Optimization and Differential Equation

Elvis Cabrera

May 11 2019

1 Introduction

I can use scipy.optimize to optimize the dimensions of a rectangular box without a lid while keeping the surface area fixed. I will need the equation of a rectangular box with the height, length, and width as parameters to be determined. The surface area will be used to find the dimensions because there is only so much 'cardboard' (surface area) to be used.

I would also like to use odeint to solve the differential equation for a non-steady state oscillation. I will plot the behavior of a linear oscillator subjected to a quick, discontinuous force. This impulsive force can be from a step or spike (delta function).

2 Optimization

The volume of the box without a lid is xyz. Where x is the length, y is the width, and z is the height of the box. The surface area will be used as a constraint because we will find the dimensions of the box that keeps the surface area fixed to 100 meters squared. The equation for the surface area of a lidless, rectangular box is 2xz + 2yz + xy. The equations used for this part of the project is

$$Volume = xyz$$

$$SurfaceArea = 2xz + 2yz + xy = 100$$

rearranging the equation for volume, we get

$$V = (100xy - x^2y^2)/(2x + 2y)$$

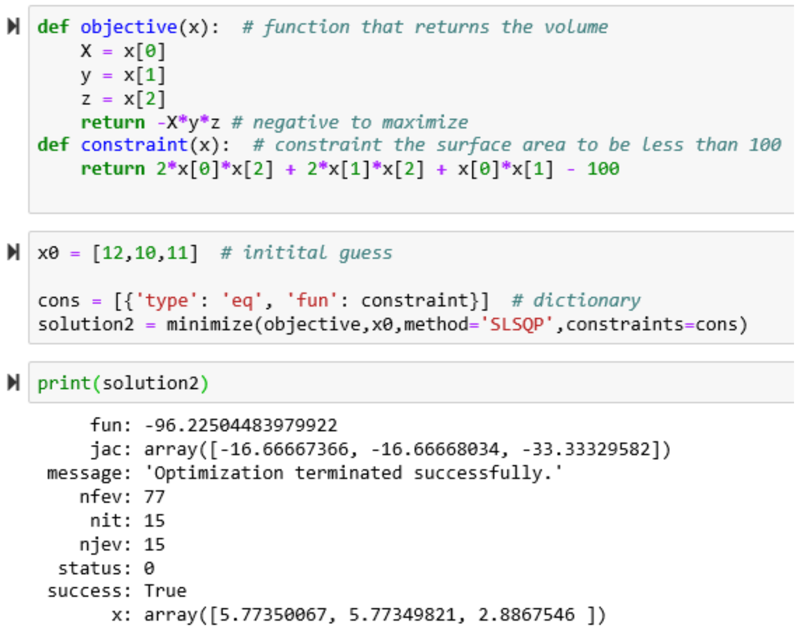
partial differentiating with respect to y and x we get two equations

$$200 - 4xy - 2x^2 = 0$$

$$200 - 4xy - 2y^2 = 0$$

solving this system of equations gives

$$y = x$$



```
def objective(x):  # function that returns the volume
    X = x[0]
    y = x[1]
    z = x[2]
    return -X*y*z # negative to maximize
def constraint(x):  # constraint the surface area to be less than 100
    return 2*x[0]*x[2] + 2*x[1]*x[2] + x[0]*x[1] - 100
```

```
x0 = [12,10,11]  # initital guess

cons = [{'type': 'eq', 'fun': constraint}]  # dictionary
solution2 = minimize(objective,x0,method='SLSQP',constraints=cons)
```

```
print(solution2)
```

```
     fun: -96.22504483979922
     jac: array([-16.66667366, -16.66668034, -33.33329582])
 message: 'Optimization terminated successfully.'
    nfev: 77
     nit: 15
    njev: 15
  status: 0
 success: True
       x: array([5.77350067, 5.77349821, 2.8867546 ])
```

Figure 1: One of the codes I used to solve for the optimal dimension of the lidless box with surface area constrained to 100 units squared.

plugging this particular solution into the equation for the surface area gives

$$z = (100 - x^2)/4x$$

this yields the following dimensions for the box

$$x = y = \sqrt{200/6} = 5.77$$

$$z = (400/24)\sqrt{6/200} = 2.88$$

I showed that this problem can be solved using calculus. This problem can also be solved numerically in Python. I will optimize the dimensions of the box in Python using the function minimize from the package scipy.optimize with the Sequential Least Squares Programming algorithm.

As can be seen in Figure 1, I was able to optimize the dimensions of the box, keeping the surface area fixed. The minimize function takes in the objective function, initial guess for the parameters, the algorithm used to minimize (SLSQP), and the surface area constraint (cons). I defined an objective function that returns the volume of the box. There is a negative attached to it to maximize the volume. I originally did not include it and got the following answer for the dimensions of the box: array([1.03581798e+13, -2.58448925e+13, 3.87672890e+13]). Obviously, this was wrong. I attached a negative on the volume and to my astonishment, it worked. My thought process was the following.

If the function ‘minimize’ minimizes parameters then if I attach a negative, I’ll go the other direction, so maximize.

3 Differential Equation

I will investigate the behavior of an oscillator that is subjected to a discontinuous driving force. In particular, I will plot the time evolution of this oscillator after it encountered a sharp force for a short moment.

The differential equation describing the motion of a damped oscillator is

$$\ddot{x} + 2\beta\dot{x} + \omega_0^2 x = F(t)/m$$

Where β is the damping coefficient and ω_0 is the angular frequency. Here, F(t) is the applied force that could either be a step force function or an impulse force function. The general solution is a sum of a complementary solution and particular solution. Where the particular solution depends on the impulsive force function. The complementary solution is a product of an exponential and sinusoidal function.

$$x(t) = exp(-\beta t)(A_1 cos\omega_1 t + A_2 sin\omega_2 t)$$

where

$$\omega_1 = \sqrt{\omega_0^2 - \beta^2}$$

and

$$\beta < \omega_0$$

I will solve this differential equation in Python with the scipy.integrate package using the function odeint.

The differential equation I used is

$$\ddot{x} = -2\beta\dot{x} - \omega_0^2 x$$

setting $\omega_0 = 4 rad/s$ and $\beta = 1$. I also plugged in the general solution for x(t).

$$x(t) = exp(-t)cos3.87t$$

Initially, I wanted to solve the second order differential equation for a damped oscillator in response to a forcing function in the form of a Dirac delta function. I tried using the special function scipy.signal.unitimpulse() from SciPy, but odeint cannot differentiate a broken function. The delta function is not a proper function in the sense that it is differentiable in small increment steps. So the integrator odeint has trouble differentiating this function. Odeint differentiates continuous functions. I decided to plug in the general solution for x(t) into the second order differential equation. The general form is a product of an exponential and a sinusoidal function of time. I set the forcing function to zero because I am interested in the motion of the oscillator after the impulse. The time evolution of the oscillator after the impulsive force (whichever one it may have been) follow a similar damped sinusoidal curve with the amplitude decreasing as time progresses.

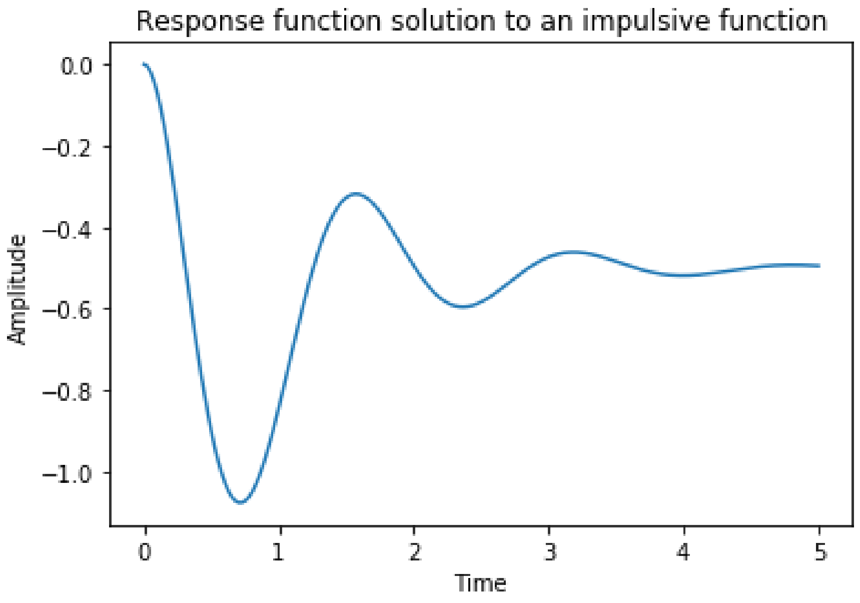


Figure 2: The above plot shows the time evolution of a damped harmonic oscillator that came into contact with a short lived, discontinuos driving force.

4 Conclusion

I optimized the length, width, and height for a box without a lid keeping the surface area constant to 100 meters squared. I solved this optimization problem two ways. First, I used calculus to optimize the dimensions. After finding the dimensions using pen and paper I wanted to solve it numerically in Python using the built in minimize function from the package scipy.optimize. After coding it up I found the dimensions of the box to be exactly those that I found by using calculus.

I investigated the time evolution of a damped oscillator that experienced an impulsive driving force that could've taken the form of a Heaviside step function or an impulse delta function. The amplitude of the oscillator drops as time progresses. I plotted the differential equation in Python and solved it with the scipy.integrate package using the function odeint.

References

[1] Stewart J. *Calculus Early Transcendentals Seventh Ed.* Books/Cole, 2012.

[2] B. Overland. *Python Without Fear.* Pearson, 2018.

[3] E.A. Blanco-Silva F.J. Rojas G. S. J., Christensen. *Concepts in Thermal Physics Second Ed.* PACKT Publishing, 2010.

[4] Marion J.B. Thornton, S.T. *Classical Dynamics of Particles and Systems Fifth Ed.* Thomson Books/Cole, 2004.

[1] [3] [2] [4]