Numerical Elements -- Kicking a Football

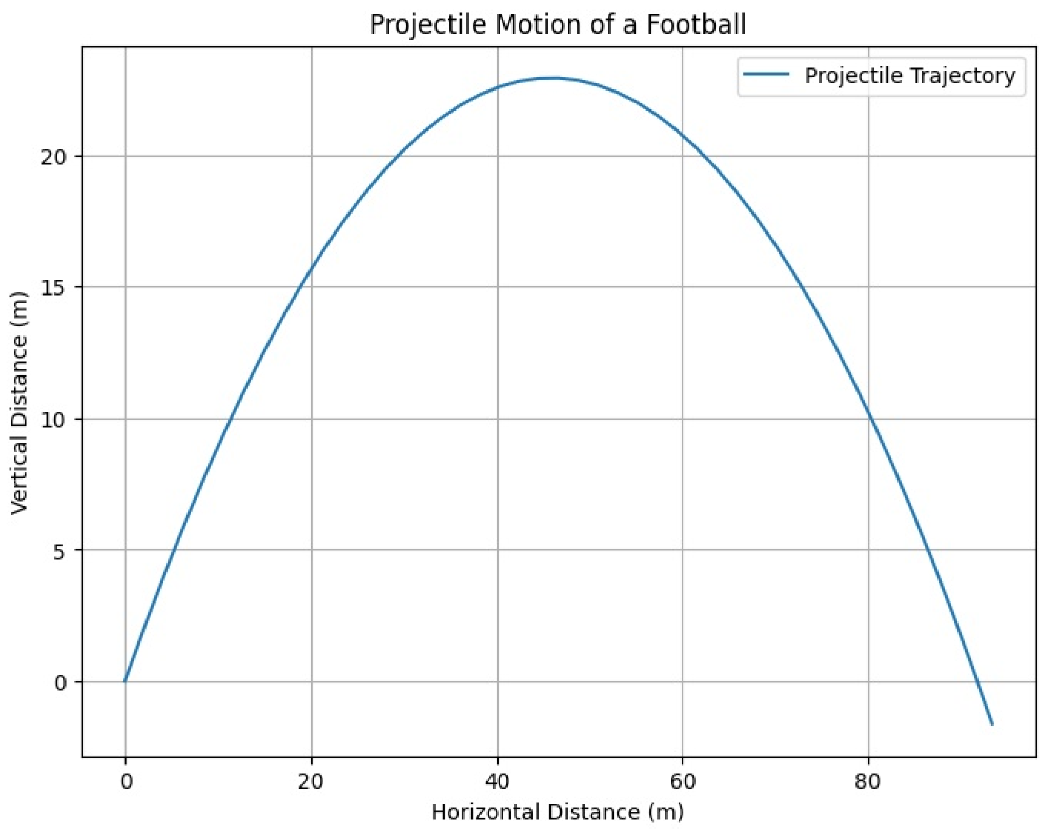
A 1 kg football is kicked from the ground with an initial velocity of $v_0 = 30 m/s$ at an angle of $\theta = 45$ above the horizontal.

a. (5pts) First, let's assume that air resistance is negligible and $g = 9.81 m/s^2$.

1. Initialize any constants and define the initial conditions. **Remember:** `np.cos()` and `np.sin()` use radians and not degrees.
2. Find the trajectories of the motion for the projectile analytically (e.g. *x(t)* and *y(t)*).
3. Use these equations to calculate: (i) the maximum height reached by the ball, (ii) the total time of flight, and (iii) the horizontal range of the ball.
4. Using this information from part 2, write a program to simulate the projectile motion and compute the same quantities numerically. You should use a time step of $\Delta t = 0.1$
5. Plot the trajectory of the projectile (e.g. y vs. x) and print the maximum height, the total time of flight, and the horizontal range of the ball.

In [4]:

```python
#Part_A_5pts

import numpy as np
import matplotlib.pyplot as plt


# Constants
g = 9.81
v0 = 30
theta_deg = 45
theta_rad = np.radians(theta_deg)
m = 1

# Initial conditions
v0x = v0 * np.cos(theta_rad)
v0y = v0 * np.sin(theta_rad)



t_max_height = v0y / g
y_max = (v0y**2) / (2 * g)

t_flight = 2 * v0y / g

range_ = v0x * t_flight

delta_t = 0.1
t_values = np.arange(0, t_flight + delta_t, delta_t)

x_values = v0x * t_values
y_values = v0y * t_values - 0.5 * g * t_values**2

# Plotting the trajectory
plt.figure(figsize=(8, 6))
plt.plot(x_values, y_values, label="Projectile Trajectory")
plt.title("Projectile Motion of a Football")
plt.xlabel("Horizontal Distance (m)")
plt.ylabel("Vertical Distance (m)")
plt.grid(True)
plt.legend()
plt.show()

# Printing the results
max_height = y_max
total_time = t_flight
horizontal_range = range_

(max_height, total_time, horizontal_range)
```

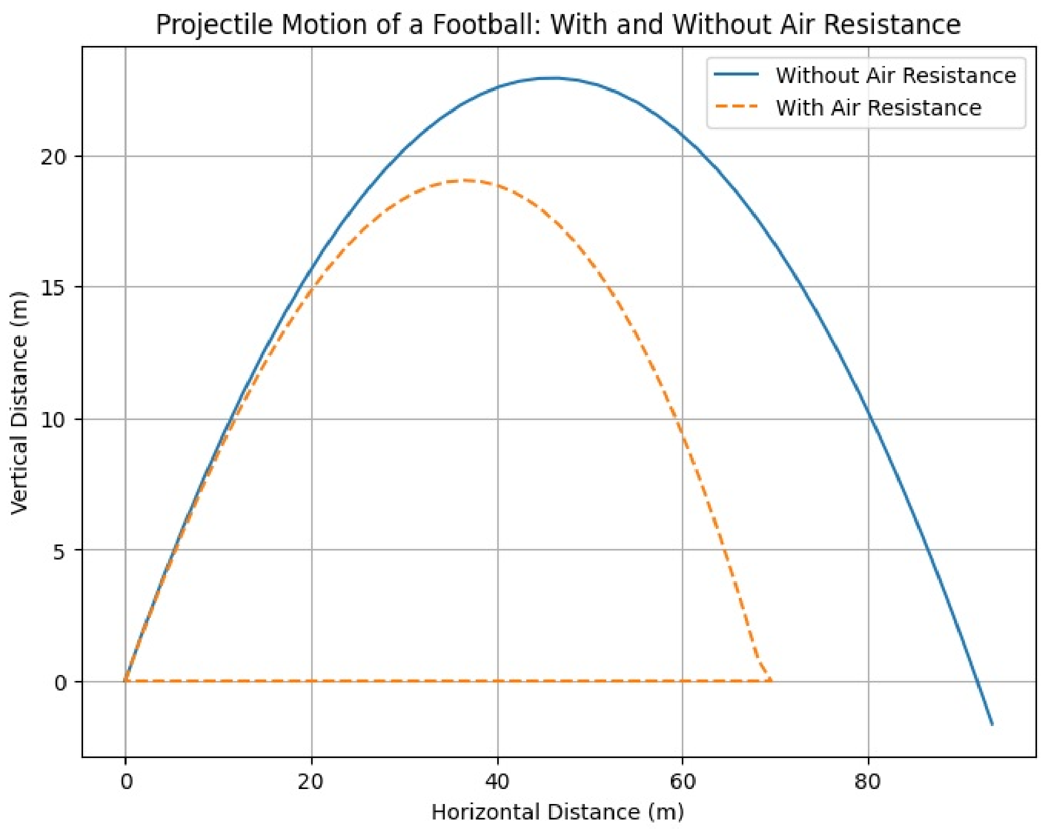



```
Out[4]: (22.935779816513755, 4.324812117348913, 91.74311926605502)
```

b. (5pts) Now consider the ball experiences a linear drag force proportional to its velocity, modeled as $F_d = -b\mathbf{v}$, where $b = 0.1 kg/s$ is the drag coefficient.

1. Initialize any new constants. Again, **remember:** `np.cos()` and `np.sin()` use radians and not degrees.
2. Find the trajectories of the motion for the projectile analytically (e.g. *x(t)* and *y(t)*). You can reference your notes from class lecture and/or homework problems.
3. Using this information from part 2, write a program to simulate the projectile motion and compute the same quantities numerically. You should use a time step of $\Delta t = 0.1$
4. Plot the trajectory of the projectile (e.g. y vs. x) on the same plot as the trajectory of the football assuming no air resistance.

```
In [5]: #Part_B_5pts


b = 0.1

t_values_drag = np.arange(0, t_flight + delta_t, delta_t)
x_values_drag = np.zeros_like(t_values_drag)
y_values_drag = np.zeros_like(t_values_drag)

vx = v0x
vy = v0y

for i in range(1, len(t_values_drag)):
    t = t_values_drag[i]
    # Update velocities
    vx = vx * np.exp(-b * delta_t / m)
    vy = (vy + g * m / b) * np.exp(-b * delta_t / m) - g * m / b
    # Update positions
    x_values_drag[i] = x_values_drag[i-1] + vx * delta_t
    y_values_drag[i] = y_values_drag[i-1] + vy * delta_t

    if y_values_drag[i] < 0:
        y_values_drag[i] = 0
        break

plt.figure(figsize=(8, 6))
plt.plot(x_values, y_values, label="Without Air Resistance")
plt.plot(x_values_drag, y_values_drag, label="With Air Resistance", linestyle='--')
plt.title("Projectile Motion of a Football: With and Without Air Resistance")
plt.xlabel("Horizontal Distance (m)")
plt.ylabel("Vertical Distance (m)")
plt.grid(True)
plt.legend()
plt.show()
```

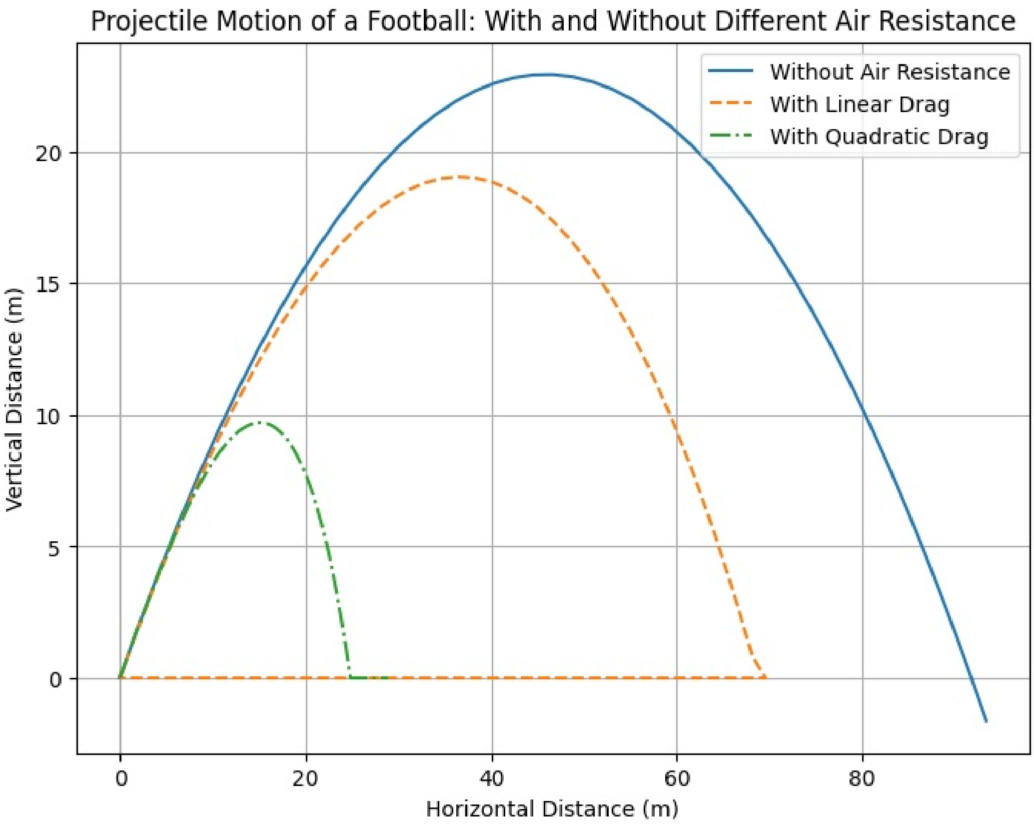


c. (10pts) Now consider that the football experiences a quadratic drag force, modeled as $F_d = -c_d v^2 \hat{v}$, where $c_d = 0.05 kg/m$ is the drag coefficient.

1. Initialize any new constants. Again, **remember:** `np.cos()` and `np.sin()` use radians and not degrees.
2. Find the trajectories of the motion for the projectile analytically (e.g. *x(t)* and *y(t)*). You can reference your notes from class lecture and/or homework problems.
3. Here we will finally use python to help us solve our coupled differential equations. Using the trajectories from part 2 and your knowledge about the relationship between position and velocity, complete the program to simulate the projectile motion. For more information on the `solve_ivp` function from `scipy.integrate` see the link **here**.
4. Plot the trajectory of the projectile (e.g. y vs. x) on the same plot as above.

In [8]:
```
from scipy.integrate import solve_ivp


cd = 0.05


def equations(t, state):

    x, y_pos, vx, vy = state


    v = np.sqrt(vx**2 + vy**2)


    dvx_dt = -cd * v * vx / m
    dvy_dt = -cd * v * vy / m - g

    return [vx, vy, dvx_dt, dvy_dt]


y0 = [0, 0, v0x, v0y]


t_span = (0, t_flight)
t_eval = np.linspace(t_span[0], t_span[1], 500)

# Solve the ODE
sol = solve_ivp(equations, t_span, y0, t_eval=t_eval)


x_num = sol.y[0]
y_num = sol.y[1]


y_num = np.where(y_num < 0, 0, y_num)

plt.figure(figsize=(8, 6))
```

```python
plt.plot(x_values, y_values, label="Without Air Resistance")
plt.plot(x_values_drag, y_values_drag, label="With Linear Drag", linestyle='--')
plt.plot(x_num, y_num, label="With Quadratic Drag", linestyle='-.')
plt.title("Projectile Motion of a Football: With and Without Different Air Resistance")
plt.xlabel("Horizontal Distance (m)")
plt.ylabel("Vertical Distance (m)")
plt.grid(True)
plt.legend()
plt.show()
```



d. (20pts) Now that we have code to solve coupled differential equations, let's explore a real world scenario.

The Denver Broncos are scheduled to play against the Kansas City Chiefs on November 10th. The field goal kicker for the Broncos is investigating his maximum distance he can successfully kick the football to score points for his team. Let's assume that the football experiences a quadratic drag force, modeled as $F_d = -\frac{1}{2}\rho c_d A v^2 \hat{v}$ in both cities and the field goal kicker strikes the football with an initial velocity of $v_0 = 30 m/s$ and an angle of $\theta = 45$ from the horizontal each time he kicks.

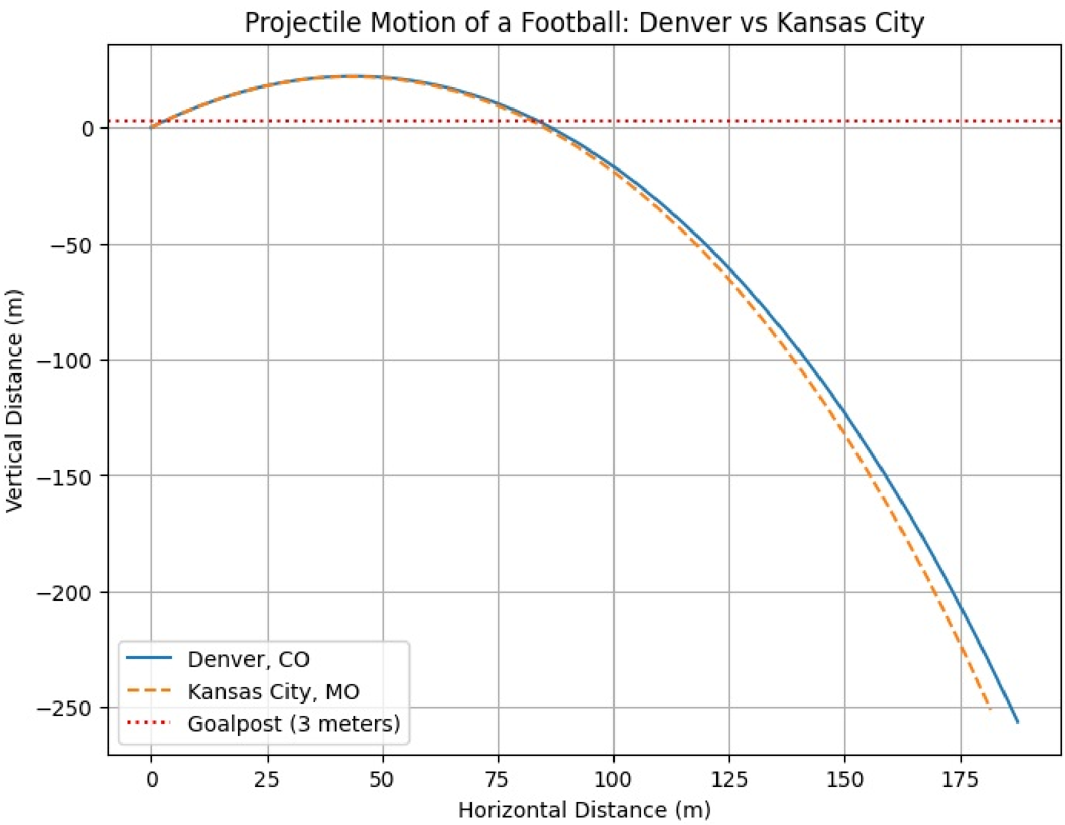
Using the `solve_ivp` code from above, plot the trajectories of the football in Denver, CO and Kansas City, MO. You will need to use the internet to find the elevation and air density for each city and assume that $c_d = 0.05 kg/m$ and the cross-sectional area of a football is $A = 0.038 m^2$.

1. If the goal post is 3 meters above the ground, what is the longest field goal the kicker can successfully make in Denver?
2. If the goal post is 3 meters above the ground, what is the longest field goal the kicker can successfully make in Kansas City?
3. Discuss how environmental factors, such as altitude and air density, impact the projectile motion.

In [9]:

```python
#Part_D_20pts

cd = 0.05
A = 0.038
rho_denver = 0.909
rho_kc = 1.168
g = 9.81
v0 = 30
theta_deg = 45
theta_rad = np.radians(theta_deg)


v0x = v0 * np.cos(theta_rad)
v0y = v0 * np.sin(theta_rad)

def equations_with_drag(t, state, rho):
    x, y_pos, vx, vy = state

    v = np.sqrt(vx**2 + vy**2)

    dvx_dt = -0.5 * rho * cd * A * v * vx / m
    dvy_dt = -0.5 * rho * cd * A * v * vy / m - g
```

```
    return [vx, vy, dvx_dt, dvy_dt]

y0 = [0, 0, v0x, v0y]

t_span = (0, 10)
t_eval = np.linspace(t_span[0], t_span[1], 500)


sol_denver = solve_ivp(equations_with_drag, t_span, y0, t_eval=t_eval, args=(rho_denver,))
sol_kc = solve_ivp(equations_with_drag, t_span, y0, t_eval=t_eval, args=(rho_kc,))

x_denver = sol_denver.y[0]
y_denver = sol_denver.y[1]

x_kc = sol_kc.y[0]
y_kc = sol_kc.y[1]

goalpost_height = 3
x_denver_final = x_denver[y_denver >= goalpost_height][-1]
x_kc_final = x_kc[y_kc >= goalpost_height][-1]

plt.figure(figsize=(8, 6))
plt.plot(x_denver, y_denver, label="Denver, CO")
plt.plot(x_kc, y_kc, label="Kansas City, MO", linestyle='--')
plt.axhline(y=goalpost_height, color='r', linestyle=':', label="Goalpost (3 meters)")
plt.title("Projectile Motion of a Football: Denver vs Kansas City")
plt.xlabel("Horizontal Distance (m)")
plt.ylabel("Vertical Distance (m)")
plt.grid(True)
plt.legend()
plt.show()

(x_denver_final, x_kc_final)
```



Out[9]: (83.18787960802355, 81.86099273435197)

Processing math: 100%