

In [1]: `pip install pandas`

Requirement already satisfied: pandas in c:\users\samil\anaconda3\lib\site-packages (2.0.3)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: numpy>=1.20.3; python_version < "3.10" in c:\users\samil\anaconda3\lib\site-packages (from pandas) (1.24.4)
 Requirement already satisfied: tzdata>=2022.1 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2023.3)
 Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2.8.2)
 Requirement already satisfied: pytz>=2020.1 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2020.1)
 Requirement already satisfied: six>=1.5 in c:\users\samil\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.15.0)

In [2]: `pip install plotly`

Requirement already satisfied: plotly in c:\users\samil\anaconda3\lib\site-packages (5.18.0)
 Requirement already satisfied: packaging in c:\users\samil\anaconda3\lib\site-packages (from plotly) (20.4)
 Requirement already satisfied: tenacity>=6.2.0 in c:\users\samil\anaconda3\lib\site-packages (from plotly) (8.2.3)
 Requirement already satisfied: pyparsing>=2.0.2 in c:\users\samil\anaconda3\lib\site-packages (from packaging->plotly) (2.4.7)
 Requirement already satisfied: six in c:\users\samil\anaconda3\lib\site-packages (from packaging->plotly) (1.15.0)
 Note: you may need to restart the kernel to use updated packages.

In [3]: `pip install matplotlib`

Requirement already satisfied: matplotlib in c:\users\samil\anaconda3\lib\site-packages (3.7.4)Note: you may need to restart the kernel to use updated packages.
 Requirement already satisfied: packaging>=20.0 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (20.4)
 Requirement already satisfied: fonttools>=4.22.0 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (4.47.2)
 Requirement already satisfied: importlib-resources>=3.2.0; python_version < "3.10" in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (6.1.1)
 Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (1.2.0)
 Requirement already satisfied: pyparsing>=2.3.1 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (2.4.7)
 Requirement already satisfied: python-dateutil>=2.7 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (2.8.2)
 Requirement already satisfied: pillow>=6.2.0 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (7.2.0)
 Requirement already satisfied: numpy<2,>=1.20 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (1.24.4)
 Requirement already satisfied: contourpy>=1.0.1 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (1.1.1)
 Requirement already satisfied: cycler>=0.10 in c:\users\samil\anaconda3\lib\site-packages (from matplotlib) (0.10.0)
 Requirement already satisfied: six in c:\users\samil\anaconda3\lib\site-packages (from packaging>=20.0->matplotlib) (1.15.0)
 Requirement already satisfied: zipp>=3.1.0; python_version < "3.10" in c:\users\samil\anaconda3\lib\site-packages (from importlib-resources>=3.2.0; python_version < "3.10"->matplotlib) (3.1.0)

In [4]: `import matplotlib.pyplot as plt
 from mpl_toolkits.mplot3d import Axes3D
 from matplotlib.animation import FuncAnimation
 import pandas as pd
 import numpy as np
 import matplotlib.pyplot as plt`

C:\Users\Samil\anaconda3\lib\site-packages\pandas\core\computation\expressions.py:20: UserWarning: Pandas requires version '2.7.3' or newer of 'numexpr' (version '2.7.1' currently in stalled).
 from pandas.core.computation.check import NUMEXPR_INSTALLED

In [5]: `raw_dataset=pd.read_csv("Doublet_EAF_All.csv",sep=",")`

```
In [6]: Doublet_EAF_All = raw_dataset.copy()
Doublet_EAF_All.head()
```

```
Out[6]:
```

| | Date | Longitude | Latitude | Depth | Magnitude |
|---|---------------------|-----------|----------|-------|-----------|
| 0 | 08/01/2024 13:19:12 | 38.7525 | 38.2842 | 8.97 | 4.4 |
| 1 | 07/01/2024 15:58:00 | 37.2725 | 38.3222 | 6.99 | 3.9 |
| 2 | 06/01/2024 12:10:09 | 38.5897 | 38.1694 | 8.76 | 4.0 |
| 3 | 05/01/2024 14:03:03 | 37.4519 | 38.3753 | 6.99 | 3.6 |
| 4 | 30/12/2023 13:26:24 | 39.0192 | 38.4564 | 9.36 | 4.2 |

```
In [7]: Doublet_EAF_All.shape
```

```
Out[7]: (6108, 5)
```

```
In [8]: x = Doublet_EAF_All.iloc[:,1].values
y = Doublet_EAF_All.iloc[:,2].values
z = Doublet_EAF_All.iloc[:,3].values
colors = Doublet_EAF_All.iloc[:,4].values
sizes = Doublet_EAF_All.iloc[:,4].values*50
```

```

In [20]: import plotly.graph_objects as go

# Obtain high-resolution world map data online
fig = go.Figure(go.Choroplethmapbox(
    geojson="https://raw.githubusercontent.com/johan/world.geo.json/master/countries.geo.json",
    locations=["USA", "CAN", "MEX", "RUS", "CHN"], # Example country codes (USA, Canada, Mexico, Russia, China)
    z=[1, 1, 1, 1, 1], # Values to be assigned to countries (all set to 1)
    colorscale='Jet', # Color scale name (Viridis, YlGnBu, Jet, etc.)
    zmin=2,
    zmax=8,
    marker_opacity=0.9, # Opacity of country borders
    marker_line_width=1, # Thickness of country borders
))

# Create sample earthquake data
earthquake_data = {
    'Longitude': x,
    'Latitude': y,
    'Magnitude': colors,
}

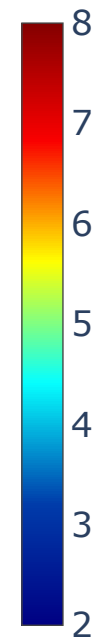
# Add earthquake data with Scatter plot
fig.add_trace(go.Scattermapbox(
    lat=earthquake_data['Latitude'],
    lon=earthquake_data['Longitude'],
    mode='markers',
    marker=dict(
        size=earthquake_data['Magnitude'] * 2, # Set point sizes based on Magnitude value
        color=earthquake_data['Magnitude'], # Set color scale based on Magnitude value
        colorscale='Jet', # Color scale name (Viridis, YlGnBu, Jet, etc.)
    ),
))

# Specify map layout and style
fig.update_layout(
    mapbox_style="open-street-map", # Set map style (for other styles: "open-street-map", "stamen-terrain", etc.)
    mapbox_zoom=6, # Set map zoom level
    mapbox_center={"lat": 37.000, "lon": 37.0000}, # Set map center (center of the USA)
)

# Increase resolution and font size
fig.update_layout(
    width=845, # Set width to increase resolution
    height=640, # Set height to increase resolution
    font=dict(
        size=25 # Set font size for English comments
    )
)

# Display the plot
fig.show()

```



```
In [10]: import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import ipywidgets as widgets
from ipywidgets import interactive
from IPython.display import display

# İnteraktif işlev

def plot_3d_scatter(elev, azim, zoom, theta):
    fig = plt.figure(figsize=(12, 10))
    ax = fig.add_subplot(111, projection='3d')

    ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')
    cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))
    cbar.set_label('Magnitude')

    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    ax.set_zlabel('Depth_km')
    ax.dist = zoom # Zoom ayarı
    ax.azim = theta # Maus ile çevirme
    plt.show()

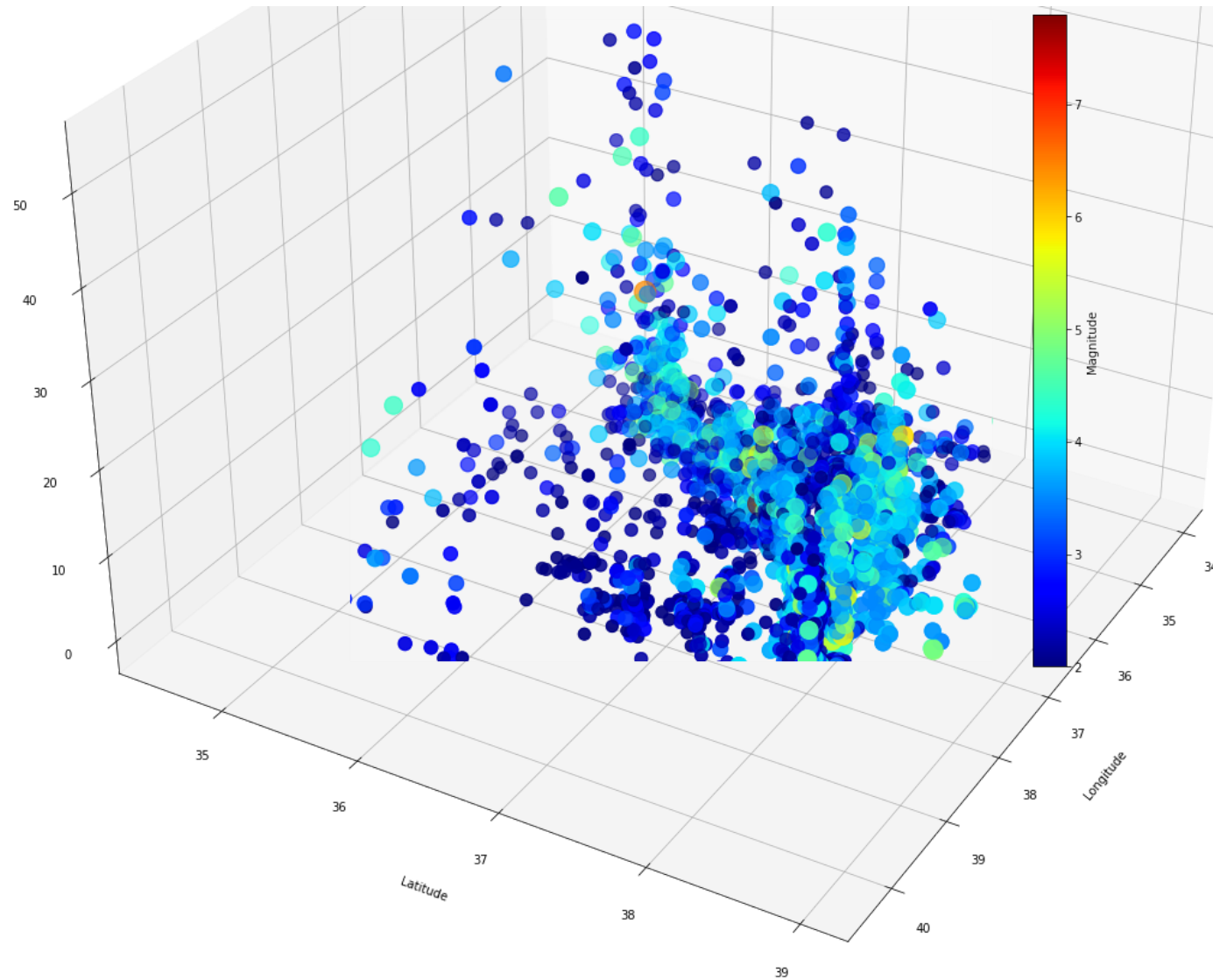
# İnteraktif widget'ı oluşturma
elev_slider = widgets.IntSlider(min=0, max=180, value=30, description='Elevation:')
azim_slider = widgets.IntSlider(min=0, max=360, value=30, description='Azimuth:')
zoom_slider = widgets.FloatSlider(min=1, max=10, value=5, description='Zoom:')
theta_slider = widgets.IntSlider(min=0, max=360, value=30, description='Theta:')
interactive_plot = interactive(plot_3d_scatter, elev=elev_slider, azim=azim_slider, zoom=zoom_slider, theta=theta_slider)

# Widget'ı görüntüleme
display(interactive_plot)
```

Elevation: 30
Azimuth: 30
Zoom: 5.00
Theta: 30

<ipython-input-10-fbd5a5764876>:24: MatplotlibDeprecationWarning:

The dist attribute was deprecated in Matplotlib 3.6 and will be removed two minor releases later.



```
In [11]: x = Doublet_EAF_All.iloc[:,1].values  
y = Doublet_EAF_All.iloc[:,2].values  
z = Doublet_EAF_All.iloc[:,0].values  
colors = Doublet_EAF_All.iloc[:,4].values  
sizes = Doublet_EAF_All.iloc[:,4].values*30
```

```
array(['08/01/2024 13:19:12', '07/01/2024 15:58:00',  
      '06/01/2024 12:10:09', ..., '11/02/2018 16:28:42',  
      '11/02/2018 06:45:36', '08/02/2018 11:44:48'], dtype=object)
```

```
from datetime import datetime
```

```
# Zaman damgalarını içeren bir liste oluştur
timestamps = z
```

```
# Zaman damgalarını saniyeye dönüştür
seconds = [datetime.timestamp(datetime.strptime(timestamp, '%d/%m/%Y %H:%M:%S')) for timestamp in timestamps]
```

```
seconds = [datetime.timestamp(datetime.strptime(timestamp, '%d/%m/%Y %H:%M:%S')) for timestamp in timestamps]
```

```
print(seconds) # Saniye cinsinden zaman damgalarını görüntüle
```

[1704709152.0, 1704632280.0, 1704532209.0, 1704452583.0, 1703931984.0, 1703891113.0, 1703745245.0, 1703652912.0, 1703544181.0, 1703500670.0, 1703386343.0, 1703172273.0, 1703006572.0, 1702868845.0, 1702582378.0, 1702416969.0, 1701678003.0, 1701379129.0, 1701124355.0, 1700993608.0, 1700744098.0, 1700740197.0, 1700739968.0, 1700454964.0, 1700430904.0, 1700315757.0, 1700078325.0, 1700076730.0, 1700049365.0, 1700003667.0, 1699486334.0, 1698672347.0, 1698595709.0, 1698298332.0, 1698097173.0, 1697924927.0, 1697785222.0, 1697783010.0, 1697779810.0, 1697570056.0, 1697540393.0, 1697473858.0, 1697446261.0, 1697355251.0, 1697311914.0, 1697274130.0, 1697256274.0, 1697251735.0, 1696822133.0, 1696724678.0, 1696705306.0, 1696661816.0, 1696508598.0, 1696395494.0, 1696312368.0, 1696307118.0, 1696236533.0, 1696196633.0, 1695946458.0, 1695946333.0, 1695871104.0, 1695866493.0, 1695701680.0, 1695695575.0, 1695636299.0, 1695446541.0, 1695305870.0, 1694786091.0, 1694208004.0, 1694019701.0, 1693820561.0, 1693791013.0, 1693731362.0, 1693585831.0, 1693479994.0, 1693278184.0, 1693029523.0, 1693024478.0, 1692926556.0, 1692892392.0, 1692844523.0, 1692765627.0, 1692752555.0, 1692699689.0, 1692688652.0, 1692615272.0, 1692504353.0, 1692473214.0, 1692275106.0, 1692243949.0, 1692185818.0, 1691842699.0, 1691836801.0, 1691836756.0, 1691822569.0, 1691743371.0, 1691678880.0, 1691646494.0, 1691589706.0, 1691557506.0, 1691547023.0, 1691491622.0, 1691384961.0, 1691336938.0, 1691152574.0, 1691027775.0, 1691025345.0, 1690829617.0, 1690806196.0, 1690719197.0, 1690377915.0, 1690368250.0, 1690358117.0, 1690323011.0, 1690261074.0, 1690260513.0, 1690259598.0, 1690253089.0, 1690227572.0, 1690136531.0, 1690098517.0, 1690026189.0, 1690011431.0, 1689983013.0, 1689969798.0, 1689927804.0, 1689725147.0, 1689550795.0, 1689431568.0, 1689240741.0, 1688858855.0, 1688744323.0, 1688525712.0, 1688355110.0, 1688282159.0, 1688202252.0, 1688162406.0, 1688087575.0, 1688034882.0, 1688005687.0, 1687950889.0, 1687881725.0, 1687858806.0, 1687820923.0, 1687611819.0, 1687608393.0, 1687594809.0, 1687529686.0, 1687529124.0, 1687433264.0, 1687400462.0, 1687385505.0, 1687277671.0, 1687274567.0, 1687209270.0, 1687188690.0, 1687126895.0, 1687123818.0, 1687111161.0, 1687091788.0, 16870858523.0, 1687053377.0, 1687028575.0, 1686947374.0, 1686844831.0, 1686833917.0, 1686818380.0, 1686797257.0, 1686795263.0, 1686793042.0, 1686682912.0, 1686556864.0, 1686484622.0, 1686465656.0, 1686448033.0, 1686270167.0, 1686184721.0, 1686151388.0, 1686103624.0, 1686011489.0, 1685972447.0, 1685957962.0, 1685831495.0, 1685820320.0, 1685777661.0, 1685736440.0, 1685680480.0, 1685587333.0, 1685551600.0, 1685466571.0, 1685437533.0, 1685434832.0, 1685426715.0, 1685332022.0, 1685317919.0, 1685242279.0, 1685212565.0, 1685148322.0, 1685117110.0, 1685093350.0, 1685046939.0, 1685034996.0, 1685033946.0, 1685016125.0, 1685000821.0, 1684988493.0, 1684951909.0, 1684895412.0, 1684793566.0, 1684789064.0, 1684781269.0, 1684780371.0, 1684762978.0, 1684760818.0, 1684728802.0, 1684689382.0, 1684684576.0, 1684677447.0, 1684673471.0, 1684665702.0, 1684662373.0, 1684622402.0, 1684563361.0, 1684541010.0, 1684533324.0, 1684473816.0, 1684455281.0, 1684137492.0, 1684071033.0, 1684055169.0, 1683996885.0, 1683990179.0, 1683975195.0, 1683918767.0, 1683917884.0, 1683916991.0, 1683869784.0, 1683869601.0, 1683806224.0, 1683733176.0, 1683679627.0, 1683593535.0, 1683500100.0, 1683456633.0, 1683404100.0, 1683373456.0, 1683337803.0, 1683300610.0, 1683237770.0, 1683145578.0, 1683063565.0, 1683004514.0, 1682927540.0, 1682832501.0, 1682740000.0, 1682655000.0, 1682570000.0, 1682485000.0, 1682400000.0, 1682315000.0, 1682230000.0, 1682145000.0, 1682060000.0, 1681975000.0, 1681890000.0, 1681810000.0, 1681730000.0, 1681650000.0, 1681570000.0, 1681490000.0, 1681410000.0, 1681330000.0, 1681250000.0, 1681170000.0, 1681090000.0, 1681010000.0, 1680930000.0, 1680850000.0, 1680770000.0, 1680690000.0, 1680610000.0, 1680530000.0, 1680450000.0, 1680370000.0, 1680290000.0, 1680210000.0, 1680130000.0, 1680050000.0, 1679970000.0, 1679890000.0, 1679810000.0, 1679730000.0, 16796

seconds

```
[1704709152.0,
1704632280.0,
1704532209.0,
1704452583.0,
1703931984.0,
1703891113.0,
1703745245.0,
1703652912.0,
1703544181.0,
1703500670.0,
1703386343.0,
1703172273.0,
1703006572.0,
1702868845.0,
1702582378.0,
1702416969.0,
1701678003.0,
1701379129.0,
1701124355.0,
1700882688.0]
```



```
In [28]: x = Doublet_EAF_All.iloc[:,1].values  
y = Doublet_EAF_All.iloc[:,2].values  
z = seconds  
colors = Doublet_EAF_All.iloc[:,4].values  
sizes = Doublet_EAF_All.iloc[:,4].values*20
```

```
In [30]: import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import ipywidgets as widgets
from ipywidgets import interactive
from IPython.display import display
from datetime import datetime

# Zaman verilerini saniyeden yıla dönüştüren fonksiyon
def seconds_to_year(seconds):
    timestamp = datetime.fromtimestamp(seconds)
    return timestamp.year

# İnteraktif işlev
def plot_3d_scatter(elev, azim, zoom, theta):
    fig = plt.figure(figsize=(15, 12))
    ax = fig.add_subplot(111, projection='3d')

    ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')
    cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))
    cbar.set_label('Magnitude')

    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    ax.set_zlabel('Time (year)') # "Time" için değişiklik burada
    ax.dist = zoom # Zoom ayarı
    ax.azim = theta # Maus ile çevirme

    # Zaman etiketlerini güncelleme
    time_ticks = np.linspace(min(z), max(z), 5)
    time_labels = [seconds_to_year(t) for t in time_ticks]
    ax.set_zticks(time_ticks)
    ax.set_zticklabels(time_labels)

    plt.show()

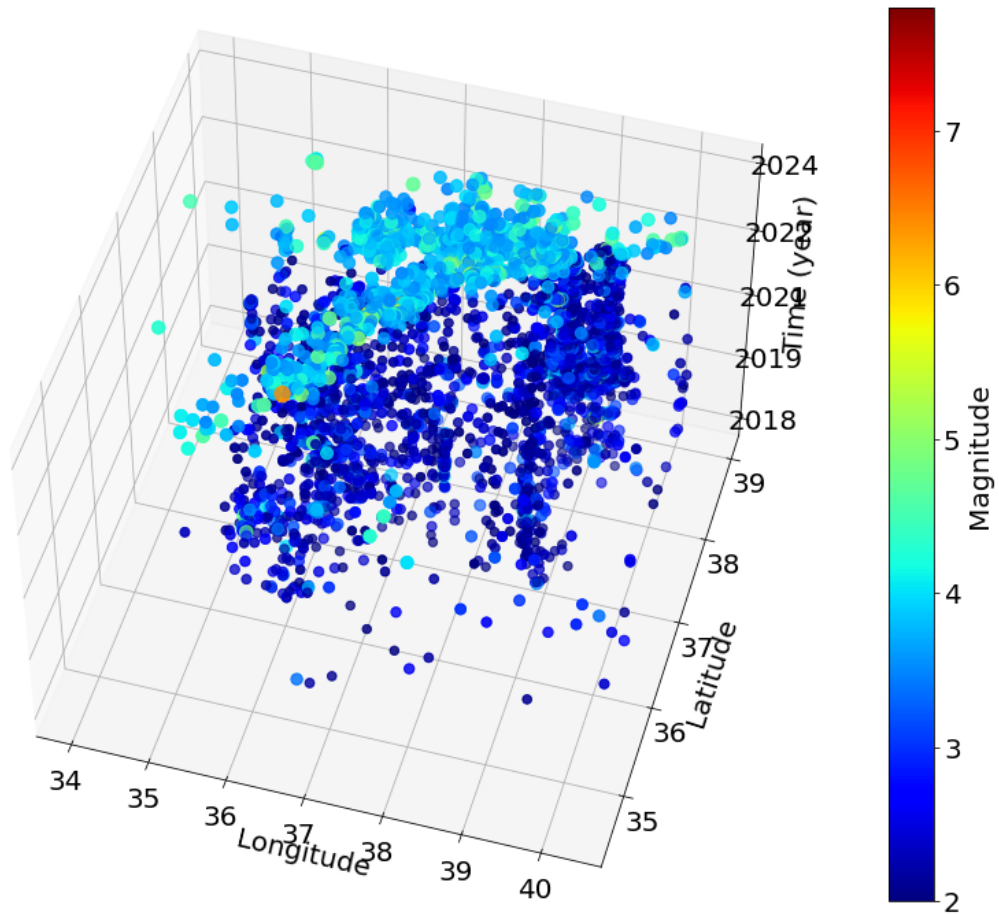
# İnteraktif widget'ı oluşturma
elev_slider = widgets.IntSlider(min=0, max=180, value=30, description='Elevation:')
azim_slider = widgets.IntSlider(min=0, max=360, value=30, description='Azimuth:')
zoom_slider = widgets.FloatSlider(min=1, max=10, value=5, description='Zoom:')
theta_slider = widgets.IntSlider(min=0, max=360, value=30, description='Theta:')
interactive_plot = interactive(plot_3d_scatter, elev=elev_slider, azim=azim_slider, zoom=zoom_slider, theta=theta_slider)

# Widget'ı görüntüleme
display(interactive_plot)
```

Elevation: 49
Azimuth: 30
Zoom: 9.80
Theta: 286

<ipython-input-30-6d9bab95096>:28: MatplotlibDeprecationWarning:

The dist attribute was deprecated in Matplotlib 3.6 and will be removed two minor releases later.



```
In [17]: x = Doublet_EAF_All.iloc[:,1].values  
y = Doublet_EAF_All.iloc[:,2].values  
z = seconds  
colors = Doublet_EAF_All.iloc[:,4].values  
sizes = Doublet_EAF_All.iloc[:,4].values*30
```

```
In [18]: import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import ipywidgets as widgets
from ipywidgets import interactive
from IPython.display import display

# İnteraktif işlev

def plot_3d_scatter(elev, azim, zoom, theta):
    fig = plt.figure(figsize=(12, 10))
    ax = fig.add_subplot(111, projection='3d')

    ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')
    cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))
    cbar.set_label('Magnitude')

    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    ax.set_zlabel('Time(year)')
    ax.dist = zoom # Zoom ayarı
    ax.azim = theta # Maus ile çevirme
    plt.show()

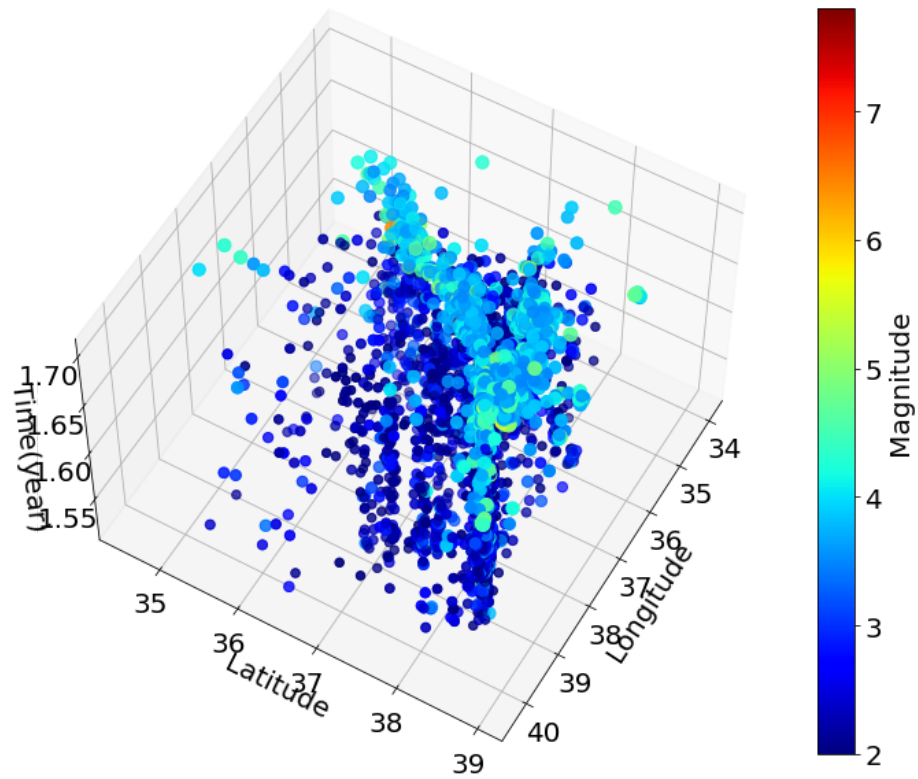
# İnteraktif widget'ı oluşturma
elev_slider = widgets.IntSlider(min=0, max=180, value=30, description='Elevation:')
azim_slider = widgets.IntSlider(min=0, max=360, value=30, description='Azimuth:')
zoom_slider = widgets.FloatSlider(min=1, max=10, value=5, description='Zoom:')
theta_slider = widgets.IntSlider(min=0, max=360, value=30, description='Theta:')
interactive_plot = interactive(plot_3d_scatter, elev=elev_slider, azim=azim_slider, zoom=zoom_slider, theta=theta_slider)

# Widget'ı görüntüleme
display(interactive_plot)
```

Elevation: 55
Azimuth: 30
Zoom: 10.00
Theta: 30

<ipython-input-18-9d017fbc56d8>:24: MatplotlibDeprecationWarning:

The dist attribute was deprecated in Matplotlib 3.6 and will be removed two minor releases later.



```
In [19]: import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import ipywidgets as widgets
from ipywidgets import interactive
from IPython.display import display
from datetime import datetime

# Zaman verilerini saniyeden yıla dönüştüren fonksiyon
def seconds_to_year(seconds):
    timestamp = datetime.fromtimestamp(seconds)
    return timestamp.year

# İnteraktif işlev
def plot_3d_scatter(elev, azim, zoom, theta):
    fig = plt.figure(figsize=(15, 12))
    ax = fig.add_subplot(111, projection='3d')

    ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')
    cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))
    cbar.set_label('Magnitude')

    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    ax.set_zlabel('Time (year)') # "Time" için değişiklik burada
    ax.dist = zoom # Zoom ayarı
    ax.azim = theta # Maus ile çevirme

    # Zaman etiketlerini güncelleme
    time_ticks = np.linspace(min(z), max(z), 5)
    time_labels = [seconds_to_year(t) for t in time_ticks]
    ax.set_zticks(time_ticks)
    ax.set_zticklabels(time_labels)

    plt.show()

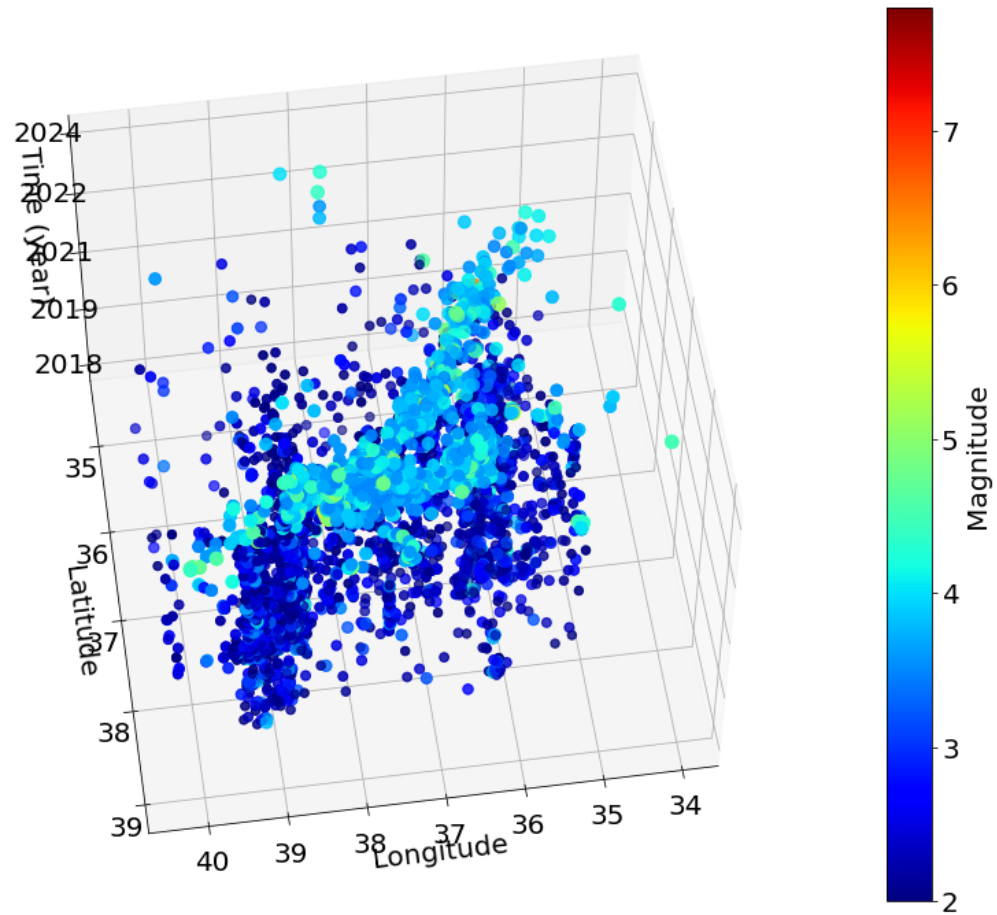
# İnteraktif widget'ı oluşturma
elev_slider = widgets.IntSlider(min=0, max=180, value=30, description='Elevation:')
azim_slider = widgets.IntSlider(min=0, max=360, value=30, description='Azimuth:')
zoom_slider = widgets.FloatSlider(min=1, max=10, value=5, description='Zoom:')
theta_slider = widgets.IntSlider(min=0, max=360, value=30, description='Theta:')
interactive_plot = interactive(plot_3d_scatter, elev=elev_slider, azim=azim_slider, zoom=zoom_slider, theta=theta_slider)

# Widget'ı görüntüleme
display(interactive_plot)
```

Elevation: 53
Azimuth: 30
Zoom: 10.00
Theta: 82

<ipython-input-19-6d9bab95096>:28: MatplotlibDeprecationWarning:

The dist attribute was deprecated in Matplotlib 3.6 and will be removed two minor releases later.




```
In [21]: import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import ipywidgets as widgets
from ipywidgets import interactive
from IPython.display import display
from datetime import datetime

# Zaman verilerini saniyeden yıla dönüştüren fonksiyon
def seconds_to_year(seconds):
    timestamp = datetime.fromtimestamp(seconds)
    return timestamp.year

# İnteraktif işlev
def plot_3d_scatter(elev, azim, zoom, theta):
    fig = plt.figure(figsize=(15, 12))
    ax = fig.add_subplot(111, projection='3d')

    ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')
    cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))
    cbar.set_label('Magnitude')

    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    ax.set_zlabel('Time (year)') # "Time" için değişiklik burada
    ax.dist = zoom # Zoom ayarı
    ax.azim = theta # Maus ile çevirme

    # Zaman etiketlerini güncelleme
    time_ticks = np.linspace(min(z), max(z), 5)
    time_labels = [seconds_to_year(t) for t in time_ticks]
    ax.set_zticks(time_ticks)
    ax.set_zticklabels(time_labels)

    plt.show()

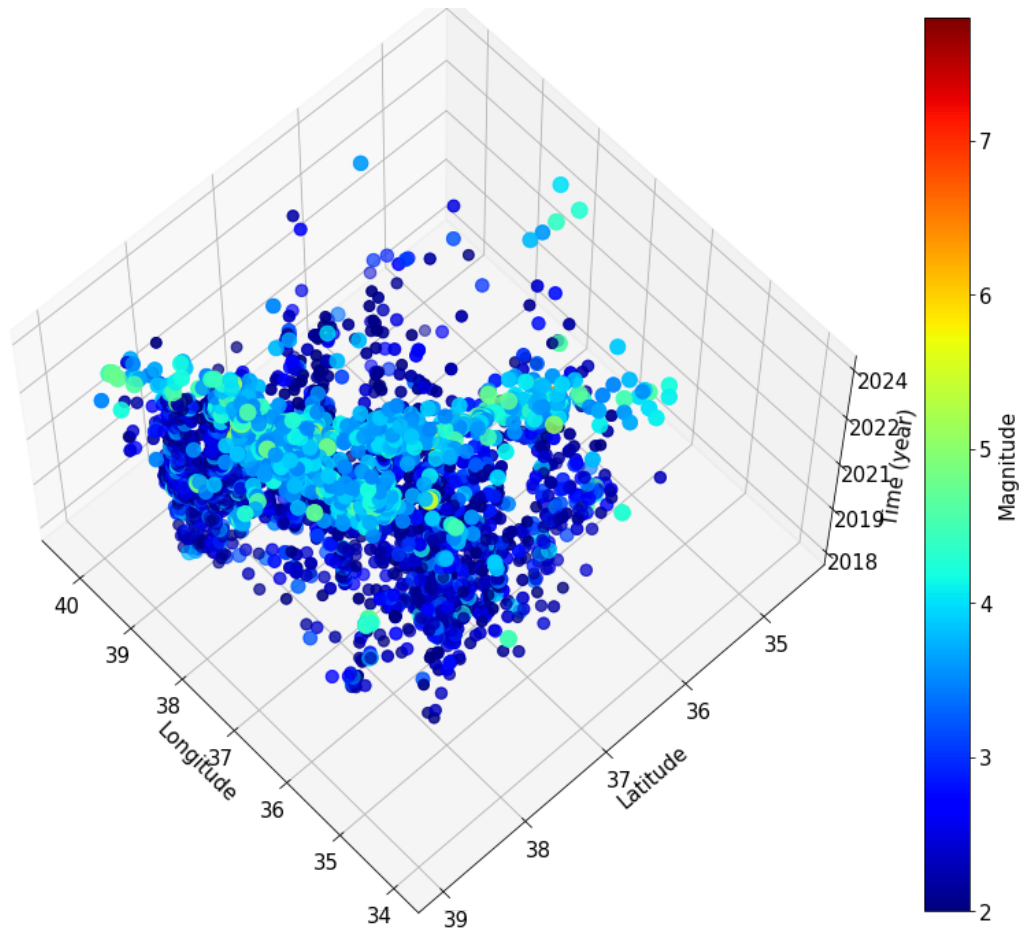
# İnteraktif widget'ı oluşturma
elev_slider = widgets.IntSlider(min=0, max=180, value=30, description='Elevation:')
azim_slider = widgets.IntSlider(min=0, max=360, value=30, description='Azimuth:')
zoom_slider = widgets.FloatSlider(min=1, max=10, value=5, description='Zoom:')
theta_slider = widgets.IntSlider(min=0, max=360, value=30, description='Theta:')
interactive_plot = interactive(plot_3d_scatter, elev=elev_slider, azim=azim_slider, zoom=zoom_slider, theta=theta_slider)

# Widget'ı görüntüleme
display(interactive_plot)
```

Elevation: 61
Azimuth: 68
Zoom: 10.00
Theta: 137

<ipython-input-21-6d9bab95096>:28: MatplotlibDeprecationWarning:

The dist attribute was deprecated in Matplotlib 3.6 and will be removed two minor releases later.



In []:

