Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

Лабораторная работа №1

«Баллистическая задача»

по курсу «Моделирование»

Выполнила:

студентка 4 курса,

группы ИУ9-82

Козлова А. А.

Проверила:

Домрачева А. Б.

2018 г

Постановка задачи:

Используя математические модели Галилея и Ньютона, определить дальность полета снаряда , брошенного под углом к горизонту α с начальной скоростью V_0 .

<u>Тестовый пример:</u>

Снаряд - свинцовый шарик диаметром $d = 0.1$ м.

1. Модель Галилея

Математическая модель Галилея, основанная на следующих допущениях:

1. Земля – инерциальная система;
2. ускорение свободного падения $g = const$
3. сопротивление воздуха отсутствует.

В этом случае составляющие скорости V движения тела по осям X и Y равны:

$$\begin{aligned} V_x &= V_0 \cos\alpha \\ V_y &= V_0 \sin\alpha - g\,t \end{aligned}$$, где $0 < \alpha < \frac{\pi}{2}$

а их пути являются функциями времени.

$$\begin{aligned} x &= V_0 \cos\alpha \cdot t \\ y &= V_0 \sin\alpha \cdot t - \frac{g t^2}{2} \end{aligned}$$ где t – время движения.

Определяя t из первого уравнения системы $t = \frac{x}{V_0 \cos\alpha}$ и подставляя его во второе, получаем уравнение траектории тела, представляющего собой параболу:

$$y = V_0 \sin\alpha \frac{x}{V_0 \cos\alpha} - \frac{g x^2}{2 V_0^2 \cos\alpha^2}$$

$$y = x\, tg\,\alpha - \frac{g x^2}{2 V_0^2 \cos\alpha^2}$$

Из условия $y = 0$ получаем дальность полета тела

$$l = \frac{V_0}{g} \sin 2\alpha \quad (1)$$

Однако, как показывает практика, результаты, получаемые на основе этой модели, оказываются справедливыми лишь при малых скоростях движения тела. С увеличением скорости V_0 дальность полета становится меньше величины, даваемой формулой (1). Такое расхождение эксперимента с расчетной формулой (1) говорит о неточности модели Галилея, не учитывающей сопротивление.

1.2. Реализация

Язык программирования: Python.

```
def modelGaliley(v_0,alfa,g,h):
  x_arr = []
 y_arr = []
 t = x = y = 0.0
 while ( y >= 0.0):
   x = v_0 * math.cos(alfa) *t
   y = v_0 * math.sin(alfa) *t - g*pow(t,2)/2
   x_arr.append(x)
   y_arr.append(y)
   t += h
 return x_arr[:len(x_arr)-1], y_arr[:len(y_arr)-1]
```

2. Модель Нютона.

2.1 Описание.

В математической модели Ньютона все ограничения совпадают за исключением силы сопротивления воздуха

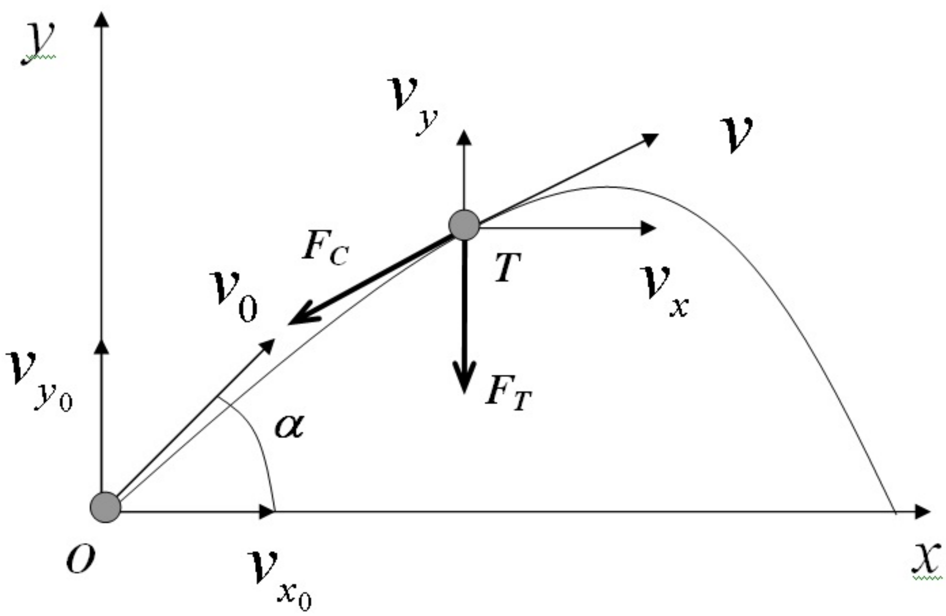
$\overline{F_c} = -\beta \overline{v^2}$

$\beta = \frac{C\rho S}{2}$, где

C - безразмерный коэффициент обтекаемости воздухом, зависящий от формы

ρ - плотность воздуха

S - площадь



Составляющие равнодействующей всех сил по осям X и Y равны:

$$R_x = -\beta V_x \sqrt{V_x^2 + V_y^2}$$
$$R_y = -\beta V_y \sqrt{V_x^2 + V_y^2} - mg$$

Согласно 2-ому закону Ньютона:

$$m\frac{\partial V_x}{dt} = -\beta V_x \sqrt{V_x^2 + V_y^2}$$
$$m\frac{\partial V_y}{dt} = -\beta V_y \sqrt{V_x^2 + V_y^2} - mg$$

Координаты точки связаны с проекциями скорости на оси координат следующим образом:

$$\frac{\partial x}{dt}=V_x$$
$$\frac{\partial y}{dt}=V_y$$

Получаем систему :

$$\begin{cases} m\frac{\partial V_x}{dt}=-\beta V_x\sqrt{V_x^2+V_y^2} \\ m\frac{\partial V_y}{dt}=-\beta V_y\sqrt{V_x^2+V_y^2}-mg \\ \frac{\partial x}{dt}=V_x \\ \frac{\partial y}{dt}=V_y \end{cases}$$

Эта система должна быть дополнена начальными условиям:

$$\begin{cases} V_x(0)=V_0\cos\alpha \\ V_y(0)=V_0\sin\alpha \\ x(0)=0 \\ y(0)=0 \end{cases}$$

Можно показать, что при $\beta=0$ оказывается частным случаем модели Ньютона.

2.2. Реализация метода.

Для решения дифференциальных уравнений был использован метод Рунге-Кутты 4 порядка.

Язык программирования: Python.

```
def f_u ( w, u):
    global beta,m
    root = math.sqrt(math.pow(u,2)+math.pow(w,2))
    return -(beta/m)*u*root

def f_w ( u,  w):
    global beta,m
    root = math.sqrt(math.pow(u,2)+math.pow(w,2))
    return -(beta/m)*w*root-g

def next_iter_runge_cutta_2D(func,u,w,h):
    if (func == for_u):
        k1 = f_u(w, u)
        k2 = f_u(w + h / 2, u + h * k1 / 2)
        k3 = f_u(w + h / 2, u + h * k2 / 2)
        k4 = f_u(w + h, u + h * k3)
        return u + (k1 + 2 * k2 + 2 * k3 + k4) * h / 6
    else:
        l1 = f_w(u,w)
        l2 = f_w(u + h/2,w + h*l1/2)
        l3 = f_w(u + h/2, w + h*l2/2)
        l4 = f_w(u + h, w + h*l3)
        return w + (l1+2*l2+2*l3+l4)*h/6

def modelNewton(u,w,x,y,h):
    x_arr = []
    y_arr = []
    t = 0.0
    while( y >= 0.0):
        u = next_iter_runge_cutta_2D(for_u,u,w,h)
        w = next_iter_runge_cutta_2D(for_w,u,w,h)
        x += u*h
        y += w*h
        x_arr.append(x)
        y_arr.append(y)
    return x_arr[:len(x_arr)-1], y_arr[:len(y_arr)-1]
```

3. Тестирование.

Для тестирования полученной программы были выбраны следующий начальные данные:

стальной щарик диаметром $d=0.1$ м выпущен с начальной скоростью V_0 под углом α к горизонту.

1) Масса шарика была найдена по формуле

$m=\rho\frac{4}{3}\pi\frac{d^2}{4}$, где $\rho=11300$ - плотность свинца

$m=4{,}44\,кг$

2) Коэффициент β был найдем по следующей формуле:

$\beta=\frac{C\rho S}{2}$, где

$C\approx 0{,}15$ - значение коэффициента обтекаемости воздухом для сферы

$\rho=1{,}3\frac{кг}{м^3}$ - плотность воздуха

$S=\pi\frac{d^2}{4}=0{,}008\,м^2$ - площадь поперечного сечения

$\beta=0{,}00076$

3) Угол $\alpha=45$ градусов

4) Начальная скорость $V_0=30\,м/с$

5) Количество разбиений для метода Рунге-Кутты $n=70$

Результат работы программы:

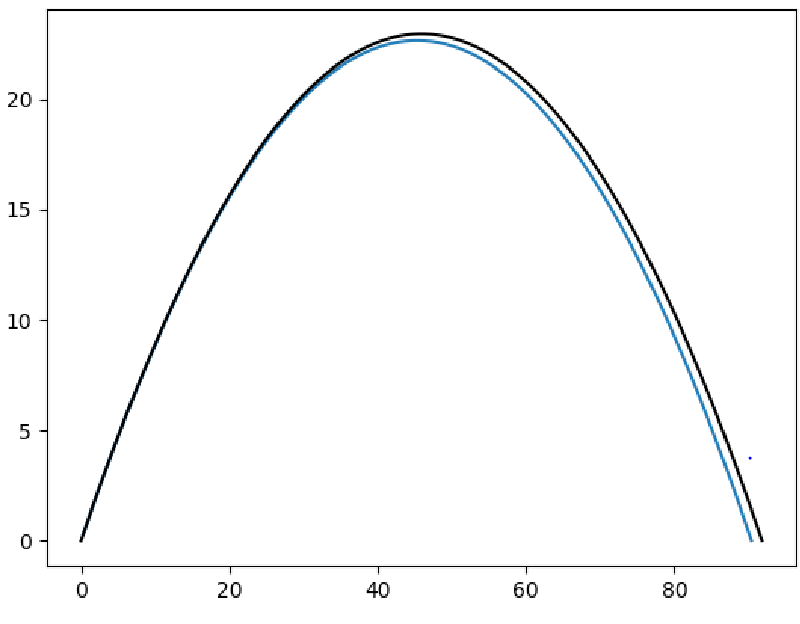


График построен по синей линии соответствует модели Галилея, с черной — модели Ньютона.

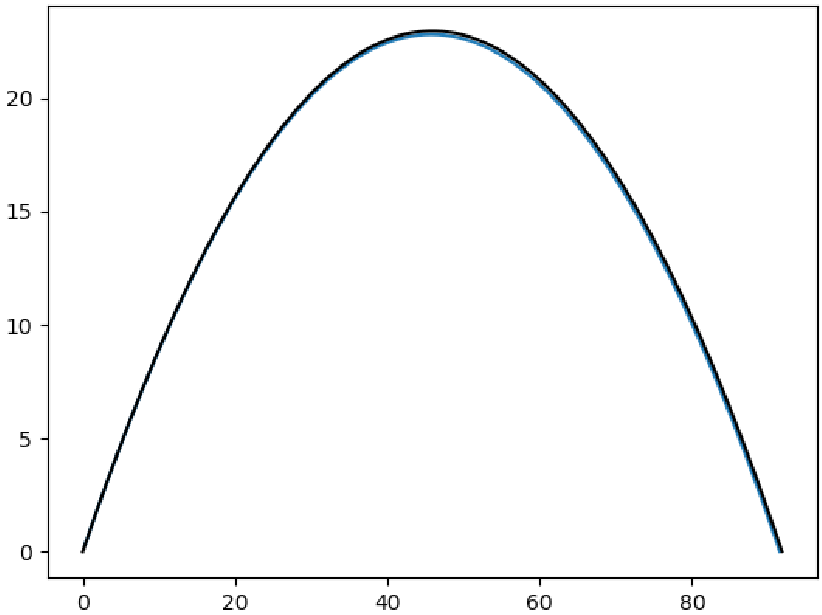
Расстояние, которое в результате пролетит шарик:

Модель Галилея	Модель Ньютона
91.8 м	90.4 м

При тех же начальных условиях и $\beta = 0$:

Модель Галилея	Модель Ньютона
91.8 м	91.5 м

График :



Вывод:

Как модель Галилея, так и модель Ньютона позволяют описать движение тела, брошенного под углом к горизонту, однако модель Ньютона дает более точный результат, так как учитывает силу сопротивления воздуха. Кроме того при $\beta = 0$ модель Галилея является частным случаем модели Ньютона.