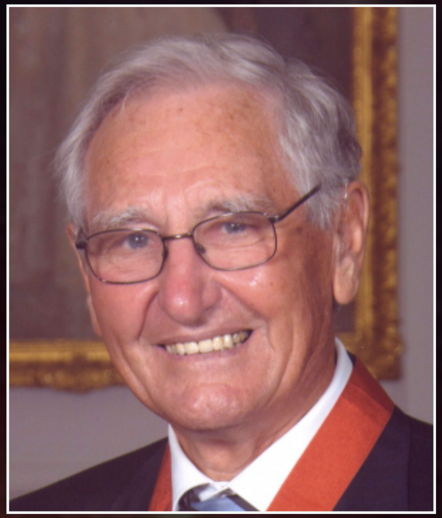
INERTIAL FRAME DRAGGING

NEAR KERR BLACK-HOLE

KERR BLACK-HOLE

- Roy Kerr (1934 -)



Discovered exact solutions of Einstein’s equation describing a spinning (uncharged) Black-Hole, which is known as Kerr Solution (1960).

- Any (isolated) Black-Hole can be described by just three quantities :

1. Mass
2. Spin
3. Electric Charge

FRAME-DRAGGING

The non-static stationary distributions of mass-energy distorts the space-time. This effect is known as Frame-Dragging.

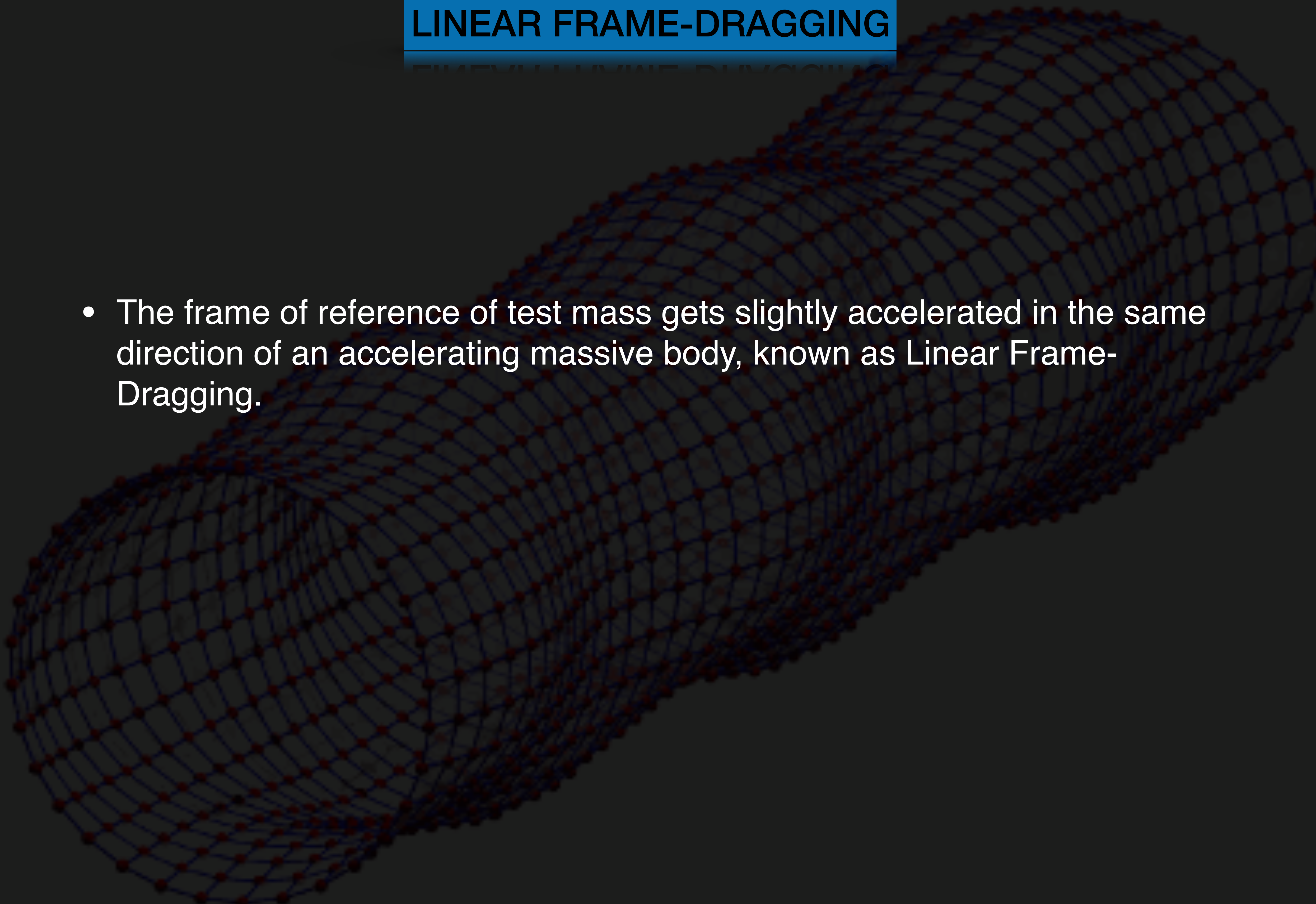
Einstein's General Theory of Relativity predicted this effect.

There are two basic types of Frame- Dragging :

1. **Linear Frame-Dragging**
2. **Rotational Frame- Dragging**

LINEAR FRAME-DRAGGING

- The frame of reference of test mass gets slightly accelerated in the same direction of an accelerating massive body, known as Linear Frame-Dragging.

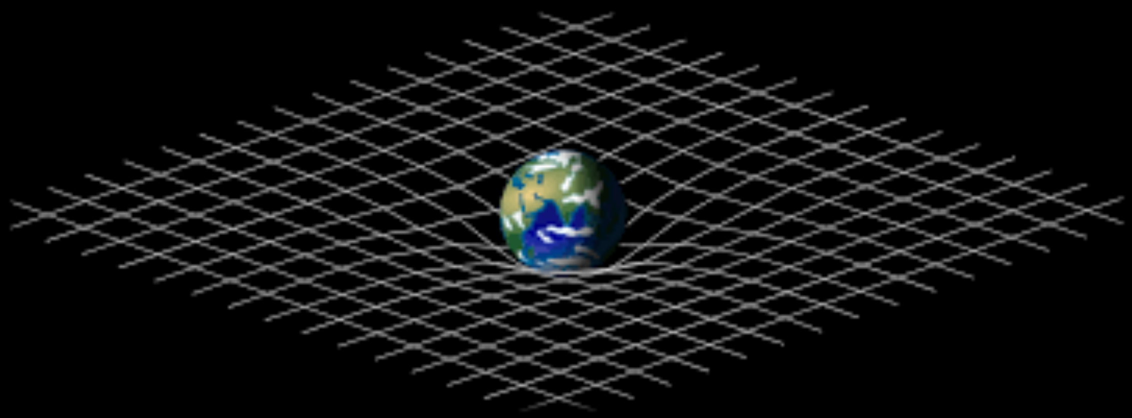


www.einstein-online.info

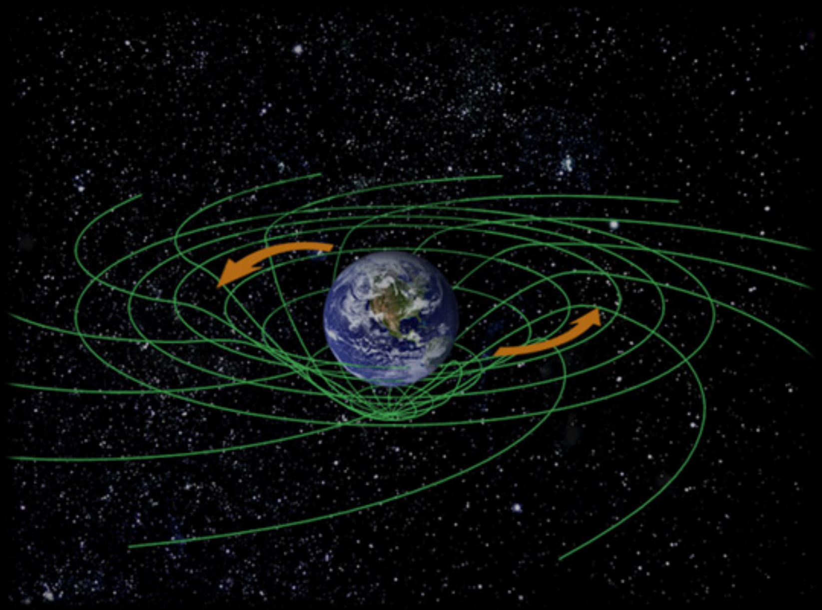
ROTATIONAL FRAME-DRAGGING

- The rotation of a massive object distorts the space-time metric locally, such that, it makes the orbit of a nearby test particle to precess around, this effect is known as Rotation Frame-Dragging.

Non-Rotating Massive Object



Rotating Massive Object



KERR SOLUTION & AZIMUTHAL VELOCITY

- **Kerr Solution line element :**

$$ds^2 = c^2\partial\tau^2 = \left(1 - \frac{2mr}{\rho^2}\right)c^2\partial t^2 + \frac{4mcraSin^2\theta}{\rho^2}\partial t\partial\phi - \frac{\rho^2}{\Delta}\partial r^2 - \rho^2\partial\theta^2 - \left((r^2 + a^2)Sin^2\theta + \frac{2mra^2Sin^4\theta}{\rho^2}\right)\partial\phi^2$$

$$\rho^2 = r^2 + a^2Cos^2\theta$$

$$\Delta = r^2 + a^2 - 2mr$$

$$a = \frac{J}{Mc}$$

$$m = \frac{GM}{c^2}$$

ϕ and t are Cyclic Coordinates

Therefore, $\dfrac{\partial \mathrm{L}}{\partial \dot{\phi}} = \text{constant}$

where, $\mathrm{L} = \dfrac{1}{2}\left[\left(1 - \dfrac{2\mathrm{mr}}{\rho^2}\right)\mathrm{c}^2\dot{\mathrm{t}}^2 + \dfrac{4\mathrm{mcraSin}^2\theta}{\rho^2}\dot{\mathrm{t}}\dot{\phi} - \dfrac{\rho^2}{\Delta}\dot{\mathrm{r}}^2 - \rho^2\dot{\theta}^2 - \left((\mathrm{r}^2 + \mathrm{a}^2)\mathrm{Sin}^2\theta + \dfrac{2\mathrm{mra}^2\mathrm{Sin}^4\theta}{\rho^2}\right)\dot{\phi}^2\right]$

$$\frac{4mcraSin^2\theta}{\rho^2}\dot{t} - 2\left((r^2 + a^2)Sin^2\theta + \frac{2mra^2Sin^4\theta}{\rho^2}\right)\dot{\phi} = constant$$

Take constant = 0

$$\frac{4mcraSin^2\theta}{\rho^2}\dot{t} - 2\left((r^2 + a^2)Sin^2\theta + \frac{2mra^2Sin^4\theta}{\rho^2}\right)\dot{\phi} = 0$$

$$\dot{\phi} \to 0 \; as \; r \to \infty$$

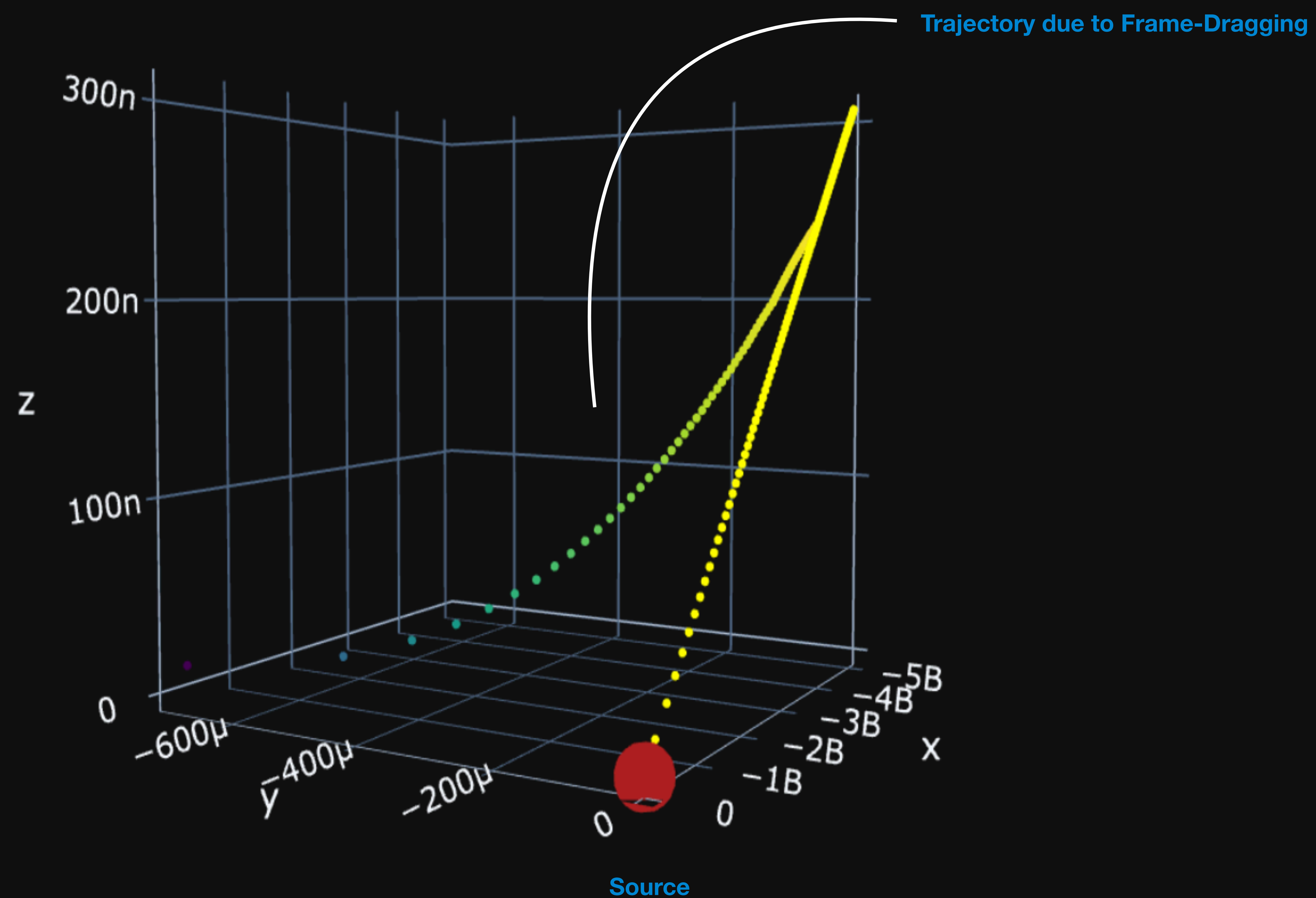
$$\Omega = \frac{d\phi}{dt} = \frac{\dot{\phi}}{\dot{t}} = \frac{2mcra}{(r^2 + a^2)\rho^2 + 2mra^2Sin^2\theta}$$

$$where, \; \rho^2 = r^2 + a^2Cos^2\theta$$

$$a = \frac{J}{Mc}$$

$$m = \frac{GM}{c^2}$$

Inertial Frame Dragging In Kerr Space-Time



GRAVITY PROBE-B EXPERIMENT

- It was a satellite-based mission (2004) by a Stanford group and NASA.
- Experimentally measure the Frame-Dragging effect due to Gravity.
- Four perfectly spherical gyroscope balls were sent to polar orbit of Earth.

$$m, a << r$$

$$\Omega \approx \frac{2mca}{r^3} = 7.64e - 15 \text{ rad/s}$$

PYTHON CODE FOR SIMULATING ROTATIONAL FRAME DRAGGING

```
import numpy as np
import astropy.units as u

from plotly.offline import init_notebook_mode
from einsteinpy.coordinates import BoyerLindquistDifferential
from einsteinpy.bodies import Body
from einsteinpy.geodesic import Geodesic
from einsteinpy.metric.kerrnewman import KerrNewman

init_notebook_mode(connected=True)

## Source ##

m = 2e30 # kg

spin_factor = 0.3 # spin_factor = J/(Mc) (metres)

## Test Particle ##

# initial position cordinates

r = 5e9                    # (metres)

theta = np.pi * 0.50       # (radians)

phi = np.pi                # (radians)

# Source
Attractor = Body(name="attractor", mass = m * u.kg, R=0 * u.m, differential = None, a = spin_factor * u.m, q = 0 * u.C, parent=None)

# Coordinate System
sph_obj = BoyerLindquistDifferential(r * u.m, theta * u.rad, phi * u.rad, 0 * u.m/u.s, 0 * u.rad/u.s, 0 * u.rad/u.s, spin_factor * u.m)

# Test Particle
Object = Body(name="testparticle", mass = 0 * u.kg, R = 0 * u.m,  differential = sph_obj, a = 0 * u.m, q = 0 * (u.C/u.kg), parent=Attractor)

# geodesic simulation
geodesic = Geodesic(body = Object, time=0 * u.s, end_lambda= ((1 * u.year).to(u.s)).value/930, step_size=((0.5 * u.min).to(u.s)).value, metric=KerrNewman)

## Each element here is w.r.t. proper-time (lambda) ##
vals = geodesic.trajectory

t = np.array(vals[:, 0])
x = np.array(vals[:, 1])
y = np.array(vals[:, 2])
z = np.array(vals[:, 3])
```

```
spin_factor=0

# Source
Attractor = Body(name="attractor", mass = m * u.kg, R=0 * u.m, differential = None, a = spin_factor * u.m, q = 0 * u.C, parent=None)

# Coordinate System
sph_obj = BoyerLindquistDifferential(r * u.m, theta * u.rad, phi * u.rad, 0 * u.m/u.s, 0 * u.rad/u.s, 0 * u.rad/u.s, spin_factor * u.m)

# Test Particle
Object = Body(name="testparticle", mass = 0 * u.kg, R = 0 * u.m,  differential = sph_obj, a = 0 * u.m, q = 0 * (u.C/u.kg), parent=Attractor)

# geodesic simulation
geodesic = Geodesic(body = Object, time=0 * u.s, end_lambda= ((1 * u.year).to(u.s)).value/930, step_size=((0.5 * u.min).to(u.s)).value, metric=KerrNewman)

## Each element here is w.r.t. proper-time (lambda) ##
vals = geodesic.trajectory

t1 = np.array(vals[:, 0])
x1 = np.array(vals[:, 1])
y1 = np.array(vals[:, 2])
z1 = np.array(vals[:, 3])

import plotly.graph_objects as go

fig = go.Figure()
fig.add_trace(
    go.Scatter3d(
        x=x,
        y=y,
        z=z,
        mode="markers",
        marker=dict(size=2, color=y, colorscale='Viridis', opacity=1),
        name="Frame Dragged Trajectory"))

fig.add_trace(
    go.Scatter3d(
        x=x1,
        y=y1,
        z=z1,
        mode="markers",
        marker=dict(size=2, color='yellow', colorscale='Viridis', opacity=1),
        name="Ordinary Trajectory"))

fig.add_trace(
    go.Scatter3d(
        x=[0],
        y=[0],
        z=[0],
        mode="markers",
        marker=dict(size=15, color='firebrick', colorscale='Viridis', opacity=1),
        name="Source"))

fig.update_layout(title={
        'text': "Inertial Frame Dragging In Kerr Space-Time",
        'y':0.9,
        'x':0.4,
        'xanchor': 'center',
        'yanchor': 'top'},
        margin=dict(l=0, r=0, b=0, t=0),
        template="plotly_dark")

fig.show()
```