Lab 3: Acceleration due to Gravity

William Woodley

Results

The position and velocity of the tennis ball are given respectively by the following kinematic formulae:

$$y(t) = -\frac{1}{2}gt^2 + v_0 t + y_0, \tag{1}$$

$$v(t) = gt + v_0. \tag{2}$$

The equation for $y(t)$ is in the form of a second-order polynomial, and the equation for $v(t)$ is in the form of a first-order polynomial [1]. Both Equation 1 and Equation 2 can be used to find values of g and v_0.

As the tennis ball fell, its motion over time was recorded, and video analysis in Logger Pro gave its position and velocity for each frame of the video. These values are given in Table 1 for the first 0.30 s. The full set of data is given in Table A1 in Appendix A.

Table 1: A sample of the position and velocity data for the tennis ball over time. The uncertainty in time is taken as the reciprocal of the frame rate of the camera, $f = 30$ fps, and the uncertainties in position and velocity are assumed to be negligible. The origin was placed at the point of release of the ball, with the $+\underline{\hat{y}}$-axis pointing downwards.

Time, t / s (± 0.03)	**Position, y / m**	**Velocity, v / ms^{-1}**
0.00	0.003	0.195
0.03	0.013	0.584
0.07	0.042	1.051
0.10	0.083	1.452
0.13	0.168	1.791
0.17	0.202	2.959
0.20	0.280	2.453
0.23	0.366	3.001
0.27	0.480	3.270
0.30	0.584	3.115
...	...	...

The data in Table 1 was imported into a Python notebook, and the `polyfit()` function from the NumPy library, described in [2], was used to fit the position-time data to a second-order polynomial, and the velocity-time data to a first-order polynomial. These fits, as well as the data, are shown in Figure 1 and Figure 2 respectively.

From the velocity-time graph in Figure 2, acceleration due to gravity was calculated to be (9.5 ± 0.4) ms^{-2}, and the initial velocity of the ball was calculated to be (0.5 ± 0.1) ms^{-1}.

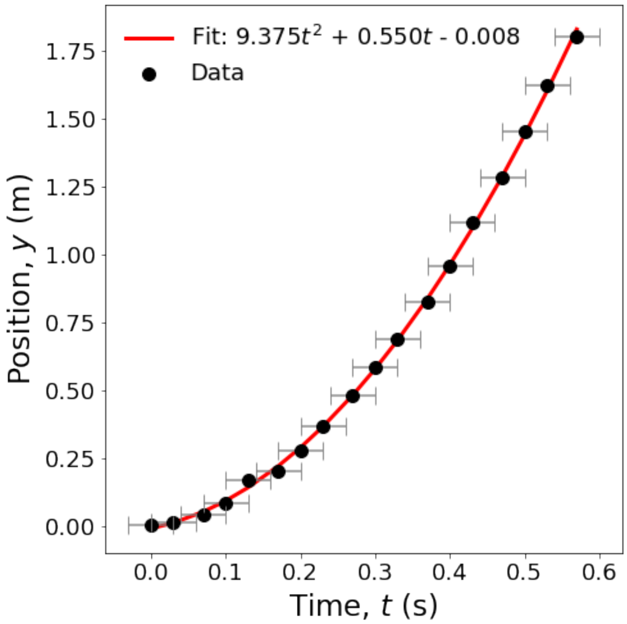

Figure 1: A plot of position vs time using the data from Table A1. Using `numpy.polyfit()`, the parameters of the curve fit were found to be, in order of the terms in Equation 1, (9.4 ± 0.3) ms^{-2}, (0.55 ± 0.08) ms^{-1}, and (0.01 ± 0.01) m.

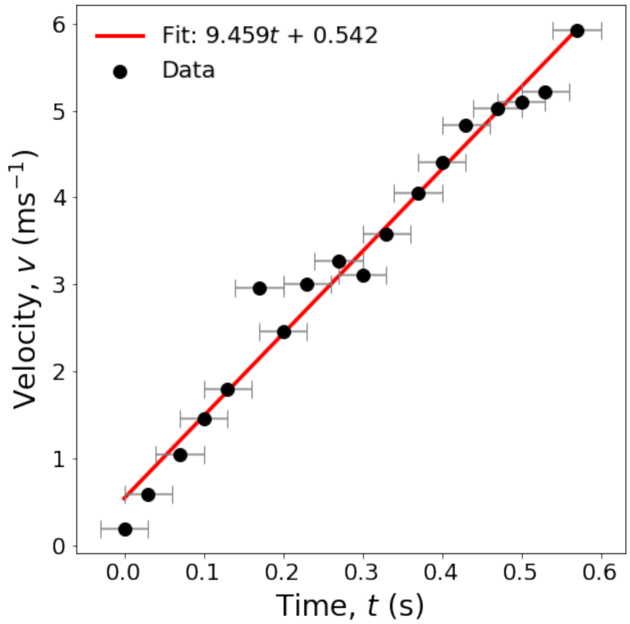

Figure 2: A plot of velocity vs time using the data from Table A1. Using `numpy.polyfit()`, the slope of the fit line was found to be (9.5 ± 0.4) ms^{-2}, and the intercept was found to be (0.5 ± 0.1) ms^{-1}.

Discussion

Acceleration due to gravity was found to be (9.5 ± 0.4) ms^{-2}. This is within one error interval of the accepted value of 9.81 ms^{-2}, and therefore has good agreement with the accepted value. The initial velocity, v_0, was found to be (0.5 ± 0.1) ms^{-1}. As the tennis ball was released from rest, the expected value is 0 ms^{-1}. Because the expected value is more than three error intervals away from the experimental value, there is no agreement between the two values.

The trends of the data in Figure 1 and Figure 2 are both as expected. The position-time graph is a second-order polynomial, as predicted by Equation 1, and the data points lie nicely on the curve. Its shape is consistent with the choice of origin and axis. The velocity-time graph is linear with positive slope, as predicted by Equation 2 and the choice of axis. Though the data does not lie as nicely on the fit line as in the position-time graph, the majority of the data points overlap the line within error; however, there is one noticeable outlier. The point at $t = 0.17$ s does not overlap with the line. This seems to be random error from a single misplaced point, as the surrounding points are also off the trend, because the velocity calculation Logger Pro does to obtain these surrounding values is dependent on the outlier value. When this data point is excluded from the fit, acceleration due to gravity is calculated as (9.6 ± 0.3) ms^{-2}, and the initial velocity is calculated as (0.44 ± 0.09) ms^{-1}. Although the initial velocity still does not agree with the expected value, both g and v_0 became closer to the expected values, and their uncertainties decreased. A plot showing the new fit line is given in Figure B1 in Appendix B.

The most significant source of error in the data collection and analysis was the blurring of the ball as it moved. Because of the low frame rate of the smartphone camera, as the ball's velocity increased, the amount of motion blur became increasingly large, making it very difficult to accurately place points. For the camera used, it is unknown whether the shutter takes the photo starting at the top of the frame or the bottom. For this reason, the points in Logger Pro were placed in the middle of the ball. While it is definitely wrong to place the points here, it risks being less wrong than placing the points at the top of the ball when they should have been placed at the bottom, or vice versa. However, because of the severe motion blurring, especially towards the end of the video, the position of the middle of the ball was difficult to judge, introducing random error into the data. The effects of this can be seen in the higher velocities, where the last four data points in Figure 2 show increasing inconsistency with the rest of the trend. In order to improve this, a camera with a higher frame rate and known shutter details should be used to minimise the amount of motion blur and so the points can be placed properly in Logger Pro.

Correct placement of the points is crucial. Although my lab partner, Classmate A, and I both used the same video, we did the Logger Pro analysis individually. Classmate A calculated a value for g of (9.84 ± 0.06) ms^{-2}. This is in good agreement with my value of (9.5 ± 0.4) ms^{-2}, but their value is both closer to the accepted value and more precise. The difference between our values is attributed only to the way we placed our points in the video analysis, showing that the human factor in this experiment does play a significant role.

References

[1] Isaac, I., *et al.* (2021). *Lab Manual PHYS 144*. Edmonton: University of Alberta, Department of Physics.

[2] Harris, C. R., Millman, K. J., van der Walt, S. J., *et al.* (2021, June 22). *numpy.polyfit*. NumPy. https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html

Acknowledgements

I acknowledge the help of Classmate A, with whom I collected the raw data and compared my final value of g.

Appendix A

The partial data set given in Table 1 is given in full in Table A1 below.

Table A1: The full position and velocity data for the tennis ball over time. The uncertainty in time is taken as the reciprocal of the frame rate of the camera, $f = 30$ fps, and the uncertainties in position and velocity are assumed to be negligible. The origin was placed at the point of release of the ball, with the $+\underline{\hat{y}}$-axis pointing downwards.

Time, t / s (± 0.03)	**Position, y / m**	**Velocity, v / ms^{-1}**
0.00	0.003	0.195
0.03	0.013	0.584
0.07	0.042	1.051
0.10	0.083	1.452
0.13	0.168	1.791
0.17	0.202	2.959
0.20	0.280	2.453
0.23	0.366	3.001
0.27	0.480	3.270
0.30	0.584	3.115
0.33	0.688	3.582
0.37	0.823	4.049
0.40	0.958	4.400
0.43	1.116	4.828
0.47	1.280	5.023
0.50	1.451	5.100
0.53	1.620	5.217
0.57	1.799	5.918

Appendix B

As shown in Figure B1, when the outlier at $t = 0.17$ s is removed from the analysis, the slope of the fit line becomes slightly steeper, leading to a higher g value and a lower intercept value. There is less variation between the data and the fit line now, leading to smaller uncertainties on both g and v_0.

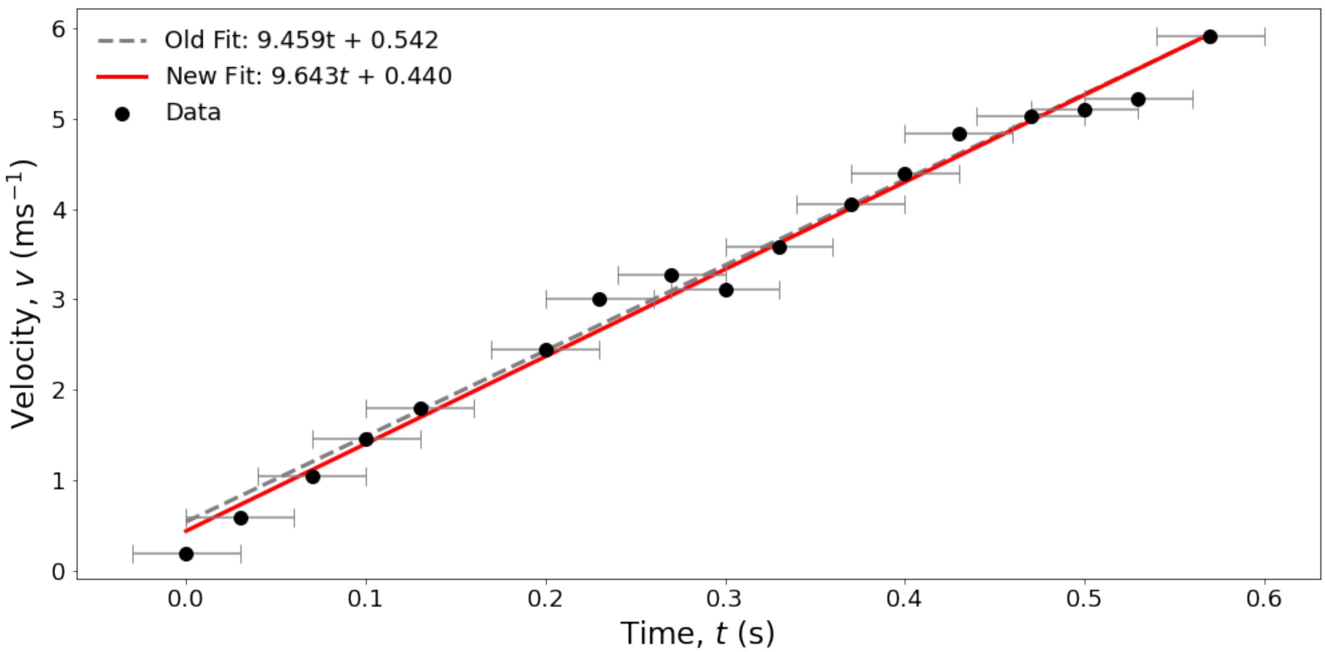

Figure B1: A plot of velocity vs time using the data from Table A1, but excluding the outlier at $t = 0.17$ s. Using `numpy.polyfit()`, the slope of the fit line was found to be (9.6 ± 0.3) ms^{-2}, and the intercept was found to be (0.44 ± 0.09) ms^{-1}. The fit including the outlier is represented by the grey dashed curve, and comes from Figure 2.