



Realistic Projectiles

Quadratic Drag

Vincent Edwards, Julia Corrales, Rachel Gossard

Mt. SAC

2025-06-09

Contents

| | | | | | |
|-----|-------------------------------------|----|-----|---------------------------------------|----|
| 1 | Background | 3 | 3 | Trajectory Shapes | 14 |
| 1.1 | Quadratic Drag Equation . | 4 | 3.1 | Trajectory as θ Varies . . . | 15 |
| 1.2 | RK4 Method for Systems . | 5 | 3.2 | Trajectory as v_0 Varies . . | 16 |
| 1.3 | Projectile Equations | 7 | 4 | Firing Range | 17 |
| 1.4 | Equation Implementation . | 8 | 4.1 | Diagram | 18 |
| 1.5 | Launching Projectiles | 9 | 4.2 | R vs θ as v_0 Varies | 19 |
| 2 | Interdependence of Motion | | 4.3 | R vs v_0 as θ Varies | 20 |
| | Components | 10 | 5 | Hitting a Fixed Target | 21 |
| 2.1 | Equations | 11 | 5.1 | Over/Under-Shooting the | |
| 2.2 | x vs t as v_{0y} Varies | 12 | | Target | 22 |
| 2.3 | y vs t as v_{0x} Varies | 13 | 5.2 | v vs θ as Target Angle | |
| | | | | Varies | 23 |

5.3 v vs θ as Target Distance
Varies 24

1 Background

1.1 Quadratic Drag Equation

1 Background 🧐

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

1.1 Quadratic Drag Equation

1 Background 🧐

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

$$\vec{r} = \begin{pmatrix} x \\ y \end{pmatrix}$$

(position)

1.1 Quadratic Drag Equation

1 Background 🧐

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

$$\vec{r} = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{(position)}$$

$$\vec{v} = \frac{d\vec{r}}{dt} \quad \text{(velocity)}$$

1.1 Quadratic Drag Equation

1 Background 🧐

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

$$\vec{r} = \begin{pmatrix} x \\ y \end{pmatrix} \quad (\text{position})$$

$$\vec{v} = \frac{d\vec{r}}{dt} \quad (\text{velocity})$$

$$\vec{g} = \begin{pmatrix} 0 \\ -g \end{pmatrix} \quad (\text{gravitation acceleration})$$

1.1 Quadratic Drag Equation

1 Background 🧐

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

$$\vec{r} = \begin{pmatrix} x \\ y \end{pmatrix} \quad (\text{position})$$

$$\vec{v} = \frac{d\vec{r}}{dt} \quad (\text{velocity})$$

$$\vec{g} = \begin{pmatrix} 0 \\ -g \end{pmatrix} \quad (\text{gravitation acceleration})$$

$$k = \text{"constant"} \quad (\text{drag constant})$$

1.1 Quadratic Drag Equation

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

$$\vec{r} = \begin{pmatrix} x \\ y \end{pmatrix} \quad (\text{position})$$

$$\vec{v} = \frac{d\vec{r}}{dt} \quad (\text{velocity})$$

$$\vec{g} = \begin{pmatrix} 0 \\ -g \end{pmatrix} \quad (\text{gravitation acceleration})$$

$$k = \text{"constant"} \quad (\text{drag constant})$$

Let $g = 1$, $k = 1$, & $v_\infty = 1$ to focus on scale-independent features

1.2 RK4 Method for Systems

1 Background 🧐

$$\frac{d\vec{u}}{dt} = \vec{f}(t, \vec{u})$$

1.2 RK4 Method for Systems

$$\frac{d\vec{u}}{dt} = \vec{f}(t, \vec{u})$$

$$\vec{k}_1 = \vec{f}(t_i, \vec{u}_i)$$

$$\vec{k}_2 = \vec{f}\left(t_i + \frac{h}{2}, \vec{u}_i + \frac{h}{2}\vec{k}_1\right)$$

$$\vec{k}_3 = \vec{f}\left(t_i + \frac{h}{2}, \vec{u}_i + \frac{h}{2}\vec{k}_2\right)$$

$$\vec{k}_4 = \vec{f}(t_i + h, \vec{u}_i + h\vec{k}_3)$$

1.2 RK4 Method for Systems

$$\frac{d\vec{u}}{dt} = \vec{f}(t, \vec{u})$$

$$\vec{k}_1 = \vec{f}(t_i, \vec{u}_i)$$

$$\vec{k}_2 = \vec{f}\left(t_i + \frac{h}{2}, \vec{u}_i + \frac{h}{2}\vec{k}_1\right)$$

$$\vec{k}_3 = \vec{f}\left(t_i + \frac{h}{2}, \vec{u}_i + \frac{h}{2}\vec{k}_2\right)$$

$$\vec{k}_4 = \vec{f}(t_i + h, \vec{u}_i + h\vec{k}_3)$$

$$\vec{u}_{i+1} = \vec{u}_i + \frac{h}{6}(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4) \quad t_{i+1} = t_i + h$$

1.2 RK4 Method for Systems

```
def calculate(t_0, u_0, h, diff, should_exit):  
    t = [t_0]  
    u = [u_0]  
  
    while not should_exit(t[-1], u[-1]):  
        k_1 = diff(t[-1], u[-1])  
        k_2 = diff(t[-1] + h/2, u[-1] + h/2 * k_1)  
        k_3 = diff(t[-1] + h/2, u[-1] + h/2 * k_2)  
        k_4 = diff(t[-1] + h, u[-1] + h * k_3)  
  
        u_next = u[-1] + h/6 * (k_1 + 2*k_2 + 2*k_3 + k_4)  
        u.append(u_next)  
        t_next = t[-1] + h  
        t.append(t_next)  
  
    return np.array(t), np.array(u)
```

1.3 Projectile Equations

$$\vec{u} = \begin{pmatrix} x \\ y \\ v_x \\ v_y \end{pmatrix}$$
$$\frac{d\vec{u}}{dt} = \begin{pmatrix} v_x \\ v_y \\ -kvv_x \\ -g - kvv_y \end{pmatrix}$$

1.4 Equation Implementation

```
def u_prime(t, u):  
    k = 1  
    g = 1  
  
    x, y, v_x, v_y = u  
    speed = np.sqrt(v_x**2 + v_y**2)  
    drag_part = k * speed  
    drag_x = drag_part * v_x  
    drag_y = drag_part * v_y  
  
    return np.array([  
        v_x,  
        v_y,  
        -drag_x,  
        -g - drag_y,  
    ])
```


1.5 Launching Projectiles

```
def below_ground(t, u):  
    y = u[1]  
    return y < 0  
  
def launch(v_0, should_exit=below_ground):  
    t_0 = 0.0  
    h = 0.001  
    v_x, v_y = v_0  
    u_0 = np.array([0, 0, v_x, v_y])  
  
    t, u = rk4.calculate(t_0, u_0, h, u_prime, should_exit)  
  
    return t, u
```

2 Interdependence of Motion Components

2.1 Equations

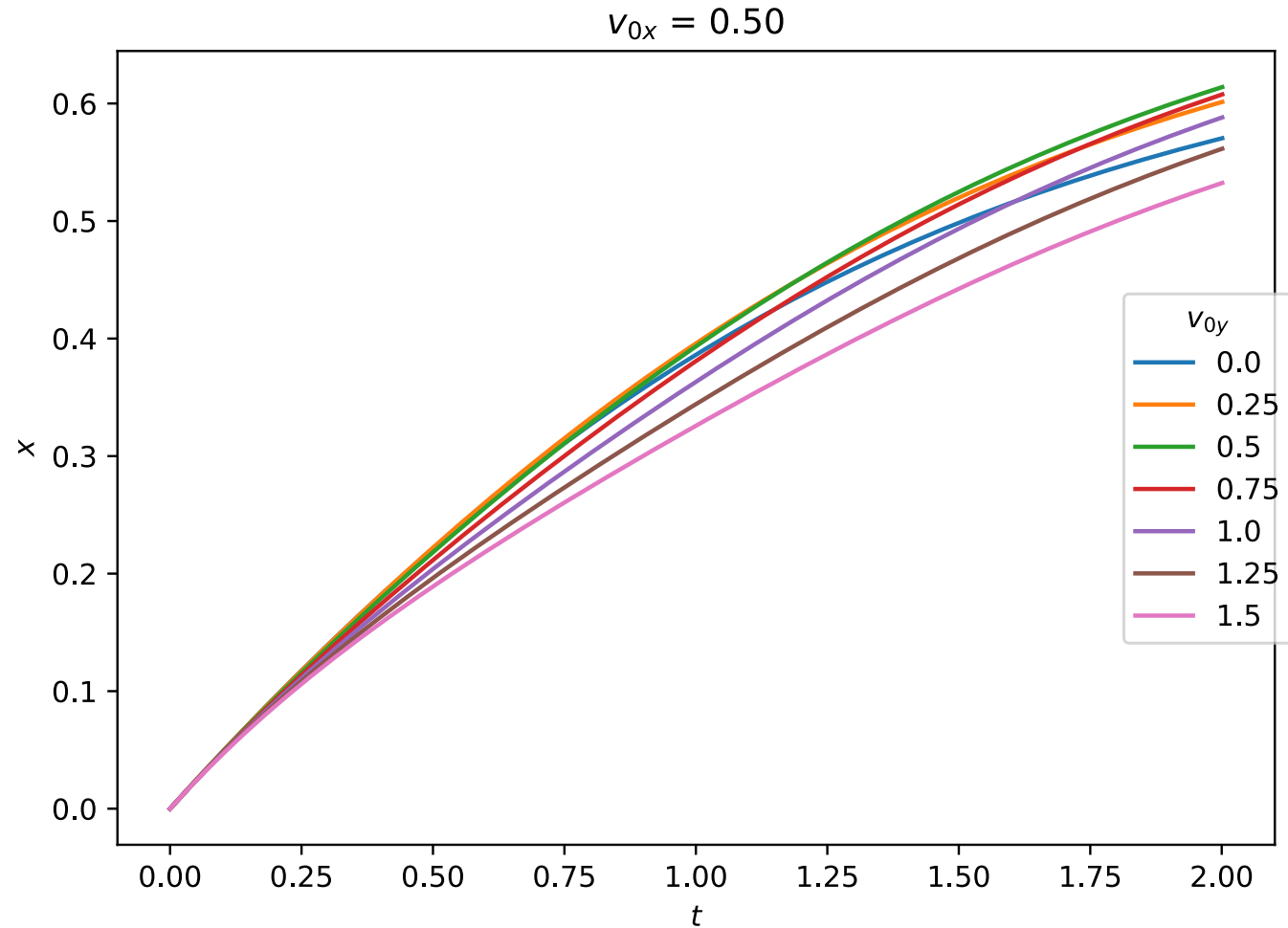
- Conditions for independence
 - $\frac{dv_x}{dt}$ does not depend on y or v_y
 - $\frac{dv_y}{dt}$ does not depend on x or v_x
- That is not the case for quadratic drag

$$\frac{dv_x}{dt} = -kvv_x = -kv_x \sqrt{v_x^2 + v_y^2}$$

$$\frac{dv_y}{dt} = -g - kvv_y = -g - kv_y \sqrt{v_x^2 + v_y^2}$$

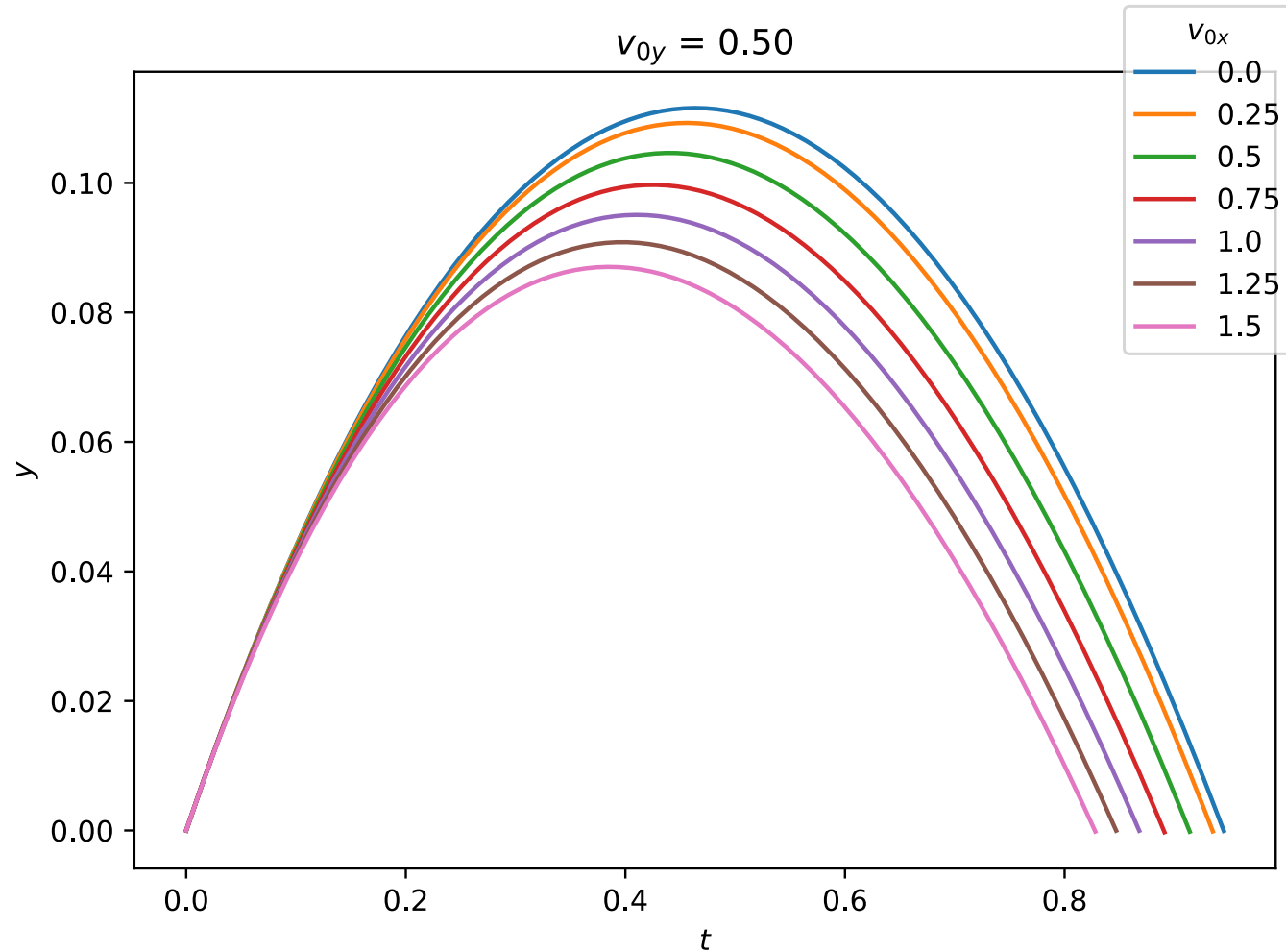
2.2 x vs t as v_{0y} Varies

2 Interdependence of Motion Components 🧐



2.3 y vs t as v_{0x} Varies

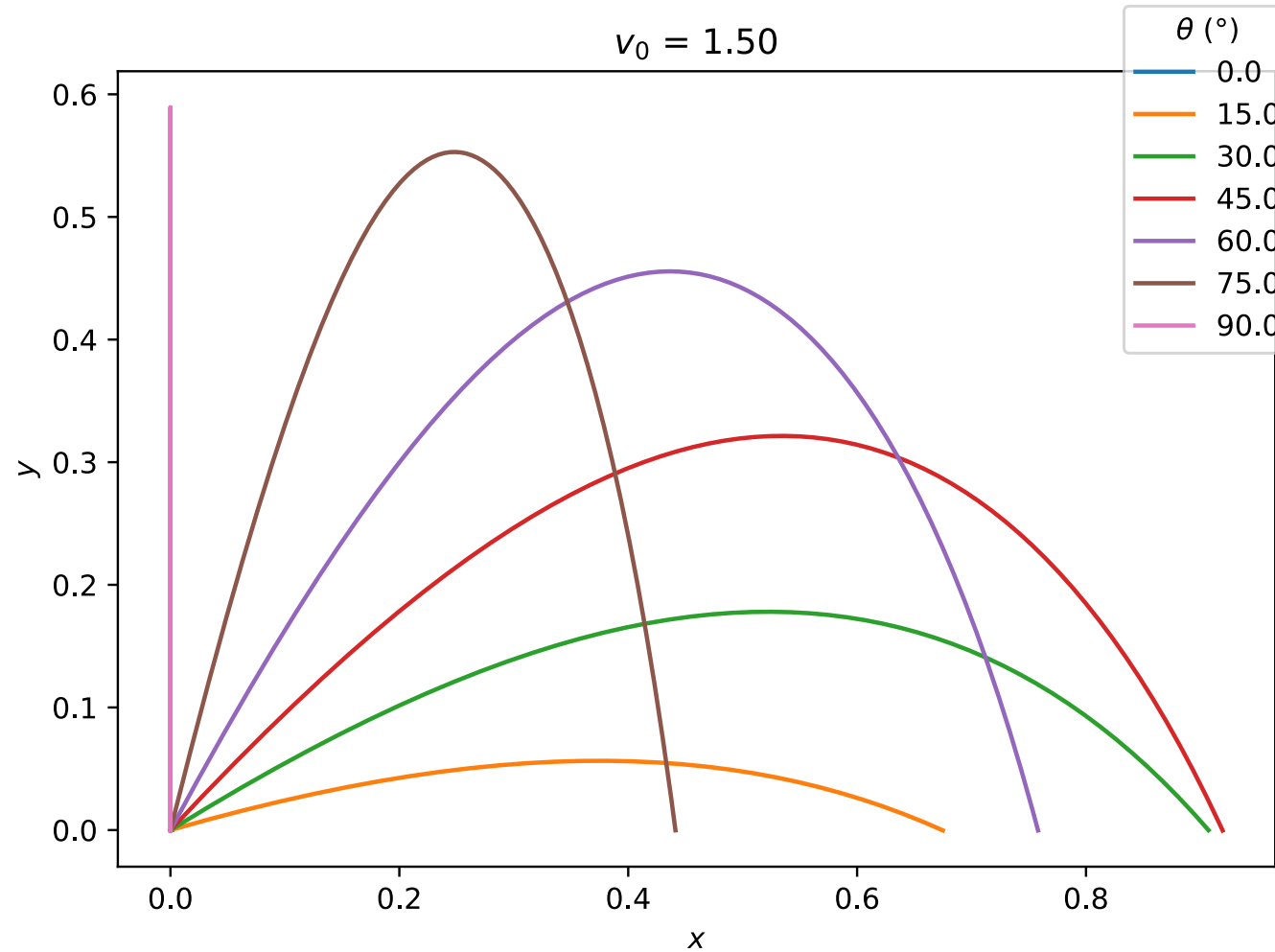
2 Interdependence of Motion Components 🧐



3 Trajectory Shapes

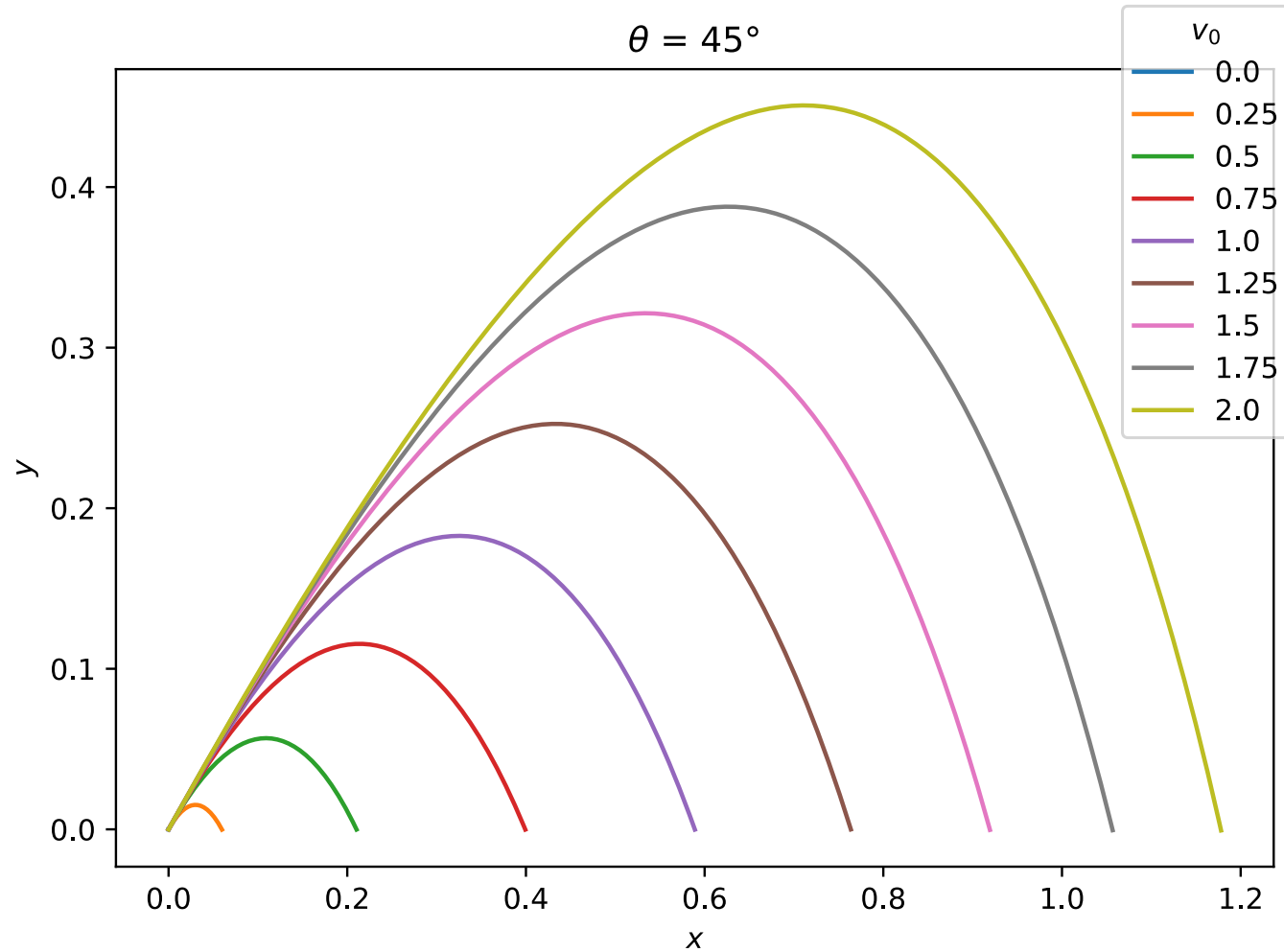
3.1 Trajectory as θ Varies

3 Trajectory Shapes 🧐



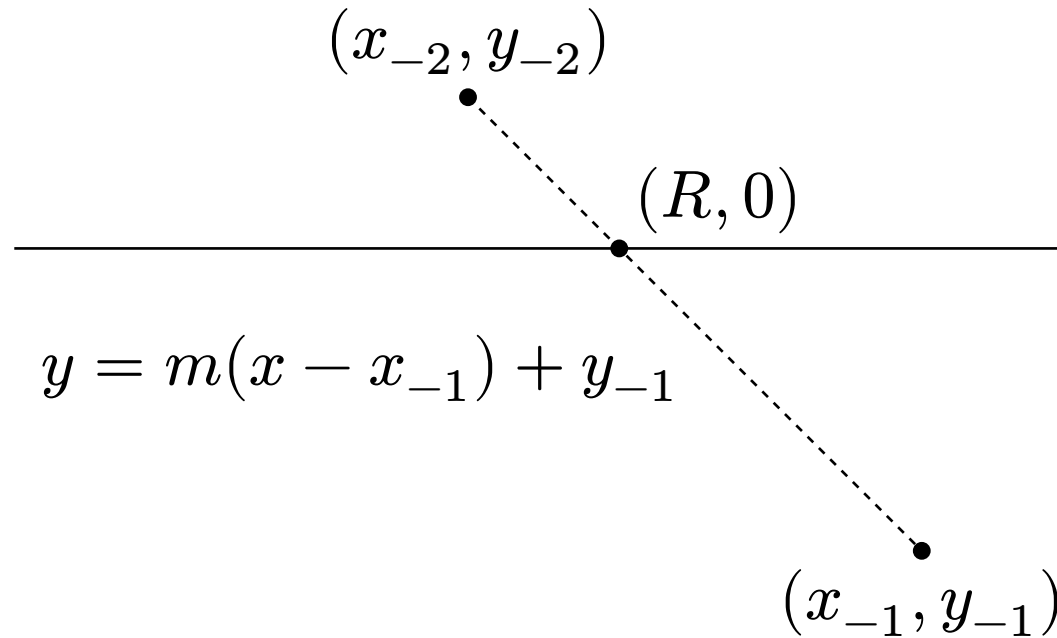
3.2 Trajectory as v_0 Varies

3 Trajectory Shapes 🧐

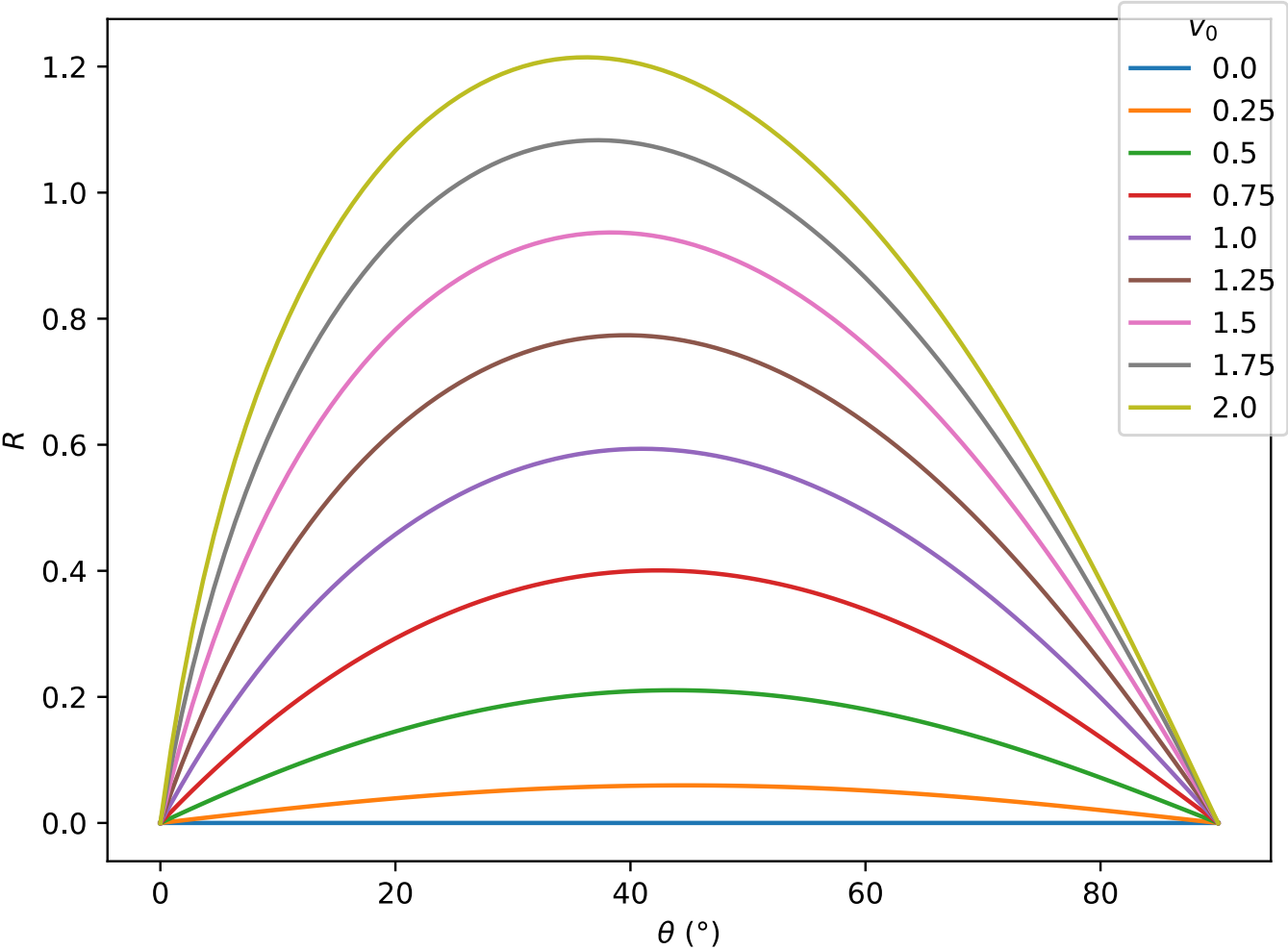


4 Firing Range

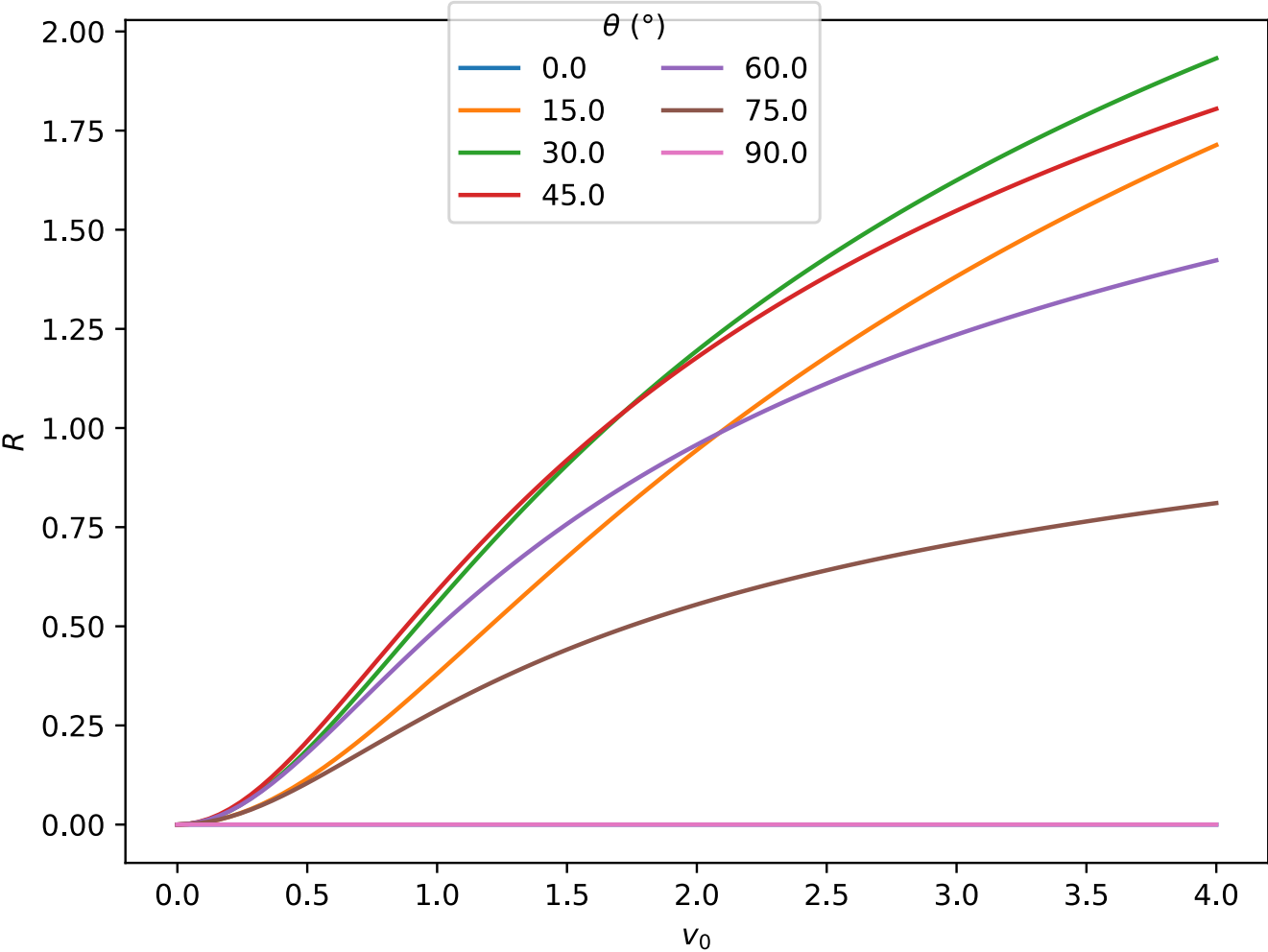
4.1 Diagram



4.2 R vs θ as v_0 Varies



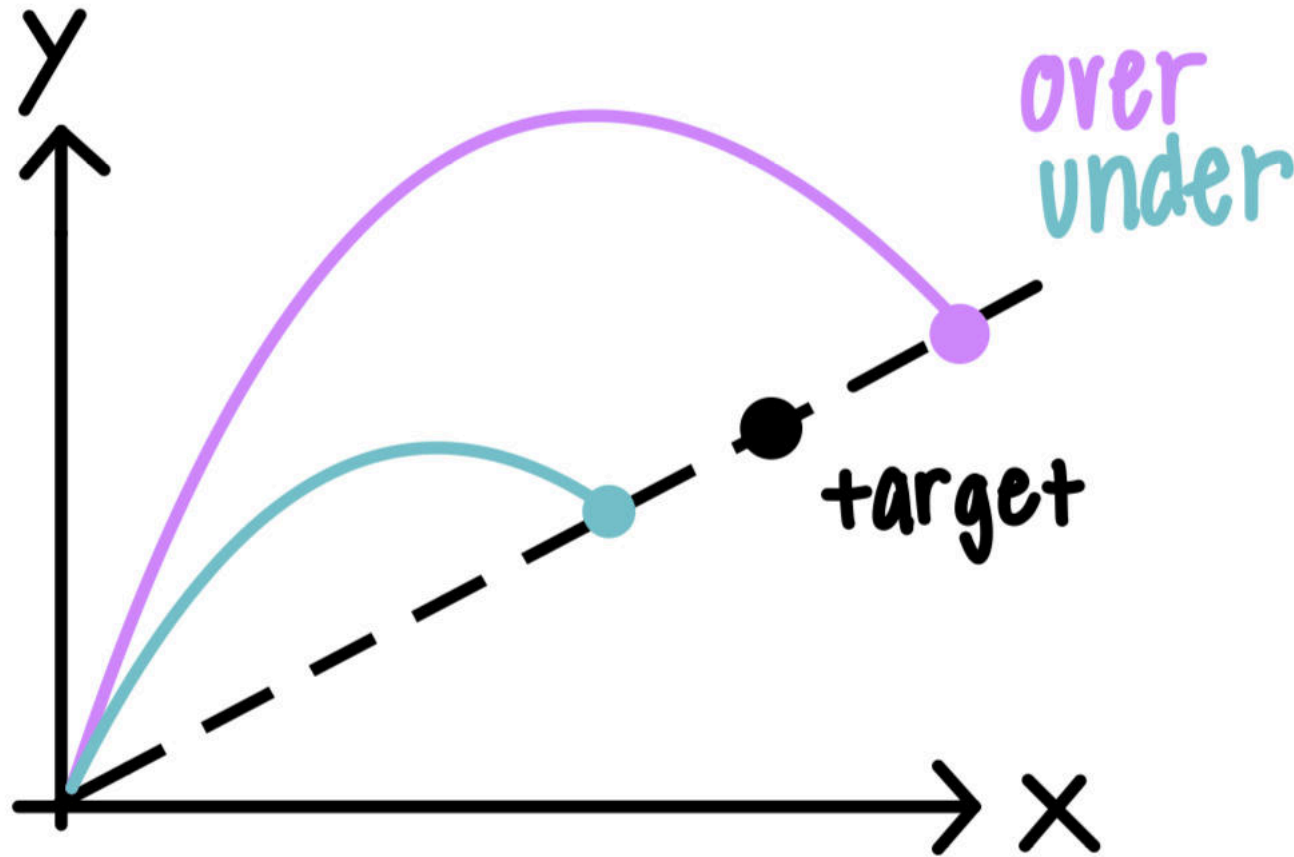
4.3 R vs v_0 as θ Varies



5 Hitting a Fixed Target

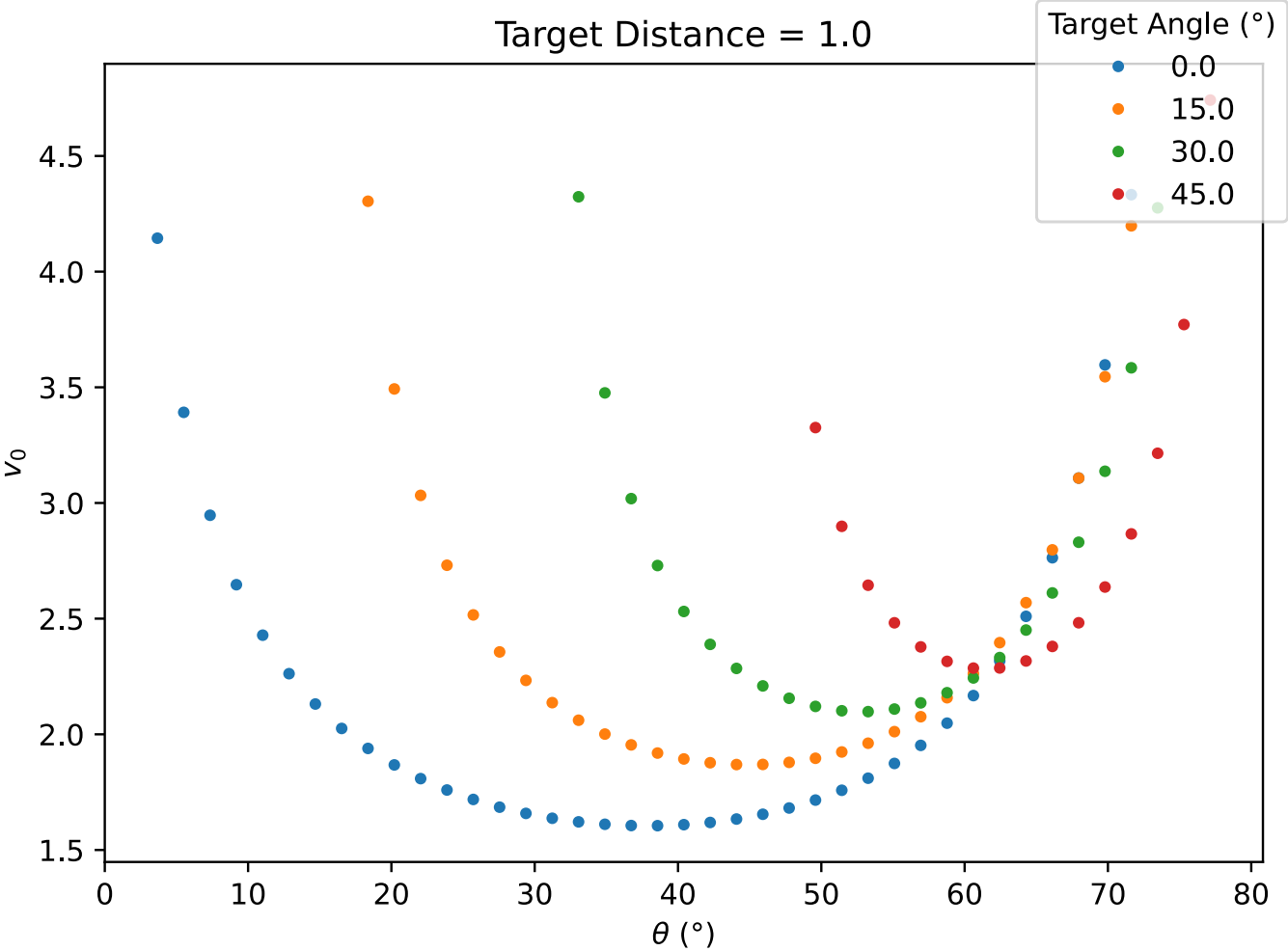
5.1 Over/Under-Shooting the Target

5 Hitting a Fixed Target 🧐



5.2 v vs θ as Target Angle Varies

5 Hitting a Fixed Target 🧐



5.3 v vs θ as Target Distance Varies

5 Hitting a Fixed Target 🧐

