Project 2

Realistic Projectiles

Matthew Lopez Tarsky
[Redacted Member #1]
[Redacted Member #2]

ENGR 285

Objective #1: The interdependence of the horizontal and vertical motion (i.e. that the horizontal and vertical motion are not independent)

In order to to see how how horizontal motion and vertical motion aren't exactly independent from a theoretical perspective, we first have have to derive the a system of equations for the differential equation:

$$\frac{d^2\hat{r}}{dt^2} = \hat{g} - f(v)\frac{\hat{v}}{v}$$

$$Note: \hat{r},\ \hat{g}, and\ \hat{v}\ are\ vectors,\ not\ unit\ vectors$$

For most air resistance models, it uses a quadratic drag to model air resistance. This drag that we know is described using the function:

$$f(v) = kv^2$$

So our differential model now looks more like this:

$$\frac{d^2\hat{r}}{dt^2} = \hat{g} - kv \cdot \hat{v}$$

We can now "split" the equation into their respective x and y direction formulas:

$$\frac{d^2x}{dt^2} = -\ kv \cdot v_x \qquad\qquad \frac{d^2y}{dt^2} = -\ g\ -\ kv \cdot v_y$$

Knowing that speed is the magnitude of the velocity vector:

$$v = \sqrt{v_x^2 + v_y^2}$$

Along with knowing that:

$$\frac{d^2x}{dt^2} = \frac{dv_x}{dt} \qquad\qquad \frac{d^2y}{dt^2} = \frac{dv_y}{dt}$$

We now have our full and drawn out system of equations:

$$\frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

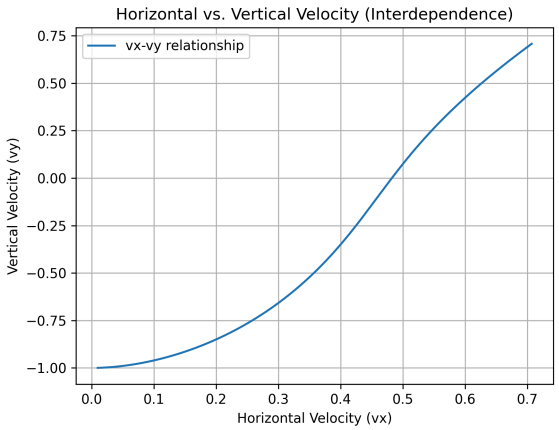
$$\frac{dv_x}{dt} = -\,k \cdot v_x\sqrt{v_x^2 + v_y^2} \qquad \frac{dv_y}{dt} = -\,g\; -\,k \cdot v_y\sqrt{v_x^2 + v_y^2}$$

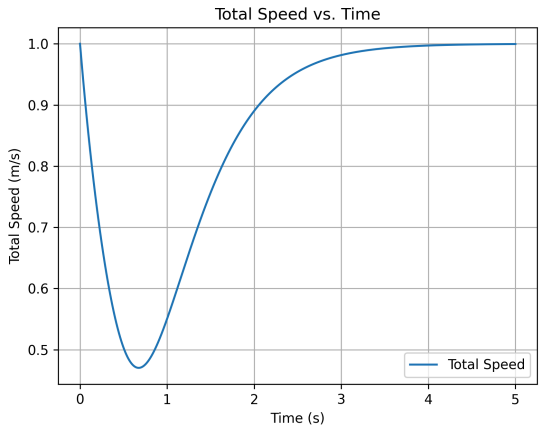
Looking at the rate of change of either the x or y velocity, it is obvious that the motion in each direction is somewhat dependent on the motion of the object in the other direction. Each velocity is dependent on the speed of the moving object, in which it is made up of the x and y components of velocity. Now let's prove this using the RK4 method.

- Our function in ou program includes this as the differential equations:
  Horizontal:
  - $\frac{dv_x}{dt} = -\,k \cdot v_x\sqrt{v_x^2 + v_y^2}$
  - $\frac{dx}{dt} = v_x$

  Vertical:
  - $\frac{dv_y}{dt} = -\,g\; -\,k \cdot v_y\sqrt{v_x^2 + v_y^2}$
  - $\frac{dy}{dt} = v_y$

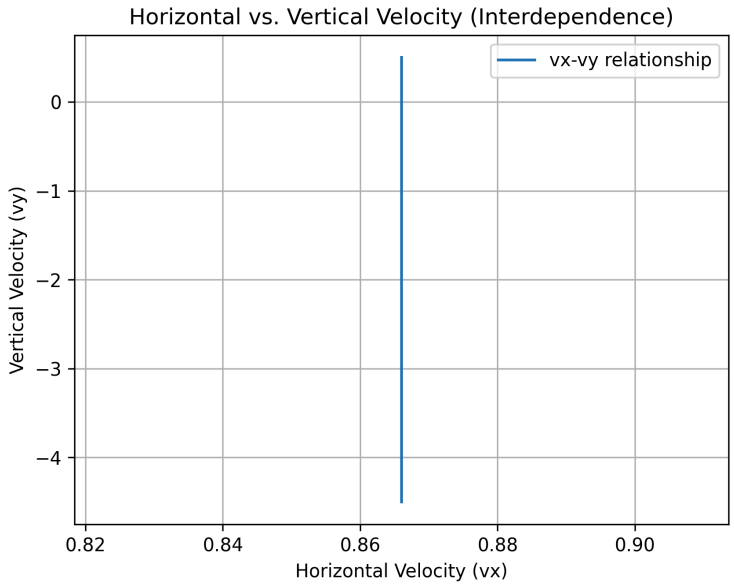
- These terms show interdependence because drag depends on the total velocity magnitude ($v = \sqrt{vx^2 + vy^2}$)
- To show how drag links the two motions we will plot:
  a. Horizontal velocity (v_x) vs. Vertical velocity (v_y).
  b. Total speed ($v = \sqrt{vx^2 + vy^2}$) over time.
- These plots show how changes in one direction affects the other.





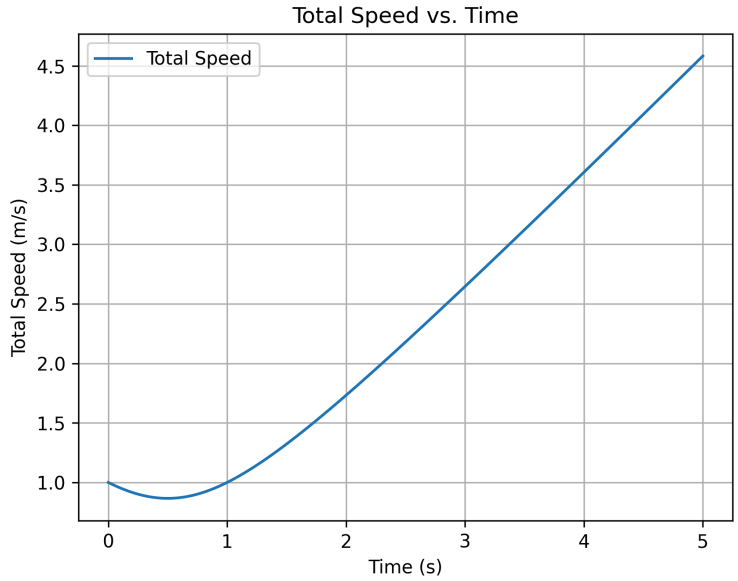
- In the graphs above, the total speed decreases over time, which can indicate energy loss due to drag.

- In the graph below, we will now set the drag constant to zero to compare a “drag free” motion, which will show how drag explicitly causes interdependence.

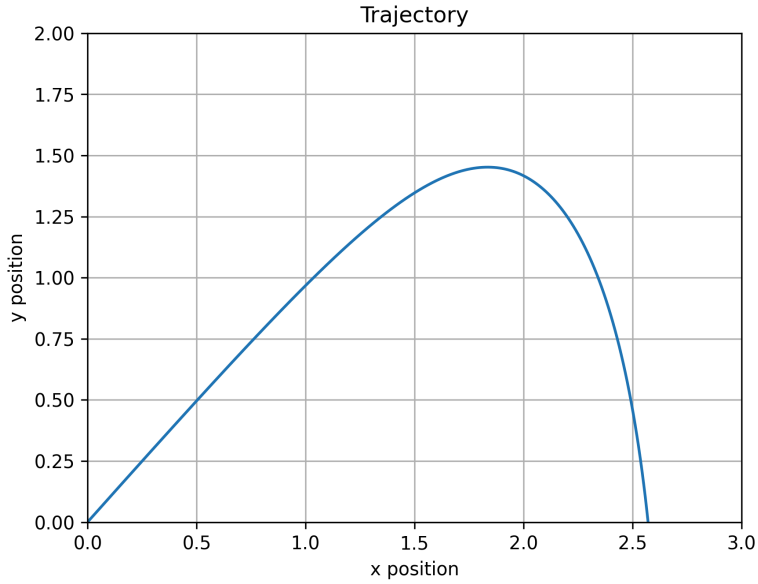


- In the plot above, without drag, the horizontal velocity is now constant since there is no change, and the vertical velocity is a linear decrease and symmetry about the apex (upward motion mirrors downward motion) because of the uniform force of gravity
- Without drag, energy is conserved, so velocity components of the vertical direction are only influenced by gravity, while the horizontal component of velocity is constant due there being zero acceleration in the horizontal direction.

Continued on the next page...

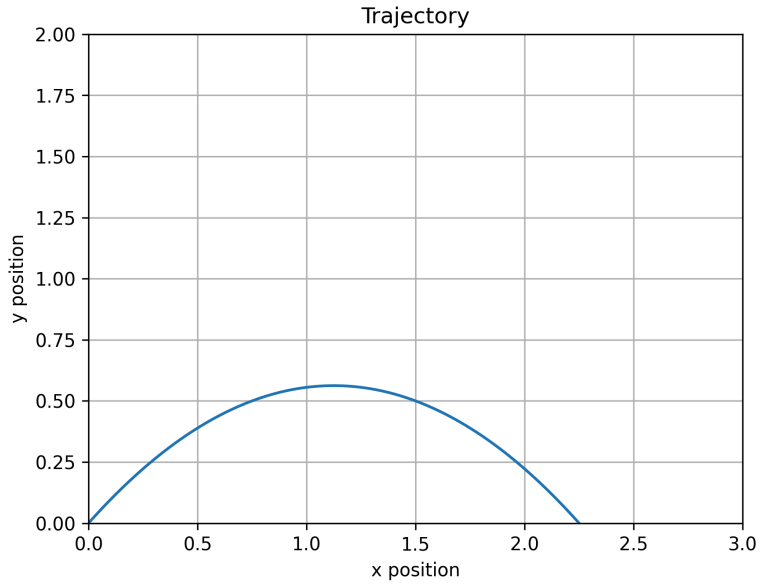


- Shown above, Now the speed over time will only increase because of gravity without any counter forces to cause drag.



- Shown in the graph above, adding back drag, observe how horizontal velocity (v_x) and vertical velocity (v_y) are interdependent due to drag.
- The shape of the plot shows how changes in one velocity affect the other.

- Its trajectory typically starts off as linear, but after it reaches a maximum, it begins to drop down parabolically, which makes sense as the drag and positions are somewhat quadratic.

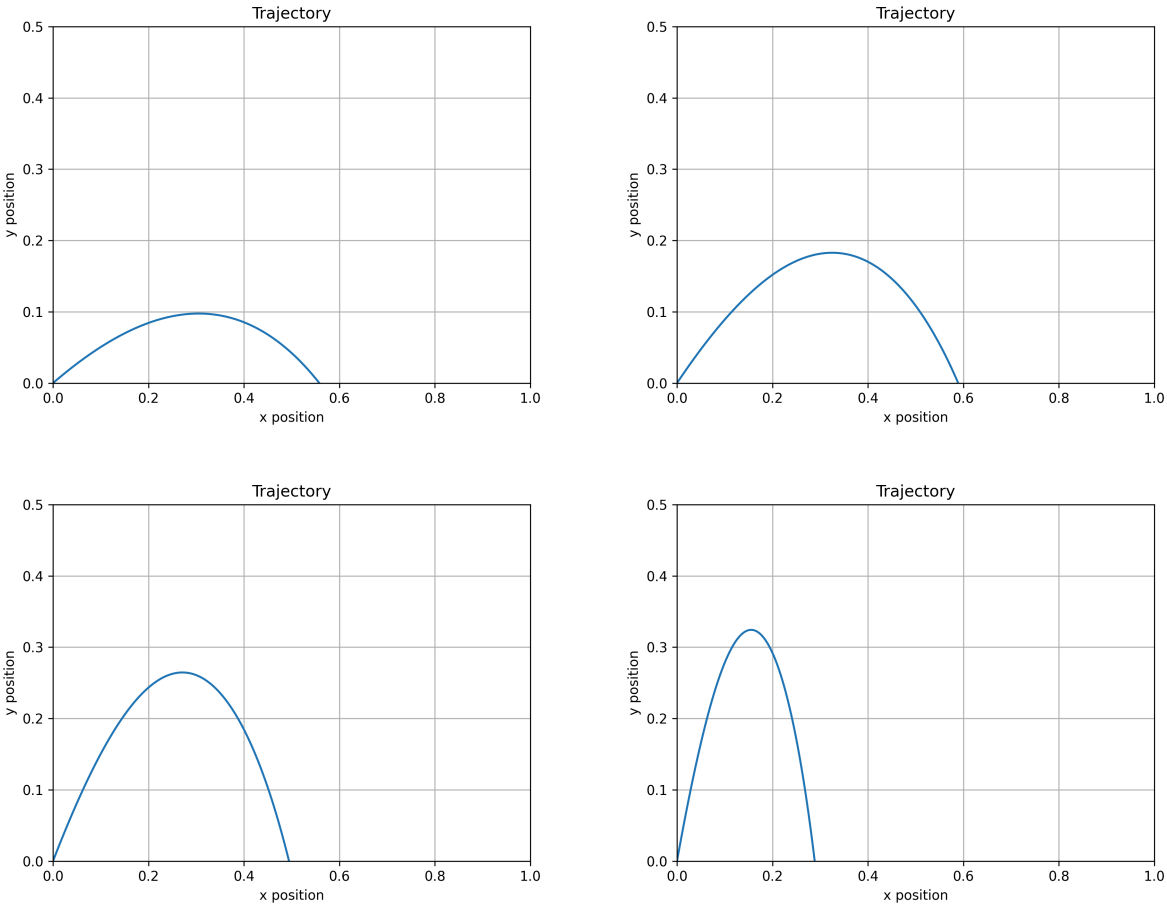


- In the graph above, this is the trajectory without drag. Its trajectory represents a parabola, which is what is expected from an object according to normal kinematics. We'll visit this more in the next objective.

Objective #2: The general shape of the trajectories, and how they appear to change based on initial speed and firing angle

- We will now compare the general shape of the trajectories, and how they appear to change based on initial speed and firing angle
- To do this we will compare the graphs of trajectories with the same initial speed, but with different launch angles, and vice versa.
- The only restriction we should have is the angles should be $0 < \theta < \frac{\pi}{2}$
- The reason for this restriction is because, if we go over $\frac{\pi}{2}$ the launching would be backwards; and if we go under 0 the launching would be into the ground.

- In the iterations below, the angles started at $\frac{\pi}{6}$ and increased by $\frac{\pi}{12}$ (until $\frac{5\pi}{12}$), while the launch speed remained 1 m/s.



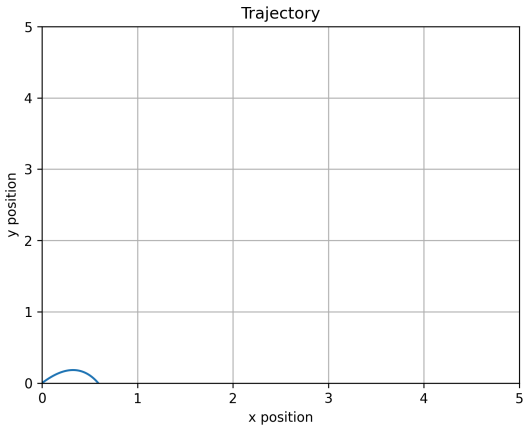
- Notice how as we increased the launching angle, the trajectory shape becomes more parabolic
- Also notice:

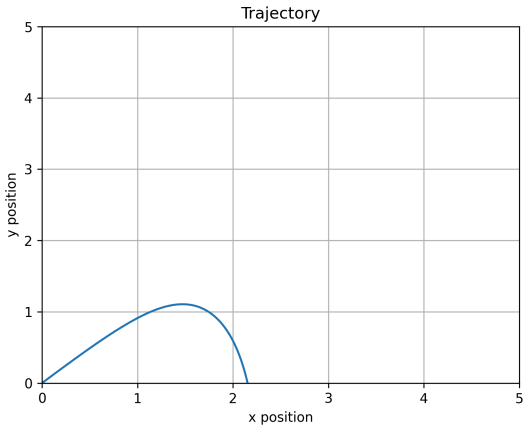
Smaller angles (e.g., $\frac{\pi}{6}$): Flatter, shorter trajectories with a focus on horizontal range.

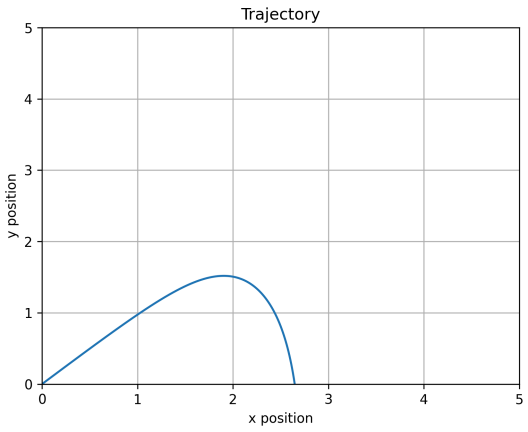
Intermediate angles (e.g., $\frac{\pi}{4}$): Parabolic trajectories with the greatest range.

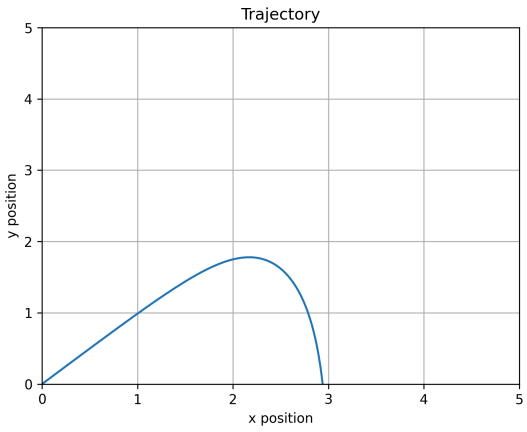
Larger angles (e.g., $\frac{\pi}{3}$): Steeper, higher trajectories with less horizontal range but greater maximum height.

- Now we will compare trajectories of the same angle, but with different launching speeds
- For the iterations below, the launching angle remained at π/4, and the launching speed started at 1 and multiplied by 5 each iteration (until 100)









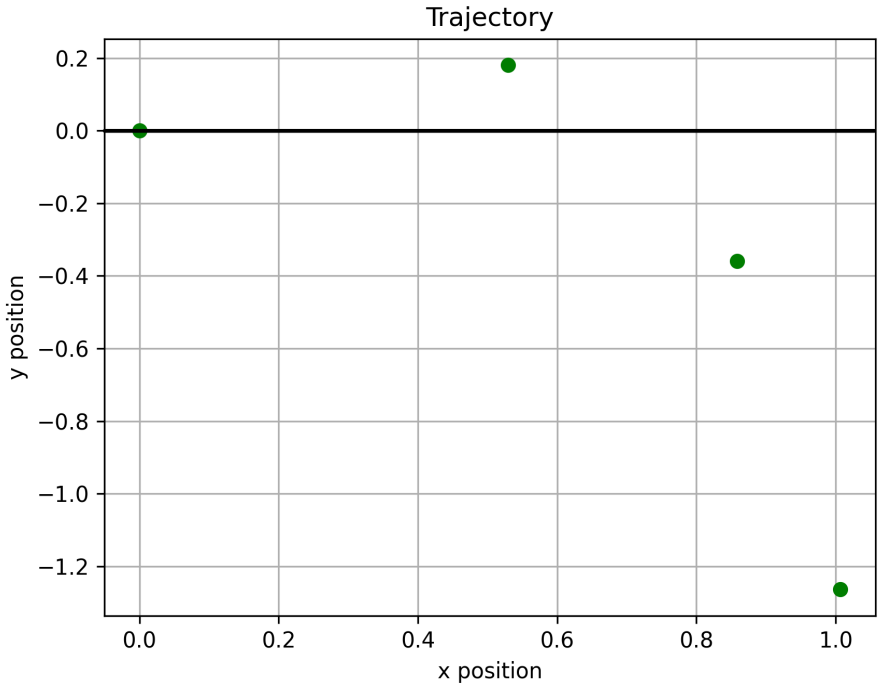
- Notice how Higher initial speeds result in longer trajectories (greater horizontal range) due to increased horizontal and vertical velocity components.
- Also, notice how graphs start off linear, and as drag and gravity cancel out the launching speed and start to have an effect, it turns into an upside down parabola.

Objective #3: How you will extract firing range from your program

Firing range is essentially how far in the x direction an object moves from its original position. (i.e. its change in x position). In normal kinematics, we just need to find the two times at which y positions are 0 (i.e. the ground) and find their respective x positions; usually with the first time the y position being zero, would be at the start. Then, we would have to find the change of position or the absolute value of the difference between these two x positions. Additionally, we can find this range by just simply programming the range equation for just any normal 2D kinematics scenario (as long as $\Delta y = 0$):

$$(R = \frac{v_o^2 sin(2\theta)}{g})$$

But of course our case is not just a simple kinematics problem we have to account for air resistance. We also don't know or can't solve for any equation(s) that will continuously plot our positions as a function of time. Our program is only numerically solving for the position of our projectile based on a system of differential equations. Due to this, our calculated position data isn't exactly continuous, meaning we need we can't exactly when the y position is zero and its respective x position. Here is an example of out plotted data:



As seen in the graph above, the bolded line represents our ground level or when y is zero. We have one data point at ground level, which is our starting position, but we don't know the exact position that the projectile hits the ground, but we at least know that it hits the ground due to two y data points having negative values (i.e. in the ground). Intuitively, we know that our projective can't "phase through" the ground, meaning that any data in which y is negative is trivial and can somewhat be ignored for the most part. Our program can't properly model the position data of the projectile as the projectile moves continuously in 3D space. Our program can only roughly approximate its position every so often. But as mentioned earlier, these projectiles realistically have continuous movement, so we can apply this known fact and approximate an effective range and assume that if small enough, each data point "linearly" connects to each other.

In order to calculate the range or effective range, we can approximate the x position of the projectile by simply finding the "average range" or the middle ground of two ranges. One of these two ranges would have a negative y value, and another having a positive y value. The moment in between these two y ranges are essentially when the y values flips from positive to negative or in other words, goes through ground level. We can use two of these y values and find their respective x values. Next, we can then use these two x values to calculate two ranges and then finally, we can find the average or the middleground of these two ranges to calculate the effective range between these ranges.

Of course this isn't the actual range, just a close approximation of it, but since we know that movement is continuous, the actual range is between the two ranges we calculated earlier and so we can calculate an error to see how far we can actually be from the true range. This is simple to do as we can show that this error is just the half difference between the two calculated ranges mentioned earlier.

Continued on the next page...

Now that we have explained our method, here is an example how our approximation method works:

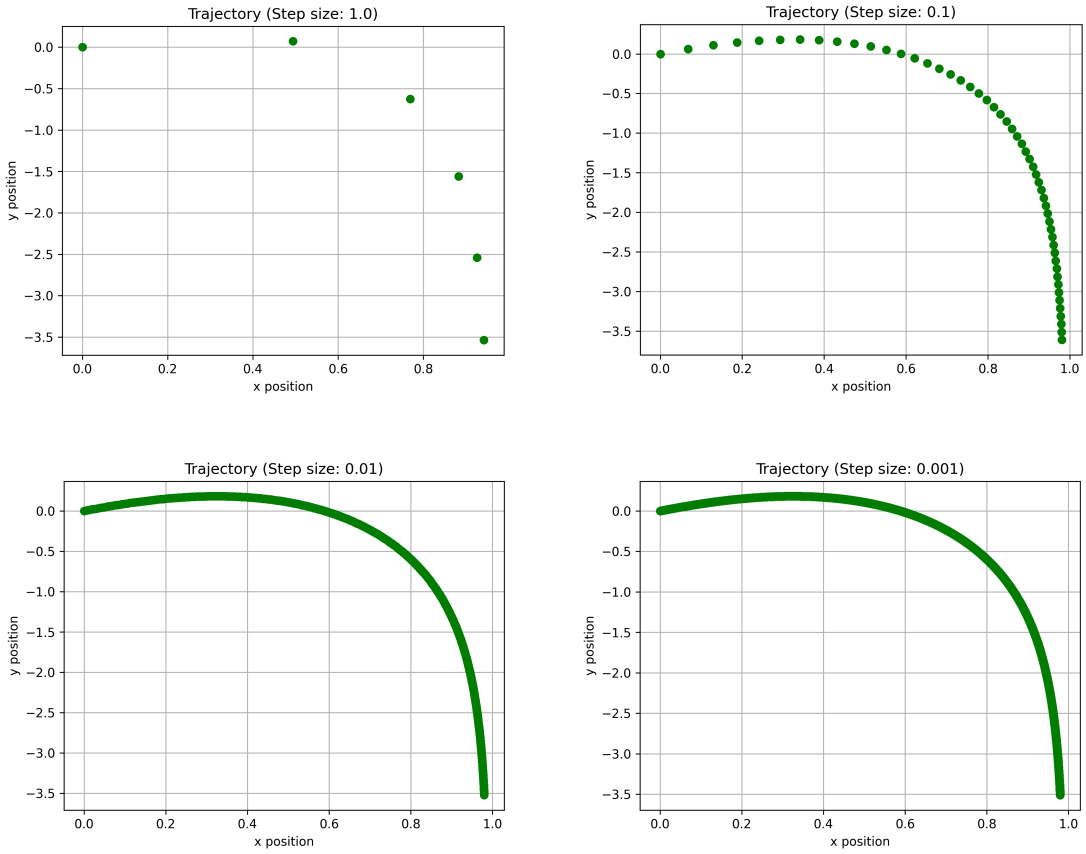
	Range 1 $y > 0$	Range 2 $y < 0$	Effective Range $\left\|\frac{R2+R1}{2}\right\|$	Error in Range $\left\|\frac{R2-R1}{2}\right\|$
Example 1	12.40 m	13.50 m	12.95 m	± 0.55 m
Example 2	7.9 m	9.5 m	8.7 m	± 0.8 m

Our method does have some limitations. First off, our method assumes that our projectiles will have at least two points in which its y position is zero, so if the projectile never touches the ground, it will never calculate the range. We found that this only occurs if the program isn't given an adequate amount of time to calculate the range of the object.

There can also be some rounding errors for some calculations that we suspect is due to the how program does calculates, For example, when the program calculates $cos(\frac{\pi}{2})$, which should be zero, it approximates its value to be: (6.123233995736766e-17), Which is very small and is practically zero, but is not exactly zero, so when graphing, you'll expect to see just only vertical movement, but due to this approximation, it causes the projectile to also have horizontal movement, which can propagate and due to interdependence of the horizontal and vertical movement, causes the projectile to to go "off course." We found that round this value helped fix this

Continued on the next page...

Additionally, we found that decreasing the step size or having a small step size, reduced this error, meaning we had a better approximation of the true range. Here are some examples:



Step Size	1.0	0.1	0.01	0.001
Calculated Range	0.6316 m	0.6041 m	0.5891 m	0.5889 m
Error in Range	± 0.1376 m	± 0.01675 m	± 0.001724 m	± 0.0001725 m

As shown above, the smaller the step size, the smaller the error. This makes sense as the program will produce more and more points in small increments that are closer to ground level.

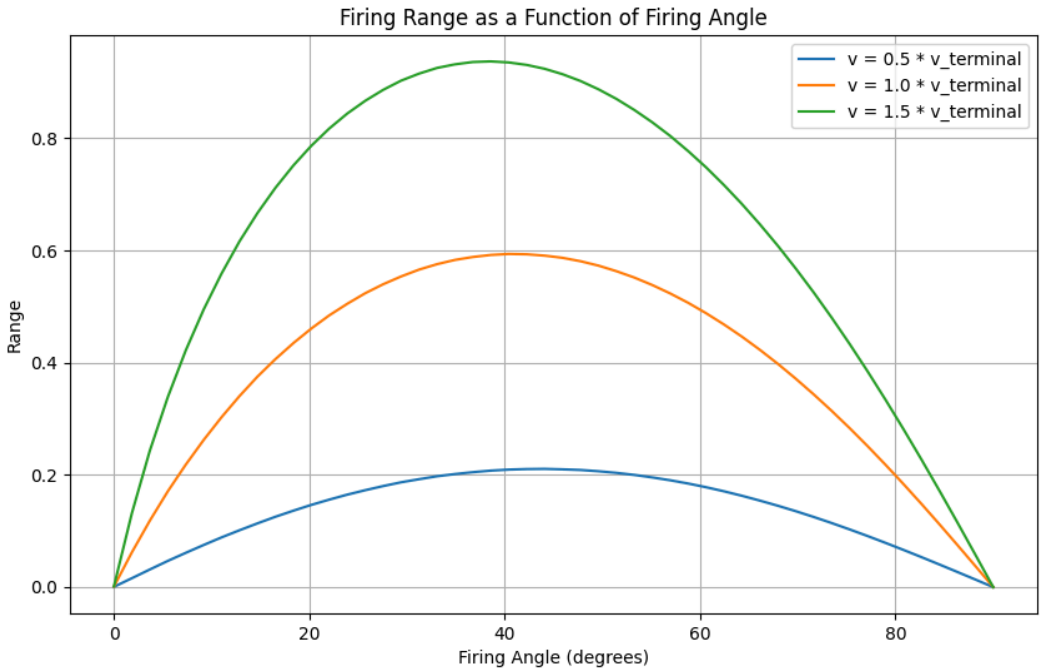
Here is the portion of our code that does this:

```
xList = [] #Empty list to store x values

    #Finds the time where y values go from negative to positive, (i.e. pass
through the x axis)
    #Appends corresponding x values to list
    for i in range(len(t_values)):
        if i == 0:
            xList.append(x_values[0])
        elif i != len(t_values)-1:
            if y_values[i] > 0 and y_values[i+1] < 0 :
                xList.append(x_values[i])
                xList.append(x_values[i+1])
        elif i == len(t_values)-1:
            xList.append(x_values[i])
            xList.append(x_values[i])
    #Finds the two bounding ranges that are less or more than the "true"
range
    Range1 = xList[1] - xList[0]
    Range2 = xList[2] - xList[0]

    #Calculates the effective or the average range between the two bounding
ranges and finds their error.
    effRange = abs(Range1 + Range2)/2
    errorRange = abs(Range2 - Range1)/2 #Decreasing step size reduces the
error of the range
```

It uses a loop that iterates through each y value and compares it to the next one, and if they are both different signs (i.e. negative and positive), their corresponding x values are added to a list. It adds the first x value, which is the starting position and is zero. It also adds the last value in the list just in case the program terminates just before it “pierces” the ground.

Finally, we can’t conclude if our code is 100% flawless, but throughout the rest of our testing for this report, we used careful consideration and observations to account and consider any out of the ordinary situations.

Objective #4: The firing range as a function of firing angle for various fixed initial speeds (above and below terminal speed)



Firing Angles: Angles range from 0 to $\frac{\pi}{2}$

Initial Speeds: $\frac{1}{2}v_{Terminal}$, $\frac{2}{2}v_{Terminal}$, $\frac{3}{2}v_{Terminal}$

Blue Line:

When the initial speed is half of the terminal speed, this is the projectile's firing range. The relatively low speed of the projectile prevents great horizontal motion, making this curve lower and narrower than the others. Since this velocity is much smaller than the terminal velocity, drag is present but weak, and thus the trajectory is more similar to a typical parabolic motion.

Green Line:

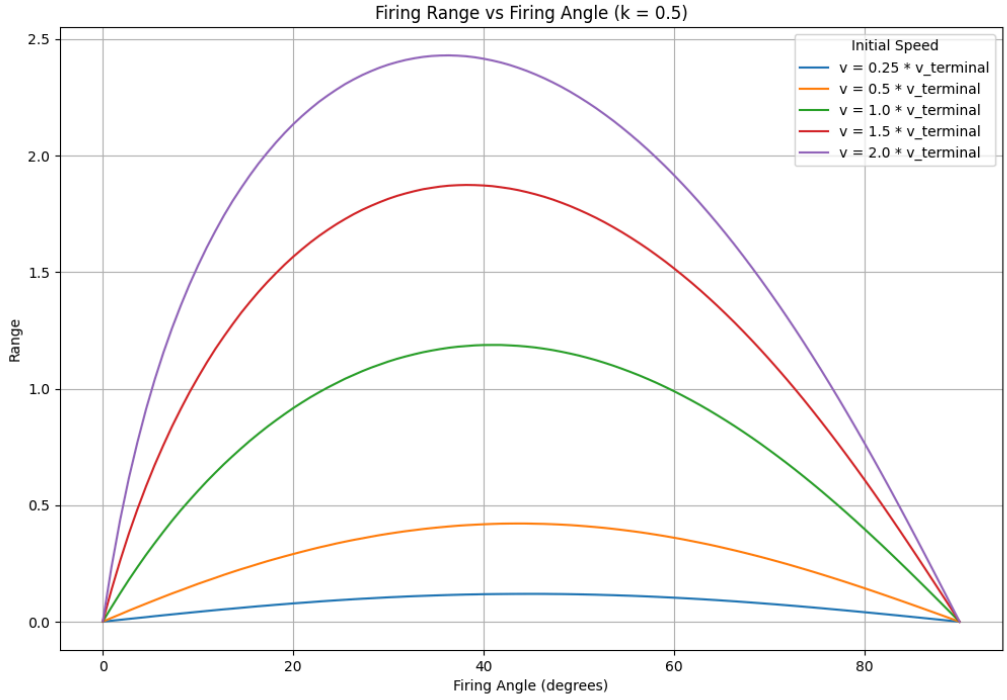
When the initial speed is 1.5 times the terminal speed, this is the projectile's firing range. Since the projectile has an appreciable range at initial velocity before drag overcomes the motion, the curve is the highest and broadest.

The drag is much more pronounced at this speed and the range decreases considerably beyond the ideal firing angle.

Orange Line:

When the initial speed and the $v_{Terminal}$ are equal, this is the projectile's firing range. Drag has a significant impact on the trajectory at this balancing point. Because the drag reaches equilibrium with gravity sooner than in the $\frac{3}{2}v_{Terminal}$ case, the range is shorter.

Due to its limited vertical height, the projectile's motion for shallow angles is nearly horizontal and the ranges are relatively small. Because the high initial velocity overcomes drag, the green curve has the maximum range for the shallow angle launches. This is the optimum angle in ideal conditions. Since drag reduces the effectiveness of steeper launch angles, the orange and green curves have a maximum below 45°. At steeper angles, the bullet travels less horizontally because most of its velocity is concentrated vertically. Drag slows down the climb and the fall of the projectile, which forces the green curve to sharply decline for $\frac{3}{2}v_{Terminal}$. In aiming, angles near 45° result in the greatest ranges for projectiles projected at speeds below the $v_{Terminal}$. For projectiles at high speeds (faster than $v_{Terminal}$), the angles should ideally just be below 45° because drag will strongly affect them.



Continued on the next page...

Blue curve:

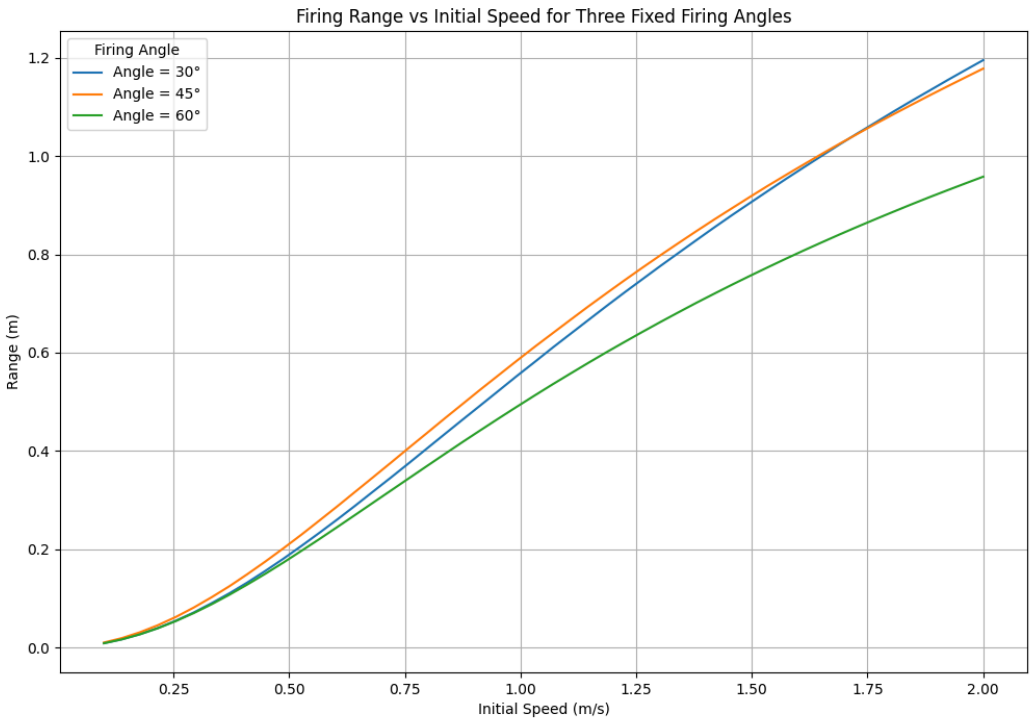
This speed is so much lower than the $v_{Terminal}$ that drag forces are negligible compared with gravity. It is pretty much similar to the ideal projectile motion. The ideal angle for drag-free conditions is 45°, where the peak of the firing range occurs. Because of the low takeoff speed, which limits the horizontal displacement, the range is rather restricted.

Purple Curve:

Because the drag forces predominate throughout the motion, the range at double the $v_{Terminal}$ is considerably less than the others. The ideal firing angle is much less than 45° and the firing range is flatter. The range is the greatest because of the high starting speed, but the returns drop when compared to lower speeds.

The ideal firing angle for maximum range without drag is 45°. With drag, the ideal angle decreases below 45°, with the amount depending on the initial speed and the drag constant. Drag lessens the projectile's horizontal displacement, particularly at greater starting speeds.
Drag becomes restrictive to the higher velocity achieving longer ranges, as shown in this flattening of the curves at the higher speeds. At the lower speeds, shooting range is limited because there is not enough horizontal velocity. At higher speeds, drag reduces the possible advantages of a higher starting velocity.

Objective #5: The firing range as a function of initial speed for various fixed firing angles



This graph illustrates how a projectile's firing range varies with initial velocity for three fixed firing angles: 30, 45, and 60.

Blue curve:

The horizontal component of the velocity is so large at this angle that it is especially effective in converting speed into range, particularly at low speeds. At higher speeds, drag begins to erode the horizontal velocity, resulting in decreasing range returns.

Orange Curve:

Because it balances vertical and horizontal motion, 45 degrees is the perfect angle for maximum range when air resistance is absent. Because it minimizes the impact of drag on vertical motion and preserves a reasonable balance between vertical and horizontal motion, the 45 angle achieves the longest range across the majority of beginning speeds. Even when returns start to decline, the curve is still higher than the 30 and 60 angles.

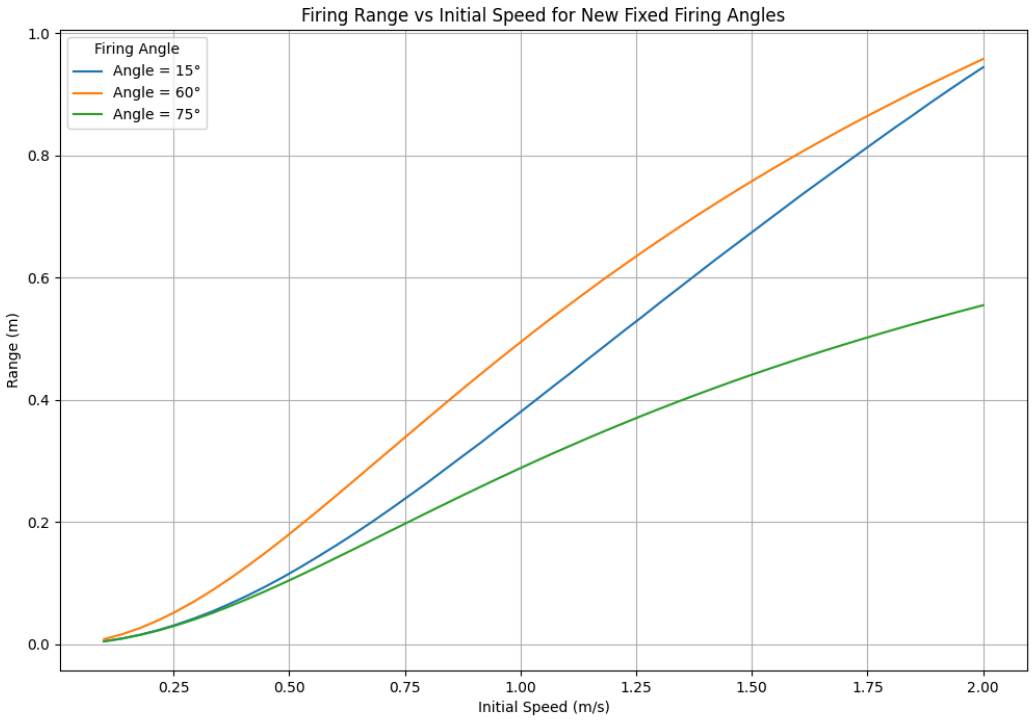
Green Curve:

> The dominance of the vertical component of velocity is responsible for less effective horizontal motion. Drag drastically cuts down the range at higher speeds due to its effect on vertical motion. The curve would flatten much faster compared to 30 or 45.

For small initial velocities ($v < \frac{1}{2}v_{Terminal}$), all angles reveal similar range behavior. The angle of 45 has a slight superiority due to the balanced horizontal and vertical motions. As the speed approaches the $v_{Terminal}$ ($v \approx v_{Terminal}$), 45 yields the maximum range, while the second best goes to 30 since it has a larger horizontal velocity component. Drag limits the range for all angles at higher speeds ($v > v_{Terminal}$), while the curves flatten due to diminishing returns from increased starting speed. Relatively steep angles, like 60, are inefficient compared with shallower angles because drag impacts the vertical motion disproportionately. Overall, 45 always yields the greatest range, especially at medium-range speeds; under strong drag influence, though, 30 becomes competitive at higher speeds.

For optimizing range over a broad variety of starting speeds, 45 is still the most effective angle. Angle differences are negligible at lower speeds, but drag flattens the curve for all angles and accentuates the inefficiency of steeper angles (60) at higher speeds.

Continued on the next page...



Blue Curve:

Range effectively uses the initial speed due to the shallow launch angle distributing the majority of the velocity horizontally. Range increases with increasing speed but eventually flattens as the drag rises with greater speeds. This angle works well at low to moderate speeds but can compete with 60 at greater speeds.

Orange Curve:

Because 60 better balances horizontal and vertical motion than 15 or 75, it reaches the longest range at low- and moderate-speed values. The curve for the most part stands tall above the others, testifying to how efficiently this angle transfers speed to range.

Green Curve:

Since most of the velocity is vertical at large speeds, the ranges are correspondingly small. For larger speeds, drag has a great effect on the vertical motion, and the curve flattens out far more quickly than it does for other angles.

Particularly at sub-$v_{Terminal}$ ($v < v_{Terminal}$). The range grows with beginning speed at all angles. This is because, at lower speeds, drag has less of an effect. The range flattens at all angles as the beginning speed rises above the $v_{Terminal}$ ($v > v_{Terminal}$). The projectile's range is limited and its horizontal velocity is diminished by drag forces.

Extension: Incorporating lift

In order to find the differential equations for a projectile with lift, we will start with the base equation we found earlier:

$$\frac{d^2\hat{r}}{dt^2} = \hat{g} - kv \cdot \hat{v}$$

This is the base equation for the movement without lift, but in order to incorporate lift, we need to add a third term along with the proportionality to the square of the speed:

$$\frac{d^2\hat{r}}{dt^2} = \hat{g} - k_R v \cdot \hat{v} + k_L v \cdot U\hat{v}$$

$$Note: \hat{r}, \hat{g}, and \hat{v} \text{ are vectors, not unit vectors. } U \text{ is } 2x2 \text{ matrix}$$
$$The \text{ constants } k_R \text{ and } k_L \text{ denote the drag and lift constants respectively}$$

As shown in the equation above, in order to find the perpendicular component of the differential equation, we just need to rotate the 2nd velocity vector by 90 degrees, and we can use a transformation matrix to do this.

$$U\hat{v} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} -v_y \\ v_x \end{bmatrix} = \hat{u}$$

$$Note: \hat{u} \text{ is a vector, not a unit vector.}$$

Now we can denote our differential equation like this:

$$\frac{d^2\hat{r}}{dt^2} = \hat{g} - k_R v \cdot \hat{v} + k_L v \cdot \hat{u}$$

And then also split it into its respective components and define the speed:

$$\frac{d^2x}{dt^2} = -k_R v_x \sqrt{v_x^2 + v_y^2} - k_L v_y \sqrt{v_x^2 + v_y^2}$$
$$\frac{d^2y}{dt^2} = -g - k_R v_y \sqrt{v_x^2 + v_y^2} + k_L v_x \sqrt{v_x^2 + v_y^2}$$

Now we have our system of differential equations:

$$\frac{d^2x}{dt^2} = (-k_R v_x - k_L v_y)\sqrt{v_x^2 + v_y^2} \qquad \frac{d^2y}{dt^2} = -g + (-k_R v_y + k_L v_x)\sqrt{v_x^2 + v_y^2}$$

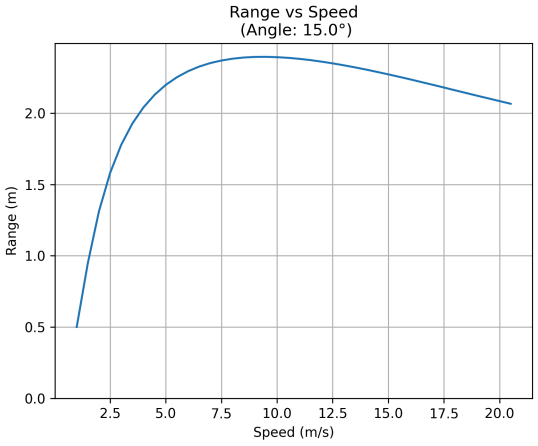
$$\frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

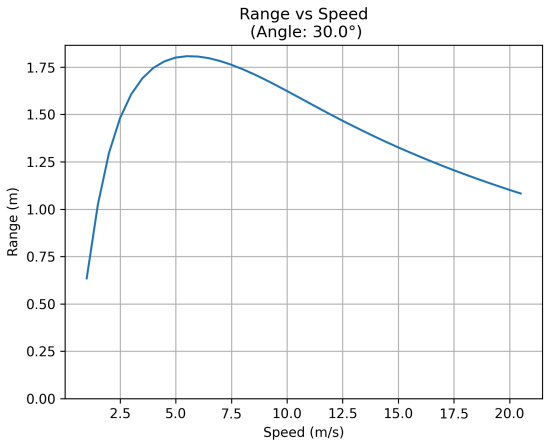
Now, in order to determine the optimal lift constants for the max ranges, we need to first rest it on a variety of speeds and angles to see how to maximize the range. We will test this by keeping all variables except one, and slowly change the one we aren't keeping constant. This will help see how the behavior of increasing or decreasing a value also increases or decreases the range.

Our three variables are the lift constant, the initial launch speed, and the launch angle. We will define the range as the value of the change in the x position relative to the starting position (i.e. the origin), so this means the going backwards or on landing on the opposite side of launch direction also counts and won't be neglected. This is essentially negative ranges are not “desired”

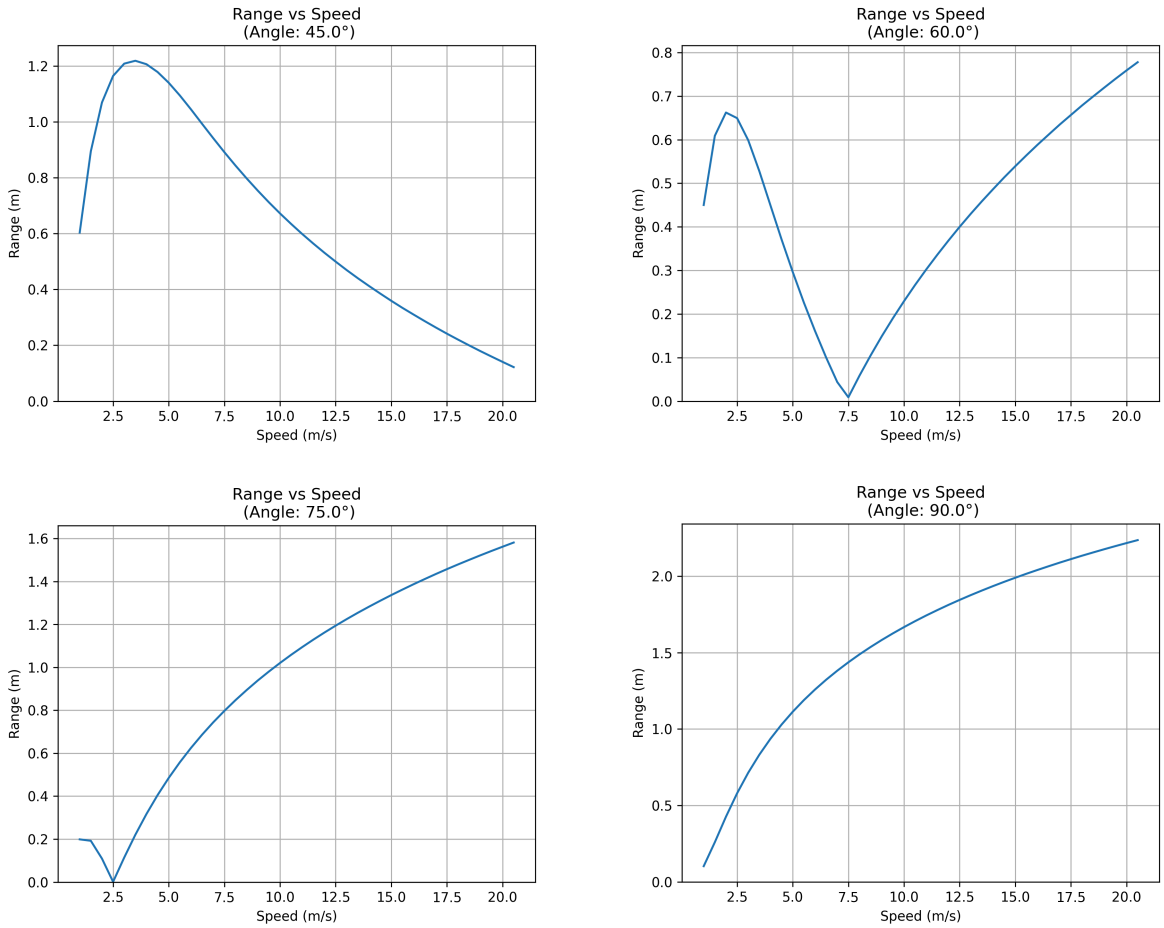
First let's start with slowly adjusting the speed ($k_L = 0.5$):

From our testing we found that increasing the launch speed did in fact increase the range but there was sometimes a maximum before the range began to decrease as the launch speed increased. There were also other times in which the graph would reach a maximum, then reach a minimum before just increasing all together were also other times where it should just directly increase. We found that this varied by angle. Here are some example graphs of the Range vs launch speed at various launch angles:





Continued on the next page...

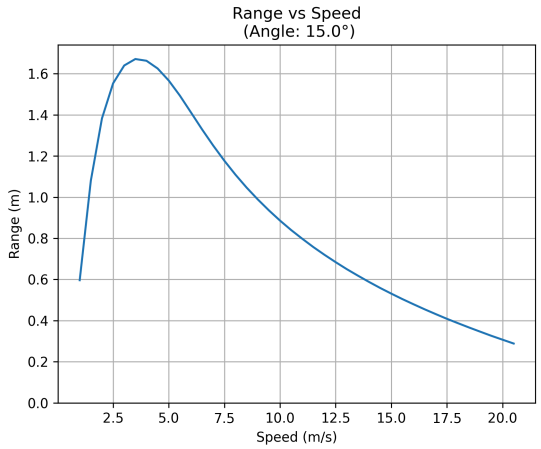


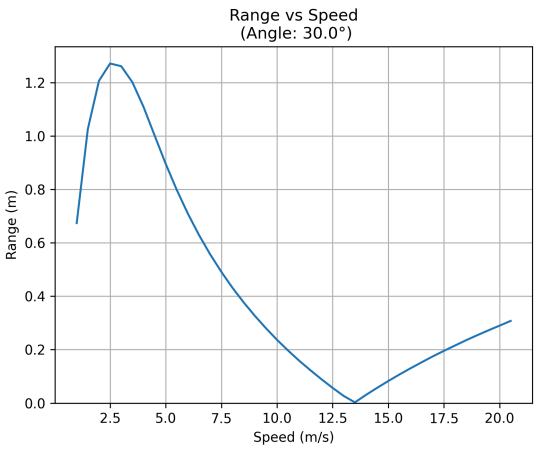
(Note: the 90° angle seems like the range increase only, but it immediately went backwards)

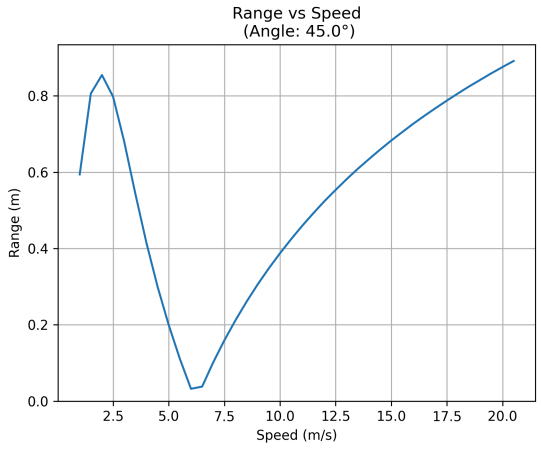
There are multiple scenarios in which increasing the speed will either decrease or increase and from the graphs shown above, increasing the speed tends to increase the range at higher angles and increasing speed will tend to decrease the range at lower angles. But overall, at lower angles, the range decreases as speed decreases, but at higher angles, they tend to flip or go "backwards" (i.e. the projectile lands behind the direction of initial motion). This makes sense because looking at our differential equations, the lift or acceleration in the horizontal direction is more prominent or greater when the vertical velocity is greater, which is true at higher launch angles. The point at which the graph has a "cusp" is where we start having "negative ranges" or start going backwards.

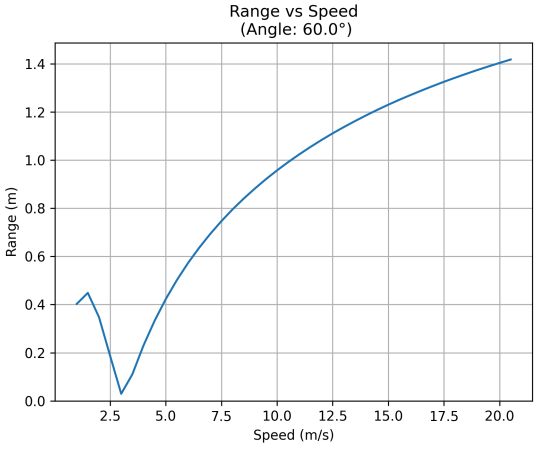
Continued on the next page...

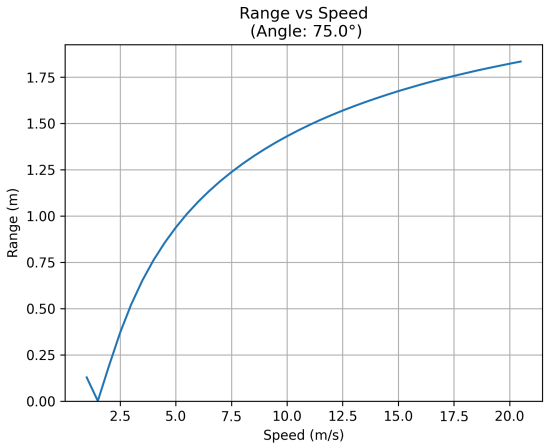
Repeating this with a new lift constant, we found similar things but with slightly different tendencies. Here are some example graphs ($k_L = 0.85$):

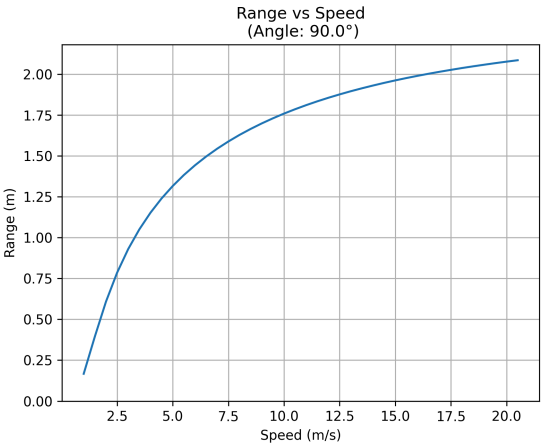












(Note: the 90° angle seems like the range increase only, but it immediately went backwards)

Continued on the next page…

We found a similar pattern, in which the projectile would begin to move backwards, but sooner than normal when adjusting the angle (i.e. then range began to go backwards at smaller angles compared to the last tested lift constant). Once again, this makes sense as the lift force is greater in the horizontal direction when the vertical velocity is higher, which is true for larger angles, along a higher lift constant, maximizes the backwards horizontal acceleration.

So, what can we conclude from this? Well there are many different kinds of optimal lift constants, but as we examined, the type of launch and angle can of course affect the which lift constant would maximize the range if you have high launch angle and a high launch speed, its best to have the smallest possible lift constant, but if you have a smaller launch speed and small launch angle, you can be more lenient with a bigger lift constant. If you have launch speed and small angle, or vice versa, it's best to have a mild or a not too big or not too small lift constant.

In reality, the lift contant is probably the hardest thing to adjust as it is based on the physicality of the projectile, followed by the launch speed, and then the launch angle, which is the easiest to adjust.