What Angle Should We Throw a Football for Maximum Range

Iydon Liang*

December 15, 2019

Abstract

We introduce projectile motion and quadratic equation through a real world problem — *what angle should we throw a football for maximum range*, then we give a proper and accurate definition of quadratic equation, from which we derive quadratic formula for solving quadratic equations. Furthermore, we list some useful applications of the quadratic equation and implement them in `Python`.

Keywords: Projectile motion; Quadratic equation; `Python` simulation.

Contents

1 Introduction **2**

1.1 Preliminary information 2

1.2 Projectile Motion 2

2 Quadratic Equation **3**

2.1 Definition 3

2.2 Quadratic Formula and Its Derivation 3

3 Applications **5**

3.1 Solution to Article Title 5

3.2 Simulation of Projectile Motion 6

4 Conclusions and Discussions **6**

*DataHub Organization, Southern University of Science and Technology, Shenzhen, China

1 Introduction

1.1 Preliminary information

We assume that the reader has already learned the parabola in the physics class and the rectangular coordinate system in the math class. If you do not have the required knowledge, it does not matter, you can follow the information provided in this article, learning while reading.

1.2 Projectile Motion

Perhaps we have been told by our parents since childhood that, 45 degrees is the ideal angle to throw a football without air resistance, to understand the reasons why 45 degrees is the best angle without air resistance, we should start with projectile motion [1].

Definition 1 ► Projectile motion

rojectile motion is a form of motion experienced by an object or particle (a projectile) that is thrown near the Earth's surface and moves along a curved path under the action of gravity only[a].

[a] In particular, the effects of air resistance are assumed to be negligible.

In the following article, we call the curved path a trajectory [2]. Through experiments or simulations, we can get the trajectories of football at different angles, as shown in Figure 1. From the picture we can see that, the footballs are thrown from the origin, the speed remains unchanged, while the angle has changed. Intuitively, 45 degrees is indeed the ideal angle to throw a football for maximum range, and complementary angles have the same range.

The trajectory was shown by Galileo to be a parabola, which is relevant to quadratic equation. No rush, we will learn the basics of quadratic equation, in the next section.

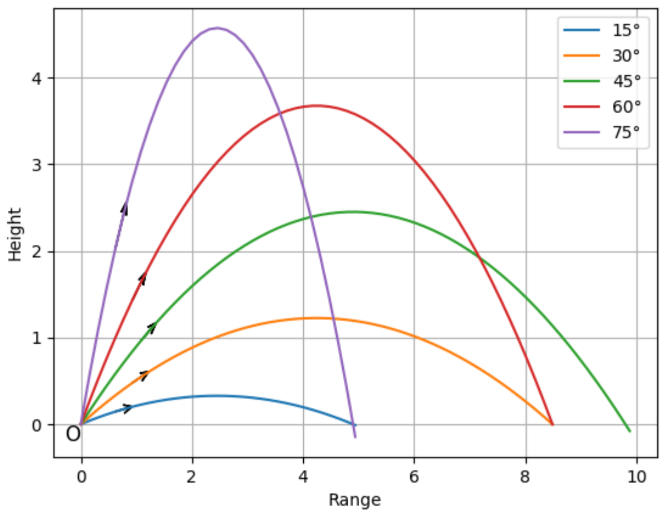

Figure 1: Parabolic trajectories at different angles

2 Quadratic Equation

2.1 Definition

Definition 2 ▶ Quadratic equation[3]

n algebra, a quadratic equation (from the Latin *quadratus* for "square") is any equation having the form

$$c_2x^2 + c_1x + c_0 = 0, \tag{1}$$

where x represents an unknown, and c_n $(n = 0, 1, 2)$ represent known numbers, with $a \neq 0$. If $a = 0$, then the equation is linear, not quadratic, as there is no c_2x^2 term. Then we can divide both sides of the equation by c_2, and replace c_1/c_2 with b, c_0/c_2 with c.

$$x^2 + bx + c = 0. \tag{2}$$

That is, every quadratic equation can be transformed into the form of Equation (2).

2.2 Quadratic Formula and Its Derivation

From Equation (2), we can complete the square to derive a general formula to solve quadratic equations. Then we have noticed that, coefficient of x^2 is 1, coefficient of x is b, if we want to complete the square form, we have

$$
\begin{aligned}
x^2 + bx + c &= 0 \\
x^2 + 2\frac{b}{2}x &= -c \\
\left(x + \frac{b}{2}\right)^2 &= -c + \frac{b^2}{4} \\
\left(x + \frac{b}{2}\right)^2 &= \frac{b^2 - 4c}{4}.
\end{aligned}
\tag{3}
$$

The left side of the Equation (3) is a squared form, which means that it may have multiple solutions. That is, if $b^2 - 4c \geq 0$,

$$
\begin{aligned}
x + \frac{b}{2} &= \pm\frac{\sqrt{b^2 - 4c}}{2} \\
x &= -\frac{b}{2} \pm \frac{\sqrt{b^2 - 4c}}{2} \\
x &= \frac{-b \pm \sqrt{b^2 - 4c}}{2}.
\end{aligned}
\tag{4}
$$

If we substitute the Equation (4) back to the Equation (2), we will find that the equation still holds. That is the quadratic formula is

$$
x = \begin{cases} \frac{-b\pm\sqrt{b^2-4c}}{2} & \text{If } b^2 - 4c \geq 0; \\ \frac{-b\pm i\sqrt{4c-b^2}}{2} & \text{If } b^2 - 4c < 0. \end{cases}
\tag{5}
$$

You can get an intuitive feel for the quadratic equation from Figure 2. Moreover, we found other interesting phenomena — vertex, which is the extreme point of the parabola, whether minimum or maximum. The x-coordinate of the vertex will be located at $x = -\frac{b}{2}$, and the y-coordinate of the vertex may be found by substituting this x-value into the function, which gives that $y = \frac{1-(b^2-4c)}{4}$.

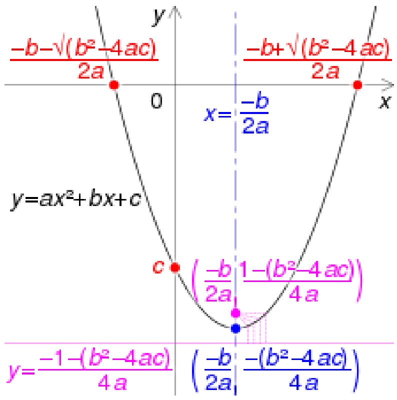

Figure 2: Visualization of quadratic equations (with $a = 1$)

3 Applications

3.1 Solution to Article Title

From the definition of quadratic equation in Section 2, we can finally answer the question, *what Angle Should We Throw a Football for Maximum Range?*

Firstly, we establish a rectangular coordinate system, as shown in Figure 3. We denote θ as shot angle, V as initial velocity, g as gravity, S as distance, t as time.

Then, we have both horizontal and vertical velocity,

$$\begin{cases} V_x & = V\cos(\theta) \\ V_y & = V\sin(\theta) - gt \end{cases} \tag{6}$$

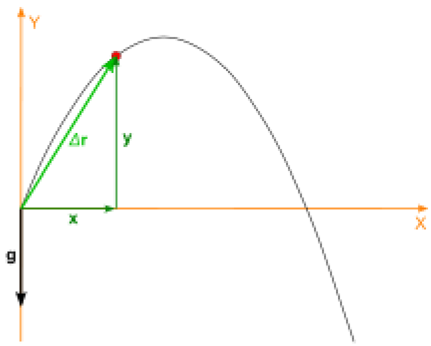

Figure 3: Displacement and coordinates of parabolic throwing

Therefore, we have both horizontal and vertical distance at time t,

$$\begin{cases} S_x &= V\cos(\theta)t \\ S_y &= V\sin(\theta)t - \frac{1}{2}gt^2 \end{cases} \tag{7}$$

From Equation (6) and Equation 6, we can find all of the coordinate (S_x, S_y) with angle θ. From Equation (7), we have $t = \frac{2V\sin(\theta)}{g}$, then we substitute the expression of t back to the S_y, we have the quadratic equation

$$S_y = \tan(\theta)S_x - \frac{g}{2V^2\cos^2(\theta)}S_x^2. \tag{8}$$

From Equation (5), we can solve this quadratic equation, the solutions are 0 and $\frac{2V^2}{g}\sin(\theta)$. Obviously, 0 is the initial position, then the football will land at $\frac{2V^2}{g}\sin(\theta)$, that is, $\frac{V^2}{g}\sin(2\theta)$.

According to knowledge of trigonometric functions, we have the maximum range of the football is $\frac{V^2}{g}$, which is taken at $\theta = 45$.

3.2 Simulation of Projectile Motion

You can use quadratic equation to simulate projectile motion, which can solve free fall problems. If you are familiar with `Python`, you can use `Python` to draw the pictures of trajectories, and you will find it easy to use Python to solve quadratic equations, especially in symbolic calculations. And the code is in Appendix 4.

4 Conclusions and Discussions

From this article, we introduce projectile motion and quadratic equation through a real world problem — *what angle should we throw a football for maximum range*, then we give a proper and accurate definition of quadratic equation, from which we derive quadratic formula for solving quadratic equations. Furthermore, we list some useful applications of the quadratic equation and implement them in `Python`.

Acknowledgement

The author would like to thank the Associate Editor and one referee for their valuable comments which have greatly improved the paper. The author's research is fully supported by the DataHub Organization of China (No. 1000001).

References

[1] Wikipedia contributors. Projectile motion — Wikipedia, the free encyclopedia[EB/OL]. 2019. https://en.wikipedia.org/w/index.php?title=Projectile_motion&oldid=928426750.

[2] Wikipedia contributors. Trajectory — Wikipedia, the free encyclopedia[EB/OL]. 2019. https://en.wikipedia.org/w/index.php?title=Trajectory&oldid=929280103.

[3] Wikipedia contributors. Quadratic equation — Wikipedia, the free encyclopedia [EB/OL]. 2019. https://en.wikipedia.org/w/index.php?title=Quadratic_equation&oldid=929716375.

[4] ALLAIN R. What angle should you throw a football for maximum range?[EB/OL]. 2008. https://www.wired.com/2008/12/what-angle-should-you-throw-a-football-for-maximum-range/.

Appendix — Quadratic Equation in Python

If you are familiar with Python, you can use Python to draw the pictures of trajectories, and you will find it easy to use Python to solve quadratic equations, especially in symbolic calculations.

quadratic_equation.py

```python
#!/usr/bin/python3
# -*- encoding: utf-8 -*-
'''
@File      : quadratic_equation.py
@Author    : Iydon Liang
@Docstring : Solve the quadratic equation.
@Require   : Python==3.7.6; sympy==1.4
@Result    : \\
Poly(C2*x**2 + C1*x + C0, x, domain='ZZ[C0,C1,C2]')
                   ________________
                  /              2
    C1          \/  -4*C0*C2 + C1
- ---- - -------------------
  2*C2              2*C2
                   ________________
                  /              2
    C1          \/  -4*C0*C2 + C1
- ---- + -------------------
  2*C2              2*C2
'''
from sympy import symbols, Poly, roots, pretty_print
from sympy.assumptions import assuming, Q

# variables declaration
degree = 2
C = symbols(f'C0:{degree+1}')  # C_0, C_1, ..., C_degree
x = symbols('x')               # x^0, x^1, ..., x^degree
# calculate roots
with assuming(Q.nonzero(C[-1]), *map(Q.complex, C[:-1])):
    quadratic_equation = Poly(reversed(C), x)
    solutions = roots(quadratic_equation)
# display results
pretty_print(quadratic_equation, use_unicode=False)
for solution in solutions:
    pretty_print(solution, use_unicode=False)
```

trajectory.py

```python
#!/usr/bin/python3
# -*- encoding: utf-8 -*-
'''
@File      : trajectory.py
@Time      : 2019/12/15
@Author    : Iydon Liang
@Contact   : liangiydon AT gmail.com
@Docstring : Draw the picture of trajectories.
@Require   : Python==3.7.6; matplotlib==3.1.2;numpy==1.17.4
@Result    : <pictures>
'''
import numpy as np
import matplotlib.pyplot as plt


θs = 15, 30, 45, 60, 75
V = g = 9.8

Sx = np.linspace(0, 2*V**2/g, 128)
for θ in np.deg2rad(θs):
    Sy = Sx * (np.tan(θ) - g*Sx/(2*(V*np.cos(θ))**2))
    Sy[np.sum(Sy>=0.)+1:] = np.nan
    plt.plot(Sx, Sy)
    idx = np.sum(Sx < .1*V**2/g*np.sin(2*θ))
    plt.arrow(Sx[idx], Sy[idx],
        Sx[idx+1]-Sx[idx], Sy[idx+1]-Sy[idx],
        head_width=.1, overhang=1)

plt.legend([f'{θ}°' for θ in θs])
plt.text(0, 0, 'O', fontsize=12,
    horizontalalignment='right', verticalalignment='top')
plt.grid()
plt.xlabel('Range')
plt.ylabel('Height')
plt.show()
```