

# **Equations of Motion**

Computational Physics

Equations of Motion

# Outline

- Equation of motion in 3 dimensions
- Projectile Motion Problem
- 3D Plotting in MATLAB
- Equation of motion for Orbits
- Orbital Properties
- $\vec{r}$  unit vector

# Motion in Three Dimensions

Independent Equations  
for each dimension

$$\frac{d^2 x}{dt^2} = \frac{F_x}{m}$$

$$\frac{d^2 y}{dt^2} = \frac{F_y}{m}$$

$$\frac{d^2 z}{dt^2} = \frac{F_z}{m}$$

Single Vector Equation

$$\frac{d^2 \vec{r}}{dt^2} = \frac{\vec{F}}{m}$$

in MATLAB:

$$r = [x, y, z]$$

$$F = [F_x, F_y, F_z]$$

# 3D Solution in MATLAB

## *Independent Equations - Euler Method*

Initialization not shown explicitly

```
% initialize t(1),  
% initialize vx(1), vy(1), vz(1), x(1), y(1),z(1)  
  
% iteration through time steps  
% dt is time step  
% vx,vy,vz are velocities in x,y,z coordinates  
for i=1:nSteps  
    t(i+1) = t(i) + dt;  
  
    vx(i+1) = vx(i) + Fx(i)/m * dt;  
    vy(i+1) = vy(i) + Fy(i)/m * dt;  
    vz(i+1) = vz(i) + Fz(i)/m * dt;  
  
    x(i+1) = x(i) + vx(i) * dt;  
    y(i+1) = y(i) + vy(i) * dt;  
    z(i+1) = z(i) + vz(i) * dt;  
end
```

Velocity Components {

Position Components {

# **3D Solution in MATLAB**

## ***Vector Equations - Euler Method***

**Initialization**

```
% note use of colon operator  
% to set initial conditions  
t(1) = 0.0;  
v(1,:) = [ vx0, vy0, vz0 ];  
x(1,:) = [ x0, y0, z0 ];
```

**Velocity Vector**

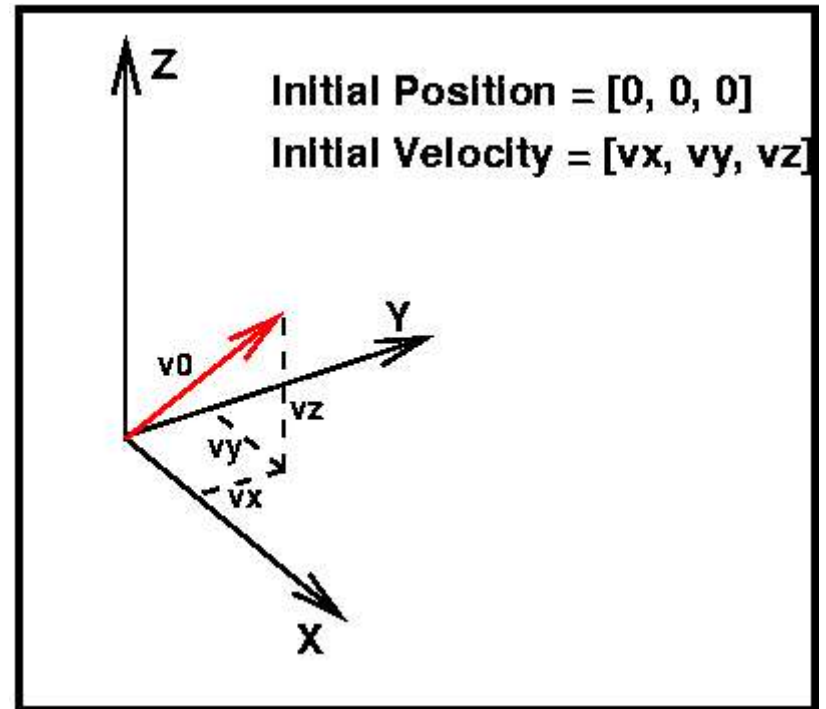
```
% iteration through time steps  
for i=1:nSteps  
    t(i+1) = t(i) + dt;
```

**Position Vector**

```
    v(i+1,:) = v(i,:) + F(i,+)/m * dt;  
  
    x(i+1,:) = x(i,:) + v(i,:) * dt;  
end
```

# Projectile Motion Problem

- Motion of particle under gravity, and eventually other realistic forces.
- Initial Conditions:
  - specify location of beginning of trajectory
  - specify initial velocity



# Equation of Motion

## *Gravity Only*

$$\frac{d^2 x}{dt^2} = \frac{F_x}{m} = 0$$

$$\frac{d^2 y}{dt^2} = \frac{F_y}{m} = 0$$

$$\frac{d^2 z}{dt^2} = \frac{F_z}{m} = -g$$

$$\frac{d^2 \vec{r}}{dt^2} = \frac{\vec{F}}{m} = -g \hat{z}$$

**Gravity is only force:  
Acceleration in -z direction**

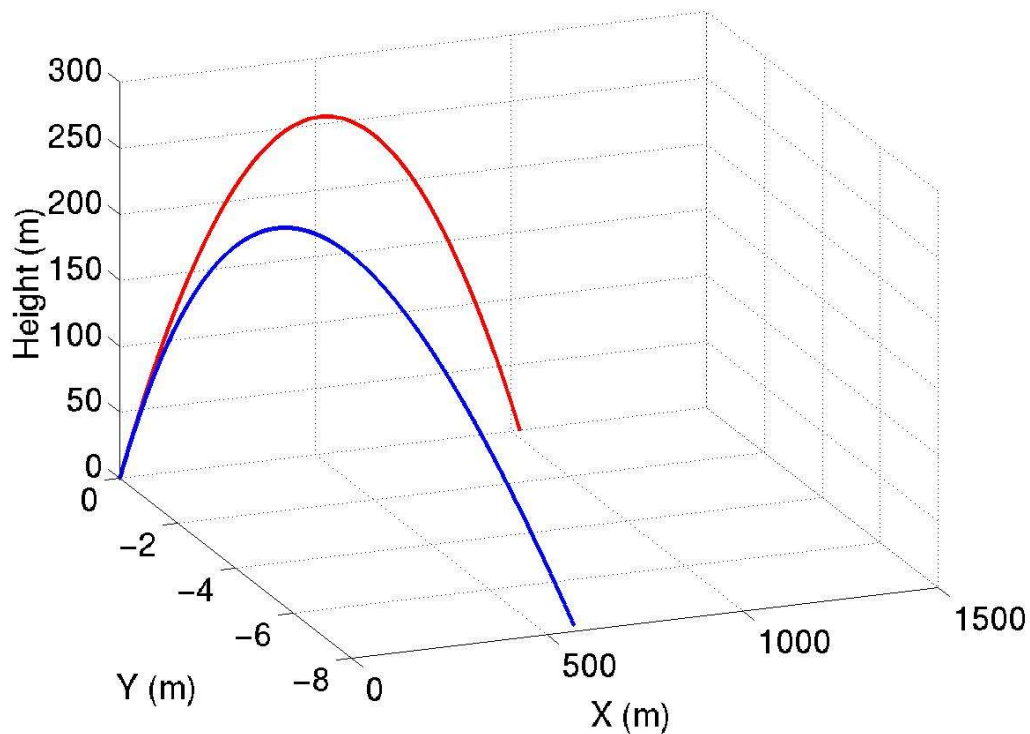


# Constants of the Motion

## *Gravity Only*

- Constants of motion are useful for evaluating whether your program works!
- No Force in X and Y directions:
  - momentum in X and Y conserved
- Force of gravity depends on position only
  - total energy is conserved
  - potential energy =  $m g z$
  - kinetic energy =  $\frac{1}{2} m |v|^2$
  - total energy:  $E = \frac{1}{2} m |v|^2 + m g z$

# Plotting in 3D in MATLAB



```
% plot3 example  
% x is array with position vrs t  
% x2 array has second solution  
  
% plot trajectory  
plot3( x(:,1), x(:,2), x(:,3), 'b');  
  
hold  
  
% plot a second solution  
plot3( x2(:,1), x2(:,2), x2(:,3), 'r');
```

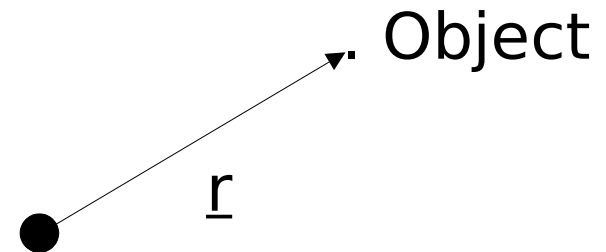
# Orbit Problem

## Equation of Motion

- Second Order ODE
- Radial Force dependent on position only:
  - Angular Momentum conserved; Motion in a plane.
  - Energy conserved.

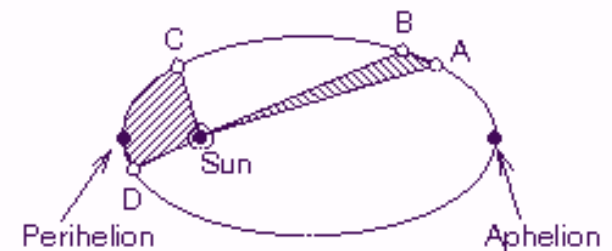
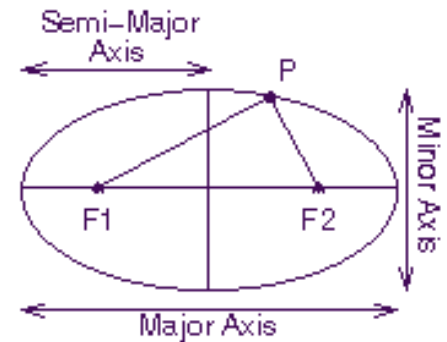
$$\frac{d^2 \vec{r}}{dt^2} = \frac{\vec{F}}{m}$$

$$\vec{F} = -\frac{G M_{\odot} m}{r^2} \hat{r}$$



# Kepler's Laws

1. The orbit of a planet around the Sun is an ellipse with the Sun at one focus of the ellipse.
2. A line joining a planet and the Sun sweeps out equal areas in equal intervals of time.
3. The squares of the sidereal periods of the planets are proportional to the cubes of their semimajor axes.



$$P^2 = a^3$$

# Initial Conditions

- 3 Dimensional, Second Order D.E.
  - 6 Numbers
  - initial position:  $r = [x, y, z]$  at time = 0
  - initial velocity:  $\dot{r} = [v_x, v_y, v_z]$  at time = 0
- Each set of initial conditions has unique orbit. Can characterize orbit with *any* six numbers that will describe it.
- Astronomers use "Orbital Elements" to specify and describe orbits.

# Orbital Elements

- Size and Shape of Orbit
  - Semimajor Axis
  - Eccentricity
- Orientation of Orbital Plane in Space
  - Inclination wrt Earth's Orbital plane
  - Longitude of Ascending Node
  - Longitude of Perihelion
- Time of Perihelion Passage

# The $\hat{r}$ unit vector.

- Gravitational Force is radial, so need unit vector in  $r$  direction to derive force.
- A convenient way to look at this, for MATLAB programs is:

$$\hat{r} = \frac{\vec{r}}{|\vec{r}|}$$