Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Санкт-Петербургский государственный архитектурно-строительный университет»**

Факультет строительный

Кафедра информационных технологий

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

Обучающийся Мельниченко Д.С.

Направление подготовки: 01.04.02 – Прикладная математика и информатика

Группа: ПМИб-3

Оценка

Дата

Преподаватель

Семенов А.А., заведующий каф. ИТ

*(подпись) (Ф.И.О., должность)*

Санкт-Петербург

2023 г.

# ОГЛАВЛЕНИЕ

Оглавление

[ОГЛАВЛЕНИЕ 2](#_Toc128653141)

[ВВЕДЕНИЕ 3](#_Toc128653142)

[ОСНОВНАЯ ЧАСТЬ 4](#_Toc128653143)

[1. Основные теоретические материалы 4](#_Toc128653144)

[2. Вычисления 5](#_Toc128653145)

[ЗАКЛЮЧЕНИЕ 7](#_Toc128653146)

[ПРИЛОЖЕНИЕ 7](#_Toc128653147)

# ВВЕДЕНИЕ

Чтобы описать реальный физический процесс математическими зависимостями используются разные приемы и методы, которые исходят из природы процесса исследования. Пожалуй, основной метод построения математических моделей основан на применении фундаментальных законов природы, таких как сохранение энергии, массы вещества, импульса, числа частиц и др. Для построения математических моделей может применяться метод, основанный на вариационных принципах. Также для построения математических моделей может применяться принцип аналогий, иерархический подход, подход, основанный на приравнивании к нулю проекций всех силовых факторов (внутренних и внешних) по соответствующим направлениям осей координат.

Более широкими возможностями обладает математическое моделирование как метод исследования различных процессов путем описания их функционирования с помощью математических соотношений. Таким образом, при изучении процессов методом математического моделирования в первую очередь необходимо построить математическое описание изучаемого процесса. Математическая модель реального процесса – некоторый математический объект, поставленный в соответствие данному физическому процессу, т. е. математическое описание физического процесса с помощью алгебраических, дифференциальных, интегральных и других уравнений. Эти уравнения обычно выражают законы сохранения основных физических величин (энергии, количества движения, массы и др.) и связывают характеристики процесса с параметрами соответствующей системы, исходной информацией и начальными условиями.

# ОСНОВНАЯ ЧАСТЬ

# Основные теоретические материалы

Суть построения математических моделей на основе вариационных принципов состоит в построении функционала полной энергии изменения (например, энергии деформации для механических систем) изучаемого процесса и минимизации этого функционала, т. е. нахождения уравнений Эйлера для этого функционала.

Далее, приведем вывод необходимых нам формул, для составления мат. модели и последующего вычисления аппроксимирующей функции и определений деформации в любой точки балки:

Выведем уравнение равновесия пластины толщиной *h*, находящейся под действием поперечной нагрузки *q*(*x*,*y*).  
Срединную поверхность (плоскость) пластины примем за координатную и сведем трехмерную задачу к двумерной относительно деформации срединной поверхности, введя следующие ограничения:

1. Пластина допускает малые прогибы, поэтому соотношения между деформациями и перемещениями (геометрические соотношения) будут линейными.
2. Справедлива гипотеза прямой нормали, согласно которой первоначально прямолинейный и нормальный к срединной поверхности элемент остается при деформировании пластины прямолинейным и нормальным. При этом перемещения в слое, отстоящем на *z* от срединной поверхности, имеют вид.
3. Для тонких пластин (*h*/*a* < 1/20) пренебрегается вертикальными напряжениями (*z* = 0).
4. Будем считать, что в пластине под действием нагрузки возникают только изгибные деформации.
5. Материал пластины изотропный и упругий (связь между напряжениями и деформациями линейная). Основные соотношения деформирования пластины состоят из геометрических и физических соотношений и функционала полной энергии деформации.

При введенных предположениях геометрические соотношения в срединной поверхности будут иметь вид (связь между деформациями и перемещениями):



С учетом ограничения 4 (учитываются только изгибные деформации)

Так как наша модель основывается на минимизации функционала полной энергии, то нам необходимо привести его формулу (c учетом наших предположений):



Здесь выражения для моментов получаются интегрированием компонент напряжений по толщине пластины:



Функционал полной потенциальной энергии деформации равен разности потенциальной энергии системы и работы внешних сил А:



Потенциальная энергия системы – это работа внутренних сил, которая в общем виде записывается как:



Исходя из принятых нами предположений мы можем записать Es как:



# Вычисления

Полученные значения, уже с первых значений были достаточно точными, и дальнейшее приближение не сильно увеличило точность нашего значения. (см. таблицу 1.)

Также, можно проанализировать графики и сделать вывод о том, что наибольшей деформации подвергается середина балки т.к. в нашем примере балка закреплена с обоих сторон. (см. рис. 1)

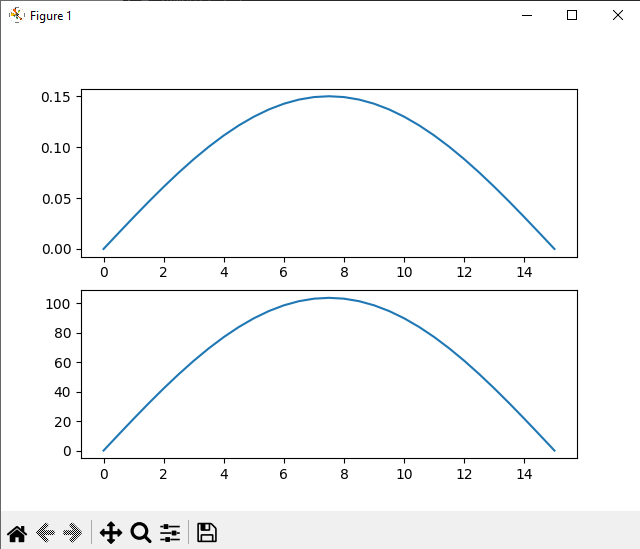


Рис. 1. Верхний – *W(x)*, Нижний – *Ϭ(x)*

|  |  |
| --- | --- |
| Зависимость: | W(x) при x= L/2 |
| N=1 | 0.150130398229424 |
| N=2 | 0.150130398229424 |
| N=3 | 0.149512577660579 |
| N=4 | 0.149512577660579 |
| N=5 | 0.149560619388012 |
| N=7 | 0.149551686776706 |
| N=9 | 0.149554229248183 |

Таблица 1 – Зависимость значений *W(x)* в середине балки от *N*

# ЗАКЛЮЧЕНИЕ

Проанализировав полученные значения, мы можем сделать вывод, что при увеличении кол-ва функций входящих в аппроксимирующую функцию увеличивается и точность решения. Задав необходимые требования к точности решения, мы можем без проблем получить необходимое приближение к реальным значениям.

Также необходимо упомянуть, что даже в такой, казалось бы, простой задаче уже при *N*=10 время вычисления необходимых *W(x)* начинает сильно расти и превышает минуту.

# ПРИЛОЖЕНИЕ

Далее, приведем код решения на языке Python и результаты:  
from sympy import \*  
import numpy as np  
import math as m  
import matplotlib as mpl  
import matplotlib.pyplot as plt  
  
N=1  
L=15  
x = Symbol('x')  
w\_coefs=[]  
E=2.1\*(10\*\*5)  
q\_T=1.34/100  
h=0.15  
  
#Settings to integral  
a = 0  
b = L  
  
#Diff w to Es  
def function\_w(x,n,l,Coef):  
 Pp = Symbol('pi')  
 Ll = Symbol('l')  
 w=[]  
 w\_2=[]  
 w\_3 = []  
 for i in range(1,n+1):  
 w.append(Coef[i-1]\*sin(m.pi\*x\*i/Ll))  
 for i in range(1, n + 1):  
 w\_2.append(w[i-1].diff(x))  
 for i in range(1, n + 1):  
 w\_3.append(w\_2[i-1].diff(x))  
 return w\_3  
def fuction\_q(x,n,l,Coef):  
 Pp = Symbol('pi')  
 Ll = Symbol('l')  
 w = []  
 w\_2 = []  
 w\_3 = []  
 for i in range(1, n + 1):  
 w.append(Coef[i - 1] \* sin(m.pi \* x \* i / Ll))  
 return w  
  
#Add w1..n  
for i in range(1, N + 1):  
 w\_coefs.append(Symbol('w'+str(i)))  
  
  
y = function\_w(x,N,L,w\_coefs)  
q = fuction\_q(x,N,L,w\_coefs)  
  
y\_result=0  
q\_result=0  
for i in range (0,N):  
 y\_result += y[i]  
 q\_result += q[i]  
  
y\_for\_sigma=y\_result  
y=y\_result\*\*2  
q=q\_result  
  
y\_1 = integrate(y,(x,a,b))  
q\_1 = integrate(q,(x,a,b))  
  
Ee = Symbol('E')  
Hh = Symbol('h')  
Qq = Symbol('q')  
Ll = Symbol('l')  
y\_1\*=Ee\*(Hh\*\*3)/24  
q\_1\*=Qq  
  
Es=y\_1  
dw = []  
for i in range (1,N+1):  
  
 buf = y\_1.diff(w\_coefs[i - 1]) - q\_1.diff(w\_coefs[i - 1])  
  
 for j in range(1,N+1):  
 if j != i :  
 buf=buf.subs('w'+str(j),0)  
  
 buf=buf.subs([(Ee,E),(Hh,h),(Qq,q\_T),(Ll,L),(pi,m.pi)])  
 dw.append(solve(buf)[0])  
  
W\_result=0.  
# W для конкретного икса  
Xx=L/2  
pi=m.pi  
for i in range(1, N + 1):  
 W\_result+= dw[i-1] \* sin((pi \* i \* Xx)/L)  
  
print(W\_result)  
# W для всех иксов  
fig,axs = plt.subplots(2)  
x\_now\_1=0  
X\_count\_1 = L/0.5  
X\_step\_1 = L/X\_count\_1  
X\_for\_graph\_1 =[]  
W\_graph =[]  
for i in range (1,int(X\_count\_1)+2):  
 W\_result=0  
 for i in range(1, N + 1):  
 W\_result+= dw[i-1] \* sin((pi \* i \* x\_now\_1)/L)  
 W\_graph.append(W\_result)  
 X\_for\_graph\_1.append(x\_now\_1)  
 x\_now\_1 +=X\_step\_1  
  
  
axs[0].plot(X\_for\_graph\_1,W\_graph)  
  
y\_for\_sigma=y\_for\_sigma.subs([(Ll,L),(pi,m.pi)])  
Sigma =[]  
  
  
z=h/2  
x\_now=0  
X\_count = L/0.5  
X\_step = L/X\_count  
X\_for\_graph=[]  
for i in range (1,int(X\_count)+2):  
 X\_for\_graph.append(x\_now)  
 E\_x =y\_for\_sigma.subs(x,x\_now)  
 x\_now+=X\_step  
 for j in range(1,N+1):  
 E\_x=E\_x.subs('w'+str(j),dw[j-1])  
  
 buf\_sigma = -1\*E\*z\*E\_x  
 Sigma.append(buf\_sigma)  
  
axs[1].plot(X\_for\_graph,Sigma)  
plt.show()  
#y\_for\_sigma=y\_for\_sigma.suba()