```python
#□□□
import pandas as pd
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature

# 1. □□□□
df = pd.read_csv('usgs_earthquakes.csv')

# 2. □□□□□□□□□□50□
df_sorted = df.sort_values(by='mag', ascending=False)
df_top50 = df_sorted.head(50)

# 3. □□□□
fig, ax = plt.subplots(figsize=(15, 10),
                       subplot_kw={'projection': ccrs.Robinson()})

# □□□□□□
ax.add_feature(cfeature.LAND, facecolor='lightgray')
ax.add_feature(cfeature.OCEAN, facecolor='lightblue')
ax.add_feature(cfeature.COASTLINE, linewidth=0.5)
ax.add_feature(cfeature.BORDERS, linestyle=':', linewidth=0.5)
ax.gridlines(draw_labels=True, linewidth=0.5, color='gray', alpha=0.5)

# 4. □□□□□□□□□□□□□□□□
scatter = ax.scatter(df_top50['longitude'],
                     df_top50['latitude'],
                     c=df_top50['mag'],
                     cmap='viridis',
                     s=df_top50['mag']**2 * 10,  # □□□□□□□
                     transform=ccrs.PlateCarree(),
                     edgecolor='black',
                     linewidth=0.5,
                     zorder=5)

# 5. □□□□
cbar = plt.colorbar(scatter, ax=ax, orientation='
```

```
vertical', pad=0.05, shrink=0.6)
cbar.set_label('Magnitude (mag)', fontsize=12)

# 6. □□□□
plt.title('Top 50 Earthquakes of 2014 by Magnitude', fontsize=16, pad=20)

# 7. □□□□□□□
plt.tight_layout()
plt.savefig('global_earthquakes_top50.png', dpi=300, bbox_inches='tight')
plt.show()

#□□□
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
from cartopy.mpl.gridliner import LONGITUDE_FORMATTER, LATITUDE_FORMATTER
import matplotlib.patches as mpatches

# ==================== 1. □□□□□□□ ====================
file_path = 'NCALDAS_NOAH0125_Trends.A198010_201509.002.nc'
ds = xr.open_dataset(file_path)

print("=== □□□□□ ===")
print(ds)
print("\n=== □□□□ ===")
for var in ds.variables:
    print(f"- {var}: {ds[var].attrs.get('long_name', '□□□')}")

print("\n=== □□□□ ===")
print(f"□□: {ds.dims}")

# □□□□□□□□□
temp_vars = []
```

```
for var in ds.variables:
    var_lower = var.lower()
    if any(keyword in var_lower for keyword in ['temp', 'tair', 't2m', 'temperature', 'air']):
        temp_vars.append(var)

if temp_vars:
    print(f"\n□□□□□□□□□: {temp_vars}")
    # □□□□□□□□□□□
    variable_name = temp_vars[0]
else:
    # □□□□□□□□□□□□□□□□□□□
    trend_vars = [var for var in ds.variables if 'Trend' in var]
    if trend_vars:
        print(f"\n□□□□□□□□□□□□□□□: {trend_vars[0]}")
        variable_name = trend_vars[0]
    else:
        # □□□□□□□
        non_coord_vars = [var for var in ds.variables if var not in ['lat', 'lon', 'time', 'climatology_bounds']]
        if non_coord_vars:
            variable_name = non_coord_vars[0]
            print(f"\n□□□□: {variable_name}")

print(f"\n□□□□□□□: {variable_name}")

# ==================== 2. □□□□ ====================
# □□□□□□□
data_var = ds[variable_name]

# □□□□□□
var_long_name = data_var.attrs.get('long_name', variable_name)
var_units = data_var.attrs.get('units', 'unknown')

print(f"□□□□: {var_long_name}")
print(f"□□: {var_units}")

# □□□□□□
```

```
108 print(f"□□□□: {data_var.dims}")
109 print(f"□□□□: {data_var.shape}")
110
111 # □□□□□□□
112 if 'lon' in ds.variables:
113     lon = ds['lon'].values
114 elif 'longitude' in ds.variables:
115     lon = ds['longitude'].values
116 else:
117     # □□□□□□□□
118     if 'lon' in data_var.coords:
119         lon = data_var['lon'].values
120     else:
121         raise KeyError("□□□□□□□□")
122
123 if 'lat' in ds.variables:
124     lat = ds['lat'].values
125 elif 'latitude' in ds.variables:
126     lat = ds['latitude'].values
127 else:
128     # □□□□□□□□
129     if 'lat' in data_var.coords:
130         lat = data_var['lat'].values
131     else:
132         raise KeyError("□□□□□□□□")
133
134 print(f"□□□□: {lon.min():.2f} □ {lon.max():.2f}")
135 print(f"□□□□: {lat.min():.2f} □ {lat.max():.2f}")
136
137 # □□□□□□□□3D□□□□□□□□□□□□□□□□□□□□
138 if 'time' in data_var.dims:
139     print(f"□□□□□□□□□□ {len(data_var['time'])} □□□□"
    )
140     # □□□□□□□□□□
141     data_2d = data_var.isel(time=0)
142     print(f"□□□□□□□□□□□□")
143 else:
144     data_2d = data_var
145
146 # □□□□□2D□
147 if len(data_2d.shape) == 2:
```

```
148     data_values = data_2d.values
149 elif len(data_2d.shape) == 1:
150     # □□□1D□□□□□□□□
151     print("□□□1D□□□□□□□...")
152     data_values = data_2d.values.reshape(len(lat),
    len(lon))
153 else:
154     print(f"□□□□: {data_2d.shape}")
155     # □□□□□□□□
156     data_values = data_2d.isel({dim: 0 for dim in
    data_2d.dims if dim not in ['lat', 'lon']}).values
157
158 # □□□□□□□
159 lon_grid, lat_grid = np.meshgrid(lon, lat)
160
161 print(f"□□□□□: {np.nanmin(data_values):.4f} □ {np.
    nanmax(data_values):.4f}")
162
163 # ==================== 3.1 □□□□ ====================
164 print("\n=== □□□□□□ ===")
165 fig1 = plt.figure(figsize=(16, 10))
166
167 # □□Robinson□□
168 proj_global = ccrs.Robinson(central_longitude=0)
169 ax1 = plt.subplot(111, projection=proj_global)
170
171 # □□□□□□
172 ax1.add_feature(cfeature.LAND, facecolor='lightgray'
    , alpha=0.2)
173 ax1.add_feature(cfeature.OCEAN, facecolor='lightblue
    ', alpha=0.2)
174 ax1.add_feature(cfeature.COASTLINE, linewidth=0.5,
    alpha=0.8)
175 ax1.add_feature(cfeature.BORDERS, linestyle=':',
    linewidth=0.3, alpha=0.5)
176
177 # □□□□□
178 gl1 = ax1.gridlines(draw_labels=True, linewidth=0.5
    , color='gray',
179                     alpha=0.5, linestyle='--')
180 gl1.top_labels = False
```

```
gl1.right_labels = False
gl1.xformatter = LONGITUDE_FORMATTER
gl1.yformatter = LATITUDE_FORMATTER
gl1.xlabel_style = {'size': 10}
gl1.ylabel_style = {'size': 10}

# □□□□□□□contourf□
# □□□□□□□□□
vmin, vmax = np.nanpercentile(data_values, [5, 95])
levels = np.linspace(vmin, vmax, 15)

cf1 = ax1.contourf(lon_grid, lat_grid, data_values,
                   transform=ccrs.PlateCarree(),
                   cmap='RdYlBu_r', levels=levels, extend='both')

# □□□□
cbar1 = plt.colorbar(cf1, ax=ax1, orientation='horizontal',
                     pad=0.08, shrink=0.8)
cbar1.set_label(f'{var_long_name} ({var_units})', fontsize=12, fontweight='bold')

# □□□□
title1 = f'Global Trends: {var_long_name}\nNCALDAS_NOAH0125 Dataset (1980-2015)'
ax1.set_title(title1, fontsize=16, fontweight='bold', pad=20)

# □□□□□
textbox_content = f"""Data Source: NASA GES DISC
Dataset: NCALDAS_NOAH0125
Variable: {variable_name}
Period: 1980-2015
Resolution: {ds.attrs.get('spatial_resolution', '1.25°')}
Map Projection: Robinson"""
ax1.text(0.02, 0.02, textbox_content, transform=ax1.transAxes,
         fontsize=9, verticalalignment='bottom', fontfamily='monospace',
```

```
214           bbox=dict(boxstyle='round', facecolor='
    white', alpha=0.9, edgecolor='gray'))
215 
216 # □□□□
217 if 'Trend' in variable_name:
218     trend_type = variable_name.replace('Trend_', '')
219     ax1.annotate(f'Highest {trend_type}\nTrend Area'
    ,
220                  xy=(150, 60), xycoords=ccrs.
    PlateCarree()._as_mpl_transform(ax1),
221                  xytext=(120, 40), textcoords=ccrs.
    PlateCarree()._as_mpl_transform(ax1),
222                  arrowprops=dict(arrowstyle='->',
    color='darkred', lw=1.5),
223                  fontsize=10, fontweight='bold',
    color='darkred',
224                  bbox=dict(boxstyle='round',
    facecolor='white', alpha=0.8))
225 
226 # □□□□
227 from matplotlib.lines import Line2D
228 
229 legend_elements = [
230     Line2D([0], [0], marker='s', color='w', label='
    Positive Trend',
231            markerfacecolor='darkred', markersize=12
    ),
232     Line2D([0], [0], marker='s', color='w', label='
    Negative Trend',
233            markerfacecolor='darkblue', markersize=12
    ),
234     Line2D([0], [0], marker='s', color='w', label='
    No Significant Trend',
235            markerfacecolor='lightgray', markersize=
    12)
236 ]
237 
238 ax1.legend(handles=legend_elements, loc='upper left'
    , fontsize=10,
239            title='Trend Categories', title_fontsize=
    11)
```

```
240
241 # □□□□□□□
242 ax1.set_xlabel('Longitude', fontsize=12, labelpad=25
    )
243 ax1.set_ylabel('Latitude', fontsize=12, labelpad=25)
244
245 plt.tight_layout()
246 plt.savefig('global_trends_map.png', dpi=300,
    bbox_inches='tight')
247 print("□□□□□□□□ 'global_trends_map.png'")
248
249 # ==================== 3.2 □□□□ ====================
250 print("\n=== □□□□□□ ===")
251 fig2 = plt.figure(figsize=(14, 10))
252
253 # □□PlateCarree□□□□□□□□□□□□□□
254 proj_regional = ccrs.PlateCarree()
255 ax2 = plt.subplot(111, projection=proj_regional)
256
257 # □□□□□□□□□□
258 region_extent = [80, 150, 0, 60]  # [lon_min,
    lon_max, lat_min, lat_max]
259 ax2.set_extent(region_extent, crs=proj_regional)
260
261 # □□□□□□
262 ax2.add_feature(cfeature.LAND, facecolor='lightgray'
    , alpha=0.2)
263 ax2.add_feature(cfeature.OCEAN, facecolor='lightblue
    ', alpha=0.2)
264 ax2.add_feature(cfeature.COASTLINE, linewidth=1.0)
265 ax2.add_feature(cfeature.BORDERS, linestyle=':',
    linewidth=0.8, alpha=0.7)
266 ax2.add_feature(cfeature.LAKES, facecolor='lightblue
    ', alpha=0.5)
267
268 # □□□□□□□□□□□□
269 try:
270     ax2.add_feature(cfeature.NaturalEarthFeature('
    cultural', 'admin_0_countries',
271                                                 '
    50m', edgecolor='black',
```

```
272
    facecolor='none', linewidth=0.5))
273 except:
274     pass
275
276 # □□□□□
277 gl2 = ax2.gridlines(draw_labels=True, linewidth=0.5
    , color='gray',
278                     alpha=0.7, linestyle='--')
279 gl2.top_labels = False
280 gl2.right_labels = False
281 gl2.xformatter = LONGITUDE_FORMATTER
282 gl2.yformatter = LATITUDE_FORMATTER
283 gl2.xlabel_style = {'size': 10, 'weight': 'bold'}
284 gl2.ylabel_style = {'size': 10, 'weight': 'bold'}
285
286 # □□□□□□
287 lon_mask = (lon >= region_extent[0]) & (lon <=
    region_extent[1])
288 lat_mask = (lat >= region_extent[2]) & (lat <=
    region_extent[3])
289
290 if np.any(lon_mask) and np.any(lat_mask):
291     lon_regional = lon[lon_mask]
292     lat_regional = lat[lat_mask]
293
294     # □□□□□□□□□
295     lat_indices = np.where(lat_mask)[0]
296     lon_indices = np.where(lon_mask)[0]
297     data_regional = data_values[np.ix_(lat_indices,
    lon_indices)]
298
299     # □□□□□□
300     lon_grid_regional, lat_grid_regional = np.
    meshgrid(lon_regional, lat_regional)
301
302     # □□□□
303     cf2 = ax2.contourf(lon_grid_regional,
    lat_grid_regional, data_regional,
304                        transform=ccrs.PlateCarree(),
305                        cmap='RdBu_r', levels=15,
```

```
305 extend='both', alpha=0.8)
306 else:
307     # □□□□□□□□□□□□□□□□□□□□
308     print("□□: □□□□□□□□□□□□□□□")
309     cf2 = ax2.contourf(lon_grid, lat_grid,
    data_values,
310                        transform=ccrs.PlateCarree(),
311                        cmap='RdBu_r', levels=15,
    extend='both', alpha=0.8)
312
313 # □□□□
314 cbar2 = plt.colorbar(cf2, ax=ax2, orientation='
    vertical',
315                      pad=0.03, shrink=0.8)
316 cbar2.set_label(f'{var_long_name} ({var_units})',
    fontsize=12, fontweight='bold')
317
318 # □□□□
319 title2 = f'Regional Trends: East Asia\n{
    var_long_name}\nNCALDAS_NOAH0125 Dataset (1980-2015
    )'
320 ax2.set_title(title2, fontsize=16, fontweight='bold'
    , pad=20)
321
322 # □□□□□
323 textbox_content_regional = f"""Region: East Asia
324 Longitude: {region_extent[0]}°E to {region_extent[1]
    }°E
325 Latitude: {region_extent[2]}°N to {region_extent[3]}
    °N
326 Projection: PlateCarree
327 Variable: {variable_name}"""
328 ax2.text(0.02, 0.98, textbox_content_regional,
    transform=ax2.transAxes,
329          fontsize=9, verticalalignment='top',
    fontfamily='monospace',
330          bbox=dict(boxstyle='round', facecolor='
    white', alpha=0.9, edgecolor='gray'))
331
332 # □□□□□□□□□□□□□□□
333 annotations = [
```

```
334     ('Tibetan Plateau\nHigh Elevation', 90, 35, 85,
    30),
335     ('East China\nDense Population', 120, 30, 125,
    25),
336     ('Siberia\nCold Region', 110, 55, 105, 50)
337 ]
338 
339 for text, x, y, tx, ty in annotations:
340     ax2.annotate(text, xy=(x, y), xycoords=ccrs.
    PlateCarree()._as_mpl_transform(ax2),
341                  xytext=(tx, ty), textcoords=ccrs.
    PlateCarree()._as_mpl_transform(ax2),
342                  arrowprops=dict(arrowstyle='->',
    color='darkgreen', lw=1.5, alpha=0.8),
343                  fontsize=9, fontweight='bold',
    color='darkgreen',
344                  bbox=dict(boxstyle='round,pad=0.3'
    , facecolor='white', alpha=0.8))
345 
346 # ????
347 legend_elements_regional = [
348     mpatches.Patch(facecolor='darkred', alpha=0.8,
    label='Strong Positive'),
349     mpatches.Patch(facecolor='red', alpha=0.8, label
    ='Moderate Positive'),
350     mpatches.Patch(facecolor='lightgray', alpha=0.8
    , label='Near Zero'),
351     mpatches.Patch(facecolor='blue', alpha=0.8,
    label='Moderate Negative'),
352     mpatches.Patch(facecolor='darkblue', alpha=0.8,
    label='Strong Negative')
353 ]
354 
355 ax2.legend(handles=legend_elements_regional, loc='
    lower right', fontsize=9,
356            title=f'{trend_type if "Trend" in
    variable_name else "Value"} Range',
357            title_fontsize=10)
358 
359 # ???????
360 ax2.set_xlabel('Longitude (°E)', fontsize=12,
```

```
360 labelpad=15, fontweight='bold')
361 ax2.set_ylabel('Latitude (°N)', fontsize=12,
    labelpad=20, fontweight='bold')
362
363 plt.tight_layout()
364 plt.savefig(f'regional_trends_{variable_name}.png',
    dpi=300, bbox_inches='tight')
365 print(f"□□□□□□□□ 'regional_trends_{variable_name}.
    png'")
366
367 # ==================== 4. □□□□□□
     ====================
368 plt.show()
369
370 # □□□□□
371 ds.close()
372
373 print("\n=== □□□□ ===")
374 print(f"1. □□□□: global_trends_map.png")
375 print(f"2. □□□□: regional_trends_{variable_name}.png
    ")
376 print(f"\n□□□□□: {variable_name}")
377 print(f"□□□□: {var_long_name}")
```