Санкт-Петербургский политехнический университет

Высшая школа теоретической механики, ФизМех

Направление подготовки

«01.03.03 Механика и математическое моделирование»

Отчет по индивидуальной работе №**05**

тема " Метод конечных элементов "

дисциплина "Вычислительная механика"

Выполнил студент гр. 90301 **М. А.Бенюх**

Преподаватель: Е.Ю. Витохин

Санкт-Петербург

2021

Содержание:

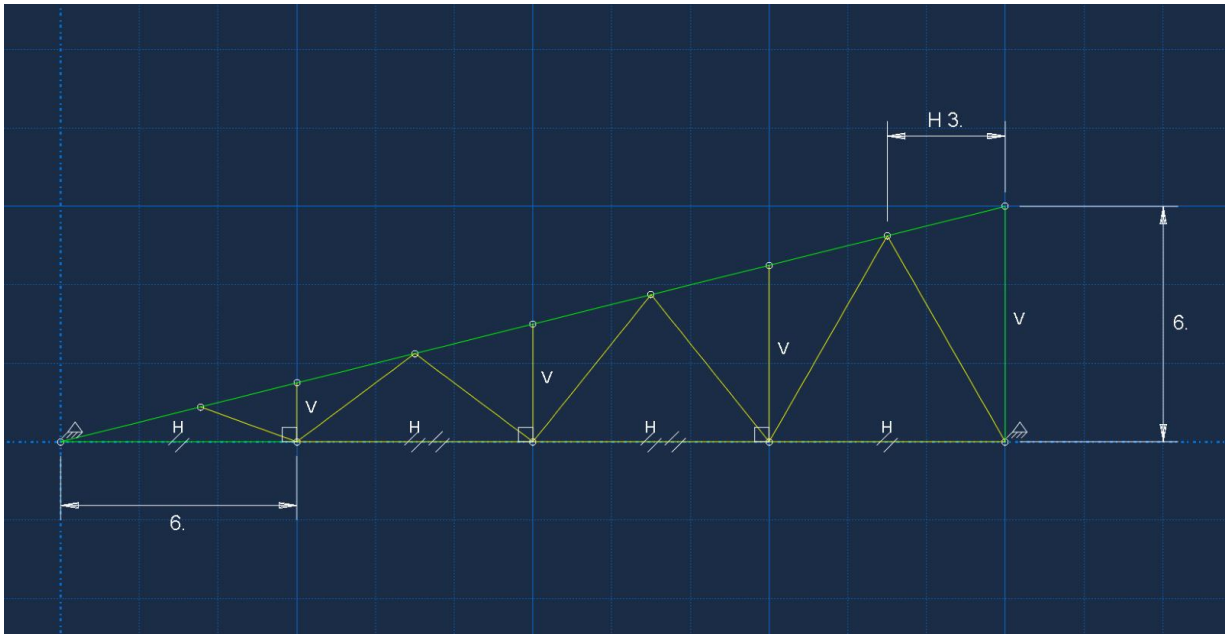
1. Формулировка задачи.

Произвести расчет плоских фермы под действием нагрузки **F**. В вариантах 1 – 4, 9, 11 – 13, 15 – 17, 19, 20 нагрузку следует прикладывать на верхний пояс. В остальных случаях нагрузка прикладывается на нижний пояс. Во всех случаях, кроме 12 и закрепляется крайне левый и правый нижний угол. Закрепление производится по горизонтальным и вертикальным степеням свободы. Требуется определить перемещения узлов фермы и усилия в стержнях.

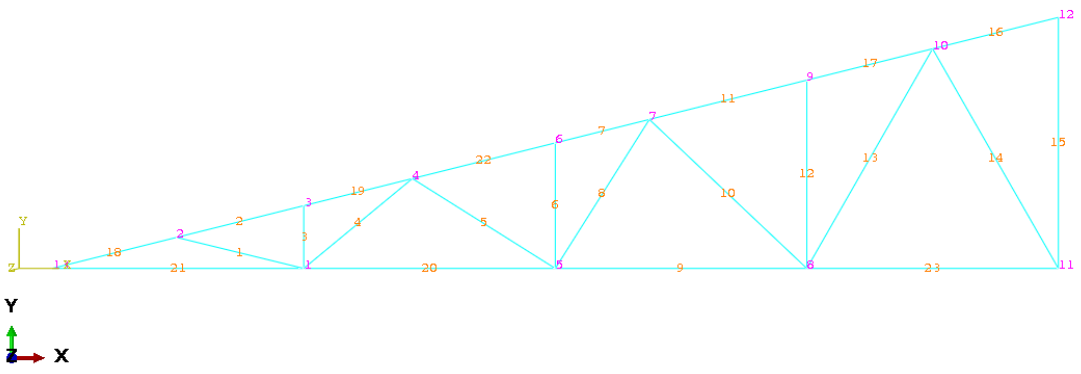
Таблица 1. Параметры задачи

Параметр	Значение
Площадь сечения **S** ($м^2$)	$1\ 10^{-4}$
Модуль Юнга **E** (Па)	$2\ 10^{11}$
Сила **F** (Н)	$1\ 10^{3}$

Вариант 3



Нумерация узлов и элементов



Получить:

1) поле U_x , U_y
2) Таблицу сравнения U_x и U_y
3) Поле усилий F_x
4) Таблицу сравнений F_x

2. Алгоритм метода.

Для каждого элемента получим локальную матрицу жесткости:

$$[k_{loc}^e] = \frac{EA}{l_e} * \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

, где $E = 2e11 -$ Модуль Юнга, $A = 0.0001 -$ Площадь сечения, $l_e -$ длина элемента

$$l_e^i = \sqrt{\left(x_2^i - x_1^i\right)^2 + \left(y_2^i - y_1^i\right)^2} - \text{для каждого } i \text{ элемента}$$

Для каждого элемента соберем матрицу перехода:

$$l_{1,2} = \frac{x_2 - x_1}{l_e}, \qquad m_{1,2} = \frac{y_2 - y_1}{l_e}$$

$$[T] = \begin{bmatrix} l_{1,2} & m_{1,2} & 0 & 0 \\ 0 & 0 & l_{1,2} & m_{1,2} \end{bmatrix}$$

Перееdем к матрице жесткости для каждого элемента в глобальной системе координат при помощи матрицы перехода:

$$\left[k_{gl}^e\right] = T' * [k_{loc}^e] * T$$

Соберем $K -$ матрицу жесткости системы:

$$num - \text{номера узлов для элемента.}$$

$$i = 1{:}\,2, j = 1{:}\,2$$

$$\begin{cases} K\left[2\ num[i], 2\ num[j]\right] += k_{loc}[2i, 2j] \\ K[2\ num[i] + 1, 2\ num[j] + 1] += k_{loc}[2i + 1, 2j + 1] \\ K[2\ num[i], 2\ num[j] + 1] += k_{loc}[2i, 2j + 1] \\ K\left[2\ num[i], +1\ 2\ num[j]\right] += k_{loc}[2i + 1, 2j] \end{cases}$$

Также обозначим вектор сил для системы:

$$[F] = \begin{bmatrix} \\ f_x^1 \\ f_y^1 \\ f_x^2 \\ f_y^2 \\ \dots \end{bmatrix}$$

Применим граничные условия:

Например, 1-й элемент закреплен:

$$K = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & k & k \\ 0 & 0 & & k & k \end{pmatrix}, \qquad F = \begin{bmatrix} 0 \\ 0 \\ f_x^2 \\ f_y^2 \\ \cdots \end{bmatrix}$$

Тогда решением основанного уравнения МКЭ: $K * U = F$ будет вектор перемещений U

3. Результаты.

Результаты данной задачи представлены в сравнительной форме Abacus и алгоритма на MATLAB.

	Abaqus		MATLAB	
№	Ux, мм	Uy, мм	Ux, мм	Uy, мм
1	1,7814304	-29,7467540	1,7814305	-29,7467541
2	3,6062300	-23,6534510	3,6062300	-23,6534516
3	3,1257952	-29,8217540	3,1257951	-29,8217541
4	0,9843379	-28,1675870	0,9843378	-28,1675860
5	2,4153716	-24,8590430	2,4153715	-24,8590427
6	-1,3084380	-25,0090440	-1,3084380	-25,0090427
7	-3,4383258	-20,4467360	-3,4383258	-20,4467361
8	1,8870375	-13,4983710	1,8870376	-13,4983701
9	-5,9282803	-13,7233700	-5,9282804	-13,7233701
10	-8,4519656	-6,2133684	-8,4519653	-6,2133686
11	0,0000000	0,0000000	0,0000000	0,0000000
12	-9,9303070	-0,3000000	-9,9303074	-3,0000000
13	0,0000000	0,0000000	0,0000000	0,0000000

Таб. 1. Перемещения U_x и U_y в узлах

Разность Ux, мм
-0,0000001
0,0000000
0,0000001
0,0000000
0,0000001
0,0000000
0,0000000
-0,0000001
0,0000001
-0,0000003
0,0000000
0,0000004
0,0000000

Таб. 2. Таблица сравнения U_x

Разность F, кН
-0,0413966
-0,0352487
-0,0080329
-0,0026250
-0,0078174
-0,0154083
-0,0456970
-0,0301187
-0,0358546
-0,0282987
-0,0338815
-0,0301640
-0,0433519
-0,0297081
-0,0273933
-0,0388533
-0,0652428
-0,6071175
-0,0063289
-0,0066354
-0,2457226
-0,0106200
-0,0506284

Таб. 3. Таблица сравнения F

	Abaqus	MATLAB
№	F, кН	F, кН
1	-2,0815383	-2,0401417
2	-12,5525420	-12,5172933
3	-0,9999998	-0,9919668
4	2,3418918	2,3445168
5	-2,7448049	-2,7369875
6	-1,0000001	-0,9845918
7	-8,3002666	-8,2545696
8	2,9113325	2,9414512
9	-1,7611132	-1,7252586
10	-3,5371379	-3,5088392
11	-4,0544907	-4,0206092
12	-1,0000000	-0,9698360
13	3,9544614	3,9978133
14	-3,9736284	-3,9439203
15	-1,0000000	-0,9726067
16	-0,0000001	0,0388532
17	-4,0544907	-3,9892479
18	-14,6367130	-14,0295955
19	-12,5525420	-12,5462131
20	2,1131367	2,1197721
21	5,9381016	6,1838242
22	-8,3002666	-8,2896466
23	-6,2901250	-6,2394966

Таб. 4. Силы в узлах

АБАКУС:

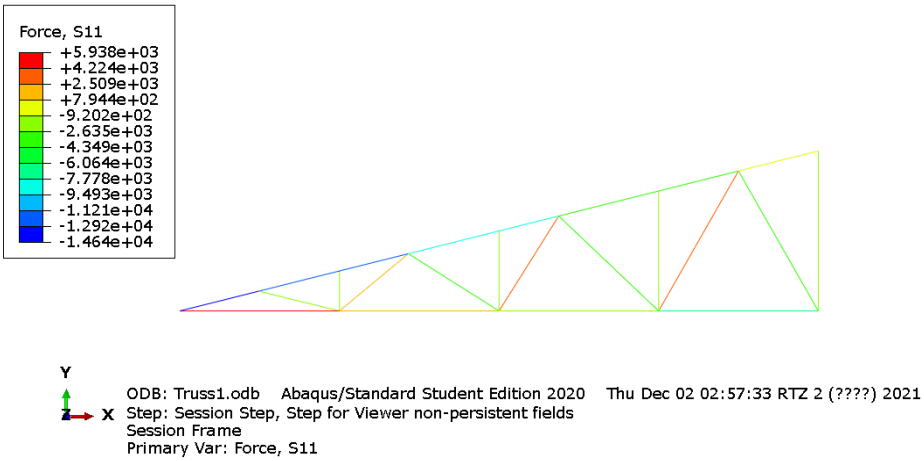


Рис. 2. Поле Сил [Н]

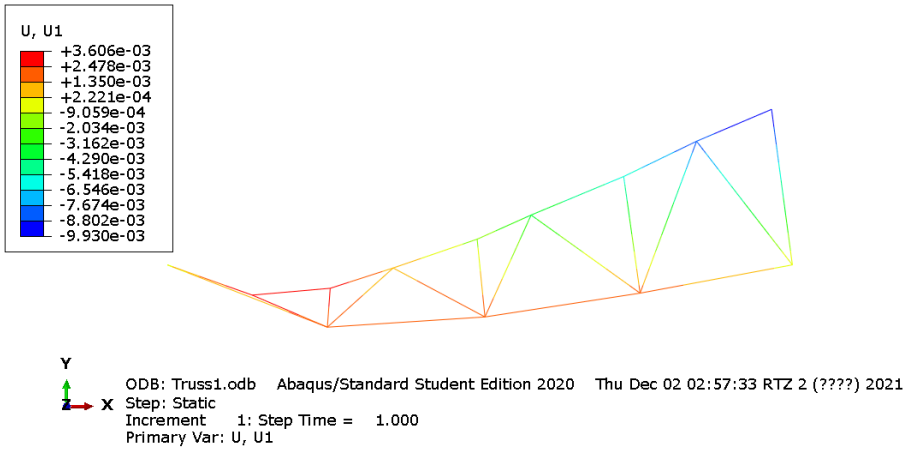


Рис. 2. Поле перемещений по X [м]

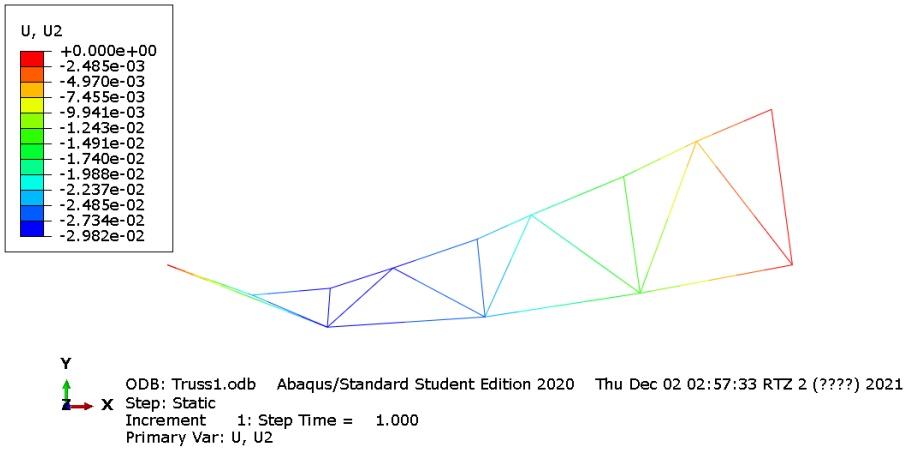


Рис. 3. Поле перемещений по Y [м]

MATLAB:

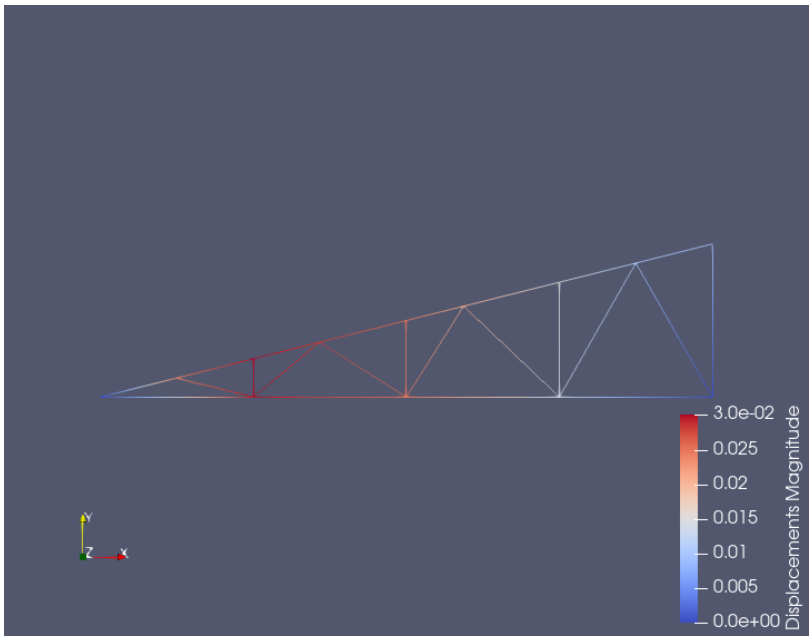


Рис. 5. Поле перемещения по X и Y [м]

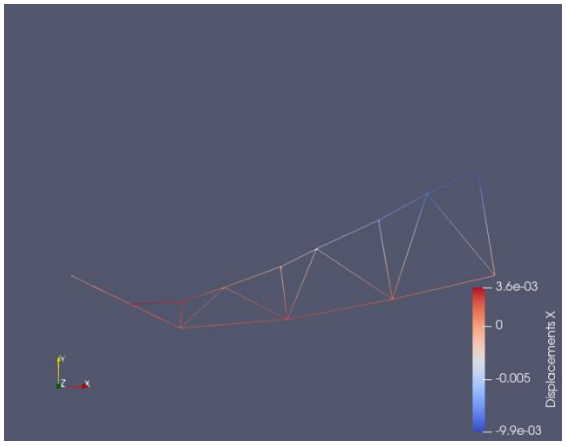


Рис. 6. Поле перемещения по X [м]

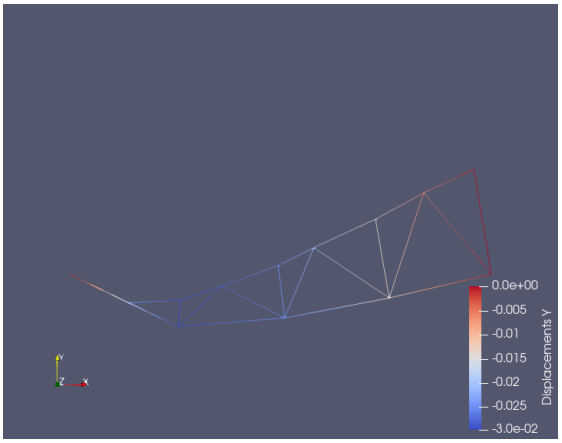


Рис. 7. Поле перемещения по Y [м]

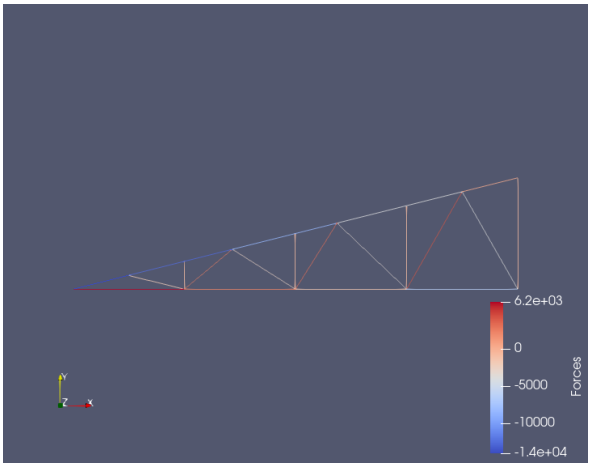


Рис. 8. Поле Сил [Н]

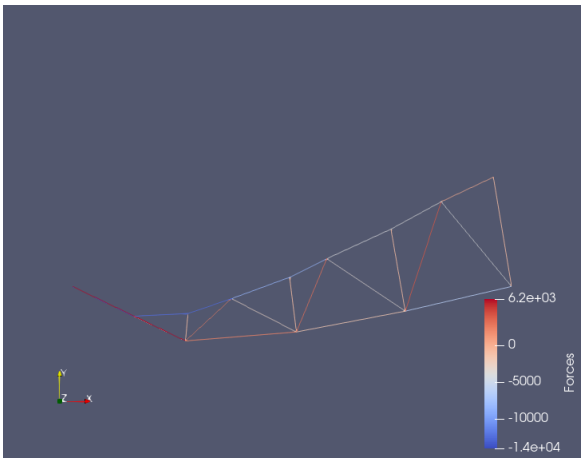


Рис. 9. Поле Сил [Н]

4. Заключение

В рамках данной задачи с помощью метода конечных элементов были получены усилия и деформации, возникающие при растяжении-сжатии упругого стержня.

В ходе работы проведен расчет плоских ферм под действием нагрузки F, которая прикладывалась на верхний пояс, а закреплены – левый и правый нижний угол.

Далее был проверен МКЭ в Abaqus и MATLAB. Были получены результаты с точностью до 10 знака, которые оказались равны.

Использование Abaqus позволяет строить ферму и получать решение и визуализацию сразу в программном пакете, тогда как для визуализации в Matlab приходилось использовать стороннее ПО и задавать координаты узлов вручную.

Также не маловажно, что алгоритм на MatLab гибок для решения не стандартных задач, в отличие от Abaqus(не возможно изменить исходный код) и стоимость разработки на MatLab значительно ниже, а в случае портирования кода например на Python – зависит только от стоимости оплаты труда сотрудника.

5. Код программы

```
Mass_node=[6.78988647,           0.;
3.75697184,  0.741771281;
6.78988647,          1.5;
9.38103199,   2.14778638;
12.7898865,           0.;
12.7898865,           3.;
15.0334682,   3.56089544;
18.7898865,           0.;
18.7898865,          4.5;
21.7898865,         5.25;
24.7898865,           0.;
24.7898865,           6.;
0.789886594,           0.;
 ]
Mass_Element=[
1, 2;
3, 2;
1, 3;
4, 1;
5, 4;
5, 6;
7, 6;
7, 5;
5, 8;
8, 7;
9, 7;
8, 9;
10,  8;
11, 10;
11, 12;
```

```
12, 10;
10,  9;
2, 13;
4, 3;
1, 5;
13,  1;
6, 4;
8, 11;
    ]
GU_el=[13,11] %закреп
N_no=length(Mass_node);
N_el=length(Mass_Element);
E=2e11; %юнг
A=0.0001; % площадь
l_e=getLenEls(Mass_Element,Mass_node) %длина каждого элемента
massK_loc=getK_loc(E,A,l_e) %*[1,-1;-1,1] не забвть
%сосредоточеная сила
% F=zeros(length(Mass_node),2);
% F(2,2)=1000 ; F(3,2)= 1000; F(5,2)=1000 ; F(9,2) = 1000;
F(11,2) = 1000;
% F1_gl = getF_gl(F,Mass_Element,Mass_node, l_e )
F=zeros(2*length(Mass_node),1);
% F(2*2)=1000 ;
% F(3*2)= -1000; F(5*2)=-1000 ; F(9*2) = -1000; %F(11*2) = 1000;
F(2*2)= -1000; F(3*2)=-1000 ; F(4*2)=-1000 ; F(6*2)=-1000 ;
F(7*2)=-1000 ; F(9*2)=-1000 ; F(10*2)=-1000 ; F(12*2)=-1000 ;
F1_gl=F

%
%матрица жесткости
K_gl=getK_gl(Mass_Element,massK_loc,Mass_node, l_e )
%вектор сил
% F1_gl= getF_gl(F,Mass_Element,Mass_node, l_e )
length(F1_gl)
length(K_gl(1,:))
%задние ГУ
[guF_gl,guK_gl]=setGuToFK(F1_gl,K_gl , GU_el)
det(guK_gl)
% guF_gl=guF_gl(1:length(guF_gl)-1)
% guK_gl=guK_gl(1:length(guK_gl)-1,:)
% guF_gl=[guF_gl(1:11),guF_gl(13:length(guK_gl)-1)]
% guK_gl=[guK_gl(1:11,:),guK_gl(13:length(guK_gl)-1,:)]
%перемещение
U=guK_gl\guF_gl
%деформации
new_len=zeros(length(Mass_Element),1);
deffs=zeros(length(Mass_Element),1);
for i=1:length(Mass_Element)
    nodes=Mass_Element(i,:);
    X1=Mass_node(nodes(1),1)+U(2*nodes(1)-1);
```

```
    X2=Mass_node(nodes(2),1)+U(2*nodes(2)-1);
    Y1=Mass_node(nodes(1),2)+U(2*nodes(1));
    Y2=Mass_node(nodes(2),2)+U(2*nodes(2));
    new_len(i)=sqrt((X2-X1)^2 +(Y2-Y1)^2);
    deffs(i)=(new_len(i)-l_e(i))/l_e(i);

end
stesses=deffs*E
Forces=stesses*A

function [F1_gl,K_gl]=setGuToFK(F1_gl,K_gl , GU_el)
    for i=1:length(F1_gl)
        for j=1:length(GU_el)
            if(i == 2*(GU_el(j)-1)+1)
                F1_gl(i)  = 0;

                F1_gl(i+1)  = 0;

                K_gl(i,:) = 0;
                K_gl(:,i) = 0;


                K_gl(i+1,:) = 0;
                K_gl(:,i+1) = 0;

                K_gl(i,i) = 1;
                K_gl(i+1,i+1) = 1;
            end
        end
    end
end
function mass_k_loc=getK_loc(E,A,l_e)
    mass_k_loc=zeros(length(l_e),1);
    for i=1: length(l_e)
        mass_k_loc(i)=E*A/l_e(i);
    end
end
function mass_l=getLenEls(Mass_Element,Mass_node)
    mass_l=zeros(length(Mass_Element),1);
    for i=1:length(Mass_Element)
        nodes=Mass_Element(i,:)
        mass_l(i)=sqrt((Mass_node(nodes(2),1)-
Mass_node(nodes(1),1))^2 +(Mass_node(nodes(2),2)-
Mass_node(nodes(1),2))^2);
    end
    mass_l
end
function T=getT(x_loc,y_loc,l)
    l_loc=(x_loc(2)-x_loc(1))/l;
    m_loc=(y_loc(2)-y_loc(1))/l;
```

```
    T=[l_loc, m_loc, 0    0;
        0,        0, l_loc, m_loc]
end
function K_gl = getK_gl(Mass_Element,massK_loc,Mass_node, l_e )
K_gl=zeros(2*length(Mass_node));
    for i =1:length(Mass_Element)
        nodes=Mass_Element(i,:)

T=getT([Mass_node(nodes(1),1),Mass_node(nodes(2),1)],[Mass_node(
nodes(1),2),Mass_node(nodes(2),2)], l_e(i));
        k_loc =[1,-1;-1,1].*massK_loc(i)
        T
        k_gl=T'*k_loc*T
        for k=1:2
            for j=1:2
                K_gl(2*(nodes(k)-1)+1+0,2*(nodes(j)-1)+1+0)
=K_gl(2*(nodes(k)-1)+1+0,2*(nodes(j)-1)+1+0)
+k_gl(2*(k-1)+1+0,2*(j-1)+1+0);
                K_gl(2*(nodes(k)-1)+1+0,2*(nodes(j)-1)+1+1)
=K_gl(2*(nodes(k)-1)+1+0,2*(nodes(j)-1)+1+1)
+k_gl(2*(k-1)+1+0,2*(j-1)+1+1);
                K_gl(2*(nodes(k)-1)+1+1,2*(nodes(j)-1)+1+0)
=K_gl(2*(nodes(k)-1)+1+1,2*(nodes(j)-1)+1+0)
+k_gl(2*(k-1)+1+1,2*(j-1)+1+0);
                K_gl(2*(nodes(k)-1)+1+1,2*(nodes(j)-1)+1+1)
=K_gl(2*(nodes(k)-1)+1+1,2*(nodes(j)-1)+1+1)
+k_gl(2*(k-1)+1+1,2*(j-1)+1+1);
            end
        end
    end
    K_gl
end
```