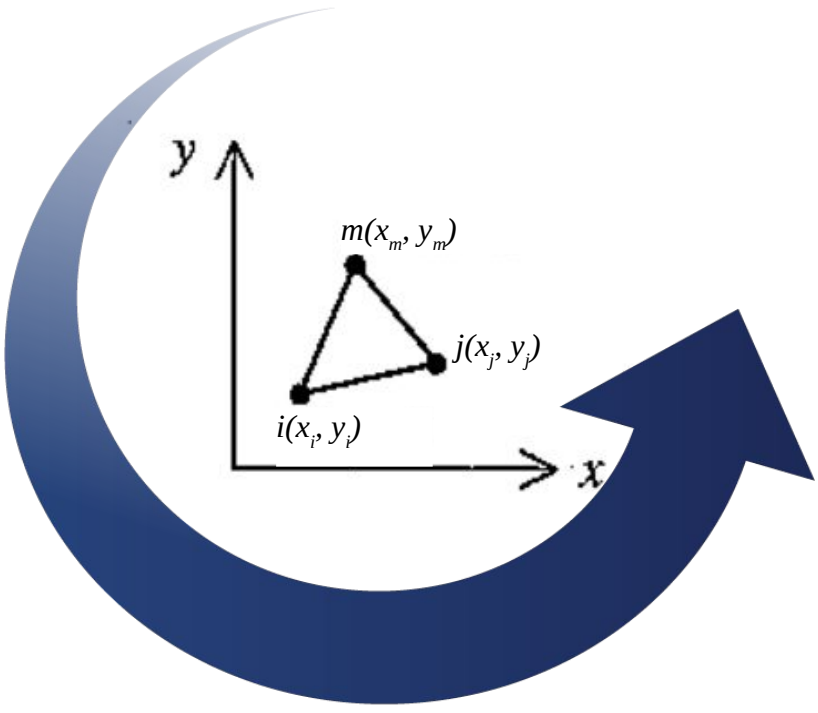
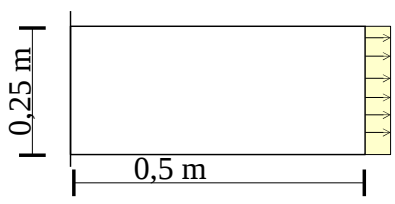
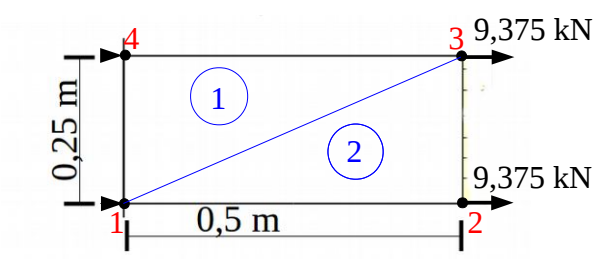
Trójkątny element liniowy





E = 210 GPa
ν = 0,3 (NU)
t = 0,025 m
w = 3000 kN / m^3

1. Dyskretyzacja



Elementy	i	j	m
1	1	3	4
2	1	2	3

2. Utworzenie lokalnych macierzy sztywności (k_1, k_2) dla każdego elementu

wymiary lokalnych macierzy sztywności?

3. Zebranie informacji z lokalnych macierzy sztywności do globalnej macierzy sztywności K

wymiary globalnej macierzy sztywności?

K*U=F

Zasadniczy układ równań liniowych (zawsze niereprezentatywny):

K – globalna macierz sztywności
U – wektor przemieszczeń węzłów układu
F – wektor sił (reakcje i obciążenia zewnętrzne) w węzłach układu

co to oznacza, że układ równań jest niereprezentatywny?

4. Uwzględnienie warunków brzegowych (informacje z warunków zadania dotyczące przemieszczeń węzłów układu i sił w węzłach układu)

po co uwzględniamy warunki brzegowe?

k*u=f

mały układ równań liniowych (reprezentatywny)

5. Rozwiązanie układu równań (reprezentatywnego)

6. Postprocessing

Wykorzystywane funkcje Matlaba (M-pliki)

1

LinearTriangleElementArea.m

```
function y = LinearTriangleElementArea(xi,yi,xj,yj,xm,ym)
%LinearTriangleElementArea   This function returns the area of the
%                            linear triangular element whose first
%                            node has coordinates (xi,yi), second
%                            node has coordinates (xj,yj), and
%                            third node has coordinates (xm,ym).
y = (xi*(yj-ym) + xj*(ym-yi) + xm*(yi-yj))/2;
```

==========

2

LinearTriangleElementStiffness.m

```
function y = LinearTriangleElementStiffness(E,NU,t,xi,yi,xj,yj,xm,ym,p)
%LinearTriangleElementStiffness   This function returns the element
%                                 stiffness matrix for a linear
%                                 triangular element with modulus of
%                                 elasticity E, Poisson's ratio NU,
%                                 thickness t, coordinates of the
%                                 first node (xi,yi), coordinates of
%                                 the second node (xj,yj), and
%                                 coordinates of the third node
%                                 (xm,ym). Use p = 1 for cases of
%                                 plane stress, and p = 2 for cases
%                                 of plane strain.
%                                 The size of the element stiffness
%                                 matrix is 6 x 6.
A = (xi*(yj-ym) + xj*(ym-yi) + xm*(yi-yj))/2;
betai = yj-ym;
betaj = ym-yi;
betam = yi-yj;
gammai = xm-xj;
gammaj = xi-xm;
gammam = xj-xi;
B = [betai 0 betaj 0 betam 0 ;
   0 gammai 0 gammaj 0 gammam ;
   gammai betai gammaj betaj gammam betam]/(2*A);
if p == 1
   D = (E/(1-NU*NU))*[1 NU 0 ; NU 1 0 ; 0 0 (1-NU)/2];
elseif p == 2
   D = (E/(1+NU)/(1-2*NU))*[1-NU NU 0 ; NU 1-NU 0 ; 0 0 (1-2*NU)/2];
end
y = t*A*B'*D*B;
```

==========

3

LinearTriangleAssemble.m

```
function y = LinearTriangleAssemble(K,k,i,j,m)
%LinearTriangleAssemble   This function assembles the element
%                         stiffness matrix k of the linear
%                         triangular element with nodes i, j,
%                         and m into the global stiffness matrix K.
%                         This function returns the global stiffness
%                         matrix K after the element stiffness matrix
%                         k is assembled.
K(2*i-1,2*i-1) = K(2*i-1,2*i-1) + k(1,1);
K(2*i-1,2*i) = K(2*i-1,2*i) + k(1,2);
K(2*i-1,2*j-1) = K(2*i-1,2*j-1) + k(1,3);
K(2*i-1,2*j) = K(2*i-1,2*j) + k(1,4);
K(2*i-1,2*m-1) = K(2*i-1,2*m-1) + k(1,5);
K(2*i-1,2*m) = K(2*i-1,2*m) + k(1,6);
K(2*i,2*i-1) = K(2*i,2*i-1) + k(2,1);
K(2*i,2*i) = K(2*i,2*i) + k(2,2);
K(2*i,2*j-1) = K(2*i,2*j-1) + k(2,3);
K(2*i,2*j) = K(2*i,2*j) + k(2,4);
K(2*i,2*m-1) = K(2*i,2*m-1) + k(2,5);
K(2*i,2*m) = K(2*i,2*m) + k(2,6);
K(2*j-1,2*i-1) = K(2*j-1,2*i-1) + k(3,1);
K(2*j-1,2*i) = K(2*j-1,2*i) + k(3,2);
K(2*j-1,2*j-1) = K(2*j-1,2*j-1) + k(3,3);
K(2*j-1,2*j) = K(2*j-1,2*j) + k(3,4);
K(2*j-1,2*m-1) = K(2*j-1,2*m-1) + k(3,5);
K(2*j-1,2*m) = K(2*j-1,2*m) + k(3,6);
K(2*j,2*i-1) = K(2*j,2*i-1) + k(4,1);
K(2*j,2*i) = K(2*j,2*i) + k(4,2);
K(2*j,2*j-1) = K(2*j,2*j-1) + k(4,3);
K(2*j,2*j) = K(2*j,2*j) + k(4,4);
K(2*j,2*m-1) = K(2*j,2*m-1) + k(4,5);
K(2*j,2*m) = K(2*j,2*m) + k(4,6);
K(2*m-1,2*i-1) = K(2*m-1,2*i-1) + k(5,1);
K(2*m-1,2*i) = K(2*m-1,2*i) + k(5,2);
K(2*m-1,2*j-1) = K(2*m-1,2*j-1) + k(5,3);
K(2*m-1,2*j) = K(2*m-1,2*j) + k(5,4);
```

```
K(2*m-1,2*m-1) = K(2*m-1,2*m-1) + k(5,5);
K(2*m-1,2*m) = K(2*m-1,2*m) + k(5,6);
K(2*m,2*i-1) = K(2*m,2*i-1) + k(6,1);
K(2*m,2*i) = K(2*m,2*i) + k(6,2);
K(2*m,2*j-1) = K(2*m,2*j-1) + k(6,3);
K(2*m,2*j) = K(2*m,2*j) + k(6,4);
K(2*m,2*m-1) = K(2*m,2*m-1) + k(6,5);
K(2*m,2*m) = K(2*m,2*m) + k(6,6);
y = K;
```

==========

4

LinearTriangleElementStresses.m

```
function y = LinearTriangleElementStresses(E,NU,t,xi,yi,xj,yj,xm,ym,p,u)
%LinearTriangleElementStresses   This function returns the element
%                                stress vector for a linear
%                                triangular element with modulus of
%                                elasticity E, Poisson's ratio NU,
%                                thickness t, coordinates of the
%                                first node (xi,yi), coordinates of
%                                the second node (xj,yj),
%                                coordinates of the third node
%                                (xm,ym), and element displacement
%                                vector u. Use p = 1 for cases of
%                                plane stress, and p = 2 for cases
%                                of plane strain.
%                                The size of the element stress
%                                vector is 3 x 1.
A = (xi*(yj-ym) + xj*(ym-yi) + xm*(yi-yj))/2;
betai = yj-ym;
betaj = ym-yi;
betam = yi-yj;
gammai = xm-xj;
gammaj = xi-xm;
gammam = xj-xi;
B = [betai 0 betaj 0 betam 0 ;
   0 gammai 0 gammaj 0 gammam ;
   gammai betai gammaj betaj gammam betam]/(2*A);
if p == 1
   D = (E/(1-NU*NU))*[1 NU 0 ; NU 1 0 ; 0 0 (1-NU)/2];
elseif p == 2
   D = (E/(1+NU)/(1-2*NU))*[1-NU NU 0 ; NU 1-NU 0 ; 0 0 (1-2*NU)/2];
end
y = D*B*u;
```

==========

5

LinearTriangleElementPStresses.m

```
function y = LinearTriangleElementPStresses(sigma)
%LinearTriangleElementPStresses   This function returns the element
%                                 principal stresses and their
%                                 angle given the element
%                                 stress vector.
R = (sigma(1) + sigma(2))/2;
Q = ((sigma(1) - sigma(2))/2)^2 + sigma(3)*sigma(3);
M = 2*sigma(3)/(sigma(1) - sigma(2));
s1 = R + sqrt(Q);
s2 = R - sqrt(Q);
theta = (atan(M)/2)*180/pi;
y = [s1 ; s2 ; theta];
```

==========