.44	4165.04	+0.16%

Table 4. Comparison of calculations' results in relation to the shear forces

Software	S(G) [kN]	S(Q) [kN]	S(V) [kN]	S(G,Q,V) [kN]	Relative difference
MATLAB	809.13	171.99	657.42	1638.55	–
	–827.88	–166.01	–659.47	–1653.36	–
ROBOT	813.00	172.81	657.43	1643.24	+0.29%
	–831.92	–166.83	–659.47	–1658.22	+0.29%
SOFiSTiK	812.52	172.51	657.14	1642.17	+0.22%
	–831.06	–166.55	–659.08	–1656.69	+0.20%

5. Summary

The discussed examples show how the Matlab package can effectively support the work of a designer. Compared to the modelling of structures in FEM programs the proposed approach has more steps but they are not time-consuming and this allows for minor modifications even at the numerical level. Despite this, it does not compete with advanced structural analysis programs, on the contrary – it is a good tool for preliminary calculations and checking the results. Moreover, it familiarizes the designer with working on code which is similar to working in some FEM softwares (e.g. SOFiSTiK). Another benefit of working in Matlab is improving mathematical intuition by solving complex systems of equations using algebraic structures and static through contact with partial results of calculations for different types of load. This approach also has its drawbacks which is a big limitation when it comes to construction types as creating computational models of higher classes than one-dimensional. In this case it is an issue beyond the reach or simply beyond the common sense of most designers.

However, a smart solution might be a cooperation between FEM software and mathematical packages (e.g. Autodesk Robot Structural Analysis Professional & Matlab or SOFiSTiK & Mathematica). This approach was used by Hawryszków in case of static calculations of two landmark footbridges in Poland (Fig. 13) [11–14]. The footbridge in the Pieniny mountains was built in 2006 and was designed as a large-span, cable-stayed structure made of glued-laminated wood [15], whereas the footbridge in Jadwisin was designed in 2005 and constructed in 2008 as an arch, steel structure with decks made of reinforced concrete. Coupling of Robot Millennium software with Matlab programming and numeric computing platform was an uncommon and innovative design idea in those times.

Fig. 13. Examples of footbridges designed with the use of Robot and Matlab cooperation [11–14]

Nowadays, FEM (Finite Element Method) and CAS (Computer Algebra System) programs offer designers much more possibilities in computer-aided calculations. Moreover, mathematical packages are very useful tools for expert or scientific calculations and can also perfectly support optimization processes.

Acknowledgments

Preparation of this paper was supported from a grant of the Polish National Agency for Academic Exchange (grant number: PPN/PPO/2019/1/00036 awarded to Mateusz Bocian) and funds of the Department of Roads, Bridges, Railways and Airports at the Wrocław University of Science and Technology. The authors would like to thank for this support.

References

[1] P.I. Kattan, *Matlab for Beginners: A Gentle Approach*. Petra Books, 2008.

[2] W. Wunderlich, W.D. Pilkey, *Mechanics of Structures. Variational and Computational Methods*. CRC Press, 2002.

[3] A. Guerra, P.D. Kiousis, "Design optimization of reinforced concrete structures", *Computers and Concrete*, 2006, vol. 3, no. 5, pp. 313-334.

[4] R.F. Kale, N.G. Gore, P.J. Salunke, "Applications of Matlab in optimization of bridge superstructures", *International Journal of Research in Engineering and Technology*, 2014, vol. 3, no. 5, pp. 34-39.

[5] A. Martins, L. Simões, J. Negrão, "Optimum design of concrete cable-stayed bridges", *Engineering Optimization*, 2016, vol. 48, no. 5, pp. 772-791, DOI: 10.1080/0305215X.2015.1057057.

[6] A. Martins, L. Simões, J. Negrão, "Optimization of cable forces on concrete cable-stayed bridges including geometrical nonlinearities", *Computers and Structures*, 2015, vol. 155, pp. 18-27, DOI: 10.1016/j.compstruc.2015.02.032.

[7] B. Czaplewski, "Projekt mostu drogowego MA-46 w ciągu autostrady A4". Opiekun: dr inż. Paweł Hawryszków, Politechnika Wrocławska, 2015 (in Polish).

[8] E.C. Hambly, *Bridge Deck Behaviour*. CRC Press, 1991.

[9] J. Hołowaty, "Numerical Approach for the Live Load Distribution in Road Bridges", *Computer Technology and Application*, 2015, vol. 6, pp. 101-106, DOI: 10.17265/1934-7332/2015.02.007.

[10] R. Bareš, Ch.E. Massonnet, *Analysis of Beam Grids and Orthotropic Plates by the Guyon–Massonnet-Bareš Method.* Lockwood; SNTL, 1968.

[11] J. Biliszczuk, P. Hawryszków, M. Sułkowski, "Kładka *Wężowisko* w Jadwisinie koło Zegrza", *Inżynieria i Budownictwo*, 2009, no. 1/2, pp. 46-48 (in Polish).

[12] J. Biliszczuk, P. Hawryszków, M. Sułkowski, "The design of *Snake Footbridge* in Jadwisin", in *Concrete structures in Poland 2000-2005*. Polish Cement Association, 2006, pp. 12-13.

[13] J. Biliszczuk, P. Hawryszków, M. Węgrzyniak, A. Maury, M. Sułkowski, "Podwieszona kładka dla pieszych z drewna klejonego w Sromowcach Niżnych", *Inżynieria i Budownictwo*, 2008, no. 1/2, pp. 5-8 (in Polish).

[14] J. Biliszczuk, P. Hawryszków, "Foot and cycling bridge over the Dunajec River in Sromowce Niżne", in *Engineering structures (Inženýrské stavby V4)*. ČKAIT, 2012, pp. 136-143.

[15] H. Zobel, T. Alkhafaji, *Mosty drewniane*. Warszawa: WKŁ, 2006 (in Polish).

Zastosowanie programu Matlab w obliczeniach statycznych konstrukcji mostowych

Słowa kluczowe: obliczenia statyczne, mosty, obwiednie sił wewnętrznych, pakiety matematyczne, Matlab

Streszczenie:

Pakiet Matlab jest środowiskiem programistycznym służącym do obliczeń w zakresie algebry liniowej, pomocnym zarówno dla naukowców, jak i dla inżynierów. Znajduje on zastosowanie zarówno

przy obliczeniach prostych, jak i bardzo złożonych. Jego główną zaletą z perspektywy inżyniera budowlanego jest prostota języka, niewymagająca dużych umiejętności programistycznych i możliwość szerokiego zastosowania w obliczeniach z zakresu statyki liniowej. Platforma matematyczna została użyta do oprogramowania obliczeń statycznych wieloprzęsłowych, ciągłych, belkowych konstrukcji mostowych. W sformułowanym podejściu teoretycznym wykorzystano metodę siłdo wyznaczania wykresów siłwewnętrznych w stanach jednostkowych, które to uporządkowane tworzą macierz wpływu. Mając macierz wpływu, wprowadzano wektor obciążeń, a następnie wyznaczano obwiednię sił przekrojowych przy użyciu specjalnej funkcji. Znając wyniki od poszczególnych obciążeń, możliwe jest wyznaczenie globalnej obwiedni sił wewnętrznych i przystąpienie do ewentualnych modyfikacji modelu. Podejście teoretyczne zostało przetestowane obliczeniowo na przykładzie alternatywnej koncepcji projektowej mostu MA-46 w ciągu autostrady A4.

Jedną z większych zalet omówionego podejścia analitycznego jest szeroki dostęp do wyników obliczeń pośrednich oraz praca z kodem zbliżonym do niektórych programów obliczeniowych (np. SOFiSTiK). Omówione podejście obliczeniowe jest dobrym sposobem do wstępnego projektowania ze względu na niewielki czas potrzebny do porównania kilku wariantów rozwiązania, a co za tym idzie, może być pomocne w optymalizacji konstrukcji.

Received: 19.05.2021, Revised: 2.08.2021