PROBLEMA 6.3 - TIR PARABOLIC

Gisela Martí Guerrero

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
x_o,y_o = 0,0            # posicions inicials
v_o = 25                 # velocitat inicial
theta = np.radians(60)   # angle
g = 9.81                 # gravetat
vx_o = v_o*np.cos(theta) # velocitat inicial x
vy_o = v_o*np.sin(theta) # velocitat inicial y
dt = 0.01
time = np.arange(0,5,dt)
methods = ["Simple", "Modified","Improved"]
```

In [3]:
```
def fx():
    return vx_o

def fy(t):
    return vy_o - g*t
```

In [4]:
```
def euler_x(x,f,dt,mode):
    if mode.lower() == "simple":
        a,b,d,g = 1,0,0,0
    elif mode.lower() == "modified":
        a,b,d,g = 0,1,0.5,0.5
    elif mode.lower() == "improved":
        a,b,d,g = 0.5,0.5,1,1

    xt = x + dt*(a*f() + b*f())
    return xt
```

In [5]:
```
def euler_y(y,t,f,dt,mode):
    if mode.lower() == "simple":
        a,b,d,g = 1,0,0,0
    elif mode.lower() == "modified":
        a,b,d,g = 0,1,0.5,0.5
    elif mode.lower() == "improved":
        a,b,d,g = 0.5,0.5,1,1

    yt = y + dt*(a*f(t) + b*f(t+g*dt))
    return yt
```

In [6]:
```
X = [[x_o] for _ in range(3)]
Y = [[y_o] for _ in range(3)]
for t in time:
    for i,method in enumerate(methods):
        x_i = X[i][-1]
        y_i = Y[i][-1]
        x_next = euler_x(x_i,fx,dt,method)
        y_next = euler_y(y_i,t,fy,dt,method)

        X[i].append(x_next)
        Y[i].append(y_next)
```

In [7]:
```
X = [x_val[:-1] for x_val in X]
Y = [y_val[:-1] for y_val in Y]
```

In [8]:
```
x_exact = vx_o*time + x_o
y_exact = -0.5*g*time**2 + vy_o*time + y_o
```

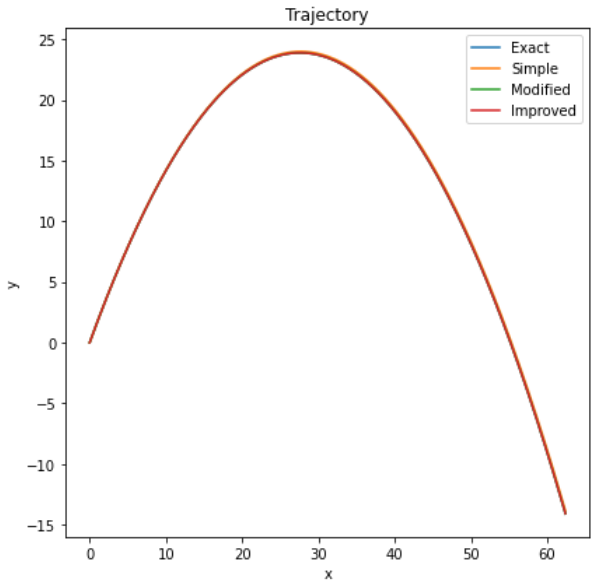
In [9]:
```
output_data = np.column_stack((time, X[0], X[1], X[2], x_exact, Y[0], Y[1], Y[2], y
header = "Time,  X_Simple,  X_Modified,  X_Improved,  X_exact,  Y_Simple,  Y_Modifi
np.savetxt("6.3_output_parabolic.txt", output_data, header=header, delimiter='\t',
```

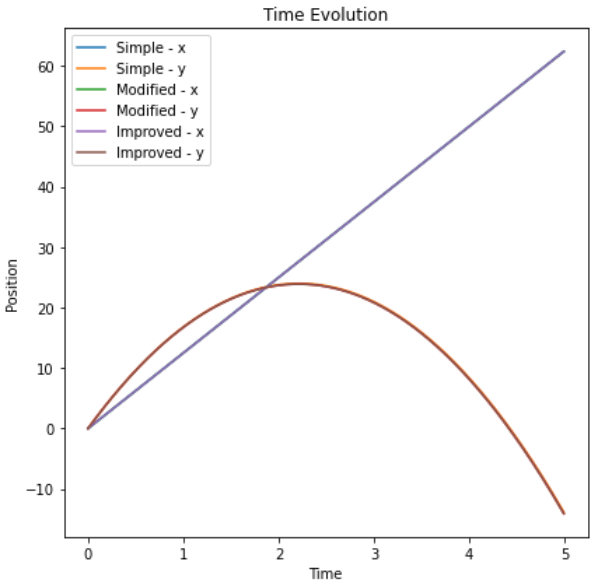
In [10]:
```
plt.figure(figsize=(12, 6))

# Plot trajectory
plt.subplot(1, 2, 1)
plt.plot(x_exact, y_exact, label='Exact')
for i, method in enumerate(methods):
    plt.plot(X[i], Y[i], label=method)
plt.title('Trajectory')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()

# Plot time evolution
plt.subplot(1, 2, 2)
for i, method in enumerate(methods):
    plt.plot(time, X[i], label=f'{method} - x')
    plt.plot(time, Y[i], label=f'{method} - y')

plt.title('Time Evolution')
plt.xlabel('Time')
plt.ylabel('Position')
plt.legend()
plt.tight_layout()
plt.show()
```


Trajectory
Exact
Simple
Modified
Improved
x
y


Time Evolution
Simple - x
Simple - y
Modified - x
Modified - y
Improved - x
Improved - y
Time
Position