

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение Высшего  
образования «Московский государственный технический университет имени  
Н.Э.Баумана (национальный исследовательский университет)» (МГТУ им Н.Э.  
Баумана)

Факультет «Робототехника и комплексная автоматизация»  
Кафедра «Системы автоматизированного проектирования» (РК6)

**Отчет**  
**По лабораторной работе №2**  
**по дисциплине «Прикладная механика»**  
**по теме «Расчет статически-неопределимой балки методом конечных**  
**элементов»**

Выполнил: студент группы РК6-34Б, Чабанов В.А.

Москва

2019

## Вариант 2

### Постановка задачи

Составить конечно-элементную программу для расчета статически-неопределимой балки и проверить корректность ее работы с использованием Siemens NX.

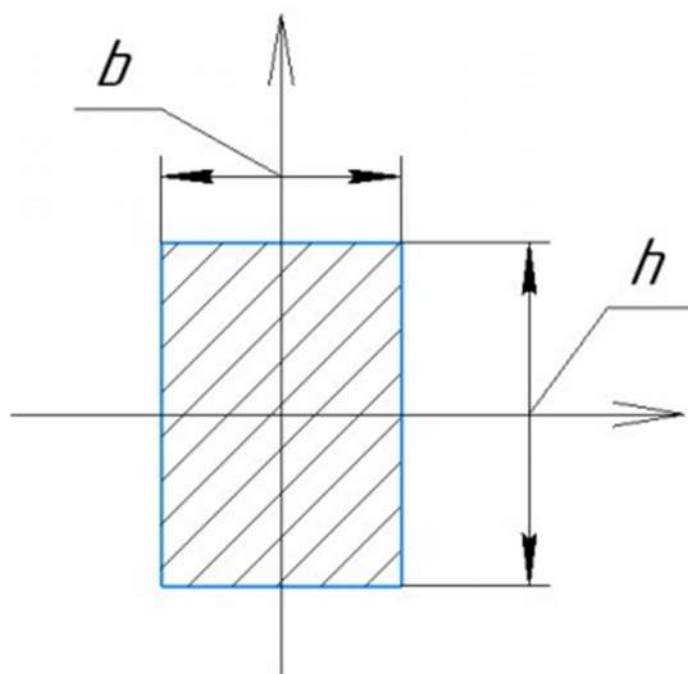
Исходные данные:

Материал балки: сталь (модуль Юнга  $E = 2 \cdot 10^{11}$  Па).

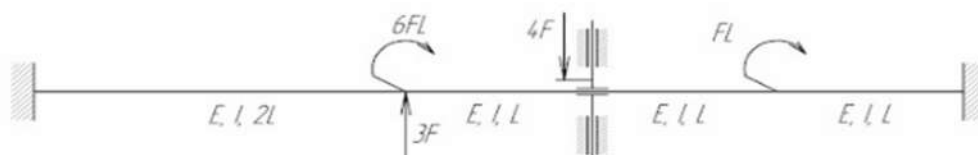
Сечение балки: прямоугольное (см. рисунок).

Геометрические параметры балки:  $l = 0.1$  м,  $b = 10$  мм,  $h = 20$  мм.

Величина нагрузки:  $F = 10$  Н.



2



### Описание алгоритма работы составленной конечно-элементной программы на языке Python

1) Задание исходных данных пользователем:

b, h - параметры сечения;

l- длина балки;

F - вектор силовых нагрузок;

E - модуль Юнга;

U - вектор ограничений, задаваемых пользователем;

n\_el - количество элементов;

N\_dof\_el - количество степеней свободы элемента.

2) Реализация 4 функций, которые выполняются последовательно:

1. `stiffness_element_matrix` - функция, формирующая матрицу жесткости каждого КЭ. На вход принимает параметры КЭ.
2. `index_matrix` - функция, формирующая матрицу индексов для балочных элементов. На вход принимает количество КЭ, количество степеней свободы каждого КЭ.
3. `ensemble` - функция, которая выполняет ансамблирование. На вход принимает матрицы жесткости КЭ, матрицу индексов, количество степеней свободы системы и количество степеней свободы отдельного элемента.
4. `boundary_conditions_modification` - функция, выполняющая наложение граничных условий. На вход принимает глобальную матрицу жесткости, полученную в результате ансамблирования, количество степеней свободы системы и вектор U.

3) В результате формируется и выводится на экран вектор и узловых перемещений.

### Результаты расчета

(вектор узловых перемещений)

Таблица 1. Результаты работы программы.

|                             |       |         |       |        |       |
|-----------------------------|-------|---------|-------|--------|-------|
| Перемещение по узлам (мм)   | 0.000 | 0.005   | 0.014 | 0.007  | 0.000 |
| Вращение по узлам (радианы) | 0.000 | 0.00018 | 0.000 | 0.0009 | 0.000 |

## Текст программы

```
import numpy as np

def index_matrix(n_el, N_dof_el):
    a = 0

    matrix = np.zeros((n_el, N_dof_el))
    for i in range(n_el):
        for j in range(N_dof_el):
            matrix[i][j] = j + 1 + a

        a = a + 2

    return matrix

def stiffness_element_matrix(E, L, J):
    K_e = [[12*E*J/np.power(L, 3), 6*E*J/np.power(L, 2), -12*E*J/np.power(L, 3),
6*E*J/np.power(L, 2)],
           [6*E*J/np.power(L, 2), 4*E*J/L, -6*E*J/np.power(L, 2), 2*E*J/L],
           [-12*E*J/np.power(L, 3), -6*E*J/np.power(L, 2), 12*E*J/np.power(L, 3), -
6*E*J/np.power(L, 2)],
           [6*E*J/np.power(L, 2), 2*E*J/L, -6*E*J/np.power(L, 2), 4*E*J/L]]

    return K_e

def ensemble(K, M_i, N_dof_system, N_dof_el):
    K_g = np.zeros((N_dof_system, N_dof_system))

    b = N_dof_el
    offset = 0

    for i in range(n_el):
        for j in range(N_dof_el):
            K_g[j + offset][offset: 4 + offset] = K_g[j + offset][offset:4 + offset] +
K[i][j]

            offset = offset + 2

    return K_g

def boundary_conditions_modification(K_g, U, N_dof_system):
    for i in range(N_dof_system):
        if (U[i] == 0):
            K_g[..., i] = 0
            K_g[i,...] = 0
            K_g[i][i] = 1

    return K_g

n_el = int(input("Введите количество элементов: "))
N_dof_el = 4
N_dof_system = int((n_el + 1) * N_dof_el / 2)
```

$E = 2e11$

$b = 10$

$h = 20$

$U = [0, 0, 1, 1, 1, 0, 1, 1, 0, 0]$

```

F = [ 0, 0, 30, -6000, -40, 0, 0, -1000, 0, 0]
J = float(b * np.power(h, 3) / 12)
K = []
for i in range(0, n_el):
    L = float(input("Введите длину элемента: "))
    K.append(stiffness_element_matrix(E, L, J))
M_i = index_matrix(n_el, N_dof_el)
K_g = ensemble(K, M_i, N_dof_system, N_dof_el)
K_mod = boundary_conditions_modification(K_g, U, N_dof_system)
u = np.linalg.inv(K_mod).dot(F)
print("Вектор узловых перемещений: ", u)

```

## **Описание выполнения расчета заданной системы в программе Siemens NX**

### **1) CAD-модуль. Создание геометрии рассчитываемого объекта.**

1. Создание нового файла модели.
2. Построение геометрии (линии) с использованием параметра “через 2 точки”.
3. Сохранение файла.

### **2) CAE-модуль. Создание конечно-элементной модели.**

1. Создание нового файла КЭ модели.
2. Установление связи с файлом геометрии: выбор параметра импортирования геометрии “Прямые”.
3. Выбор параметра “1D сетка” для создания КЭ сетки.
4. Задание размера КЭ.
5. Задание параметров КЭ (материал: AISI\_Steel\_1005, поперечное сечение: его тип и размеры).
6. Сохранение файла.

### **3) Задание граничных условий (закрепления и силовые факторы).**

1. Создание файла “SIM”.
2. Выбор типа закрепления и узла, который нужно закрепить.
3. Выбор силового внешнего фактора (сила и момент, задание их величины и направления) и узла, к которому внешний силовой фактор прикладывается.
4. Выбор отображения КЭ модели.
5. Сохранение файла.

### **4) Просмотр результатов.**

## **Результаты расчета в Siemens NX**

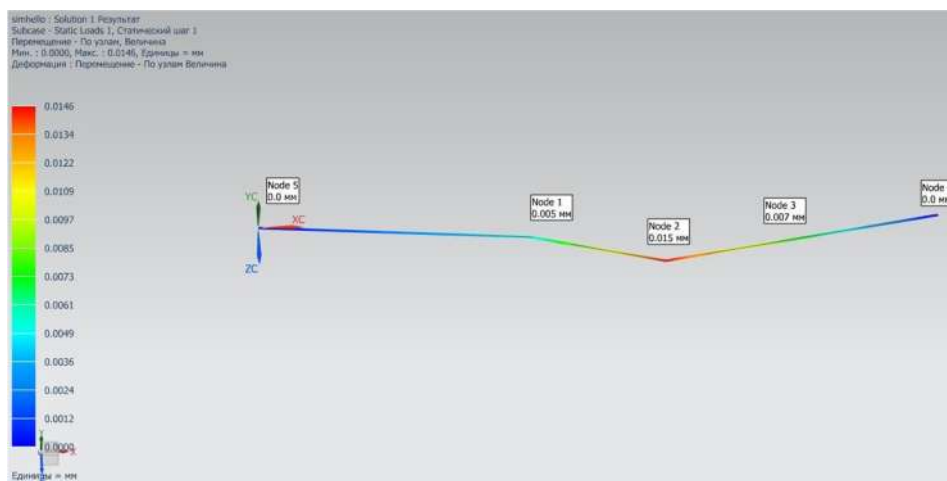


Рисунок 1. Перемещение по узлам.

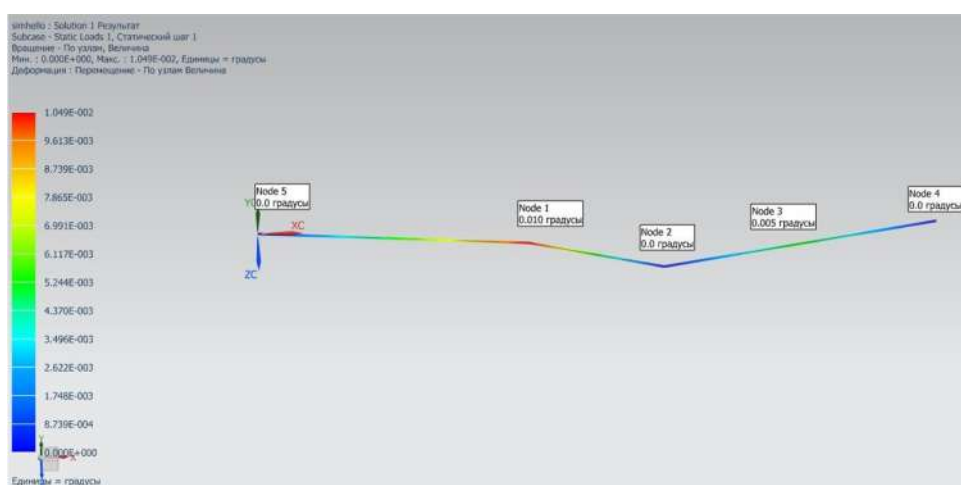


Рисунок 2. Вращение по узлам.

## Сравнение результатов

Таблица 2. Сравнение результатов работы программы и Siemens NX.

|                             |            |       |         |       |        |       |
|-----------------------------|------------|-------|---------|-------|--------|-------|
| Перемещение по узлам (мм)   | Python     | 0.000 | 0.005   | 0.014 | 0.007  | 0.000 |
|                             | Siemens NX | 0.000 | 0.005   | 0.015 | 0.007  | 0.000 |
| Вращение по узлам (радианы) | Python     | 0.000 | 0.00018 | 0     | 0.0009 | 0     |
|                             | Siemens NX | 0.000 | 0.00017 | 0.0   | 0.0008 | 0.000 |

## Вывод

Результаты работы программы практически полностью совпадают с результатами работы Siemens NX.

Следовательно, составленная программа для расчета статически-неопределимой балки работает корректно.