IB Math HL Mathematical Exploration

Calculating the volume of a plastic Coca-Cola bottle

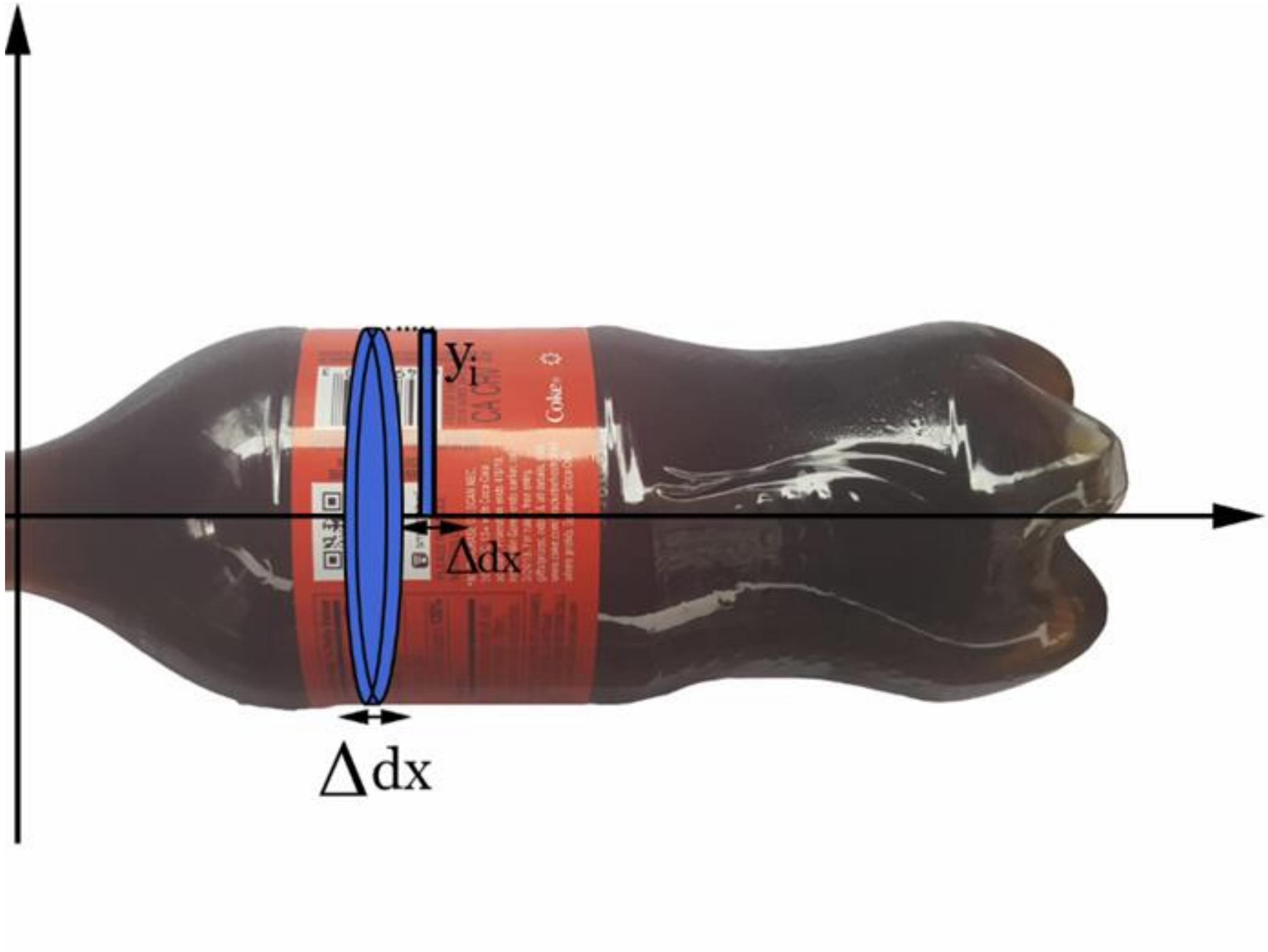

Table of Contents

Contents

Introduction

Rationale

I am not the type to grab a cup of soda over water, but I have the occasional soft drink when it's convenient for a craving. However, I notice the assortment of other soft drinks and juices, which all have unique designs plastered on standard bottle/can sizes. Despite this, I practice consumerism whenever I purchase a soft drink, as I sometimes I want to purchase an inexpensive drink with the most soda or juice. In other words, I prefer to get the most bang for my buck. I know, however, that these soft drinks are not filled to the brim, and I wonder why the allowance for such empty space when it can be replaced with soda sweetness. Further research shows that the bottles are not fully filled for good reason, which is the Carbon dioxide trapped in the bottle during the manufacturing process. If the soda was filled all the way, then the soda would burst from the pressurized Carbon dioxide[1]. Since Coca-Cola is a well-known brand of soda, then surely, they design their bottle with a purpose of aesthetic representation, as well as with an economic direction. As such, I believe there's more to the Coca-Cola bottle, which is how true is the label when it gives the bottles volume? Can the Coca-Cola bottle hold more soda if it was possible without bursting? A broad answer would be the investigation of the bottle with integration, more specifically with volumes of revolution. The area under the curve is beneficial mastering in order to ace exams, but this can be taken a step further with volumes of revolution, as the area becomes the volume that must be solved for the Coca-Cola bottle. Thus, I experience the accuracy and greatness of mathematics with functions and integration.

Aim

In my Math HL class, I learned the fundamentals of integration in order to find the integral of a function. If it's integrated in the interval [a,b] the area under the curve of the function is calculated. The skeletal formula is as follows when *f(x)* is the function:

$$A = \int_a^b f(x)dx$$

Later, I learned the application of integration, which is volumes of revolution. (also known as solids of revolution) When the function integrated in the interval [a, b] is rotated about the x-axis or y-axis, a 3-D solid is formed and the volume can be determined with the following formula:

$$V = \int_a^b \pi\left(f(x)\right)^2 dx$$

As such, the aim of my exploration is to derive the formula for volumes of revolution and then, apply it to the Coke bottle to illustrate the application of the formula with solids that deviate from normal functions, in order to find an approximated volume.

[1] Michelle Bryner, "Why Does Soda Fizz?," Live Science, last modified February 13, 2013, accessed March 25, 2019, https://www.livescience.com/32492-why-does-soda-fizz.html.

Investigating the formula for volumes of revolution

Suppose we are given a function of $y = f(x)$ shown in **Figure 1**[2].

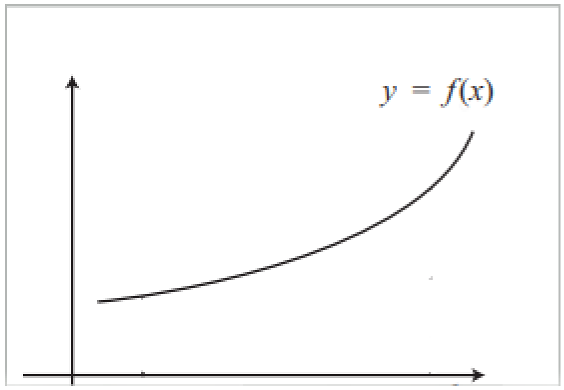

Figure 1: An example function of y = f(x)

A solid of revolution, or volumes of revolution begins with a function, and inquires the volume created must be due to the area with an additional dimension, such as depth or height. In this case, an interval [a, b] is established and an arbitrary section is taken as a thin approximation of the rectangle with a length of y_i and a width of Δx as shown in **Figure 2**[3].

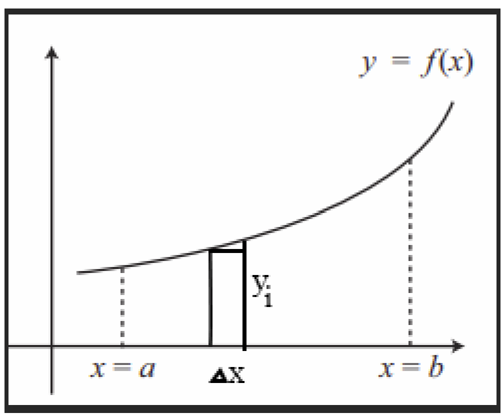

Figure 2: An example function of y = f(x)

Now we consider rotating this approximated rectangle around the x-axis and a cylindrical is formed; therefore, the length of y_i is equal to the radius of this cylinder [4]:

[2] "Volumes of solids of revolution," 2009, in *mathcentre*, 2, http://www.mathcentre.ac.uk/resources/uploaded/mc-ty-volumes-2009-1.pdf.

[3] "Volumes of solids," in *mathcentre*, 2.

[4] Morris Kline, Calculus: An Intuitive and Physical Approach, 2nd ed. (Mineola, NY: Dover Publications, 1998), 439.

Figure 3: A cylinder formed from rotation of the x-axis

This approximated area is now equal to the area of a cylinder, which is now represented as $\pi y_i^2 \Delta x$. With this as a basis for approximation of a volume, we can consider dividing the function into equal intervals of Δx within the interval [a, b].[5]

Thus, there are an n equal number of subintervals of width Δx, each with their own lengths of y_n. Now each of these are rotated around the x-axis, and is an approximation to the actual volumes of the function $y = f(x)$ when it's rotated around the x-axis as well.

The approximated volume can be represented with[6]:

$$\mathrm{V} = \pi y_1^2 \Delta x + \pi y_2^2 \Delta x + \pi y_3^2 \Delta x + \cdots + \pi y_n^2 \Delta x$$

However, a better approximation occurs with more equal n of subintervals, so Δx becomes smaller. Thus, desired volume is when the n number of subintervals approaches infinity as such to for better approximation of the volume, so that Δx approaches zero. As such, the volume of revolution is given by[7]:

$$V = \lim_{\Delta x \to 0} \sum_{x=b}^{x=b} \pi y^2 \Delta x$$

Thus, the definite integral is given for volumes of revolution:

$$V = \int \pi y^2 dx$$

[5] Kline, *Calculus: An Intuitive*, 439.
[6] Kline, *Calculus: An Intuitive*, 439.
[7] "Volumes of solids," in *mathcentre*, 3.

Graphing the Coca-Cola Bottle

To calculate the volume of the Coke bottle, I must find the functions that outline the various sections of the bottle, which the formula integrates. Before finding the functions, I found the circumference of the cylindrical section of the bottle**.** I used a piece of yarn to wrap around the bottle, and I measured this with a ruler to find the circumference of the cylindrical section. This initial step is important, for it sets the measurement for scaling the Coke bottle to the graph and scaling it as needed. Thus, I measured the circumference of the cylindrical section and used the circumference formula to find the radius.

$$Circumfrence = 2\pi r$$

In correspondence to the x-coordinate graph, the radius of the circle is given by:

$$r = \frac{Circumfrence}{2\pi}$$

The circumference for the cylindrical section gives a radius of:

$$r = \frac{23.6}{2\pi} \approx 3.8\ cm$$

The circumference and radius of the cylindrical section were recorded in **Table 1**.

	Circumference (cm)	Radius (cm)
Cylindrical section	23.6	3.8

Table 1: Circumference and Radius of the cylindrical section

The radius of the cylindrical section gives the foundation for measurement marks of the graph in relation to the scale of the Coke bottle. I used *Adobe Photoshop* to lay the photograph of the Coke bottle onto the background layer of a grid template found online. The photo opacity of the Coke bottle was reduced to #%, so that the grid lines would show clearly for the coordinates to be plotted accurately.

As such, the radius of the cylindrical section allowed me to scale the bottle as needed on the grid. To accomplish this, I set up the proportion below based on the specifications of the Coke bottle and the graph

Height of bottle = 22 cm

# of marks the height of the bottle reaches = 84

Number of total marks on the graph = 100

Thus, the interval length of a single mark on the graph is determined:

$$\frac{22\ cm}{84} = \frac{x}{100}$$

$$x \approx .262\ cm$$

This interval length was used to determine the coordinate points of the Coke bottle.

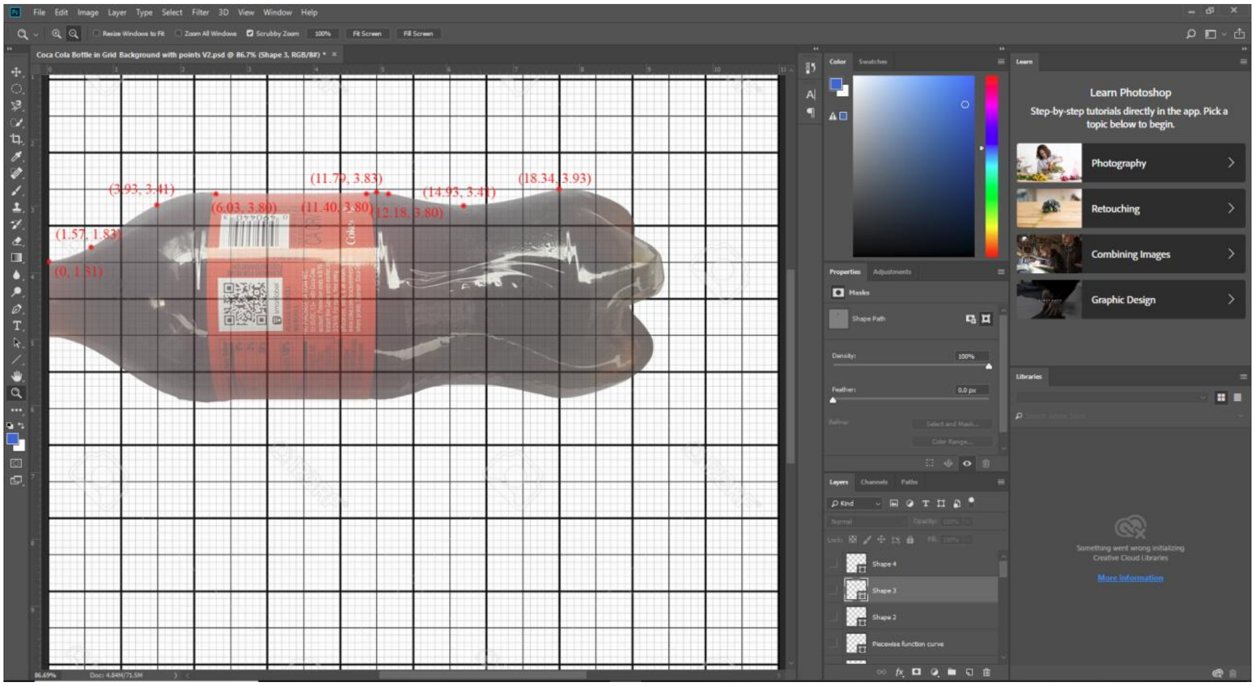

Figure 4: Plotting the graph for the Coke bottle in Adobe Photoshop

9 coordinate points were chosen to obtain a piecewise function, which will allow me to calculate the estimated volume of the Coke bottle before the Standing piece section. I chose these points based from the top to the bottom of the bottle where the shapes of the bottle curves form points of inflexion, minima, and maxima. The x-coordinate and y-coordinate points were plotted onto the graph of the Coke bottle as shown in **Figure 4**. The coordinates are listed in **Table 3**.

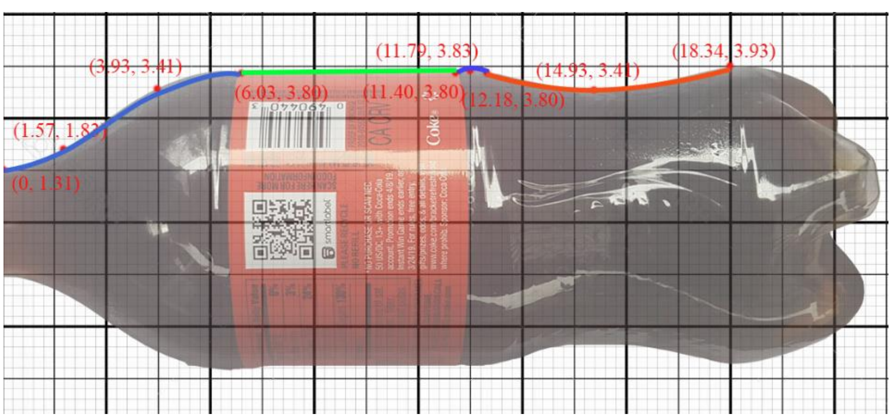

Figure 5: Graph of the Coke bottle

Coordinate	X	Y
1	0	1.31
2	1.57	1.83
3	3.93	3.41
4	6.03	3.80
5	11.40	3.80
6	11.79	3.83
7	12.18	3.80
8	14.93	3.41
9	18.34	3.93

Table 2: Coordinates of the Coke Bottle

Four different functions will be found to outline the Coke bottle. The bottle will be divided into sections as follows: Top, Middle, Bump, Bottom, and Standing Piece. For distinction, the Bottom is the last curvature section outlined in red, as shown in **Figure 5**, but the Standing Piece section is the bottom part of the bottle with the stubs not outlined.

The Standing piece is not outlined, for the stub will be cut out and graphed with its own points, as this section of the bottle will be considered the summation of the volumes of the stubs.

Finding the Functions

The four functions for the outlined sections were determined with the Lagrange interpolation formula. This formula was suitable, as it only applies to polynomials and all the data points are in unequal intervals.[8] In addition, I selected *n+1* data points based on the *nth* degree of the polynomial that was satisfactory, as the formula requires as well.[9] For instance, blue outline in **Figure 5** appears to be a cubic function; therefore, 4 data points were required. As such, the formula for a unique polynomial *P(x)* is written as[10]:

$$f(x) = \sum_{i=0}^{n} y_i \frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_1-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

Where $0 \leq i \leq n$ and $f(x_i) = y_i$

Top section:

The polynomial function for the Top section is found first. The shape of the curve appears to be a cubic function, for the graph shows a point of inflexion between the points (1.57, 1.83) and (3.93, 3.41). A cubic polynomial is a suitable choice, as the function shows opposite end behaviors; thus, a 3rd degree will be more simplistic, as a higher odd degree polynomial won't necessarily be a good fit. As such, 4 points were chosen. The 4 x-coordinates and y-coordinates are shown below:

Coordinates

i	x	y
0	0	1.31
1	1.57	1.83
2	3.93	3.41
3	6.03	3.80

Table 3: Coordinates for Top section

[8] DR. J.S.V.R. Krishna Prasad, "NUMERICAL METHODS; UNIT-II: INTERPOLATION," reading, in *NUMERICAL-ANALYSIS*, 1.

[9] "LECTURE 3 - LAGRANGE INTERPOLATION," in *CE30125* (University of Notre Dame, n.d.), 3.1, accessed March 24, 2019, https://coast.nd.edu/jjwteach/www/www/30125/pdfnotes/lecture3_6v13.pdf.

[10] "Lagrange Interpolation Formula," Art of Problem Solving, accessed March 29, 2019, https://artofproblemsolving.com/wiki/index.php/Lagrange_Interpolation_Formula.

As such, when i = 3 for a sequence of x-coordinate values {0, 1.57, 3.93, 6.03} the function $P_{1}(x)$ is given by:

$$P_1(x) = \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)}y_0 + \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)}y_1 + \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}y_2 + \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}y_3$$

$$= \frac{(x-1.57)(x-3.93)(x-6.03)}{(0-1.57)(0-3.93)(0-6.03)}(1.31) + \frac{(x-0)(x-3.93)(x-6.03)}{(1.57-0)(1.57-3.93)(1.57-6.03)}(1.83) + \frac{(x-0)(x-1.57)(x-6.03)}{(3.93-0)(3.93-1.57)(3.93-6.03)}(3.41) + \frac{(x-0)(x-1.57)(x-3.93)}{(6.03-0)(6.03-x_1)(6.03-x_2)}(3.80)$$

Then when it's expanded, we are given the function $P_{1}(x)$:

$$-0.0322632x^3 + 0.263524x^2 - 0.00299717x + 1.31$$

However, this form of the Lagrange interpolation is very tedious to expand algebraically; thus, I felt determining the coefficients would be more feasible with the fast processing of computer programming. As such, I used Python programming to determine the coefficients based on another form of the Lagrange interpolation formula. For instance, The Lagrange interpolation formula for a 3rd degree polynomial with the form $ax^3 + bx^2 + cx + d$[11]:

$$a = \frac{y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \frac{y_1}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + \frac{y_2}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \frac{y_3}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$$

$$b = \frac{-x_1y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \frac{-x_2y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \frac{-x_3y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \frac{-x_0y_1}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + \frac{-x_2y_1}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \frac{-x_3y_1}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} + \frac{-x_0y_2}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \frac{-x_1y_2}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \frac{-x_3y_2}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \frac{-x_0y_3}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} + \frac{-x_1y_3}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} + \frac{-x_2y_3}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$$

[11] Levy, Doron. "Introduction to numerical analysis." *Department of Mathematics and Center for Scientific Computation and Mathematical Modeling (CSCAMM) University of Maryland*(2010): 2-2., 33.

$$c = \frac{x_1 x_2 y_0}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} + \frac{x_1 x_3 y_0}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} + \frac{x_2 x_3 y_0}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} +$$
$$\frac{x_0 x_2 y_1}{(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)} + \frac{x_0 x_3 y_1}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} + \frac{x_2 x_3 y_1}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)} +$$
$$\frac{x_0 x_1 y_2}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} + \frac{x_0 x_3 y_2}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} + \frac{x_1 x_3 y_2}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} +$$
$$\frac{x_0 x_1 y_3}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)} + \frac{x_0 x_2 y_3}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)} + \frac{x_1 x_2 y_3}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)}$$

$$d = \frac{-x_1 x_2 x_3\, y_0}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)} + \frac{-x_0 x_2 x_3 y_1}{(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)} + \frac{-x_0 x_1 x_3 y_2}{(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)} +$$
$$\frac{-x_0 x_1 x_2 y_3}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)}$$

I then gave the x-coordinates and y-coordinates in substitution of the corresponding variables as my input data for the Python program. The code was then compiled and run, which I was given the values of *a, b, c, d* and is illustrated in **Figure 6**. The output values are precise up to 18 decimal points, which is more than precise for satisfaction, but I opted for 7 decimal places as more than enough for the precision required for the functions. In other words, the same polynomial function was given, though in a more calculative manner compared to the brute expansion of the formal Lagrange interpolation formula.

Lagrange Interpolation forumla calculations (Cubic).py

```python
#input coordinate points
x0 = float(0);
y0 = float(1.31);
x1 = float(1.57);
y1 = float(1.83);
x2 = float(3.93);
y2 = float(3.41);
x3 = float(6.03);
y3 = float(3.80);


# calculation with coordinate points

a = float(y0 / ((x0 - x1) * (x0 - x2) * (x0 - x3)) + y1 / ((x1 - x0) * (x1 - x2) * (x1 - x3)) + y2 / ((x2 -x0) * (x2 -x1) * (x2 -x3))
+ y3 / ((x3 - x0) * (x3 - x1) * (x3 - x2)));

b = float((-x1 * y0) / ((x0 - x1) * (x0 - x2) * (x0 - x3)) + (-x2 * y0) / ((x0 - x1) * (x0 - x2) * (x0 - x3)) + (-x3 * y0)
/ ((x0 - x1) * (x0 - x2) * (x0 - x3))
+ (-x0 * y1) / ((x1 - x0) * (x1 - x2) * (x1 - x3)) + (-x2 * y1) / ((x1 - x0) * (x1 - x2) * (x1 - x3)) + (-x3 * y1)
/ ((x1 - x0) * (x1 - x2) * (x1 - x3))
+ (-x0 * y2) / ((x2 -x0) * (x2 -x1) * (x2 -x3)) + (-x1 * y2) / ((x2 -x0) * (x2 -x1) * (x2 -x3)) + (-x3 * y2)
/ ((x2 -x0) * (x2 -x1) * (x2 -x3))
+ (-x0 * y3) / ((x3 - x0) * (x3 - x1) * (x3 - x2)) + (-x1 * y3) / ((x3 - x0) * (x3 - x1) * (x3 - x2)) + (-x2 * y3)
/ ((x3 - x0) * (x3 - x1) * (x3 - x2)));

c = float((x1* x2 * y0) / ((x0 - x1) * (x0 - x2) * (x0 - x3)) + (x1 * x3 * y0) / ((x0 - x1) * (x0 - x2) * (x0 - x3)) + (x2 * x3 * y0)
/ ((x0 - x1) * (x0 - x2) * (x0 - x3))
+ (x0 * x2 * y1) / ((x1 - x0) * (x1 - x2) * (x1 - x3)) + (x0 * x3 * y1) / ((x1 - x0) * (x1 - x2) * (x1 - x3)) + (x2 * x3 * y1)
/ ((x1 - x0) * (x1 - x2) * (x1 - x3))
+ (x0 * x1 * y2) / ((x2 -x0) * (x2 -x1) * (x2 -x3)) + (x0 * x3 * y2) / ((x2 -x0) * (x2 -x1) * (x2 -x3)) + (x1 * x3 * y2)
/ ((x2 -x0) * (x2 -x1) * (x2 -x3))
+ (x0 * x1 * y3) / ((x3 - x0) * (x3 - x1) * (x3 - x2)) + (x0 * x2 * y3) / ((x3 - x0) * (x3 - x1) * (x3 - x2)) + (x1 * x2 * y3)
/ ((x3 - x0) * (x3 - x1) * (x3 - x2)));

d = float((-x1 * x2 * x3 * y0) / ((x0 - x1) * (x0 - x2) * (x0 - x3))
+ (-x0 * x2 * x3 * y1) / ((x1 - x0) * (x1 - x2) * (x1 - x3))
+ (-x0 * x1 * x3 * y2) / ((x2 -x0) * (x2 -x1) * (x2 -x3))
+ (-x0 * x1 * x2 * y3) / ((x3 - x0) * (x3 - x1) * (x3 - x2)));

#display output
print ("the value of a is %f/n",a);
print ("the value of b is %f/n",b);
print ("the value of c is %f/n",c);
print ("the value of d is %f/n",d);
```

IPython console

Console 1/A

```
In [16]: runfile('C:/Users/antho/.spyder-py3/Math/Lagrange Interpolation forumla calculations (Cubic).py', wdir='C:/Users/antho/.spyder-py3/Math')
the value of a is %f/n -0.03226316890937396
the value of b is %f/n 0.2635241044063725
the value of c is %f/n -0.002997167790486199 3
the value of d is %f/n 1.31
```

Figure 6: Screenshot of code and its compiled output

Middle section:

I applied the same method for finding the polynomial for the Top section to find the function of Middle section. However, this section is a straight line formed between the points (6.03, 3.80) and (11.40, 3,80). As such, the slope is zero and only requires two points for the Lagrange interpolation formula. Despite the case the function is equal to 3.80, I'll still use the Lagrange interpolation formula for congruency in treating this section as a volume of revolution. The 2 x-coordinates and y-coordinates are shown below:

Coordinates

i	x	y
0	6.03	3.80
1	11.40	3.80

Table 4: Coordinates for Middle section

The Lagrange interpolation formula for a 1st degree polynomial with the form ax + b:

$$a = \frac{y_0}{(x_0 - x_1)} + \frac{y_1}{(x_1 - x_0)}$$

$$b = \frac{-x_1 y_0}{(x_0 - x_1)} + \frac{-x_0 y_1}{(x_1 - x_0)}$$

I utilized the Lagrange interpolation formula and used the HP Prime Graphical Display Calculator (GDC) to expand the algebraic expressions and obtained the polynomial:

$$P_{2\,(x)} = 0x + 3.80$$

Bump section

I applied the same method for finding the polynomial for the Top section to find the function of Bump section. The shape of the curve appears to be a quadratic function, for the graph shows a maximum at approximately (11.79, 3.83). Thus, 3 points were chosen. The 3 x-coordinates and y-coordinates are shown below:

Coordinates

i	x	y
0	11.40	3.80
1	11.79	3.83
2	12.18	3.80

Table 5: Coordinates for Bump section

The Lagrange interpolation formula for a 2nd degree polynomial with the form $ax^2 + bx + c$:

$$a = \frac{y_0}{(x_0 - x_1)(x_0 - x_2)} + \frac{y_1}{(x_1 - x_0)(x_1 - x_2)} + \frac{y_2}{(x_2 - x_0)(x_2 - x_1)}$$

$$b = \frac{-x_2 y_0}{(x_0 - x_1)(x_0 - x_2)} + \frac{-x_1 y_0}{(x_0 - x_1)(x_0 - x_2)} + \frac{-x_2 y_1}{(x_1 - x_0)(x_1 - x_2)} + \frac{-x_0 y_1}{(x_1 - x_0)(x_1 - x_2)} + \frac{-x_1 y_2}{(x_2 - x_0)(x_2 - x_1)} + \frac{-x_0 y_2}{(x_2 - x_0)(x_2 - x_1)}$$

$$c = \frac{x_1 x_2 y_0}{(x_0 - x_1)(x_0 - x_2)} + \frac{x_0 x_2 y_1}{(x_1 - x_0)(x_1 - x_2)} + \frac{x_0 x_1 y_2}{(x_2 - x_0)(x_2 - x_1)}$$

I utilized the Lagrange interpolation formula and used Python to find the coefficients and obtained the polynomial:

$$P_{3\,(x)} = -0.19723866x^2 + 4.6508876x - 23.5869822$$

Lagrange Interpolation forumla calculations (Quadratic).py

```python
#input coordinate points
x0 = float(11.40);
y0 = float(3.80);
x1 = float(11.79);
y1 = float(3.83);
x2 = float(12.18);
y2 = float(3.80);

# calculation with coordinate points
a = float(y0 / ((x0 - x1) *(x0-x2)) + y1 / ((x1 - x0) * (x1 - x2)) + y2 / ((x2 - x0) * (x2 - x1)));
b = float((-x2 * y0) / ((x0 - x1)*(x0 - x2)) + (-x1 * y0) / ((x0 - x1) * (x0 - x2)) +(-x2 * y1)
/ ((x1 - x0) * (x1 - x2)) +(-x0 * y1) / ((x1 - x0) * (x1 - x2)) + (-x1 * y2) / ((x2 - x0) * (x2 - x1)) + (-x0 * y2)
/ ((x2 - x0) * (x2 - x1)));
c = float((x1 * x2 * y0) / ((x0 -x1) * (x0 - x2)) + (x0 * x2 * y1) / ((x1 - x0) * (x1 -x2)) + (x0 * x1 * y2)
/ ((x2 - x0) * (x2 -x1)));

#display output
print ("the value of a is %f/n",a);
print ("the value of b is %f/n",b);
print ("the value of c is %f/n",c);
```

IPython console

Console 1/A

```
In [17]: runfile('C:/Users/antho/.spyder-py3/Math/Lagrange Interpolation forumla calculations (Quadratic).py', wdir='C:/Users/antho/.spyder-py3/Math')
the value of a is %f/n -0.19723865877712043
the value of b is %f/n 4.650887573964468
the value of c is %f/n -23.586982248521053
```

Figure 7: Screenshot of code and its compiled output

Bottom section

I applied the same method for finding the polynomial for the Bump section to find the function of the Bottom section. The shape of the curve appears to be a quadratic function, for the graph shows a minimum at approximately (14.93, 3.41). Thus, 3 points were chosen. The 3 x-coordinates and y-coordinates are shown below:

Coordinates

i	x	y
0	12.18	3.80
1	14.93	3.41

2	18.34	3.93

Table 6: Coordinates for Bump section

Since the Bottom section models a Quadratic function, the same Lagrange interpolation formula was used for the Bump section.

I utilized the Lagrange interpolation formula and used Python to find the coefficients and obtained the polynomial:

$$P_{4\,(x)} = 0.0401618x^2 - 1.2306055x + 12.8306704$$

The graph for the Coke bottle can be represented by the piecewise function:

$$f(x) = \begin{cases} -0.0322632x^3 + 0.263524x^2 - 0.00299717x + 1.31 & 0 \leq x \leq 6.03 \\ 0x + 3.80 & 6.03 \leq x \leq 11.40 \\ -0.19723866x^2 + 4.6508876x - 23.5869822 & 11.40 \leq x \leq 12.18 \\ 0.0401618x^2 - 1.2306055x + 12.8306704 & 12.18 \leq x \leq 18.34 \end{cases}$$

The piecewise functions were graphed using the Desmos online graph calculator: https://www.desmos.com/calculator

The four functions were superimposed on onto the image of the Coke bottle with *Adobe Photoshop*. This is shown below in **Figure 8**. The green curve traced on the Coke bottle represents the piecewise function determined above. As shown, the green curve models the shape of the Coke bottle as close as possible up until the bottom indentures begin. Thus, the mixture of a 3rd, 1st, and 2nd degree polynomials were suitable, rather than using only 1st and 2nd degree polynomials for simplicity.

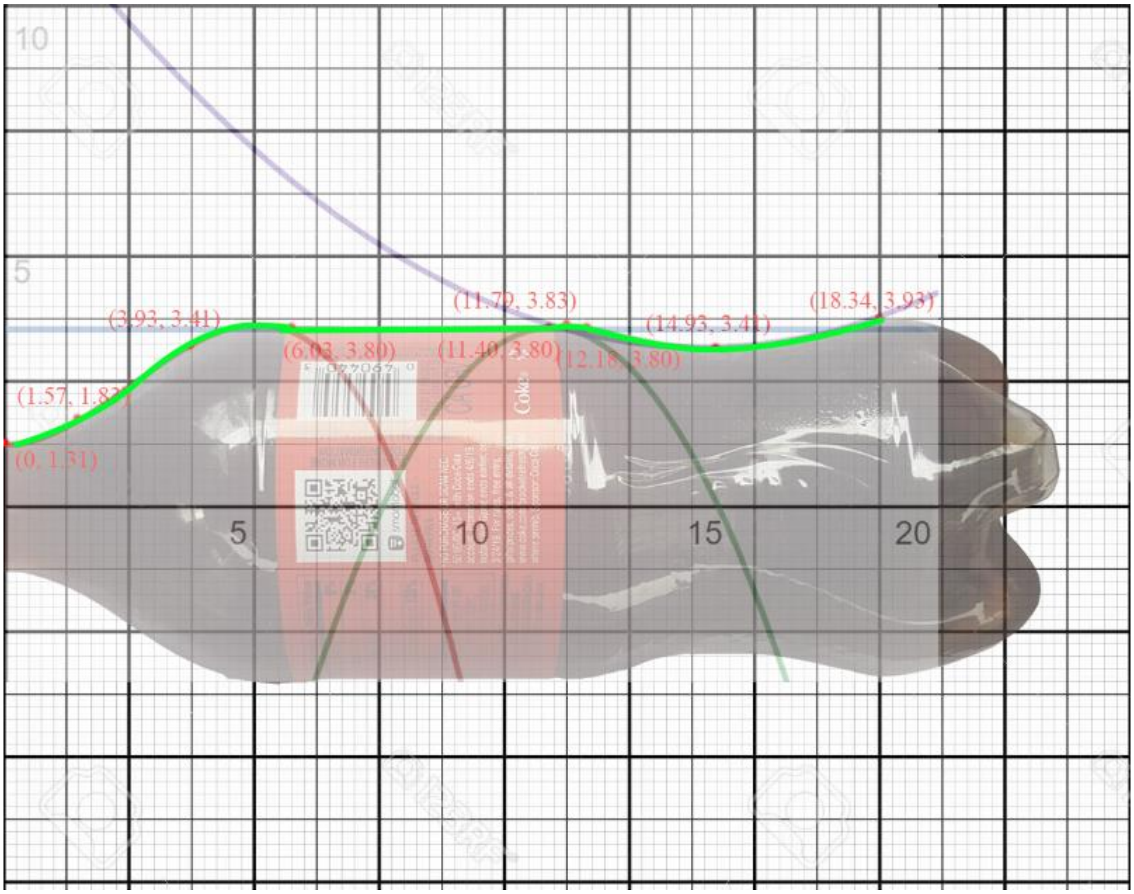

Figure 8: Superimposed image of the polynomial functions on the Coke bottle

Standing Piece section

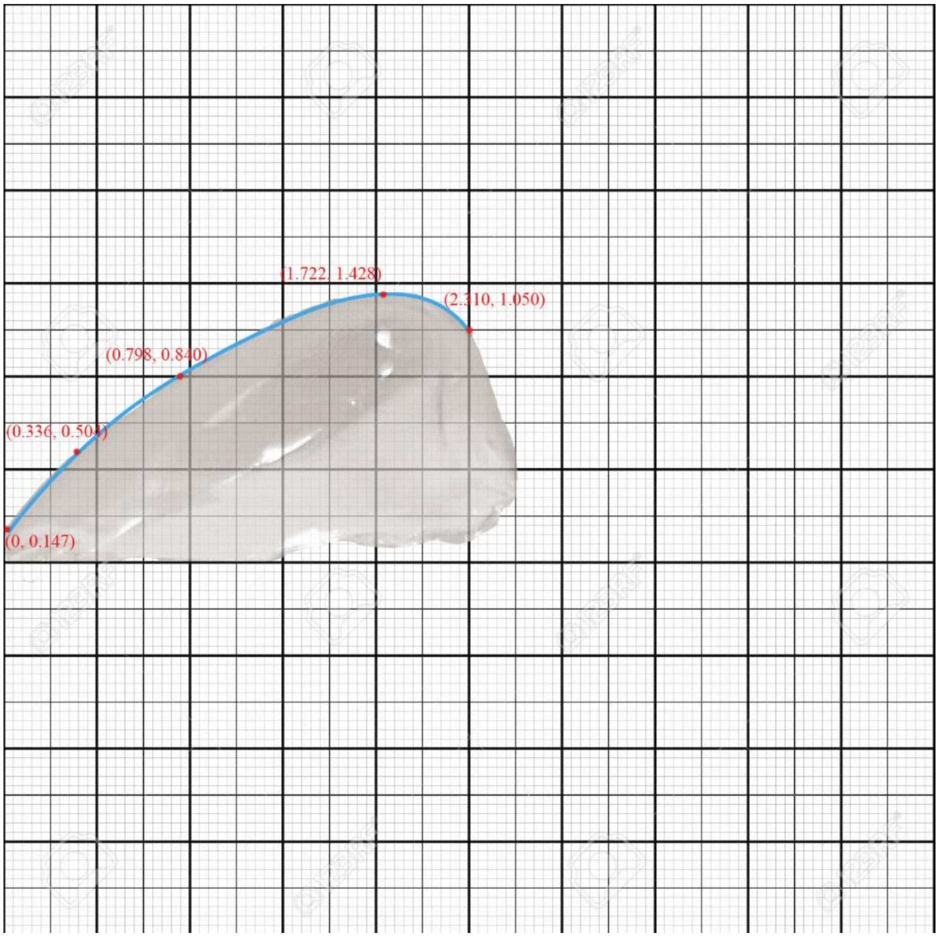

Figure 9: Graph of the Standing Piece section

I applied the same method for finding the polynomial for the other sections of the Coke bottle to find the function of the Standing Piece section. Unlike the other sections of the Coke bottle, the Standing Piece shows an uneven curvature, which a polynomial degree less than 4 would not be suitable for outlining it. As such, a 4^{th} degree polynomial, or Quartic function, is more suitable compared to a Quadratic, as there is no general symmetry[12] with a maximum at approximately (1.722, 1.428). Again, suitability is chosen over accuracy, as a higher degree polynomial won't necessarily fit with the piece, especially when measurements are less than 3 cm. Therefore, 5 points were chosen for the Lagrange interpolation formula. The 5 x-coordinates and y-coordinates are shown below:

Coordinates

i	x	y
0	0	0.147

[12] Jon Davidson, "Fourth Degree Polynomials," last modified 1998, accessed March 30, 2019, https://www.sscc.edu/home/jdavidso/math/catalog/polynomials/fourth/fourth.html.

1	0.336	0.504
2	0.798	0.840
3	1.722	1.428
4	2.310	1.050

Table 7: Coordinates for the Standing Piece section

The Lagrange interpolation formula for a 4th degree polynomial with the form $ax^4 + bx^3 + cx^2 + dx + e$:

$$a = \frac{y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \frac{y_1}{(x_1-x_0)(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{y_2}{(x_2-x_0)(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{y_3}{(x_3-x_0)(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{y_4}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

$$b = \frac{-y_0(x_1+x_2+x_3+x_4)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \frac{-y_1(x_0+x_2+x_3+x_4)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{-y_2(x_0+x_1+x_3+x_4)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{-y_3(x_0+x_1+x_2+x_4)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{-y_4(x_0+x_1+x_2+x_3)}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

$$c = \frac{y_0(x_1x_2+x_1x_3+x_1x_4+x_2x_3+x_2x_4+x_3x_4)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \frac{y_1(x_0x_2+x_0x_3+x_0x_4+x_2x_3+x_2x_4+x_3x_4)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{y_2(x_0x_1+x_0x_3+x_0x_4+x_1x_3+x_1x_4+x_3x_4)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{y_3(x_0x_1+x_0x_2+x_0x_4+x_1x_2+x_1x_4+x_2x_4)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{y_4(x_0x_1+x_0x_2+x_0x_3+x_1x_2+x_1x_3+x_2x_3)}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

$$d = \frac{-y_0(x_1x_2x_3+x_1x_2x_4+x_1x_3x_4+x_2x_3x_4)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \frac{-y_1(x_0x_2x_3+x_0x_2x_4+x_0x_3x_4+x_2x_3x_4)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{-y_2(x_0x_1x_3+x_0x_1x_4+x_0x_3x_4+x_1x_3x_4)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{-y_3(x_0x_1x_2+x_0x_1x_4+x_0x_2x_4+x_1x_2x_4)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{-y_4(x_0x_1x_2+x_0x_1x_3+x_0x_2x_3+x_1x_2x_3)}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

$$e = \frac{y_0x_1x_2x_3x_4}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \frac{y_1x_0x_2x_3x_4}{(x_1-x_0)(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{y_2x_0x_1x_3x_4}{(x_2-x_0)(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{y_3x_0x_1x_2x_4}{(x_3-x_0)(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{y_4x_0x_1x_2x_3}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

Note: I chose to algebraically manipulate as needed for the Lagrange interpolation formula, as there would've been more terms to deal with in the programming; thus, made it succinct.

I utilized the Lagrange interpolation formula and used Python to find the coefficients and obtained the polynomial:

$$P_{5}(x) = -0.260272x^4 + 0.949198x^3 - 1.23156x^2 + 1.37902x + 0.147$$

Lagrange Interpolation forumla calculations (4th power).py

```python
#input coordinate points
x0 = float(0);
y0 = float(0.147);
x1 = float(0.336);
y1 = float(0.504);
x2 = float(0.798);
y2 = float(0.840);
x3 = float(1.722);
y3 = float(1.428);
x4 = float(2.310);
y4 = float(1.050);

# calculation with coordinate points

a = float(y0 / ((x0 -x1) * (x0 -x2) * (x0 - x3) * (x0 - x4)) + y1 / ((x1 -x0) * (x1 -x2) * (x1 - x3) * (x1 - x4))
+ y2 / ((x2 -x0) * (x2 -x1) * (x2 - x3) * (x2 - x4)) + y3 / ((x3 -x0) * (x3 -x1) * (x3 - x2) * (x3 - x4))
+ y4 / ((x4 -x0) * (x4 -x1) * (x4 - x2) * (x4 - x3)));

b = float((y0 * (-x1 - x2 - x3 - x4)) / ((x0 -x1) * (x0 -x2) * (x0 - x3) * (x0 - x4))
+ (y1 * (-x0 - x2 - x3 - x4)) / ((x1 -x0) * (x1 -x2) * (x1 - x3) * (x1 - x4))
+ (y2 * (-x0 - x1 - x3 - x4)) / ((x2 -x0) * (x2 -x1) * (x2 - x3) * (x2 - x4))
+ (y3 * (-x0 - x1 - x2 - x4)) / ((x3 -x0) * (x3 -x1) * (x3 - x2) * (x3 - x4))
+ (y4 * (-x0 - x1 - x2 - x3)) / ((x4 -x0) * (x4 -x1) * (x4 - x2) * (x4 - x3)));

c = float((y0 * (x1 *x2 + x1 * x3 + x1 * x4 + x2 * x3 + x2 * x4 + x3 * x4)) / ((x0 -x1) * (x0 -x2) * (x0 - x3) * (x0 - x4))
+ (y1 * (x0 *x2 + x0 * x3 + x0 * x4 + x2 * x3 + x2 * x4 + x3 * x4)) / ((x1 -x0) * (x1 -x2) * (x1 - x3) * (x1 - x4))
+ (y2 * (x0 *x1 + x0 * x3 + x0 * x4 + x1 * x3 + x1 * x4 + x3 * x4)) / ((x2 -x0) * (x2 -x1) * (x2 - x3) * (x2 - x4))
+ (y3 * (x0 *x1 + x0 * x2 + x0 * x4 + x1 * x2 + x1 * x4 + x2 * x4)) / ((x3 -x0) * (x3 -x1) * (x3 - x2) * (x3 - x4))
+ (y4 * (x0 *x1 + x0 * x2 + x0 * x3 + x1 * x2 + x1 * x3 + x2 * x3)) / ((x4 -x0) * (x4 -x1) * (x4 - x2) * (x4 - x3)));

d = float((-y0 * (x1 * x2 * x3 + x1 * x2 * x4 + x2 * x3 * x4 + x1 * x3 * x4)) / ((x0 -x1) * (x0 -x2) * (x0 - x3) * (x0 - x4))
+ (-y1 * (x0 * x2 * x3 + x0 * x2 * x4 + x2 * x3 * x4 + x0 * x3 * x4)) / ((x1 -x0) * (x1 -x2) * (x1 - x3) * (x1 - x4))
+ (-y2 * (x0 * x1 * x3 + x0 * x1 * x4 + x1 * x3 * x4 + x0 * x3 * x4)) / ((x2 -x0) * (x2 -x1) * (x2 - x3) * (x2 - x4))
+ (-y3 * (x0 * x1 * x2 + x0 * x1 * x4 + x1 * x2 * x4 + x0 * x2 * x4)) / ((x3 -x0) * (x3 -x1) * (x3 - x2) * (x3 - x4))
+ (-y4 * (x0 * x1 * x2 + x0 * x1 * x3 + x0 * x2 * x3 + x1 * x2 * x3)) / ((x4 -x0) * (x4 -x1) * (x4 - x2) * (x4 - x3)));

e = float(y0 * x1 * x2 * x3 * x4 / ((x0 -x1) * (x0 -x2) * (x0 - x3) * (x0 - x4))
+ y1 * x0 * x2 * x3 * x4 / ((x1 -x0) * (x1 -x2) * (x1 - x3) * (x1 - x4))
+ y2 * x0 * x1 * x3 * x4 / ((x2 -x0) * (x2 -x1) * (x2 - x3) * (x2 - x4))
+ y3 * x0 * x1 * x2 * x4 / ((x3 -x0) * (x3 -x1) * (x3 - x2) * (x3 - x4))
+ y4 * x0 * x1 * x2 * x3 / ((x4 -x0) * (x4 -x1) * (x4 - x2) * (x4 - x3))
);

#display output
print ("the value of a is %f/n",a);
print ("the value of b is %f/n",b);
print ("the value of c is %f/n",c);
print ("the value of d is %f/n",d);
print ("the value of e is %f/n",e);
```

IPython console

Console 2/A

```
the value of a is %f/n -0.26027204600132686
the value of b is %f/n 0.9491983611622309
the value of c is %f/n -1.2315630671837838
the value of d is %f/n 1.3790174064881828
the value of e is %f/n 0.147
```

Figure 10: Screenshot of code for a 4^{th} degree polynomial and its compiled

The graph for the Standing Piece section of the coke bottle can be approximately estimated by the function:

$$f(x) = -0.260272x^4 + 0.949198x^3 - 1.23156x^2 + 1.37902x + 0.147 \qquad 0 \le x \le 2.310$$

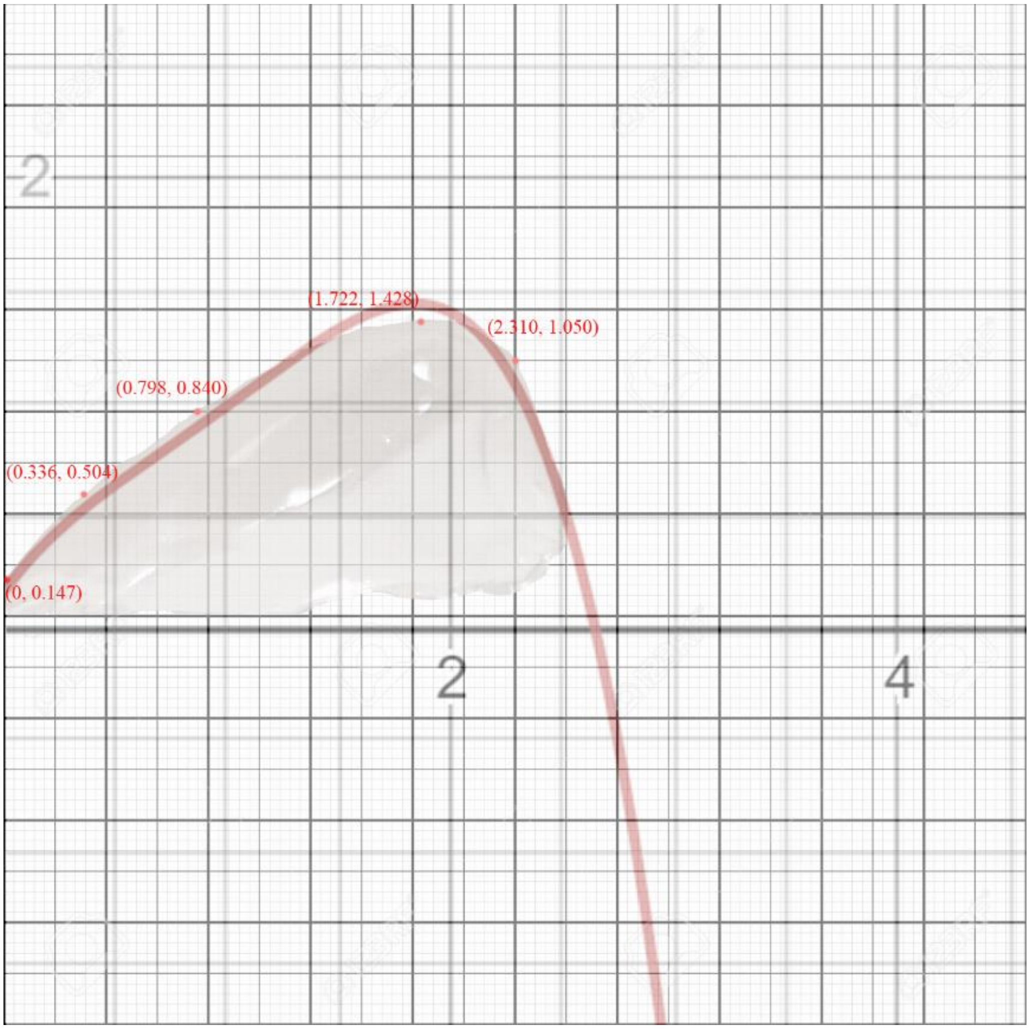

Figure 11: Superimposed image of the polynomial on the piece section of the Coke

Calculating the Volume of the Coca-Cola Bottle

With the collected piecewise functions, the volume for the Coke bottle was determined since the formula for the volume of revolution I derived is given by $V = \int \pi y^2 dx$.

Top section:

To find the volume of revolution for the Top section, the equation of the function $P_{1}(x) = -0.0322632x^3 + 0.263524x^2 - 0.00299717x + 1.31$ must be substituted for y. Additionally, the volume of revolution will is in the interval [0, 6.03].

$$\int_{0}^{6.03} \pi y^2 dx$$

y is substituted with $P_{1\,(x)} = -0.0322632x^3 + 0.263524x^2 - 0.00299717x + 1.31$:

$$y = -0.0322632x^3 + 0.263524x^2 - 0.00299717x + 1.31$$

Then y is squared:

$$y^2 = -0.0010409x^6 - 0.0170004x^5 + 0.0696382x^4 - 0.0861092x^3 + 0.6904329x^2 - 0.0078526x + 1.716100$$

Since π is an irrational number it can be multiplied as a constant to the integral of the polynomial function squared:

$$\pi\int_0^{6.03}(-0.0010409x^6 - 0.0170004x^5 + 0.0696382x^4 - 0.0861092x^3 + 0.6904329x^2 - 0.0078526x + 1.716100)\,dx$$

$$\pi\left(0.0001487x^7 - 0.0028340x^6 + 0.0139277x^5 - 0.0215273x^4 + 0.2301443x^3 - 0.0039263x^2 + 1.7161x \Big]_0^{6.03}\right)$$

$$\pi\left[\left(0.0001487(6.03)^7 - 0.0028340(6.03)^6 + 0.0139277(6.03)^5 - 0.0215273(6.03)^4 + 0.2301443(6.03)^3 - 0.0039263(6.03)^2 + 1.7161(6.03)\right) - 0\right]$$

Thus, I'm given:

$V_T = 92.396$

≈ 92.4 cm^3

Middle section:

To find the volume of revolution for the Middle section, the equation of the function $P_{2\,(x)} = 0x + 3.80$ must be substituted for y. Additionally, the volume of revolution will is in the interval [6.03, 11.40].

$$V_m = \int_{6.03}^{11.40} \pi(0x + 3.80)^2 dx$$

I then used the HP Prime calculator to evaluate V_m. Thus, I'm given:

$V_m = 243.608$ cm^3

≈ 243.6 cm^3

Bump section:

To find the volume of revolution for the Bump section, the equation of the function $P_{3\,(x)} = -0.19723866x^2 + 4.6508876x - 23.5869822$ must be substituted for y. Additionally, the volume of revolution will is in the interval [11.40, 12.18].

$$V_{Bu} = \int_{11.40}^{12.18} \pi(-0.19723866x^2 + 4.6508876x - 23.5869822)^2 dx$$

I then used the HP Prime calculator to evaluate V_{Bu}. Thus, I'm given:

$$V_{Bu} = 35.758 \text{ cm}^3$$

$$\approx 35.8 \text{ cm}^3$$

Bottom section:

To find the volume of revolution for the Bottom section, the equation of the function $P_{4\,(x)} = 0.0401618x^2 - 1.2306055x + 12.8306704$ must be substituted for y. Additionally, the volume of revolution will is in the interval [12.18, 18.34].

$$V_B = \int_{12.18}^{18.34} \pi(0.0401618x^2 - 1.2306055x + 12.8306704)^2 dx$$

I then used the HP Prime calculator to evaluate V_B. Thus, I'm given:

$$V_B = 241.535 \text{ cm}^3$$

$$\approx 241.5 \text{ cm}^3$$

Standing Piece section:

To find the volume of revolution for the Piece section, the equation of the function $P_{5\,(x)} = -0.260272x^4 + 0.949198x^3 - 1.23156x^2 + 1.37902x + 0.147$ must be substituted for y. Additionally, the volume of revolution will is in the interval [0, 2.310].

$$V_P = \int_{0}^{2.310} \pi(-0.260272x^4 + 0.949198x^3 - 1.23156x^2 + 1.37902x + 0.147)^2 dx$$

I then used the HP Prime calculator to evaluate for the Standing Piece section. Thus, I'm given:

$$8.190 \text{ cm}^3$$

$$\approx 8.2 \text{ cm}^3$$

However, its important to note that the Coke bottle has 5 Standing Piece sections; thus, the volume calculated must be multiplied by 5:

$$(8.2 \text{ cm}^3) * 5$$

$$V_P = 41 \text{ cm}^3$$

Total Volume:

The total volume for the Coke bottle can be given by the sum of the volumes of the Top, Middle, Bump, Bottom section and Piece sections. Thus, I'm given:

$$V_{Total} = V_T + V_M + V_{Bu} + V_B + V_P$$

$$= 92.4 + 243.6 + 35.8 + 241.5 + 41$$

$$= 654.3 \text{ cm}^3$$

Thus, the total volume for the Coke bottle can be approximated as 654.3 cm^3

Percent error:

The percentage error is an important aspect of measuring quantities, as it gives a confidence for how far off the investigated value is from the exact value.

First, the volume of the Coke bottle is established given by the information of the Coca-Cola company[13]:

$$\text{Volume given} = 20 \text{ fl oz}$$

An online measurement converter was used to convert this measurement to milters[14]:

$$\text{Volumein in militers} = 591.4$$

The converted volume will be the theoretical value in with the formula for percent calculation error as follows[15]:

$$\frac{|value_{experimental} - value_{theoretical}|}{value_{theoretical}} * 100 = \%error$$

$$\frac{|654.3 - 591.4|}{591.4} * 100 = 10.6\%$$

Conclusion

Despite the small size the Coca-Cola bottle is, it was still a worthy feat of sweet terror with the math that followed. For one, the derivation was a conceptual food to digest in regards to its application, as normally its taught under the conditions of functions common to a student mathematician; therefore, the study of volumes of revolution happens in a step process. However, the Coke bottle did not match a single or smooth function that could be easily plugged in the formula for volumes of revolution. Instead, I decided to undertake the reverse process of finding the functions myself, for the mathematical functions I daily learn in math class would be very inaccurate considering the symmetry and curvature that outlines the bottle itself. Thus, I had to learn the Lagrange interpolation formula, which was a knew mathematical concept I developed confidence to learn it. Initially, I was intimidated by the various "x" variables, subscripts, and conditions that followed with the formula, but I knew I could only see behind this mental façade through self-learning. As such, I watched videos online and various online sources, both educational and informal, in order to understand the pattern and formation of the Lagrange interpolation. Knowing this pattern and having an understanding of this formula allowed me to confidently utilize the coordinate points I plotted and couple it with the computational prowess of programming to relieve myself from the brute tactic algebraic expansion. Thus, the formula for volumes of revolution showed its strength when the *"y"* was all that was required for me to solve prior to integration. The formula took care of itself when I saw the volumes of the different sections of the Coke bottle come together and, ultimately, come with a close approximation to the official volume of the bottle. Despite the 10% error

[13] Coca-Cola, "Coca-Cola - 20 oz | Coca- Cola Product Facts," Coca-Cola Product Information & FAQs | Coca-Cola Product Facts, accessed April 1, 2019, https://www.coca-colaproductfacts.com/en/products/coca-cola/original/20-oz/.

[14] "Convert oz to ml - Conversion of Measurement Units," Convert Units - Measurement Unit Converter, accessed April 1, 2019, https://www.convertunits.com/from/oz/to/ml.

[15] "Percent Error Formula," Department of Astronomy and Physics | University of Iowa, last modified 2017, accessed March 25, 2019, http://astro.physics.uiowa.edu/ITU/glossary/percent-error-formula/.

in my calculations, an argument can be made with the fact that the amount of soda I see in the bottle on the shelf and the volume the label says don't necessarily pair up with one another. In other words, the label may be underestimating the true potential in regards to the volume a Coca-Cola bottle can hold; thus, may be overtaken by the superpower of mathematical interpolation and integration.

Bibliography

Bryner, Michelle. "Why Does Soda Fizz?" Live Science. Last modified February 13, 2013. Accessed March 25, 2019. https://www.livescience.com/32492-why-does-soda-fizz.html.

Coca-Cola. "Coca-Cola - 20 oz | Coca- Cola Product Facts." Coca-Cola Product Information & FAQs | Coca-Cola Product Facts. Accessed April 1, 2019. https://www.coca-colaproductfacts.com/en/products/coca-cola/original/20-oz/.

"Convert oz to ml - Conversion of Measurement Units." Convert Units - Measurement Unit Converter. Accessed April 1, 2019. https://www.convertunits.com/from/oz/to/ml.

Davidson, Jon. "Fourth Degree Polynomials." Last modified 1998. Accessed March 30, 2019. https://www.sscc.edu/home/jdavidso/math/catalog/polynomials/fourth/fourth.html.

Kline, Morris. *Calculus: An Intuitive and Physical Approach*. 2nd ed. Mineola, NY: Dover Publications, 1998.

Kumar, M. Ramesh. *New Formulas and Methods for Interpolation, Numerical Differentiation and Numerical Integration*. https://arxiv.org/ftp/arxiv/papers/0809/0809.0465.pdf.

"Lagrange Interpolation Formula." Art of Problem Solving. Accessed March 29, 2019. https://artofproblemsolving.com/wiki/index.php/Lagrange_Interpolation_Formula.

"LECTURE 3 - LAGRANGE INTERPOLATION." In *CE30125*. University of Notre Dame, n.d. Accessed March 24,

2019. https://coast.nd.edu/jjwteach/www/www/30125/pdfnotes/lecture3_6v13.pdf.

Levy, Doron. "Introduction to numerical analysis." *Department of Mathematics and Center for Scientific Computation and Mathematical Modeling (CSCAMM) University of Maryland*(2010): 2-2.

"Percent Error Formula." Department of Astronomy and Physics | University of Iowa. Last modified 2017. Accessed March 25, 2019. http://astro.physics.uiowa.edu/ITU/glossary/percent-error-formula/.

Prasad, DR. J.S.V.R. Krishna. "NUMERICAL METHODS; UNIT-II: INTERPOLATION." Reading. In *NUMERICAL-ANALYSIS*.

"Volumes of solids of revolution." 2009. In *mathcentre*, 1-7. http://www.mathcentre.ac.uk/resources/uploaded/mc-ty-volumes-2009-1.pdf.