Exercise 2- Precipitation and Freshwater flux

November 13, 2019

1 Exercise 2: Precipitation and Water cycle

Brief recap of today's lecture:
** Fluxes of water cycle**

- Precipitation
- Evaporation/ Evapotranspiration
- Infiltration
- Runoff

** Reservoirs** - Ocean - Ice and Glaciers - Rivers and Lakes - Ground water - Atmospheric water vapour and Clouds

This exercise focuses on **precipitation and evaporation** as two important fluxes for the water cycle. For human beings, precipitation is an essential water resource, since it affects for example agriculture and fresh water reservoirs. In addition, it is linked to everyday weather and storms. So there are several reasons, why climate scientists are interested in precipitation:

- understand which other factors in the climate system control precipitation patterns (knowledge can be integrated in climate models)
- prediction of heavy precipitation events/storms
- climatological trends of drying and wetting - find out where is it getting wetter, where is it getting dryier?

In addition to station measurements of precipitation, rain and snow can be monitored with modern satellite techniques. This has the advantage of giving us a much better spatial coverage, even for regions where measurements are more difficult (e.g. ocean).

1.1 Global Precipitation Measurement Mission

We will now have a look at precipitation from an international satellite network, which carries the currently most advanced satellite sensors for precipitation estimates.

More info: https://www.nasa.gov/mission_pages/GPM/main/index.html

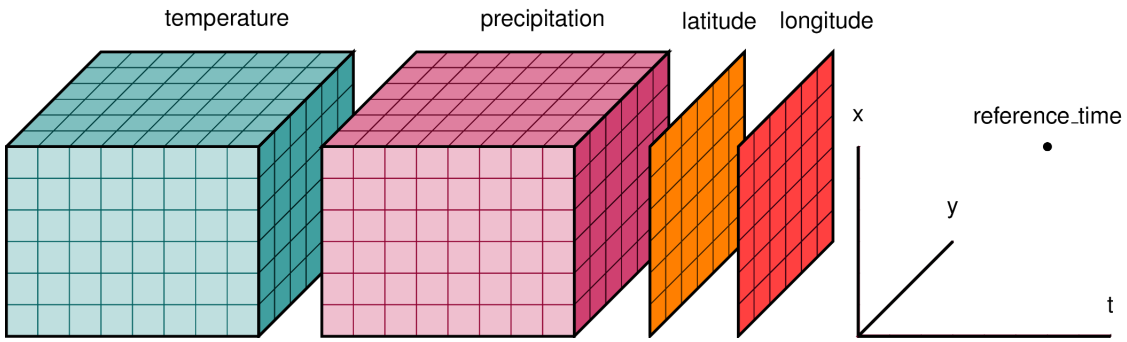
The data can be downloaded from: https://earthdata.nasa.gov/eosdis/daacs/gesdisc

The data format of the satellite observations is **netCDF4**, which is a very common data format for gridded climate data which you will for sure work with if you continue in the field of climate science!

1.2 This is how you can think of gridded climate data

In [153]:
```
from IPython.display import Image
Image(working_dir + 'datastructure.png')
```

Out[153]:



1.3 Read in data

In [154]:
```
# import python libraries (always the very first you need to do)
from netCDF4 import Dataset
import numpy as np
import matplotlib.pyplot as plt

# Ignore warnings (e.g. when using np.nanmean)
import warnings
warnings.filterwarnings('ignore')
```

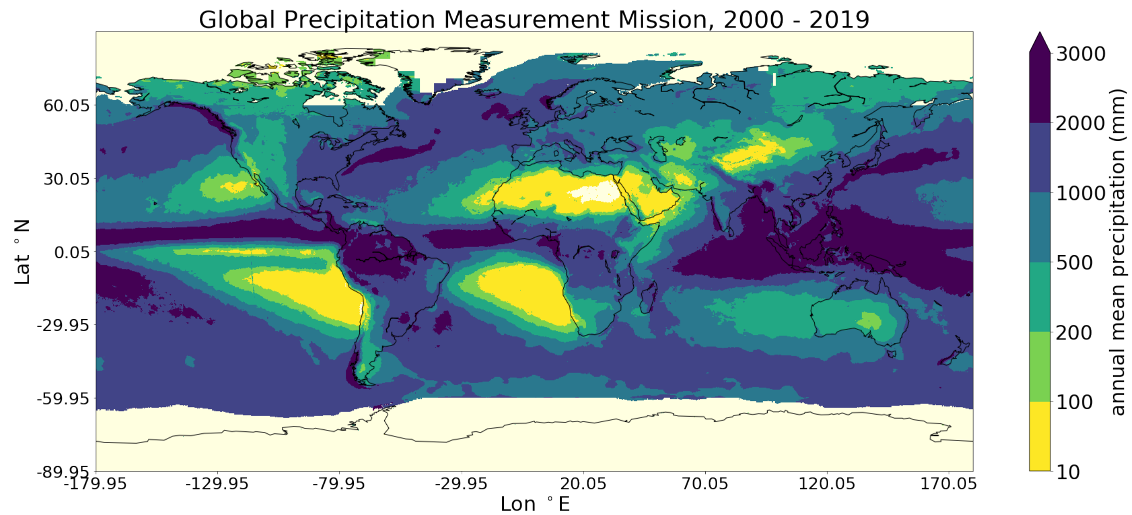
In [20]:
```
# defint your working directory
working_dir = '/home/juli/gpm_global/'
# read in file
file= working_dir + 'gpm_monmean.nc4'
ds= Dataset(file)
precip=np.array(ds['precipitation']).T
#precip[precip < 0]= np.nan
# compute annual mean based on seasonal values  (and convert unit from mm/hr to total p
annual_precip = np.nanmean(precip, axis = 2 )*24*365
lons = np.array(ds['lon'])
lats = np.array(ds['lat'])
```

In [11]:
```
# define here an output directory, where you want to save plots
output_dir= '/home/juli/Desktop/phd/teaching/GV1410/exercises/'
```

1.4 Plot the data

In [113]:
```
import matplotlib.colors as colors
from mpl_toolkits.basemap import Basemap
```

```python
# plot figure
plt.figure(figsize=(30,12))
cmap = plt.cm.get_cmap('viridis_r')

# colors
bounds= np.array([10,100, 200, 500,1000, 2000, 3000])
norm = colors.BoundaryNorm(boundaries=bounds, ncolors= 256)

#map
m = Basemap(projection='cyl', resolution = 'c')
lon, lat =np.meshgrid(lons, lats)
xi,yi = m(lon,lat)
cs = m.pcolormesh(xi,yi, annual_precip, cmap=cmap, norm = norm )
m.drawcoastlines()
cmap.set_under(color='lightyellow')

# latitude and longitude labels on axes
xlabels= lons[::500]
ylabels= lats[::300]
plt.xticks(xlabels, xlabels, fontsize=25)
plt.yticks(ylabels,ylabels, fontsize=25)
plt.xlabel('Lon $^\circ$E',  fontsize=30)
plt.ylabel('Lat $^\circ$N',  fontsize=30)

# colorbar
cbar = plt.colorbar(extend= 'max')
cbar.set_label('annual mean precipitation (mm)', fontsize= 30)
cbar.set_ticks(bounds)
#labels =
#cbar.set_ticklabels()
plt.rcParams.update({'font.size': 30})
plt.title('Global Precipitation Measurement Mission, 2000 - 2019')

plt.savefig(output_dir+ 'gpm.png')
plt.show()
```



1.5 Look at different seasonal cycles of precipitation!

In the figure above, you can see the annual mean derived from monthly global precipitation data for the period 2000 - 2019. Since our data array actually contains the monthly values (each grid cell has a time dimension!), we can look at seasonal cycles at different locations around the globe.

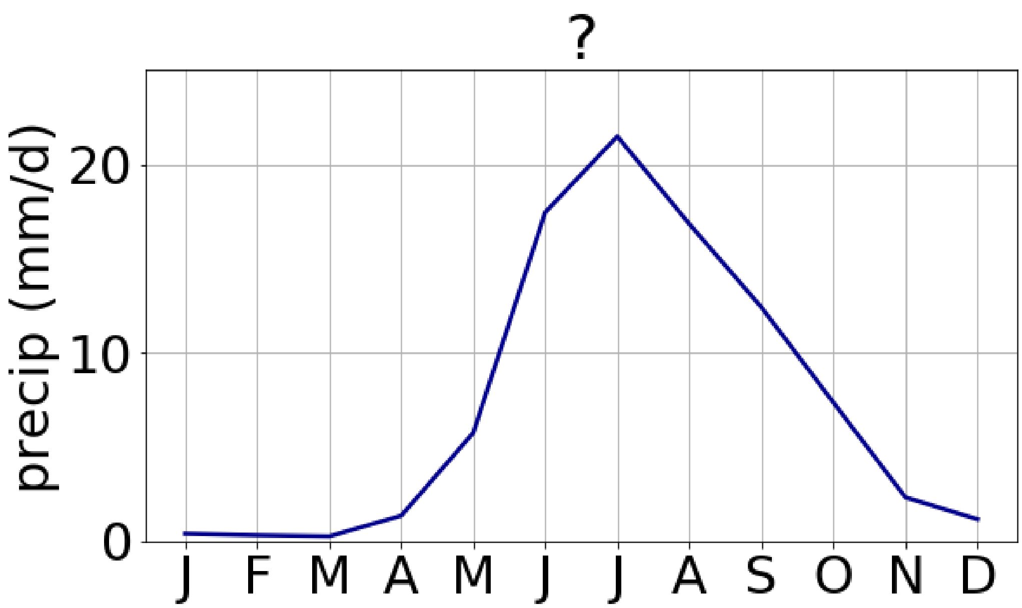
Look at the 5 seasonal cycles of precipitation and assign them to the right region!

Can you guess to which region they belong? Discuss within your group to which region you would attribute the following seasonal curves.

- Indian subcontinent
- Arabian peninsula
- Amazonas rainforest region
- Equatorial Pacific (west of Phillipines)
- Southwest coast of Norway (close to Bergen)

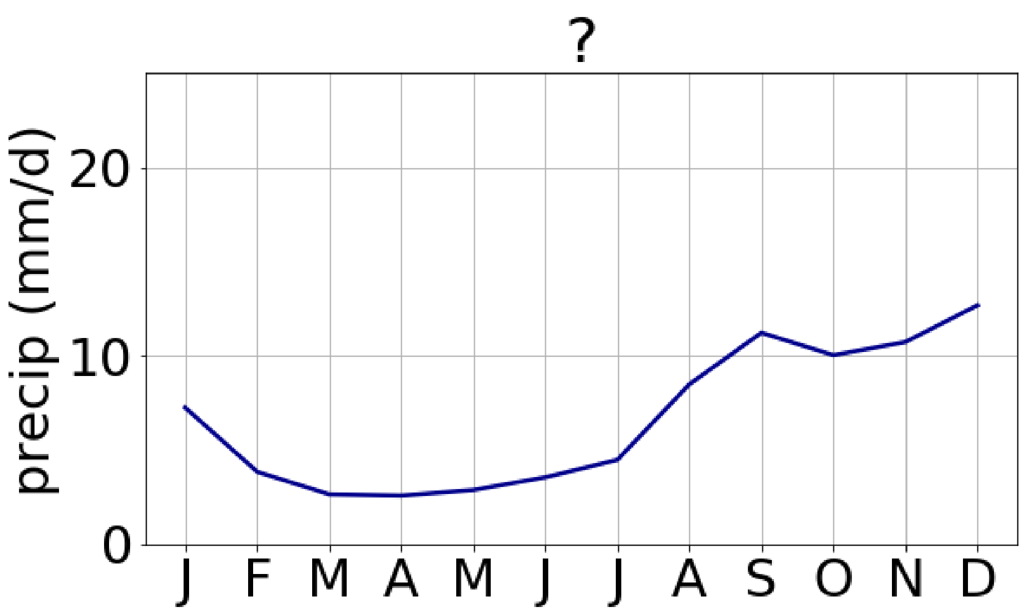
```
In [134]: Image(output_dir + 'plot1.png')
```

Out[134]:



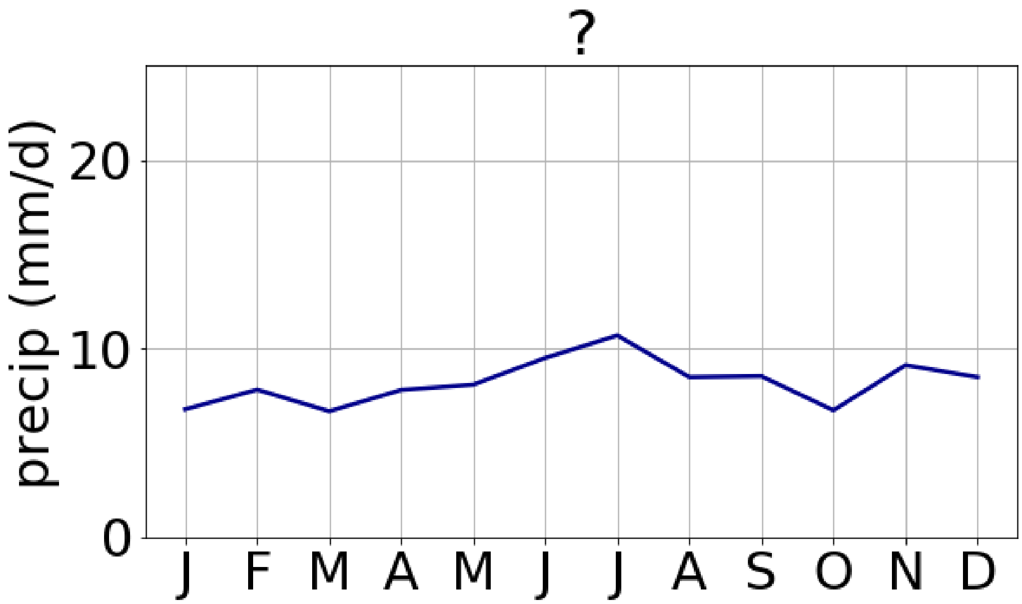

```
In [135]: Image(output_dir + 'plot2.png')
```

Out[135]:



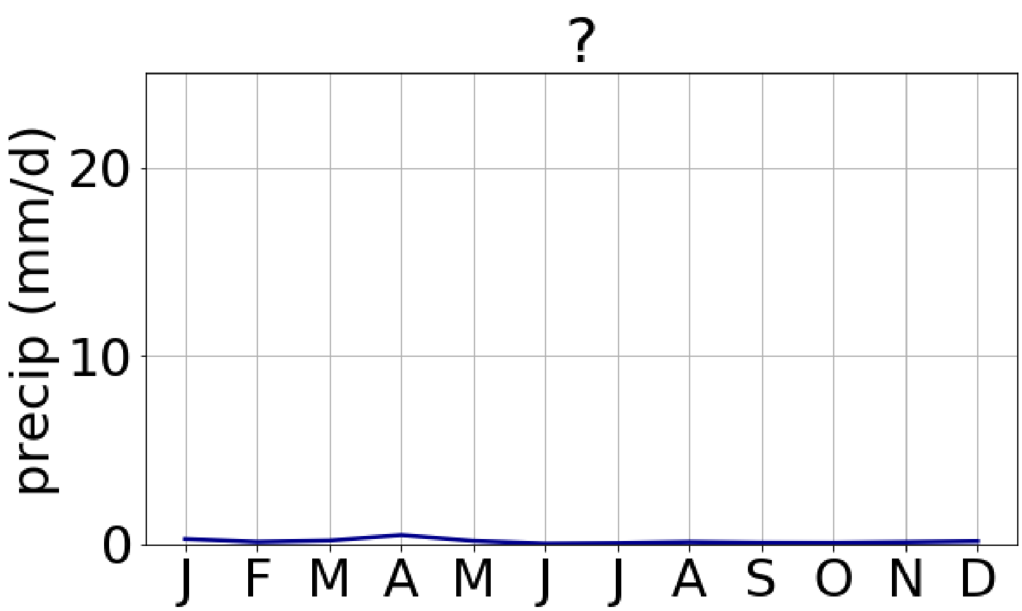

```
In [136]: Image(output_dir + 'plot3.png')
```

Out[136]:



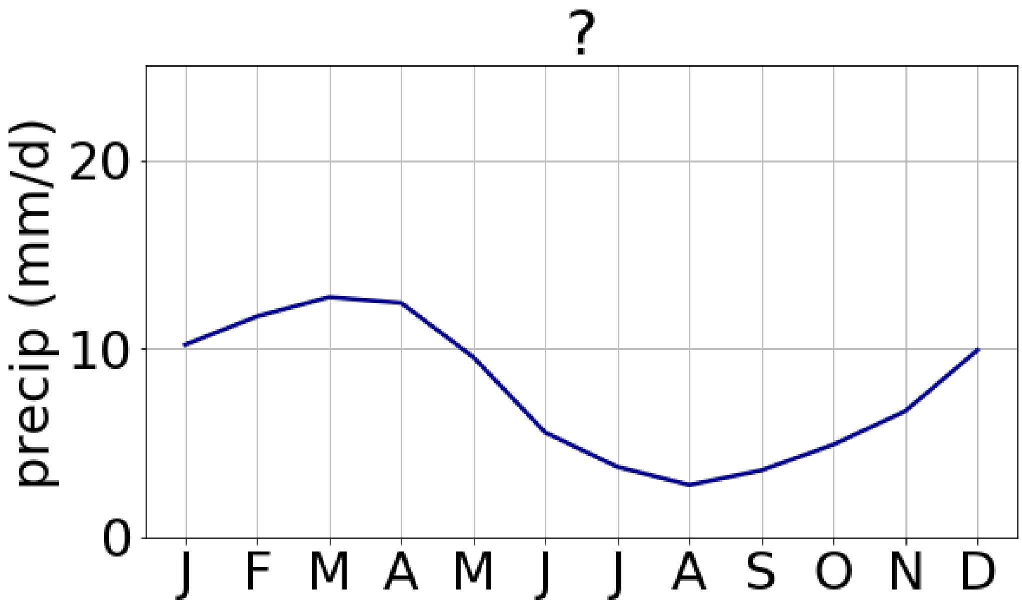

```
In [137]: Image(output_dir + 'plot4.png')
```

Out[137]:




```
In [138]: Image(output_dir + 'plot5.png')
```

Out[138]:



1.6 Your turn : Use the function 'seasonalcycle' to select your own locations!

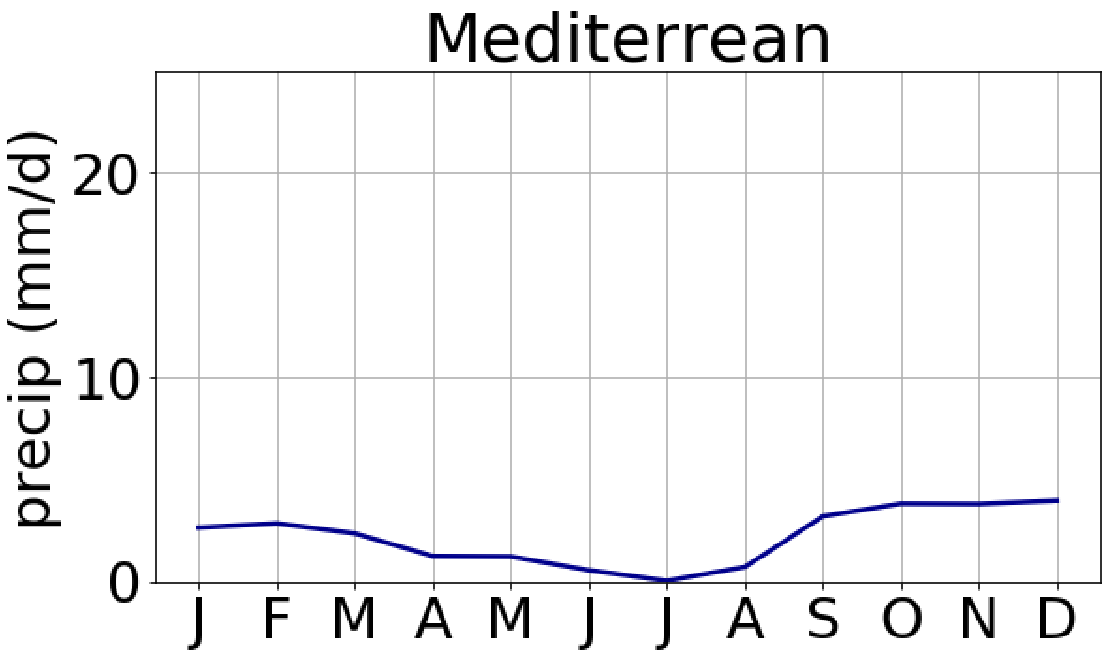
- Choose 3 locations and look at the seasonal cycle
- Discuss in your small group why the seasonal curves look like they look like:

Can you find examples for summer peaks vs. winter peaks? Very strong vs. very flat seasonal variability? Can you find some general differences, e.g. with latitude, ocean vs. land, . . . ?

You can use the function below to extract and plot the seasonal cycle for specific latitudes and longitudes. All you need to do is to call the function according to the example below- just give a lat and lon values as input. The plot automatically will be saved in your working directory.

1.7 Example:

In [155]:
```
# change here the lat and lon values and the name of the region, then run the cell
monthly_precip = seasonalcycle(lat=36, lon =19,location='Mediterrean',  out='mediterra
```



1.8 Look inside: here is the code of the function

(Really no magic)

```
In [ ]: # This function takes one latitude and one longitude as input,
        # finds the closest corresponding value in our data matrix of monthly
        # precipitation and the monthly values for this point

        def seasonalcycle(lat, lon, location= ' ', out= 'plot.png'):
            # find nearest value in lon and lat array
            lon_idx=(np.abs(lons - lon)).argmin()
            lat_idx= (np.abs(lats - lat)).argmin()
            monthly_precip= precip[lat_idx, lon_idx, :]

            ######### plot ##########
            plt.figure(figsize=(9,5))
            plt.plot(monthly_precip*24, color= 'darkblue',linewidth= '2.5')
            xlabels= ['J', 'F', 'M', 'A', 'M', 'J', 'J', 'A', 'S', 'O', 'N', 'D']
            plt.ylabel('precip (mm/d)')
            plt.ylim(0,25)
            plt.xticks(np.arange(12), xlabels)
            plt.grid()
            plt.title(str(location))
            plt.savefig(output_dir + out)
            plt.show()
```

```
    # output dataarray which contains monthly precip values
    return monthly_precip
```

---

2 Freshwater fluxes: E- P

The most important ocean-atmosphere interaction is evaporation from the ocean: most of the moisture in our atmosphere actually comes from the ocean! We can use satellite observations not only to estimate precipitation but also evaporation. One example for a dataset which provides climatological data for heat and freshwater fluxes between the ocean and the atmosphere is the **Hamburg Ocean Atmosphere Parameters and Fluxes From Satellite Data** project (HOAPS).

The data can be downloaded at: https://climatedataguide.ucar.edu/climate-data/hoaps-hamburg-ocean-atmosphere-parameters-and-fluxes-satellite-data

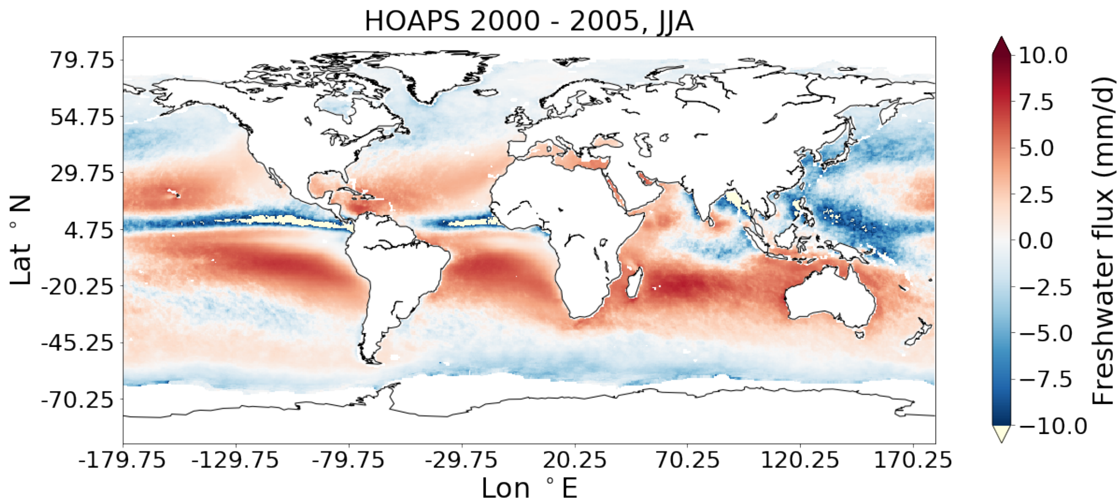
Now, we will have a look at the freshwater budget (evaporation - precipitation) over the ice-free ocean for summer and winter.

In [7]:
```
summerfile= '/home/juli/HOAPS/EMP_jja.nc'
winterfile = '/home/juli/HOAPS/EMP_djf.nc'
summer_emp= Dataset(summerfile)['budg'][0]
winter_emp= Dataset(winterfile)['budg'][0]
lats= Dataset(winterfile)['lat']
lons= Dataset(winterfile)['lon']
```

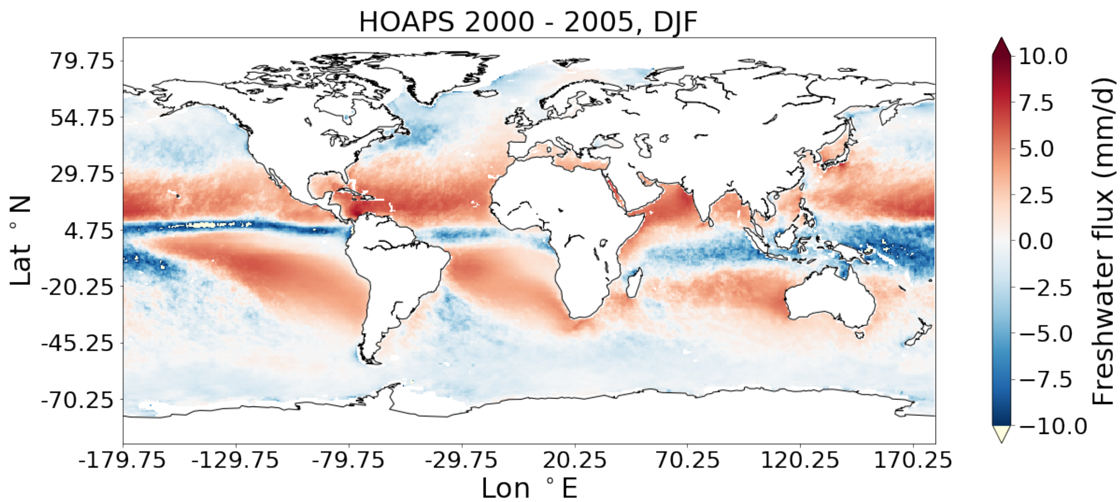
In [8]:
```
# plot figure

def plot_flux(emp, label= ' '):
    plt.figure(figsize=(22,8))
    cmap = plt.cm.get_cmap('RdBu_r')

    #map
    m = Basemap(projection='cyl', resolution = 'c')
    lon, lat =np.meshgrid(lons, lats)
    xi,yi = m(lon,lat)
    cs = m.pcolormesh(xi,yi, emp, cmap=cmap , vmin = -10, vmax= 10 )
    m.drawcoastlines()
    cmap.set_under(color='lightyellow')

    # latitude and longitude labels on axes
    xlabels= lons[::100]
    ylabels= lats[::50]
    plt.xticks(xlabels, xlabels, fontsize=25)
    plt.yticks(ylabels,ylabels, fontsize=25)
    plt.xlabel('Lon $^\circ$E',  fontsize=30)
    plt.ylabel('Lat $^\circ$N',  fontsize=30)

    # colorbar
```

```
    cbar = plt.colorbar(extend= 'both')
    cbar.set_label('Freshwater flux (mm/d)', fontsize= 30)
    plt.rcParams.update({'font.size': 25})
    plt.title(label)
    plt.savefig(output_dir)
    plt.show()
```

```
In [9]: plot_flux(summer_emp, 'HOAPS 2000 - 2005, JJA')
```




```
In [10]: plot_flux(winter_emp, label= 'HOAPS 2000 - 2005, DJF')
```



2.1 Questions

- What do the negative values mean, what do the positive values mean?
- What differences between summer and winter can you detect? What could be the reason for those differences? Which regions are most affected by this seasonality?
- In general, which flux seems to dominate over the ocean? Why?
- How would you expect the plots to look like over land ?
- Which fluxes do we miss to explain the whole water cycle?