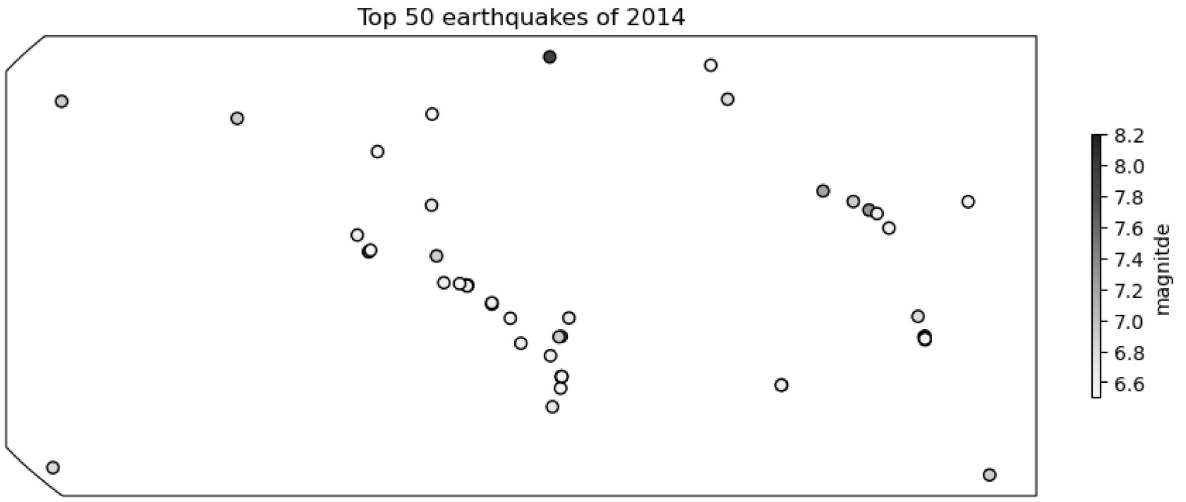
In [2]:
```
# import pandas
import pandas as pd
# import numpy
import numpy as np
import xarray as xr
import matplotlib.ticker as mticker
import cartopy.crs as ccrs
import cartopy.feature as cfeature
# import matplotlib
from matplotlib import pyplot as plt
# make plots appear and be stored within the notebook
%matplotlib inline
```

In [30]:
```
# 1
a = pd.read_csv('usgs_earthquakes.csv')
b=a.sort_values('mag',ascending=False).head(50)
```

In [38]:
```
# Create and define the size of a figure object
plt.figure(figsize=(10,10), dpi=100)


# Create an axes with an basic PlateCarree projection style
proj = ccrs.Robinson(central_longitude=180)
ax = plt.axes(projection=proj)
lon = b['longitude']
lat = b['latitude']
plt.scatter(lon,lat,marker='o',linewidth=1, edgecolor='black',transform=ccrs.PlateC

# Set title
ax.set_title('Top 50 earthquakes of 2014')
# Configure the Colorbar
from matplotlib import ticker
cb = plt.colorbar(fraction = 0.008,aspect = 30,label = 'magnitde')
tick_locator = ticker.MaxNLocator(nbins=9)
cb.locator = tick_locator
cb.update_ticks()
plt.show()
```



In [39]:
```
# 2
# 2.1
c=xr.open_dataset('200301_202006-C3S-L3_GHG-PRODUCTS-OBS4MIPS-MERGED-v4.3.nc', engin
c
```

Out[39]: xarray.Dataset

► Dimensions: (**time**: 210, bnds: 2, **lat**: 36, **lon**: 72, pressure: 10)

▼ Coordinates:

time	(time)	datetime64[ns]	2003-01-16T12:0...
lat	(lat)	float64	-87.5 -82.5 -77.5 ....
lon	(lon)	float64	-177.5 -172.5 ... 1...

▼ Data variables:

time_bnds	(time, bnds)	datetime64[ns]	...
lat_bnds	(lat, bnds)	float64	...
lon_bnds	(lon, bnds)	float64	...
pre	(pressure)	float64	...
pre_bnds	(pressure, bnds)	float64	...
land_fraction	(lat, lon)	float64	...
xch4	(time, lat, lon)	float32	...
xch4_nobs	(time, lat, lon)	float64	...
xch4_stderr	(time, lat, lon)	float32	...
xch4_stddev	(time, lat, lon)	float32	...
column_averagin...	(time, pressure, lat, lon)	float32	...
vmr_profile_ch4_...	(time, pressure, lat, lon)	float32	...

► Attributes: (28)

In [41]:
```
c.sel(time=slice('2013','2020')).mean(dim='time')
```

Out[41]: xarray.Dataset

► Dimensions: (**lat**: 36, bnds: 2, **lon**: 72, pressure: 10)

▼ Coordinates:

lat	(lat)	float64	-87.5 -82.5 -77.5 ... 82.5 87.5
lon	(lon)	float64	-177.5 -172.5 ... 172.5 177.5

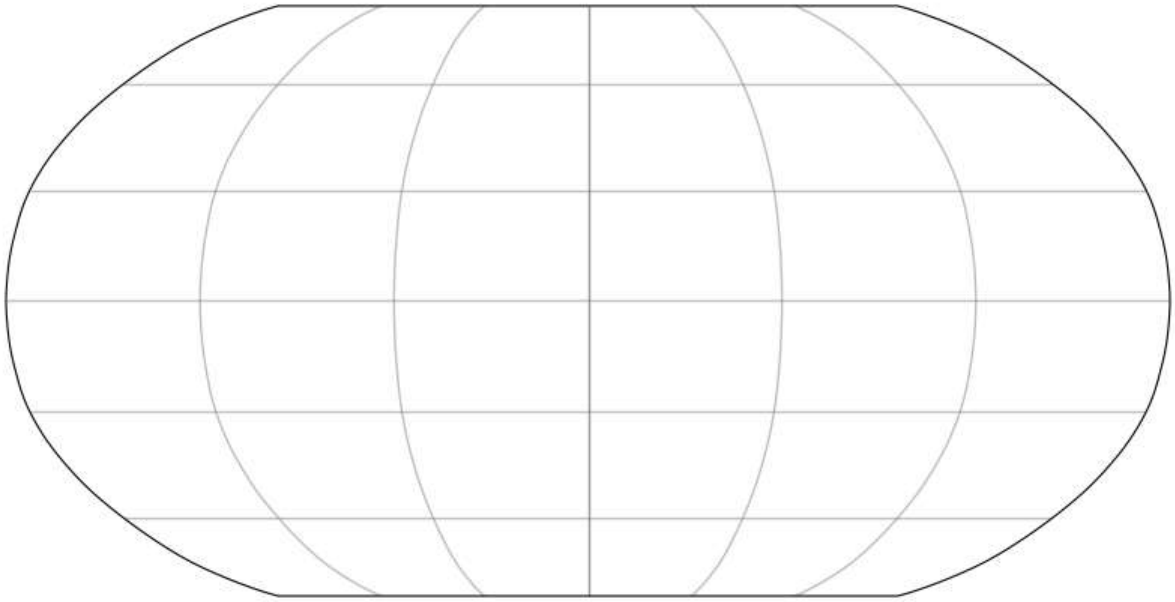
▼ Data variables:

lat_bnds	(lat, bnds)	float64	-90.0 -85.0 -85.0 ... 85.0 90.0
lon_bnds	(lon, bnds)	float64	-180.0 -175.0 ... 175.0 180.0
pre	(pressure)	float64	0.95 0.85 0.75 ... 0.25 0.15 0.05
pre_bnds	(pressure, bnds)	float64	1.0 0.9 0.9 0.8 ... 0.2 0.1 0.1 0.0
land_fraction	(lat, lon)	float64	0.9982 0.9998 0.9998 ... 0.0 0.0
xch4	(lat, lon)	float32	nan nan nan nan ... nan nan na...
xch4_nobs	(lat, lon)	float64	nan nan nan nan ... nan nan na...
xch4_stderr	(lat, lon)	float32	nan nan nan nan ... nan nan na...
xch4_stddev	(lat, lon)	float32	nan nan nan nan ... nan nan na...
column_averagin...	(pressure, lat, lon)	float32	nan nan nan nan ... nan nan na...
vmr_profile_ch4_...	(pressure, lat, lon)	float32	nan nan nan nan ... nan nan na...

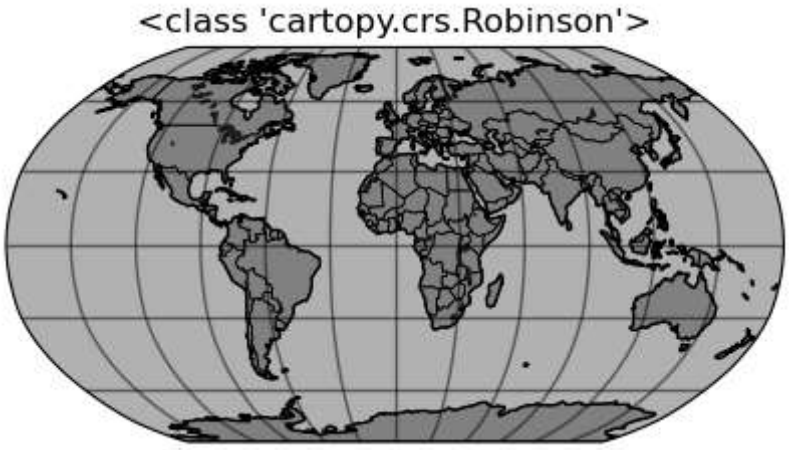
► Attributes: (0)

In [3]:
```
plt.figure(figsize=(10,10), dpi=100)
# Create an axes with Orthographic projection style
proj = ccrs.Robinson(central_longitude=180)
ax = plt.axes(projection=proj)
# np.arange(-90,91,30)np.arange(-180, 181, 30)
gl = ax.gridlines(crs=ccrs.PlateCarree(), linewidth=1, color='grey', alpha=0.5)
gl.ylocator = mticker.FixedLocator()
gl.xlocator = mticker.FixedLocator()
c.plot.contourf(ax=ax, transform=ccrs.PlateCarree(),cmap='magma',
                        vmin=0, vmax=15, levels=20,cbar_kwargs={'fraction': 0.008,'as

ax.add_feature(cfeature.COASTLINE, edgecolor='white', zorder=1)
ax.set_title('Average Pressure from 2013-2020')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In [3], line 7
      5 # np.arange(-90,91,30)np.arange(-180, 181, 30)
      6 gl = ax.gridlines(crs=ccrs.PlateCarree(), linewidth=1, color='grey', alpha=
0.5)
----> 7 gl.ylocator = mticker.FixedLocator()
      8 gl.xlocator = mticker.FixedLocator()
      9 c.plot.contourf(ax=ax, transform=ccrs.PlateCarree(),cmap='magma',
     10                         vmin=0, vmax=15, levels=20,cbar_kwargs={'fraction':
0.008,'aspect' : 30,'label' : 'magnitde'})

TypeError: __init__() missing 1 required positional argument: 'locs'
```



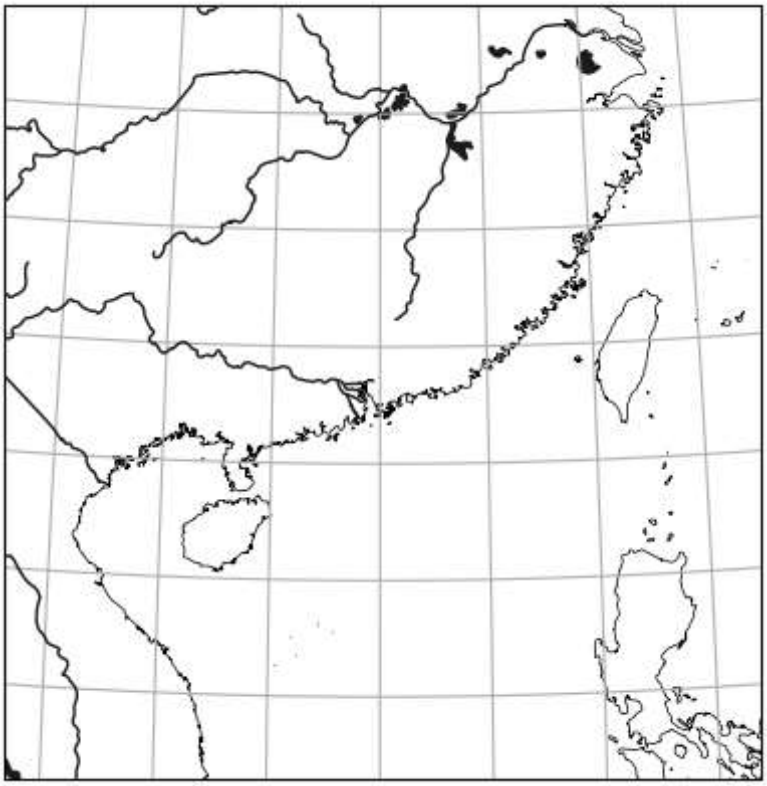
In [6]:
```
def plot_map(my_projection):
    # Create and define the size of a figure object
    plt.figure(figsize=(5,5), dpi=100)

    # Create an axes with Orthographic projection style
    ax = plt.axes(projection=my_projection)

    # Add natural features to axes using cartopy.feature (cfeature)
    ax.add_feature(cfeature.OCEAN, zorder=0)
    ax.add_feature(cfeature.LAND, edgecolor='black', facecolor='grey', zorder=1)
    ax.add_feature(cfeature.LAKES, edgecolor='blue', facecolor='blue', zorder=2)

    # Add border lines over countries
    ax.add_feature(cfeature.NaturalEarthFeature(category='cultural',
                                                name='admin_0_countries',
                                                scale='110m',
                                                facecolor='none',
                                                edgecolor='black',
                                                linewidth=0.5))

    # Add lat/lon gridlines, draw gridlines
    gl = ax.gridlines(crs=ccrs.PlateCarree(), linewidth=1, color='black', alpha=0.5

    # Manipulate latitude and longitude gridline numbers and spacing
    gl.ylocator = mticker.FixedLocator(np.arange(-90,91,30))
    gl.xlocator = mticker.FixedLocator(np.arange(-180, 181, 30))

    # Add title
    ax.set_title(f'{type(my_projection)}')

# Set a list of projections
projections = [ccrs.Robinson()]

# Loop the projections and call the plotting function
for proj in projections:
    plot_map(proj)
```



In [4]:
```
# Create and define the size of a figure object
plt.figure(figsize=(5,5), dpi=100)

# Set Orthographic projection style
central_lon, central_lat = 114.06, 22.54 # Shenzhen
proj = ccrs.Orthographic(central_lon, central_lat)

# Create an axes with Orthographic projection style
ax = plt.axes(projection=proj)

# Set a region and plot
extent = [central_lon-10, central_lon+10, central_lat-10, central_lat+10]
ax.set_extent(extent)

# Add features to axes using cartopy.feature (cfeature)
ax.add_feature(cfeature.LAKES, edgecolor='blue', facecolor='blue', zorder=2)
ax.add_feature(cfeature.RIVERS, edgecolor='blue', zorder=3)

# Add features to axes using methods
ax.coastlines(resolution='10m', linewidth=0.5)
ax.gridlines()
```

Out[4]:
```
<cartopy.mpl.gridliner.Gridliner at 0x2771a186310>
```

```
D:\ANACONDA\envs\cper\lib\site-packages\cartopy\crs.py:245: ShapelyDeprecationWarnin
g: __len__ for multi-part geometries is deprecated and will be removed in Shapely 2.
0. Check the length of the `geoms` property instead to get the  number of parts of a
multi-part geometry.
  if len(multi_line_string) > 1:
D:\ANACONDA\envs\cper\lib\site-packages\cartopy\crs.py:297: ShapelyDeprecationWarnin
g: Iteration over multi-part geometries is deprecated and will be removed in Shapely
2.0. Use the `geoms` property to access the constituent parts of a multi-part geomet
ry.
  for line in multi_line_string:
D:\ANACONDA\envs\cper\lib\site-packages\cartopy\crs.py:364: ShapelyDeprecationWarnin
g: __len__ for multi-part geometries is deprecated and will be removed in Shapely 2.
0. Check the length of the `geoms` property instead to get the  number of parts of a
multi-part geometry.
  if len(p_mline) > 0:
```



In [2]:
```
pip install pdfkit
```

```
Collecting pdfkit
  Downloading pdfkit-1.0.0-py3-none-any.whl (12 kB)
Installing collected packages: pdfkit
Successfully installed pdfkit-1.0.0
Note: you may need to restart the kernel to use updated packages.
```