Exercise Three: Freefall With Drag

Imad Abziouad
HH Wills Physics Laboratory, University of Bristol, Bristol, BS8 1TL, UK
Ia17692@bristol.ac.uk

1 Introduction

The aim of this exercise was to utilise the Euler method of differentiation to solve the second order differential for a falling object under drag given by equation one using python.

The second order differential equation was separated into two first order differential equations and then solved individually, for given values of time along the object's descent. This was then compared with an analytical and more mathematically accurate method to test whether the Euler method had functioned properly.

The Euler method was then improved by finding the gradient at the midpoint of each change to increase its accuracy and conserve energy.

$$m\frac{dv_y}{dt} = -mg - k\left|v_y\right|v_y \qquad (1)$$

The second order ODE describes the motion of the object, where m is the mass of the falling object, dv/dt is the rate of change of velocity, g is the acceleration due to gravity, k is a drag constant, and v is the velocity of the object at a given point.

2 Theory

2.1 The standard Euler method

The Euler method works in a very similar way to numerical differentiation and integration. The differential is replaced by a fraction of two finite and infinitesimal values, so that the equation being differentiated can be calculated to a good accuracy within the small limit of difference. The second order ODE to solve can then be rearranged to form the Euler method for an increment, equations two, three and four.

$$v_{y,n+1} = v_{y,n} - \Delta t\left(g + \frac{k}{m}\left|v_y\right|v_y\right) \qquad (2)$$

$$y_{n+1} = y_n + \Delta t \cdot v_{y,n} \qquad (3)$$

$$t_{n+1} = t_n + \Delta t \qquad (4)$$

With these equations the velocity and position of a falling object can be estimated quickly and without much effort, by repeatedly incrementing equation four into equations two and three to calculate new values of y and v.

2.2 The analytical method

The analytical method performed the same function as the Euler method but was solved directly from the second order ODE in equation one, it is a general solution of the ODE and so should provide a more accurate representation of freefall under gravity without any drag. Values of time are inserted into equations five and six via equation seven, and plotted against the time for graphs of velocity and displacement during the fall.

$$y = y_o - \frac{m}{2k}log_e(cosh^2(M)) \qquad (5)$$

$$v_y = -\sqrt{\frac{mg}{k}}\tanh(M) \qquad (6)$$

$$\text{Where } M = t\sqrt{\frac{kg}{m}} \qquad (7)$$

2.3 The modified Euler method

This builds up on the standard Euler method, but makes some crucial and beneficial advantages. The Euler method is applied twice for each step, once to calculate the derivative between the step and once for the gradient at the midpoint at that step, allow for a closer fit and a less divergent derivative calculation. This effect is commonly called the leapfrog method, because it jumps between calculating the midpoint and the next step, repeating this process until termination.

$$v_{mid} = v_o + \frac{\Delta t}{2}f(x_o y_o) \qquad (8)$$

$$t_{mid} = t_o + \frac{\Delta t}{2} \qquad (9)$$

$$v = v_o + \Delta t \cdot f(x_{mid} y_{mid}) \qquad (10)$$

V_0 and t_0 are the initial velocity and time at the beginning of the fall, v_{mid} and t_{mid} are the midpoints of each velocity and time step, and the function f is the original velocity function given in equation two. Equations eight and nine must be computed and put into equation ten to produce an accurate value of velocity at each time interval during the freefall. The displacement can be calculated with equation three again, having adjusted the definitions accordingly.

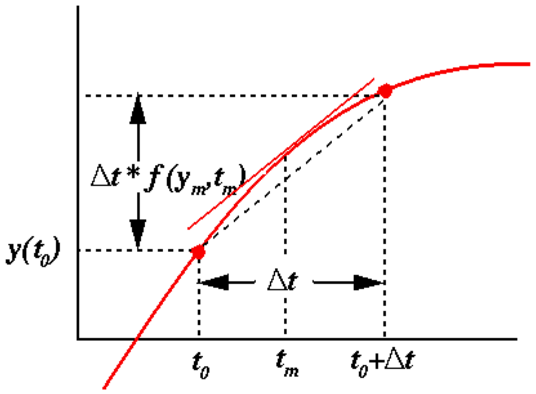

Fig. 1. Graph showing the difference in technique between the standard Euler method using only Δt, and the modified Euler method using both Δt and t_m.

2.4 Density as a function

For a more realistic re-enactment of freefall, the fact that air density varies with height above the ground must be taken into consideration. An equation for how density varies with height can be found and generalised with the exponential equation found in equation 11, where y is the height and the best value of h is 7640m.

$$\rho(y) = \rho_o e^{(-y/h)} \quad (11)$$

3 METHOD

3.1 The standard Euler method

First it was necessary to define values and initialise lists and loop values. These stayed the same across the entire exercise.

```
t = 0
y = 1000
v = 0.0
dt = 0.01
k = 0.624
m = 70
C = 1.15
A = 0.8
ylist = []
xlist = []
zlist = []
```

Now it was possible to create a loop to cycle through values of time in timestep dt and end only when the object had hit the ground, i.e. height was zero. Here the initial height is portrayed as y for convenience.

```
While y > 0:
        dv = v - dt*(9.81 + 0.1*abs(v)*v)
        t += dt
        y += (dt*v)
        v = dv
```

As the loop progressed through time, the values of velocity, position and time were constantly being added to the end of three distinct lists via the following lines of code. These would then make it easy for python and matplotlib to plot them as two graphs, a velocity-time graph and a displacement-time graph.

```
xlist.append(t)
ylist.append(-v)
zlist.append(y)
```

Here, velocity can be seen as negative. This was a decision to ensure that the graph plotted values of velocity that were positive, as a falling object's velocity with respect to a positive gravitational acceleration is negative.

3.2 The analytical method

The analytical method was implemented in an identical fashion to that presented in section 2.2. A similar loop to that done for the standard Euler method was implemented, with height above the ground as y, and y0 this time as the initial height.

```
M = t*np.sqrt((k*g)/m)
y = y0 - (m/k)*np.log(np.cosh(M))
v = - np.sqrt((m*g)/k)*np.tanh(M)
t += dt
```

These values were again collected in lists and plotted as two separate graphs using matplotlib.

3.3 The modified Euler method

This uses the same principles as the normal Euler method, only it was necessary to modify the equation to make one extra step of calculations. The velocity equation was first redefined as a function f of time step and previous velocity, in accordance with section 2.3.

```
def f(dt,v):
        dv = v - dt*(9.81 + 0.0089*abs(v)*v)
        return dv
```

Then it was necessary to redefine the loop to integrate into it a middle step that would increase the precision of the calculation process.

```
while y > 0:
    tmid = dt/2
    vmid = f(tmid,v)
    y += (dt*vmid)
    dv = f(dt,vmid)
    t += dt
    v = dv
```

The values of v and y were then collected into lists and plotted against the time of the fall.

3.4 Modified Euler method with $\rho(t)$

This section was used to replicate Felix Baumgartner's fall. In reality, the density of air is never constant, pressure and most of all temperature have a big effect on how much air exists in a particular volume of atmosphere. Equation 11 was used within a new function k(y) to ensure that the k value which holds the air density was accurately responding to the conditions of the elevation of the falling object.

```
def k(y):
    K = (Cd*rho0*np.exp(-y/7640)*A)/2
    return K
```

This new line of code was put into what was almost identical code to that used in part 3.3, save for the redefining of the change in velocity equation, where a constant k was replaced with a function, k(y).

```
dv = v - dt*(9.81 + (k(y)/70)*abs(v)*v)
```

The height was the same that Baumgartner jumped from, at y = 39,045m, with a starting velocity of v = 0.

To find out whether Baumgartner did in fact break the speed barrier, a new equation was found for the speed of air with altitude. The speed of sound in air depends mainly on temperature. The equation for temperature with altitude is approximated by T = -131.21+0.00299*y. The equation for speed with respect to air temperature is c_{air} = 331.3 + 0.66*T. Combining the two equations, a graph could be plotted of the speed of sound along with the velocity of Felix Baumgartner during his drop.

```
z = 331.3 + 0.66*(-131.21+0.00299*y)
```

3.5 Customisable drop

The last part of the code was simply allow the reader to be able to input their own values, and customise a freefall drop under gravity and drag to their liking. This was done by creating a menu system whereby the programme would ask the user to enter a value in SI units, and the user would then respond by doing so, until the height of the drop, the starting velocity, the accuracy of the calculation, the mass of the object and the surface area were all fulfilled. An example of this system is displayed below.

```
Input_M = input('Enter weight of falling object: ')
m = float(Input_M)
```

4 RESULTS

4.1 The standard Euler method

This part gave a drop duration of 32.47 seconds. The graphs of velocity and displacement can be seen in Fig. 2. and Fig. 3.

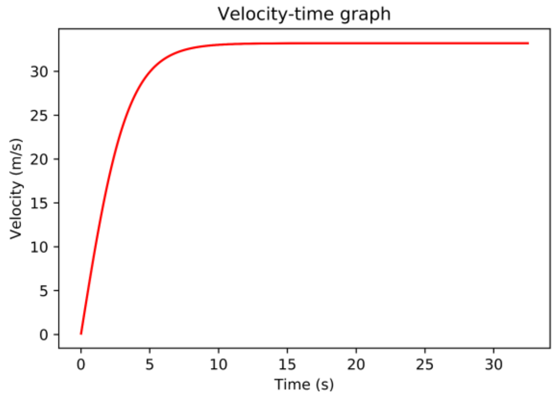

Fig. 2. Velocity time graph for 1000m drop using the standard Euler method.

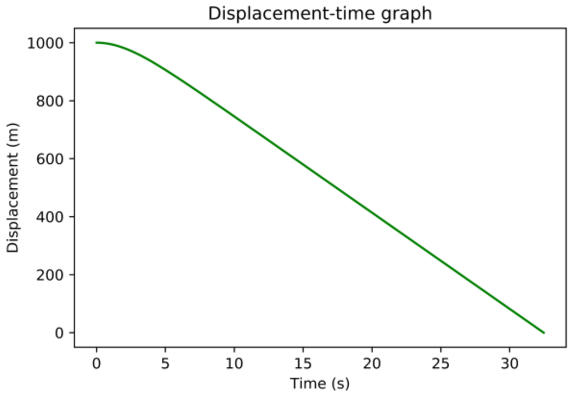

Fig. 3. Displacement time graph for 1000m drop using the standard Euler method.

The maximum velocity reached was 33.20m/s, which is roughly what is to be expected from a fall of 1km.

4.2 The analytical method

The analytical method gave a drop duration of 32.41 seconds, similar to that of the standard Euler method. Graphs of this drop an be seen in Fig. 4. and Fig. 5.

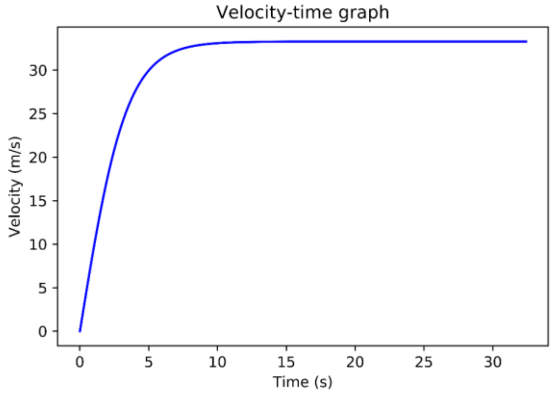

Fig. 4. Velocity time graph for 1000m drop using the analytical method.

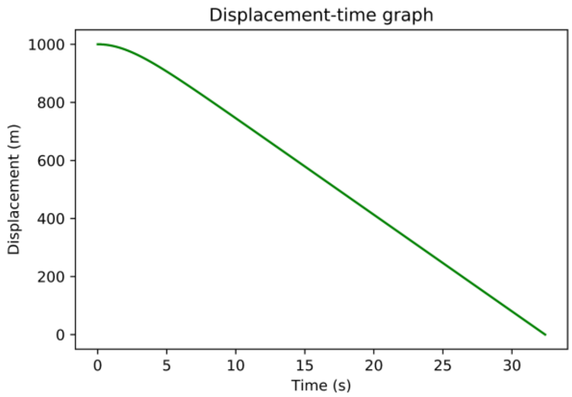

Fig. 5. Displacement time graph for 1000m drop using the analytical method.

The maximum velocity of this drop was found to be 33.28 m/s.

4.3 The modified Euler method

The improved method gave a drop duration of 31.69 seconds. The graphs of velocity and displacement can be seen in Fig. 6. and Fig. 7.

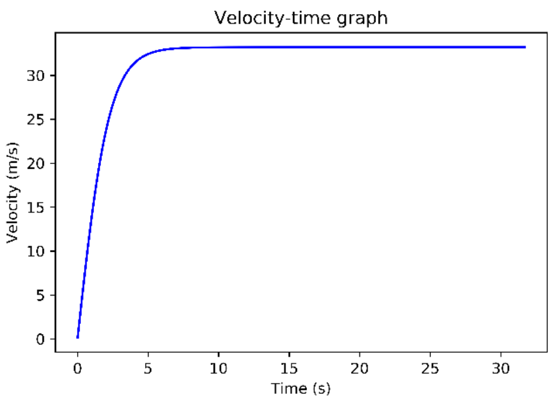

Fig. 6. Velocity time graph for 1000m drop using modified Euler method.

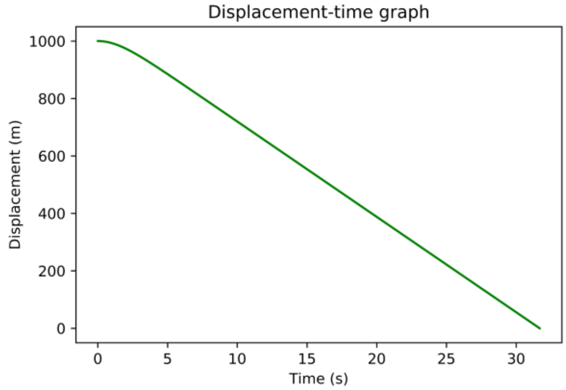

Fig. 7. Displacement time graph for 1000m drop using modified Euler method.

The maximum velocity of this drop was found to be 33.20 m/s, again both duration and maximum speed are all very similar.

4.4 Modified Euler method with $\rho(t)$

This part modelled Baumgartner's descent. With a more realistic approach of air density varying with altitude, there was a drop duration of 269.12 seconds for Felix Baumgartner. The graphs of velocity and displacement can be seen in Fig. 8. and Fig. 9.

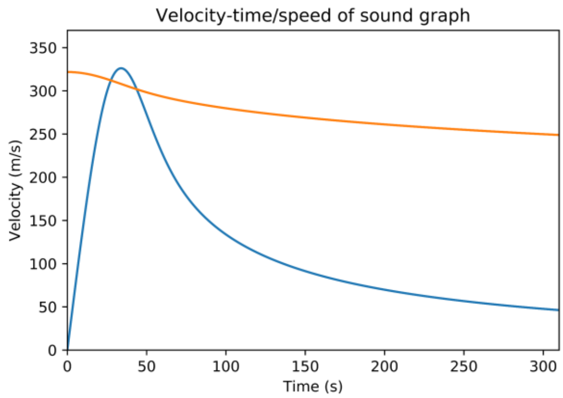

Fig. 8. Velocity time graph for the simulation of Baumgartner's descent, from a height of 39,045m. The blue line is Baumgartner's velocity, and the red line is the speed of sound in air with respect to height.

The maximum velocity reached by this simulation was 328.47 m/s. As can be seen by the graph, it suggests that Felix Baumgartner did indeed break and surpass Mach one of the speed barrier during his drop from 39,054 metres in the air.

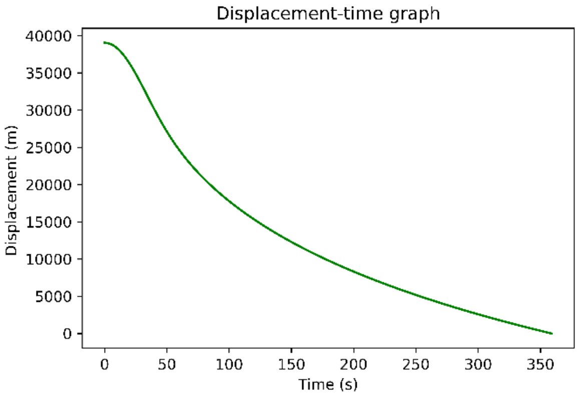

Fig. 9. Displacement time graph for the simulation of Baumgartner's descent, from a height of 39,045m.

5 ANALYSIS

Having said this, running the code in real time and previewing the calculation steps for velocity, the improved Euler method does stay closest to the values and gradient of the analytical method, which is taken to be implicitly the most accurate representation of freefall under gravity and no drag. Even from graphs in Fig. 4. And Fig. 6. it is possible to see that the gradients of the analytical and improved codes share a steeper likeness than that of the standard code in Fig. 2., which takes a more gradual approach to reach asymptotic maximum.

The fourth part of the exercise involving a simulation of Felix Baumgartner also came very close to the values achieved by Baumgartner during his descent. The time taken in reality was 259 seconds, whereas the simulation provided a duration of 269.12 seconds, a difference of only 4% error. The maximum velocity achieved by Baumgartner was 373m/s, whereas the simulation suggests a maximum of 328.47 m/s, an error of 12%. Despite this error, the simulation suggests Baumgartner did indeed break the sound barrier at around 300m/s or 32500m, which is something that did occur during his drop from the middle of the stratosphere.

The duration values for the drop simulations in the Euler method, the analytical method and the improved Euler method were all very similar. At 32.47s, 32.41s and 31.69s, the standard deviation was very small, at just 0.35σ. The graphs of velocity against time for the first three sections look almost identical, which suggests that the benefits in calculating the freefall duration of the improved Euler method and the analytical method cannot be distinguished from the standard Euler method at relatively small scales such as this.

Improvements however could still be made. A more accurate and realistic equation for drag could be used, instead of an approximation using an exponential function. The step size of the simulation could be reduced from 0.01 seconds to 0.001, adding ten times to the precision of the modified Euler method. An even better improvement would have been to include the effects of the parachute Baumgartner deployed at a certain level to stop the pressure from tearing him apart.