ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Профессор |  |  |  | Колесникова С.А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1  ЛП. Нелинейное программирование. Вариационный принцип. |
| **по дисциплине: Компьютерное моделирование** |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134К |  |  |  | Самарин Д. В. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2024

**Часть 1.**

**Содержательная постановка задачи:**

Завод производит корпуса для холодильников пяти различных марок. Для их изготовления требуются трудозатраты, металл, пластик и краска. Каждому ресурсу соответствует определённый месячный лимит. Производство холодильников приносит различную прибыль в зависимости от марки. Необходимо составить такой план производства холодильников каждой марки, который максимизирует суммарную прибыль, учитывая ограничения по ресурсам.

**Переменные задачи:**

* x1​ — количество холодильников марки 1, которое нужно произвести.
* x2​ — количество холодильников марки 2, которое нужно произвести.
* x3​ — количество холодильников марки 3, которое нужно произвести.
* x4​ — количество холодильников марки 4, которое нужно произвести.
* x5​ — количество холодильников марки 5, которое нужно произвести.

**Целевая функция:**

Цель — максимизировать прибыль от производства холодильников. Формула прибыли будет суммой произведений количества произведённых холодильников каждой марки на прибыль от продажи одного холодильника этой марки:

Max Z=40x1​+70x2​+120x3​+120x4​+50x5​

**Ограничения:**

У нас есть четыре типа ресурсов, каждый из которых имеет свои ограничения по объему.

**Ограничение по трудозатратам (чел.-ч):**

Для каждой марки холодильников требуется определенное количество трудозатрат. Общие трудозатраты не должны превышать 900 человеко-часов:

2x1​+3x2​+5x3​+4x4​+4x5​≤900

**Ограничение по металлу (м²):**

Для производства холодильников требуется металл. Общий объём металла не должен превышать 8500 м²:

2x1​+2x2​+4x3​+5x4​+0x0≤8500

**Ограничение по пластику (м²):**

Общий объём используемого пластика не должен превышать 4000 м²:

1x1​+3x2​+2x3​+0x4​+4x4≤4000

**Ограничение по краске (кг):**

Для производства холодильников также требуется краска. Общий её объём не должен превышать 5000 кг:

1x1​+2x2​+3x3​+3x4​+2x5​≤5000

**Ограничения на переменные:**

Количество произведённых холодильников каждой марки должно быть неотрицательным, то есть:

x1​,x2​,x3​,x4​,x5​≥0

**Общая формулировка задачи ЛП:**

Целевая функция:

Max Z=40x1​+70x2​+120x3​+120x4​+50x5​

Ограничения:

2x1​+3x2​+5x3​+4x4​+4x5​≤900

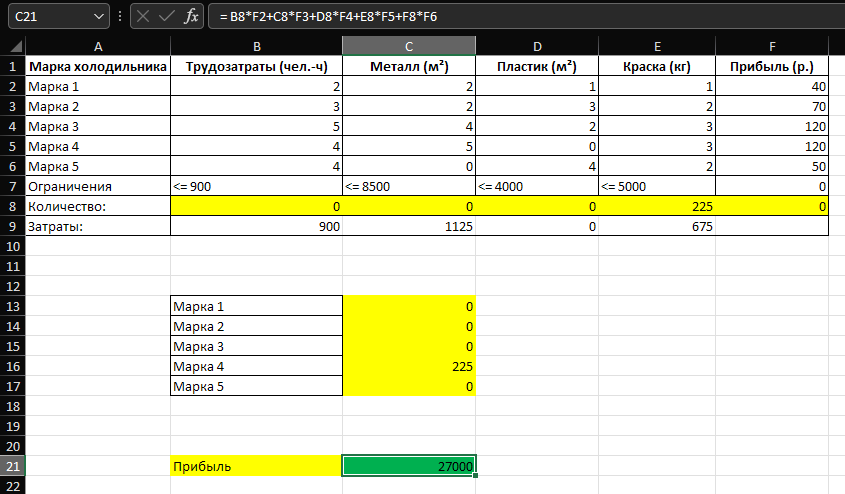
2x1​+2x2​+4x3​+5x4​+0x0≤8500

1x1​+3x2​+2x3​+0x4​+4x4≤4000

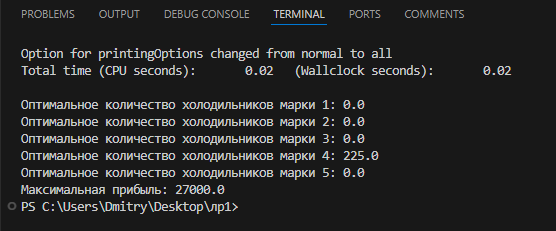
1x1​+2x2​+3x3​+3x4​+2x5​≤5000

x1​,x2​,x3​,x4​,x5​≥0

Ход работы в Excel:



Решение с помощью программирования:



Рассмотрим код:

|  |
| --- |
| import pulp  # Создаем задачу для максимизации прибыли  model = pulp.LpProblem("Maximize\_Profit", pulp.LpMaximize)  # Определяем переменные (целочисленные переменные x1, x2, x3, x4, x5)  x1 = pulp.LpVariable('x1', lowBound=0, cat='Integer')  x2 = pulp.LpVariable('x2', lowBound=0, cat='Integer')  x3 = pulp.LpVariable('x3', lowBound=0, cat='Integer')  x4 = pulp.LpVariable('x4', lowBound=0, cat='Integer')  x5 = pulp.LpVariable('x5', lowBound=0, cat='Integer')  # Целевая функция: максимизация прибыли  model += 40 \* x1 + 70 \* x2 + 120 \* x3 + 120 \* x4 + 50 \* x5, "Profit"  # Ограничения по ресурсам  model += 2 \* x1 + 3 \* x2 + 5 \* x3 + 4 \* x4 + 4 \* x5 <= 900  # Трудозатраты  model += 2 \* x1 + 2 \* x2 + 4 \* x3 + 5 \* x4 + 0 \* x5 <= 8500  # Металл  model += 1 \* x1 + 3 \* x2 + 2 \* x3 + 0 \* x4 + 4 \* x5 <= 4000  # Пластик  model += 1 \* x1 + 2 \* x2 + 3 \* x3 + 3 \* x4 + 2 \* x5 <= 5000  # Краска  # Решение задачи  model.solve()  # Вывод результатов  print(f"Оптимальное количество холодильников марки 1: {x1.varValue}")  print(f"Оптимальное количество холодильников марки 2: {x2.varValue}")  print(f"Оптимальное количество холодильников марки 3: {x3.varValue}")  print(f"Оптимальное количество холодильников марки 4: {x4.varValue}")  print(f"Оптимальное количество холодильников марки 5: {x5.varValue}")  print(f"Максимальная прибыль: {pulp.value(model.objective)}") |

**Часть 2.**

**Цель работы:**

Цель настоящей работы – освоить средства моделирования задач нелинейного программирования. Решение простейшей вариационной задачи.

**Ход работы:**

1. Ознакомиться со справочными сведениями;

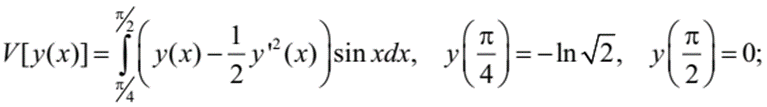
2. Записать и решить уравнение Эйлера-Лагранжа для оптимизационного функционала.

3. Разработать программу, моделирующую алгоритм поиска оптимального решения для формализованной задачи, используя вычислительный пакет MatLab и/или язык программирования Python. За аналитическое решение («в ручную») ДУ – дополнительные баллы-бонусы.

4. Подготовить и устно защитить отчет о работе.

**Рассмотрим решение:**

**Уравнение Эйлера-Лагранжа.**



Функционал, который нужно оптимизировать:

**V[y(x)] = ∫(π/4)^(π/2) (y(x) - 1/2 y^2(x)) sin(x) dx**

Граничные условия:

**y(π/4) = -ln(√2), y(π/2) = 0**

Уравнение Эйлера-Лагранжа для функционала V[y(x)]:

**d/dx(∂F/∂y') - ∂F/∂y = 0**

**Где F = (y(x) - 1/2 y^2(x)) sin(x).**

**Вывод уравнения Эйлера-Лагранжа.**

1. Вычисляем ∂F/∂y:

**∂F/∂y = sin(x) (1 – y(x))**

2. Поскольку в F не содержится y', уравнение Эйлера-Лагранжа упрощается до:

**sin(x) (1 – y(x)) = 0**

Так как sin(x) ≠ 0 на интервале (π/4, π/2), уравнение превращается в:

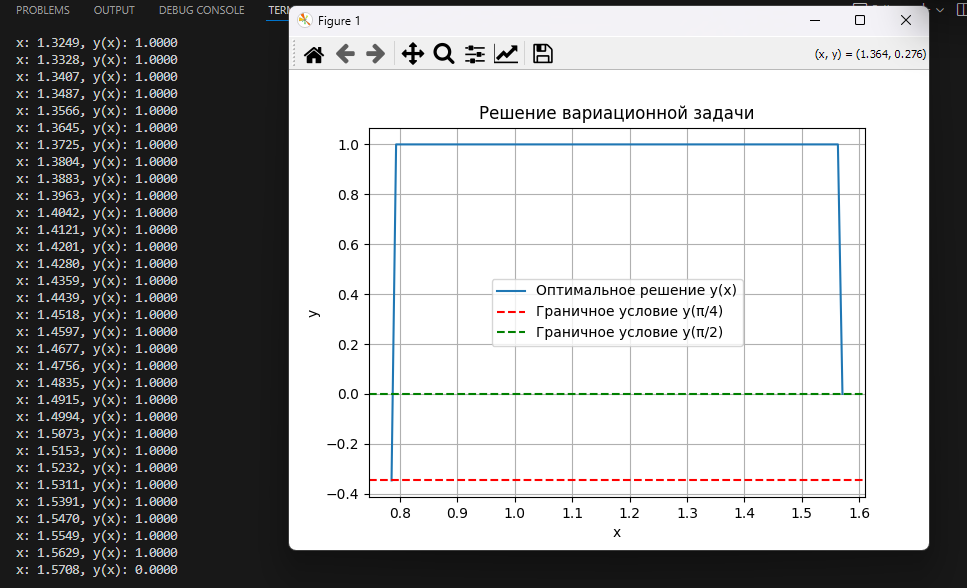
**y(x) = 1**

Таким образом, полученное решение y(x)=1 является следствием решения уравнения, но не удовлетворяет вашим граничным условиям.

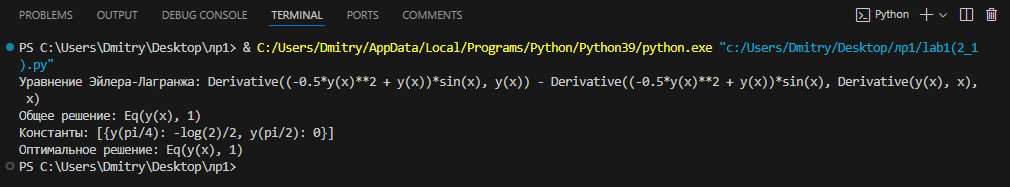
**Рассмотрим решение с помощью PY.**

В качестве итогового результата будут рассмотрены несколько методов решения поставленной задачи.

Метод 1 (с выводом графика):



Метод 2 (схожий с MatLab):



Таким образом в двух из случаев оптимальным решением является **y(x) = 1.**

Предлагаю ознакомиться с кодом для первой и второй программы соответственно:

|  |
| --- |
| import numpy as np  import matplotlib.pyplot as plt  # Параметры  x\_start = np.pi / 4  x\_end = np.pi / 2  num\_points = 100  x\_values = np.linspace(x\_start, x\_end, num\_points)  # Граничные условия  y\_start = -np.log(np.sqrt(2))  y\_end = 0  # Начальная оценка функции y  y\_values = np.ones(num\_points)  # Начнем с константы y = 1  # Итерационный процесс для нахождения оптимального решения  tolerance = 1e-6  max\_iterations = 1000  for iteration in range(max\_iterations):      # Обновление значений y      y\_new = np.zeros\_like(y\_values)        for i in range(1, num\_points - 1):          # Используем уравнение Эйлера-Лагранжа          y\_new[i] = y\_values[i] - 0.01 \* np.sin(x\_values[i]) \* (1 - y\_values[i])        # Установка граничных условий      y\_new[0] = y\_start      y\_new[-1] = y\_end        # Проверка на сходимость      if np.max(np.abs(y\_new - y\_values)) < tolerance:          break        y\_values = y\_new  # Вывод значений y(x)  for x, y in zip(x\_values, y\_values):      print(f"x: {x:.4f}, y(x): {y:.4f}")  # Визуализация результата  plt.plot(x\_values, y\_values, label='Оптимальное решение y(x)')  plt.axhline(y\_start, color='r', linestyle='--', label='Граничное условие y(π/4)')  plt.axhline(y\_end, color='g', linestyle='--', label='Граничное условие y(π/2)')  plt.title('Решение вариационной задачи')  plt.xlabel('x')  plt.ylabel('y')  plt.legend()  plt.grid()  plt.show() |

|  |
| --- |
| from sympy import symbols, Function, Derivative, dsolve, solve, sin, log, sqrt, pi  # Определяем символы  x = symbols('x')  y = Function('y')(x)  dy = Derivative(y)  # Определяем функционал  F = (y - (1/2) \* y\*\*2) \* sin(x)  # Функционал, соответствующий вашему случаю  # Находим производные  dFdy = Derivative(F, y)  dFd1y = Derivative(F, dy)  # Уравнение Эйлера-Лагранжа  L = dFdy - Derivative(dFd1y, x)  print(f'Уравнение Эйлера-Лагранжа: {L}')  # Решение уравнения  sol = dsolve(L)  print(f'Общее решение: {sol}')  # Задаем граничные условия  eq1 = sol.lhs.subs(x, pi/4) - (-log(sqrt(2)))  # y(π/4) = -ln(√2)  eq2 = sol.lhs.subs(x, pi/2)  # y(π/2) = 0  # Решаем систему уравнений для нахождения констант  coeffs = solve([eq1, eq2])  print(f'Константы: {coeffs}')  # Подставляем константы в общее решение  res = sol.subs(coeffs)  print(f'Оптимальное решение: {res}') |

**Вывод:**

Выполнив лабораторную работу №1 (1 и 2 часть), мы освоили средства моделирования задач нелинейного программирования и смогли решить вариационную задачу.