In [307]:
```
import numpy as np
import pandas as pd
import xarray as xr
import netCDF4
from matplotlib import pyplot as plt
%matplotlib inline
```

1 Niño 3.4 index

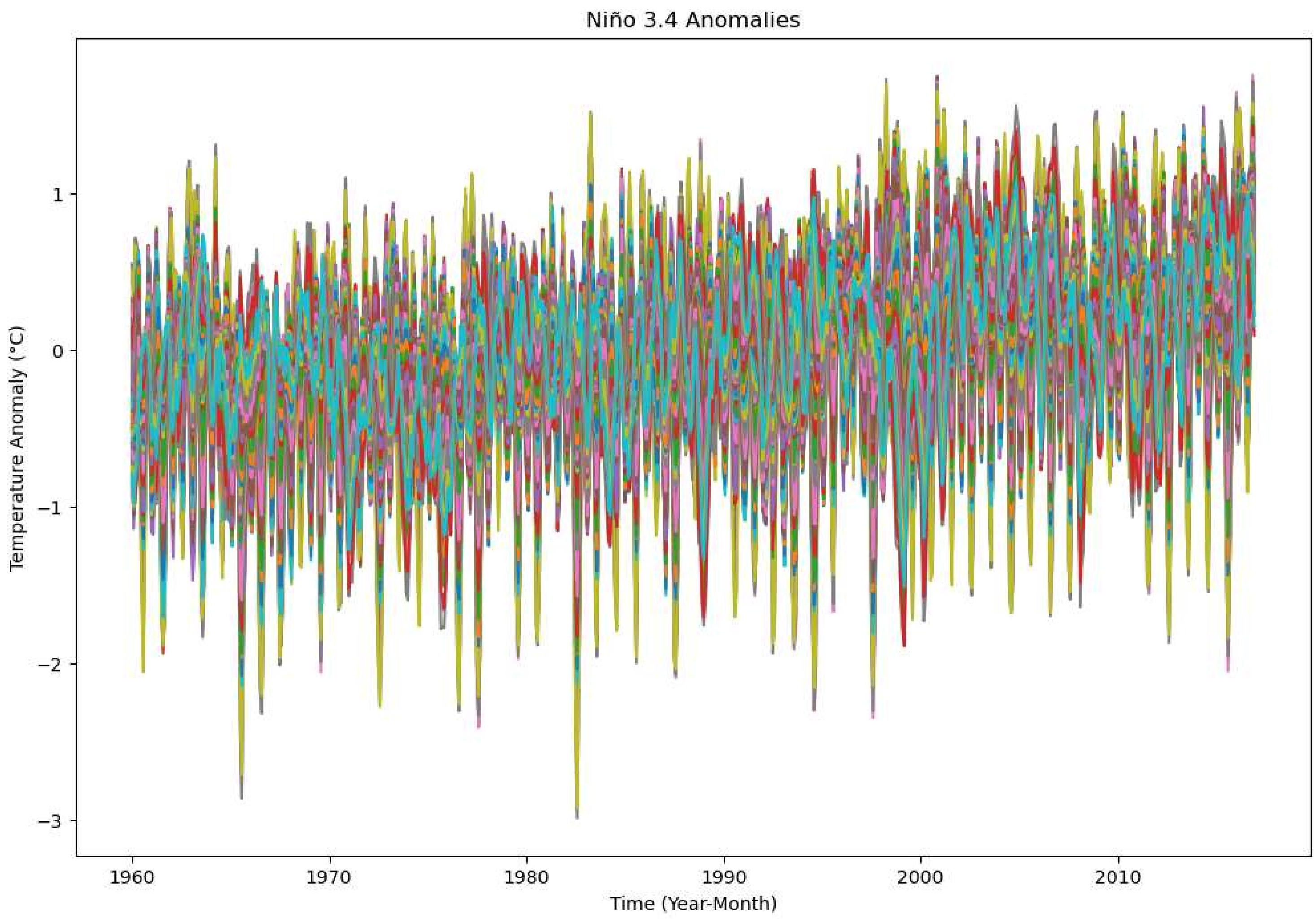
In [355]:
```
# Load the dataset
ds = nc.Dataset('NOAA_NCDC_ERSST_v3b_SST.nc')

# Extract the SST variable
sst = ds.variables['sst'][:]

# Define the Niño 3.4 region (5N-5S, 170W-120W)
lat_mask = (ds.variables['lat'][:] >= -5) & (ds.variables['lat'][:] <= 5)
lon_mask = (ds.variables['lon'][:] >= 120) & (ds.variables['lon'][:] <= 170)

# Create a 2D mask for the Niño 3.4 region
mask = np.outer(lat_mask, lon_mask)

# Extract the data from the Niño 3.4 region
nino34_region = sst[:, mask]
```

1.1

In [354]:
```
# Compute the monthly climatology
climatology = np.mean(nino34_region, axis=0)

# Compute the anomalies
anomalies = nino34_region - climatology

# Convert the time variable to datetime format
import datetime
time = ds.variables['time'][:]
time_units = ds.variables['time'].units
time_calendar = ds.variables['time'].calendar
dates = nc.num2date(time, units=time_units, calendar=time_calendar)

# Convert dates to year-month format
dates = [datetime.datetime(date.year, date.month, 1) for date in dates]
```

1.2

In [353]:
```
# Visualize the computed Niño 3.4
plt.figure(figsize=(12, 8))
plt.plot(dates, anomalies)
plt.title('Niño 3.4 Anomalies')
plt.xlabel('Time (Year-Month)')
plt.ylabel('Temperature Anomaly (° C)')
plt.show()
```



2 Earth's energy budget

In [160]:
```
# load the dataset
TOA = xr.open_dataset("CERES_EBAF-TOA_200003-201701.nc", engine="netcdf4")
TOA.data_vars
#TOA['cldarea_total_daynight_mon']
```

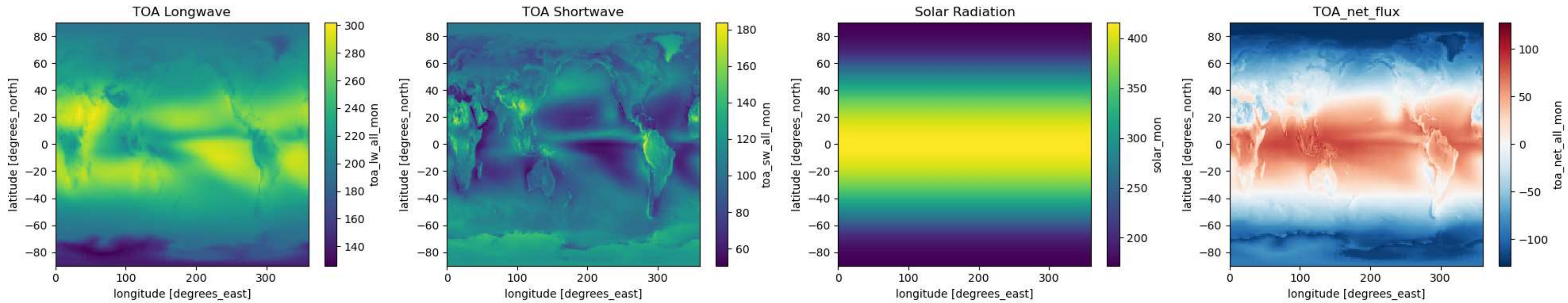
Out[160]:
```
Data variables:
    toa_sw_all_mon              (time, lat, lon) float32 ...
    toa_lw_all_mon              (time, lat, lon) float32 ...
    toa_net_all_mon             (time, lat, lon) float32 ...
    toa_sw_clr_mon              (time, lat, lon) float32 ...
    toa_lw_clr_mon              (time, lat, lon) float32 ...
    toa_net_clr_mon             (time, lat, lon) float32 ...
    toa_cre_sw_mon              (time, lat, lon) float32 ...
    toa_cre_lw_mon              (time, lat, lon) float32 ...
    toa_cre_net_mon             (time, lat, lon) float32 ...
    solar_mon                   (time, lat, lon) float32 ...
    cldarea_total_daynight_mon  (time, lat, lon) float32 ...
    cldpress_total_daynight_mon (time, lat, lon) float32 ...
    cldtemp_total_daynight_mon  (time, lat, lon) float32 ...
    cldtau_total_day_mon        (time, lat, lon) float32 ...
```

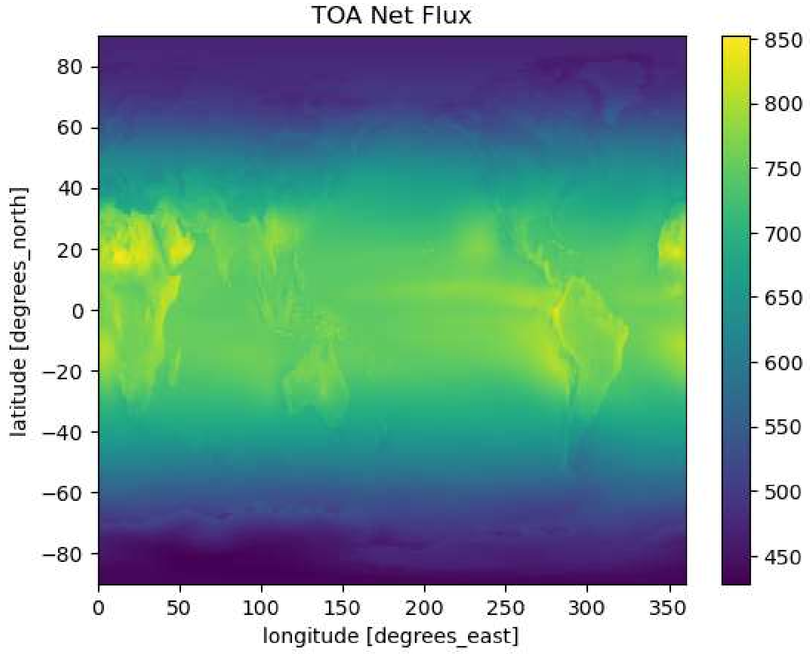
2.1

Make a plot of the time-mean TOA longwave, shortwave, and solar radiation for all-sky conditions

In [146]:

```
# Compute the time-mean TOA longwave, shortwave, and solar radiation for all-sky conditions
toa_lw_all_mon = TOA['toa_lw_all_mon'].mean(dim='time')
toa_sw_all_mon = TOA['toa_sw_all_mon'].mean(dim='time')
solar_mon = TOA['solar_mon'].mean(dim='time')
toa_net_all_mon = TOA['toa_net_all_mon'].mean(dim='time')

# Make a plot of the time-mean TOA longwave, shortwave, and solar radiation for all-sky conditions
fig, axes = plt.subplots(ncols=4, figsize=(20, 4))
toa_lw_all_mon.plot(ax=axes[0])
axes[0].set_title('TOA Longwave')
toa_sw_all_mon.plot(ax=axes[1])
axes[1].set_title('TOA Shortwave')
solar_mon.plot(ax=axes[2])
axes[2].set_title('Solar Radiation')
toa_net_all_mon.plot(ax=axes[3])
axes[3].set_title('TOA_net_flux')
plt.tight_layout()
plt.show()

# Add up the three variables above and verify (visually) that they are equivalent to the TOA net flux
net_flux = toa_lw_all_mon + toa_sw_all_mon + solar_mon
net_flux.plot()
plt.title('TOA Net Flux')
plt.show()
```





the sum of TOA longwave, shortwave, and solar radiation for all-sky conditions is not equivalent to the TOA net flux.

2.2

Calculate and verify that the TOA incoming solar, outgoing longwave, and outgoing shortwave approximately match up with the cartoon above

In [149]:

```
# Calculate the global mean values
glb_mean_toa_lw_all_mon = toa_lw_all_mon.mean(dim=['lat', 'lon']).values
glb_mean_toa_sw_all_mon = toa_sw_all_mon.mean(dim=['lat', 'lon']).values
glb_mean_solar_mon = solar_mon.mean(dim=['lat', 'lon']).values

print(f'Global mean TOA longwave radiation: {glb_mean_toa_lw_all_mon}')
print(f'Global mean TOA shortwave radiation: {glb_mean_toa_sw_all_mon}')
print(f'Global mean solar radiation: {glb_mean_solar_mon}')
```

```
Global mean TOA longwave radiation: 224.7551727294922
Global mean TOA shortwave radiation: 102.30432891845703
Global mean solar radiation: 298.3305358886719
```

As the cartoon showes, incoming solar radiation(340.4) = total reflected solar radiation(99.9) + total outgoing infrared radiation(239.9) + net absorbed(0.6)

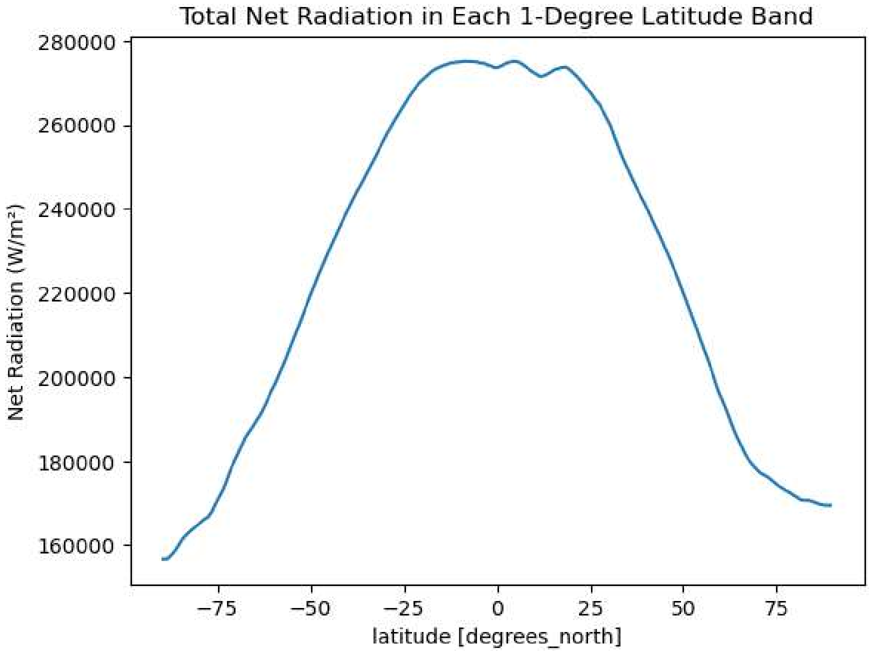
My calculation outputs are as given above, Global mean TOA longwave radiation(224.7551727294922) ≈ 239.9， Global mean TOA shortwave radiation(102.30432891845703) ≈ 99.9, and Global mean solar radiation(298.3305358886719) ≈ 340.4

2.3

Calculate and plot the total amount of net radiation in each 1-degree latitude band.

In [155]:

```
# Calculate and plot the total amount of net radiation in each 1-degree latitude band
total_net_radiation = net_flux.sum(dim='lon')
total_net_radiation.plot()
plt.ylabel('Net Radiation (W/m²)')
plt.title('Total Net Radiation in Each 1-Degree Latitude Band')
plt.show()
```



2.4

Calculate and plot composites of time-mean outgoing shortwave and longwave radiation for low and high cloud area regions.

In [175]:
```python
# define low cloud area as ≤25% and high cloud area as ≥75%
low_cloud_area = TOA['cldarea_total_daynight_mon'] <= 25
high_cloud_area = TOA['cldarea_total_daynight_mon'] >= 75

# time-mean for low and high cloud area regions
low_cloud_sw = TOA['toa_sw_all_mon'].where(low_cloud_area).mean(dim='time')
high_cloud_sw = TOA['toa_sw_all_mon'].where(high_cloud_area).mean(dim='time')
low_cloud_lw = TOA['toa_lw_all_mon'].where(low_cloud_area).mean(dim='time')
high_cloud_lw = TOA['toa_lw_all_mon'].where(high_cloud_area).mean(dim='time')

lowcld_sl = low_cloud_sw + low_cloud_lw
highcld_sl = high_cloud_sw + high_cloud_lw

# plot low and high cloud composites outgoing shortwave and longwave radiation
fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(8, 8))
low_cloud_sw.plot(ax=axs[0, 0])
axs[0,0].set_title('shortwave, low cloud')
high_cloud_sw.plot(ax=axs[0, 1])
axs[0,1].set_title('shortwave, high cloud')
low_cloud_lw.plot(ax=axs[1, 0])
axs[1,0].set_title('longwave, low cloud')
high_cloud_lw.plot(ax=axs[1, 1])
axs[1,1].set_title('longwave, high cloud')
lowcld_sl.plot(ax=axs[2,0])
axs[2,0].set_title('the composite of sw and lw, low cloud')
highcld_sl.plot(ax=axs[2,1])
axs[2,1].set_title('the composite of sw and lw, high cloud')
plt.tight_layout()
plt.show()
```

...

2.5

Calculate the global mean values of shortwave and longwave radiation, composited in high and low cloud regions.

In [310]:
```python
# Calculate the global mean values of shortwave and longwave radiation, composited in high and low cloud regions
glb_mean_sw_lowhigh = (low_cloud_sw + high_cloud_sw).mean()
glb_mean_lw_lowhigh = (low_cloud_lw + high_cloud_lw).mean()
glb_mean_low_cld = lowcld_sl.mean()     # 全球low cloud region的平均太阳辐射（短波+长波）
glb_mean_high_cld = highcld_sl.mean()   # 全球high cloud region的平均太阳辐射（短波+长波）
print(f'Global mean values of shortwave radiation in low cloud and high cloud regions: {glb_mean_sw_lowhigh.values}')
print(f'Global mean values of longwave radiation in low cloud and high cloud regions: {glb_mean_lw_lowhigh.values}')
print(f'Global mean values of shortwave and longwave radiation in low cloud region: {glb_mean_low_cld.values}')
print(f'Global mean values of shortwave and longwave radiation in high cloud region: {glb_mean_high_cld.values}')
```

```
Global mean values of shortwave radiation in low cloud and high cloud regions: 218.92471313476562
Global mean values of longwave radiation in low cloud and high cloud regions: 430.665283203125
Global mean values of shortwave and longwave radiation in low cloud region: 321.8834533691406
Global mean values of shortwave and longwave radiation in high cloud region: 330.11669921875
```

As the cartoon showes, total reflected solar radiation is 99.9 W/m^2, total outgoing infrared radiation is 239.9 W/m^2. Compared with the calculation results given above, global mean shortwave radiation is 218.92 W/m^2 for low cloud and high cloud areas, global mean longwave radiation is 430.67 W/m^2 for low cloud and high cloud areas.and Global mean values of shortwave and longwave radiation in low cloud area is 321.88 W/m^2, global mean values of shortwave and longwave radiation in high cloud area is 330.12 W/m^2.

Therefore, clouds can scatter and absorb shortwave radiation from the sun and block longwave radiation emitted from the surface. And I found that high cloud areas scatter and absorb more solar radiation than low cloud areas. In addition, clouds emit longwave thermal radiation to the surface and into space, which plays a key role in the earth's energy budget.

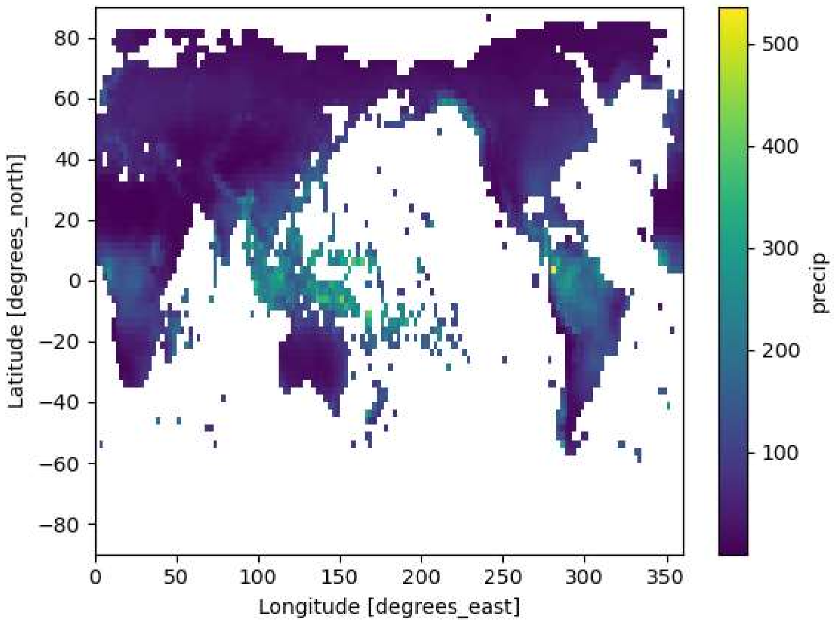
3 Explore a netCDF dataset

In [314]:

```
# Load the dataset
pp=xr.open_dataset("precip.mon.total.2.5x2.5.v7.nc",engine="netcdf4")

pp['precip'].mean(dim='time').plot()
```

Out[314]:

```
<matplotlib.collections.QuadMesh at 0x2622e9a8650>
```



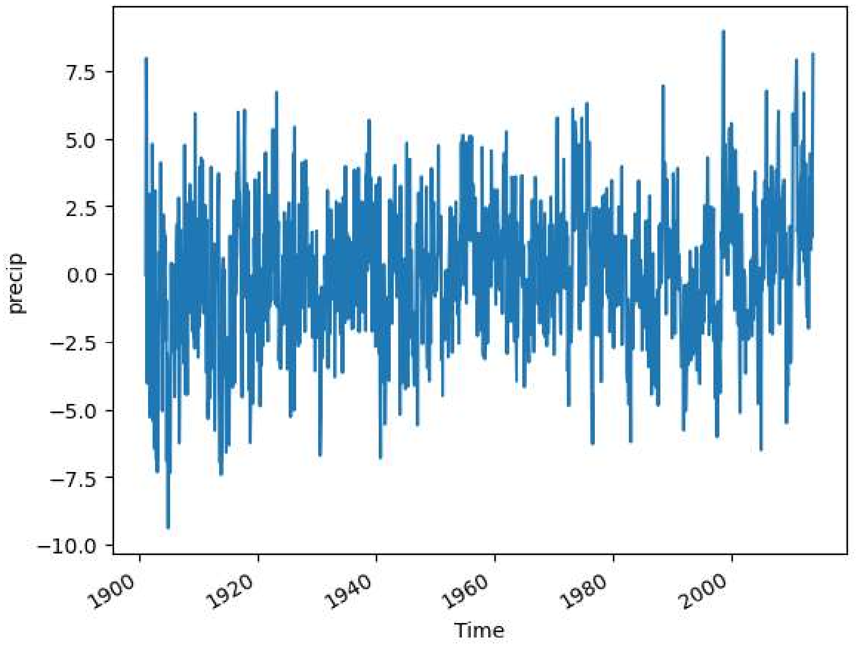
I use the monthly precipitation dataset from NOAA Physical Sciences Laboratory GPCC(Global Precipitation Climatology Centre) from 1891-present,the dataset is calculated from global station data. [https://psl.noaa.gov/data/gridded/data.gpcc.html#detail (https://psl.noaa.gov/data/gridded/data.gpcc.html#detail)](https://psl.noaa.gov/data/gridded/data.gpcc.html#detail)

3.1

Plot a time series of 'precip' with monthly seasonal cycle removed

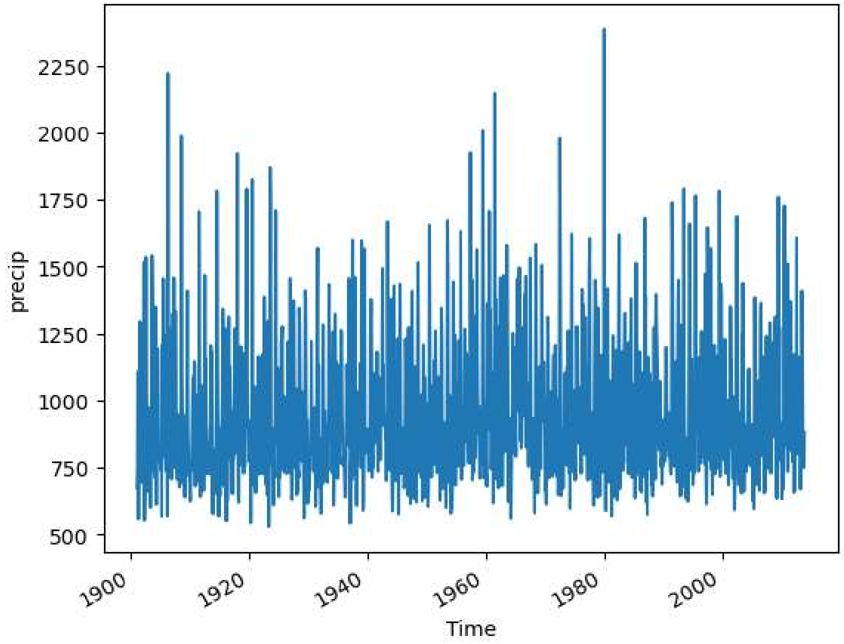
In [312]:
```
# Group data by month
mon_pp = pp.precip.groupby('time.month')
mon_pp.mean(dim='time')

# Apply mean to grouped data, and then compute the anomaly
precip_anom = mon_pp - mon_pp.mean(dim='time')
precip_anom

# Plot global mean anomalies
precip_anom.mean(dim=['lat','lon']).plot()
```

Out[312]: [<matplotlib.lines.Line2D at 0x2622eb53a10>]



In [345]:

Out[345]: [<matplotlib.lines.Line2D at 0x2622f7f3490>]

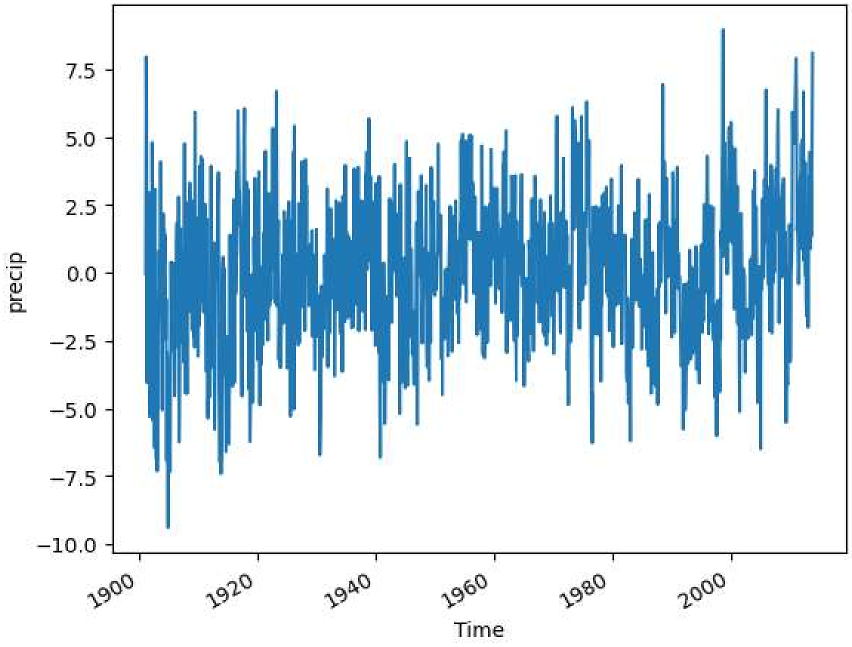


In [344]:

```
# Group data by month
mon_pp = pp.precip.groupby('time.month')

# Apply mean to grouped data, and then compute the anomaly
pp_anom = mon_pp - mon_pp.mean()
pp_anom

# Plot global mean anomalies
pp_anom.mean(dim=['lat','lon']).plot()
```

Out[344]:

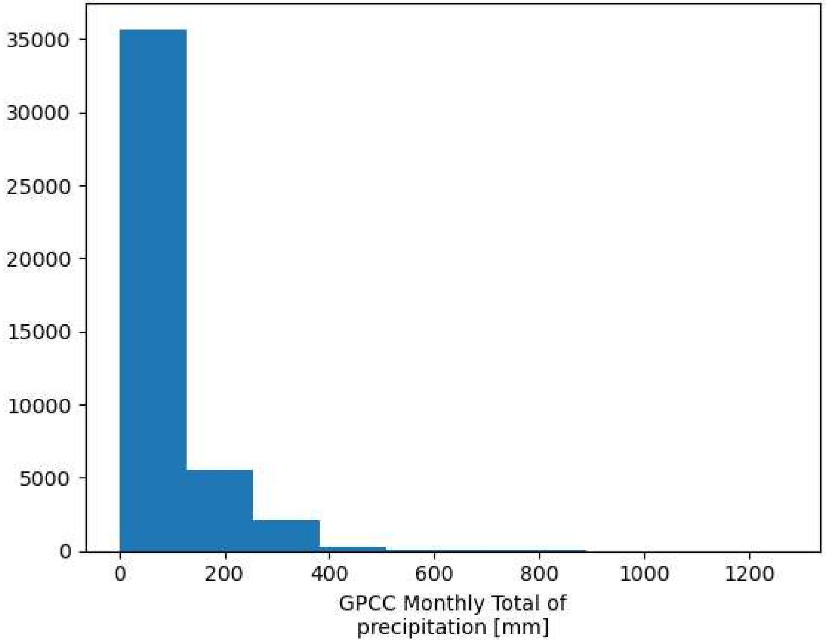
```
[<matplotlib.lines.Line2D at 0x2622ed48790>]
```

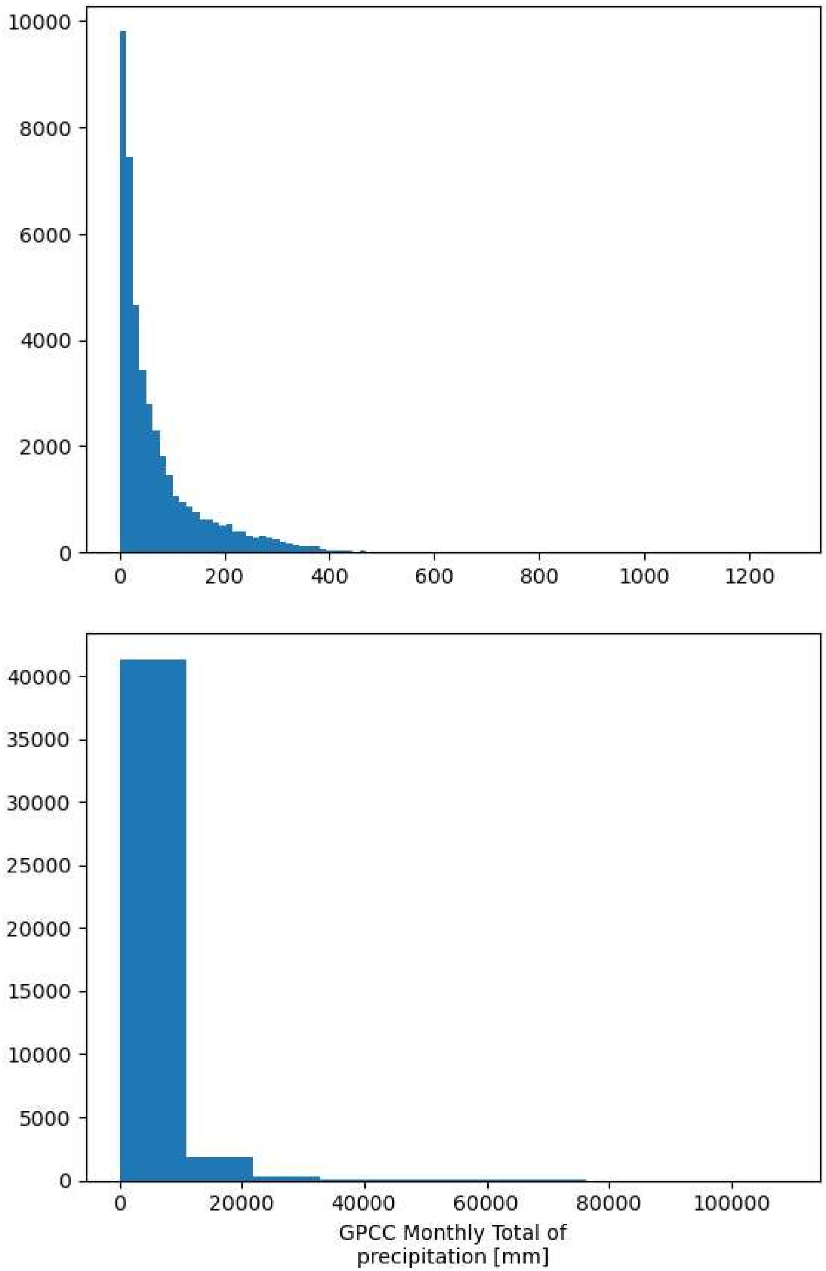


3.2
Make 5 plots using the dataset

Plot 1:
Various statistical graphs of the precipitation data

In [350]:
```
# Mean of precipitation
mon_pp.mean(dim='time').plot()
plt.show()

# Histogram of precipitation
plt.hist(mon_pp.mean(dim='time').values.flatten(), bins=100)
plt.show()

# Variance of the variable over time
mon_pp.var('time').plot()
plt.show()
```

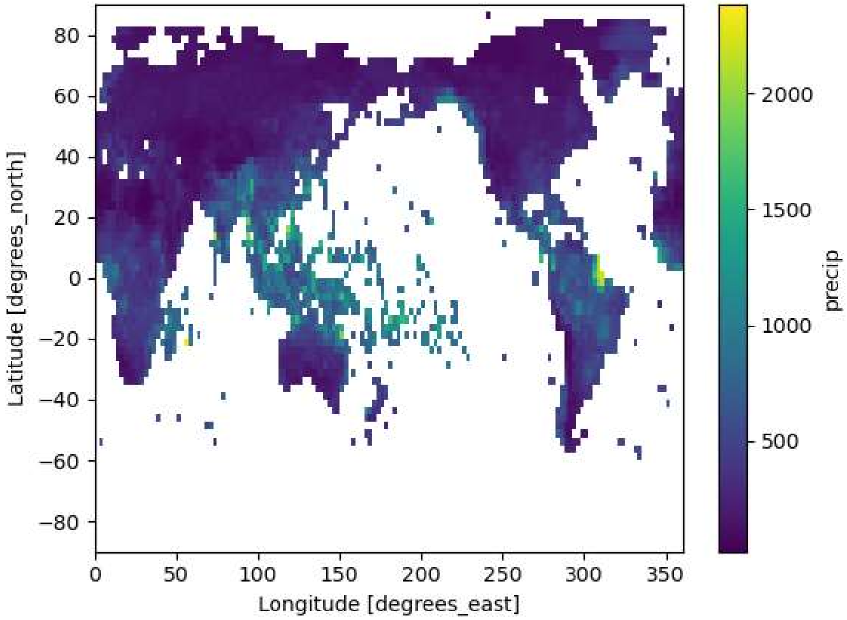



10000
8000
6000
4000
2000
0
0
200
400
600
800
1000
1200
40000
35000
30000
25000
20000
15000
10000
5000
0
0
20000
40000
60000
80000
100000
GPCC Monthly Total of
precipitation [mm]

Plot 2:

In [351]:
```
pp.precip.max(dim=['time']).plot()
```

Out[351]: <matplotlib.collections.QuadMesh at 0x2622fbdd050>

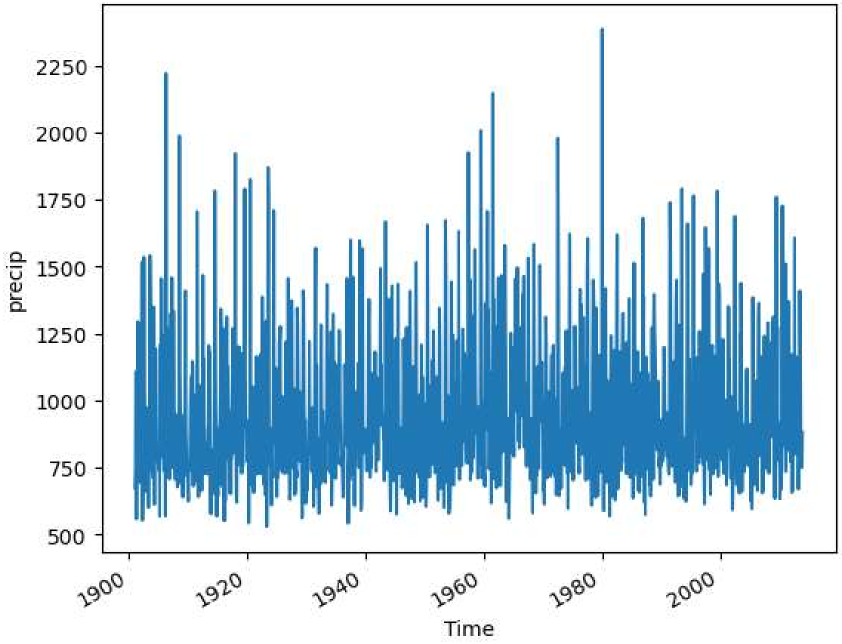


Plot 3:

Calculate the monthly max global precipitation in location

In [349]:
```
pp.precip.max(dim=['lat','lon']).plot()
```

Out[349]:
```
[<matplotlib.lines.Line2D at 0x26231673a10>]
```

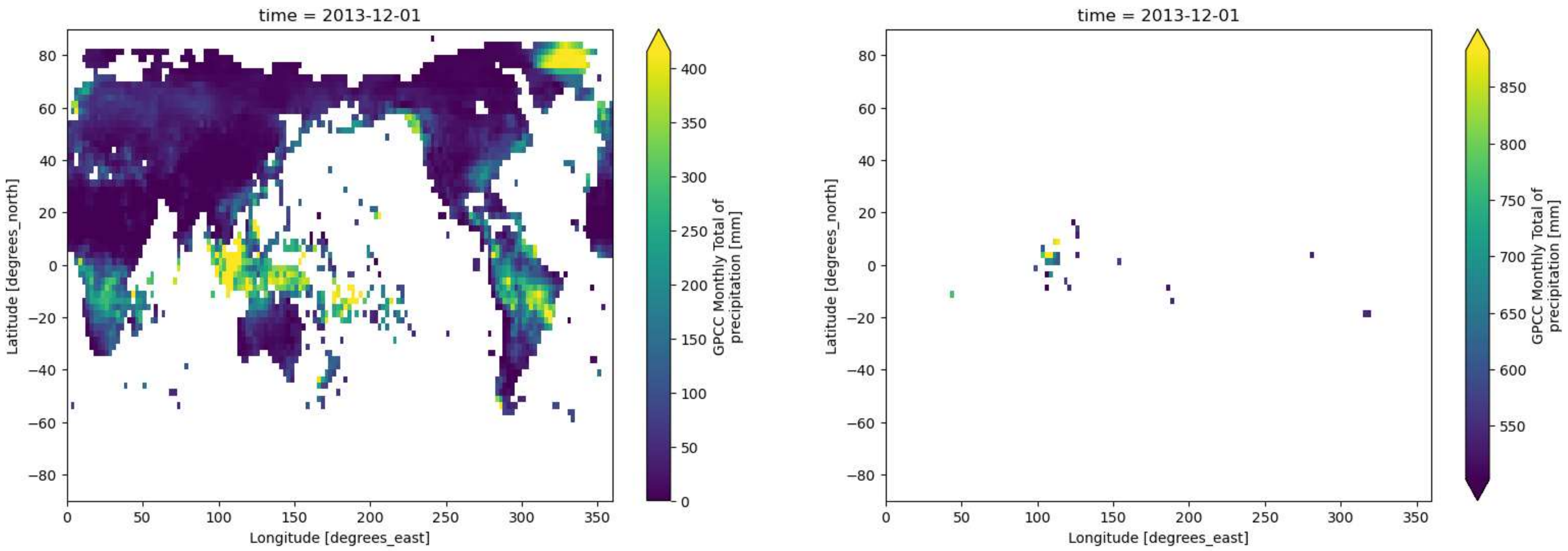


Plot 4 :

The precipitation(pp) of last year(2013) and select the area of pp>500

In [334]:
```python
# The last year pp
pp_new = pp.precip.isel(time=-1)
pp_new

# the last year pp where pp is higher than 500
masked_pp_new = pp_new.where(pp_new > 500.0)
masked_pp_new

# Plot 2 panels
fig, axes = plt.subplots(ncols=2, figsize=(19, 6))
pp_new.plot(ax=axes[0], robust=True)
masked_pp_new.plot(ax=axes[1], robust=True)
```

Out[334]:
```
<matplotlib.collections.QuadMesh at 0x2622e6f6ad0>
```



Note: the last year of the dataset is 2013.

Plot 5:

Calculate and plot zonal mean climatology

In [332]:
```
precip_clim = pp.precip.groupby('time.month').mean()
precip_clim.mean(dim='lon').transpose().plot.contourf(levels=12, robust=True, cmap='turbo')
```

Out[332]: <matplotlib.contour.QuadContourSet at 0x2621b97ae10>