02_streamflow_status_product

March 18, 2023

1 HydroSOS Streamflow Status Product Methodology

Jose Valles (jose.valles.leon@gmail.com)

1.1 One month status product

1.1.1 Importing the data and finding missing dates

[ ]:
```
# Importing the libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('classic')
%matplotlib inline

from IPython.display import HTML

sns.set()
```

Import de daily discharge from a hydrological station located in Uruguay

[ ]:
```
station_name = 'pasoroldan'
DISCHARGE_DAILY = pd.read_csv(f'../data/{station_name}_caudales.
 ↪csv',parse_dates=['Fecha'],index_col="Fecha",dayfirst=True,na_values="NA")
```

Identify the missing dates and change dataframe columns name

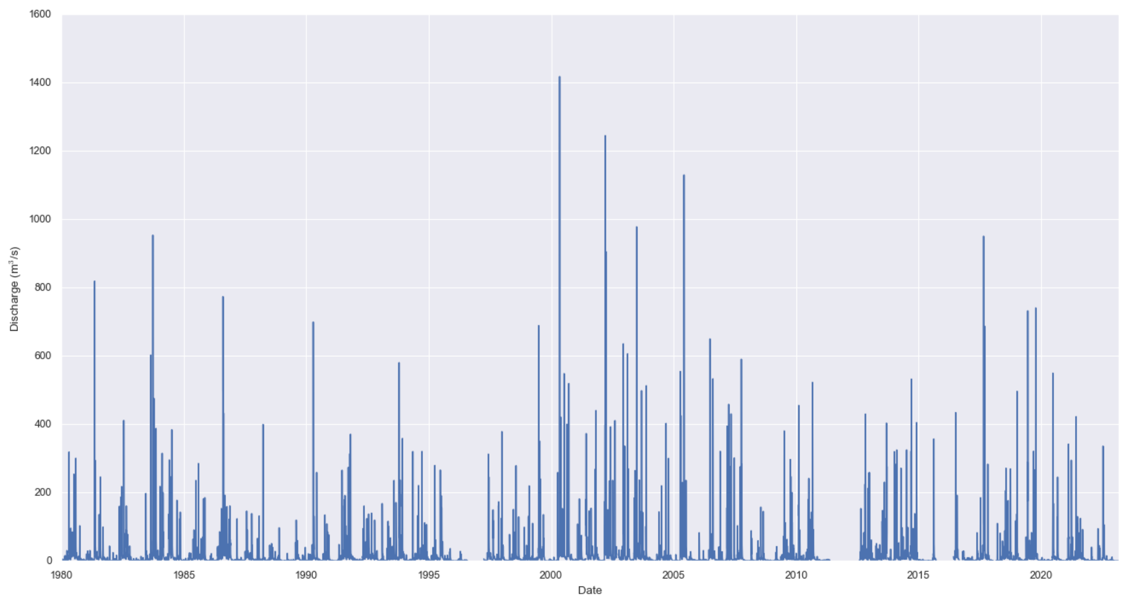
[ ]:
```
# Identify the missing data from a date range (1980 to 2023)
DISCHARGE_DAILY_date_missing = pd.date_range(start = '1980-01-01', end =␣
 ↪'2023-03-03',freq='D')
# Re-index the dataframe based on the missind date variable
DISCHARGE_DAILY = DISCHARGE_DAILY.
 ↪reindex(DISCHARGE_DAILY_date_missing,fill_value=None)
# Set index Fecha
DISCHARGE_DAILY.index.name = 'date'
# Change columns names
DISCHARGE_DAILY.columns = ['station','discharge']
# Remove station code column
```

```
DISCHARGE_DAILY = DISCHARGE_DAILY.drop(columns='station')
# Print the last 6 values
HTML(DISCHARGE_DAILY.tail(6).to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

We can visualize in a plot the daily discharge from the imported station

```
[ ]: ax1 = DISCHARGE_DAILY['discharge'].plot(figsize=(20, 10));
     ax1.set_xlabel('Date');
     ax1.set_ylabel('Discharge (m$^3$/s)');
```



1.1.2 Calculate monthly mean from daily data

First, we define a percentage of missing value. For this exercise, we use a 50% of missing data in the given month

```
[ ]: # Percentage of missing data
     max_pct_missing = 50
```

```
[ ]: # group the Dataframe in a monthly time scale
     GROUPER_DISCHARGE_MONTHLY = DISCHARGE_DAILY.groupby(pd.Grouper(freq='1MS'))

     # this function allows to calculate the percentange of missing values and label␣
      ↪the dataframe to "missing"
     NUMBER_MISSING = GROUPER_DISCHARGE_MONTHLY.apply(lambda x: pd.isnull(x).
      ↪sum()*100/len(x)).unstack(1)
     NUMBER_MISSING = NUMBER_MISSING.to_frame()
```

```
# change the column name to "missing"
NUMBER_MISSING.columns = ['number_missing']

# this function allows to calculate the percentange of missing values and label␣
↪the dataframe to "missing"
BOOL_MISSING = GROUPER_DISCHARGE_MONTHLY.apply(lambda x: pd.isnull(x).sum()*100/
↪len(x)).unstack(1) < max_pct_missing
BOOL_MISSING = BOOL_MISSING.to_frame() # Convert to DataFrame
BOOL_MISSING.columns = ['missing'] # change the column name to "missing"

# BOOL_MISSING[~BOOL_MISSING['missing']] # print the dates that does not␣
↪fulfill the criterion of null data in for each month
# BOOL_MISSING.to_clipboard()

# NUMBER_MISSING.to_clipboard() # Uncomment if you want to visualize all the␣
↪result in a CSV
```

Next, we identify the months which contains lower that the max_pct_missing value and the monthly flow is only calculated if 50% o more of recorded value in a given month

```
[ ]: # from daily to monthly
DISCHARGE_MONTHLY = DISCHARGE_DAILY.resample('M').apply(lambda x: x.mean() if x.
↪isnull().sum()*100/len(x) < max_pct_missing else np.nan)
# from monthly to 3 months
DISCHARGE_THREE_MONTHS = DISCHARGE_MONTHLY.rolling(3).apply(lambda x: x.mean()␣
↪if x.isnull().sum()*100/len(x) < max_pct_missing else np.nan)
# Create columns
DISCHARGE_MONTHLY['year'] = DISCHARGE_MONTHLY.index.year
DISCHARGE_MONTHLY['month'] = DISCHARGE_MONTHLY.index.month
DISCHARGE_MONTHLY['water_year'] = DISCHARGE_MONTHLY.index.year.
↪where(DISCHARGE_MONTHLY.index.month < 4, DISCHARGE_MONTHLY.index.year + 1)
## create column for day, month, year in the daily discharge ()
DISCHARGE_DAILY['year'] = DISCHARGE_DAILY.index.year
DISCHARGE_DAILY['month'] = DISCHARGE_DAILY.index.month
DISCHARGE_DAILY['day'] = DISCHARGE_DAILY.index.day
DISCHARGE_DAILY['monthday'] = DISCHARGE_DAILY.index.day_of_year
# The water year for this region starts in April (4)
DISCHARGE_DAILY['water_year'] = DISCHARGE_DAILY.index.year.
↪where(DISCHARGE_DAILY.index.month < 4, DISCHARGE_DAILY.index.year + 1)
## print the first results
HTML(DISCHARGE_MONTHLY.head(6).to_html(index=False))
# DISCHARGE_MONTHLY.to_clipboard() # Uncomment if you want to visualize all the␣
↪time serie
```

```
[ ]: <IPython.core.display.HTML object>
```

1.1.3 Select the period of record to estimate the percentage of average.

For this section, we will use the period 1991-2020 which is the same as climatology

```
[ ]: DISCHARGE_SELECTION = DISCHARGE_MONTHLY[(DISCHARGE_MONTHLY['year'] >= 1991) &␣
       ↪(DISCHARGE_MONTHLY['year'] < 2021)]
```

Calculate the average dischare for each month in the period of record (e.g. 1991-2020)

```
[ ]: DISCHARGE_AVERAGE = DISCHARGE_SELECTION.groupby(DISCHARGE_SELECTION.month).
       ↪mean()
     DISCHARGE_AVERAGE = DISCHARGE_AVERAGE.reindex(columns=['discharge'])
```

print the monthly average discharge

```
[ ]: HTML(DISCHARGE_AVERAGE.to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

After this, we calculate the following variables 1. we calculate the monthly mean discharge as a percentage of average 2. we rank the percentage of average of the current month 3. we count the notnull values of the current month

```
[ ]: # create empty columns in the dataframe
     DISCHARGE_MONTHLY['average_percentage'] = np.nan
     DISCHARGE_MONTHLY['rank_average'] = np.nan
     DISCHARGE_MONTHLY['non_missing'] = np.nan

     for i in range(len(DISCHARGE_MONTHLY)):
         # Extract the current month
         m = DISCHARGE_MONTHLY.month[i]
         # Extract the current year
         y = DISCHARGE_MONTHLY.year[i]
         DISCHARGE_MONTHLY.loc[DISCHARGE_MONTHLY.eval('month==@m &␣
       ↪year==@y'),'rank_average']  = DISCHARGE_MONTHLY.
       ↪query('month==@m')['discharge'].rank()
         DISCHARGE_MONTHLY.loc[DISCHARGE_MONTHLY.eval('month==@m &␣
       ↪year==@y'),'non_missing']  = DISCHARGE_MONTHLY.
       ↪query('month==@m')["discharge"].notnull().sum()
         DISCHARGE_MONTHLY.loc[DISCHARGE_MONTHLY.eval('month==@m &␣
       ↪year==@y'),'average_percentage'] = (DISCHARGE_MONTHLY['discharge'][i] -␣
       ↪DISCHARGE_AVERAGE.query('month == @m')["discharge"].item()) /␣
       ↪DISCHARGE_AVERAGE.query('month == @m')["discharge"].item()
```

1.1.4 Calculate the percentile using Weibull formula

Calculate the percentile using this formula

$$percentile = \frac{i}{N+1}$$

where i is the rank of the current month and N is the number of months in the period of record

```
[ ]: DISCHARGE_MONTHLY['percentile'] = DISCHARGE_MONTHLY['rank_average']/
       ↪(DISCHARGE_MONTHLY['non_missing']+1)
```

print the results

```
[ ]: HTML(DISCHARGE_MONTHLY.tail(5).to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

1.1.5 Assign the percentile to a category

Category	Percentile Range
High flow	0.870000 - 1.000000
Above normal	0.720000 - 0.869999
Normal range	0.280000 - 0.719999
Below normal	0.130000 - 0.279999
Low flow	0.000000 - 0.129999

```
[ ]: criteria = [DISCHARGE_MONTHLY['percentile'].between(0.87,1.00),
                 DISCHARGE_MONTHLY['percentile'].between(0.72,0.87),
                 DISCHARGE_MONTHLY['percentile'].between(0.28,0.72),
                 DISCHARGE_MONTHLY['percentile'].between(0.13,0.28),
                 DISCHARGE_MONTHLY['percentile'].between(0.00,0.13)]

     values = ['High flow','Above normal','Normal range','Below normal','Low flow']

     DISCHARGE_MONTHLY['percentile_range'] = np.select(criteria,values,None)
```

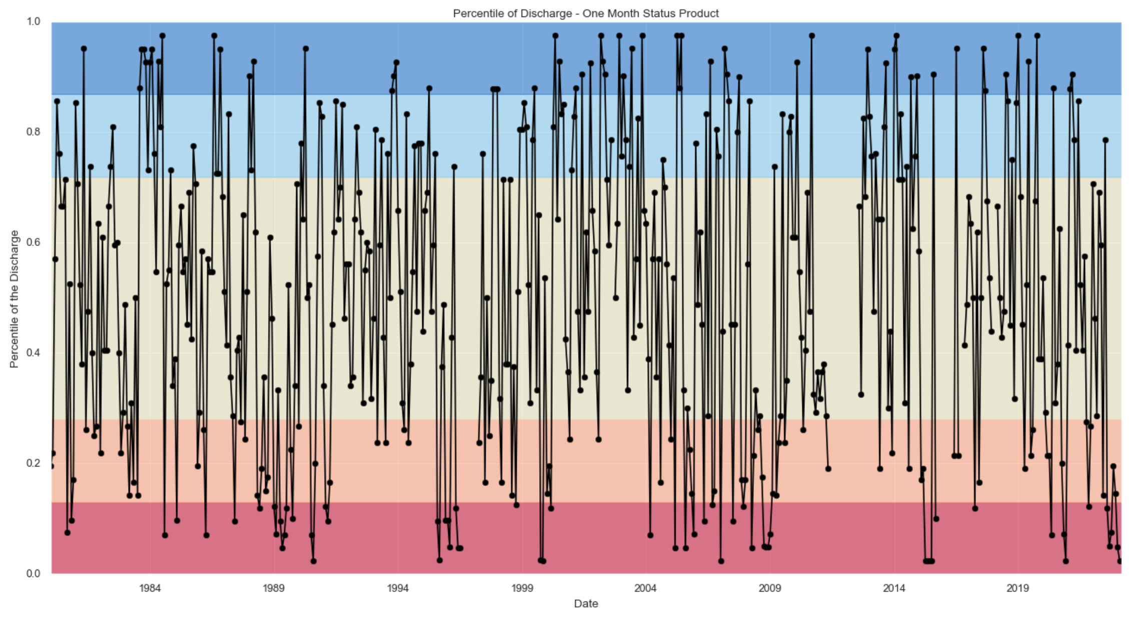
Print the results

```
[ ]: HTML(DISCHARGE_MONTHLY.tail(5).to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

Make a plot of the Percentile and visualize the percentile category

```
[ ]: ax = DISCHARGE_MONTHLY['percentile'].plot(figsize=(20,␣
       ↪10),color='black',linestyle='-', marker='o')
     ax.axhspan(0.0, 0.13, color='#CD233F',alpha=0.6)
     ax.axhspan(0.13, 0.28, color='#FFA885',alpha=0.6)
     ax.axhspan(0.28, 0.72, color='#E7E2BC',alpha=0.6)
     ax.axhspan(0.72, 0.87, color='#8ECEEE',alpha=0.6)
     ax.axhspan(0.87, 1.00, color='#2C7DCD',alpha=0.6)
     ax.set_xlabel('Date')
     ax.set_ylabel('Percentile of the Discharge')
     ax.set_title('Percentile of Discharge - One Month Status Product');
```



1.2 Three months status product

Based on the `DISCHARGE_THREE_MONTHS` variable previously calculated, we define two columns with the start month `startMonth` and end month `endMonth`

```
[ ]: DISCHARGE_THREE_MONTHS['startMonth'] = (DISCHARGE_THREE_MONTHS.index - pd.
       ↪DateOffset(months=2)).month
     DISCHARGE_THREE_MONTHS['endMonth'] = DISCHARGE_THREE_MONTHS.index.month
     DISCHARGE_THREE_MONTHS['year'] = DISCHARGE_THREE_MONTHS.index.year
```

print results

```
[ ]: HTML(DISCHARGE_THREE_MONTHS.head(6).to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

1.2.1 Select the period of record to estimate the percentage of average.

For this section, we use the same period (1991-2020)

```
[ ]: DISCHARGE_SELECTION_THREE_MONTH =␣
       ↪DISCHARGE_THREE_MONTHS[(DISCHARGE_THREE_MONTHS['year'] >= 1991) &␣
       ↪(DISCHARGE_THREE_MONTHS['year'] < 2021)]
```

Calculate the average discharge for each month in the period of record (e.g. 1991-2020)

```
[ ]: DISCHARGE_AVERAGE_THREE_MONTH = DISCHARGE_SELECTION_THREE_MONTH.
       ↪groupby(DISCHARGE_SELECTION_THREE_MONTH.startMonth).mean()
```

```
DISCHARGE_AVERAGE_THREE_MONTH = DISCHARGE_AVERAGE_THREE_MONTH.
 ↪reindex(columns=['discharge'])
```

print result

```
[ ]: HTML(DISCHARGE_AVERAGE_THREE_MONTH.to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

After this, we calculate the following variables 1. we calculate the three months mean discharge as a percentage of average 2. we rank the percentage of average of the current three month period 3. we count the notnull values of the current three month period

```
[ ]: DISCHARGE_THREE_MONTHS['average_percentage'] = np.nan
     DISCHARGE_THREE_MONTHS['rank_average'] = np.nan
     DISCHARGE_THREE_MONTHS['non_missing'] = np.nan

     for i in range(len(DISCHARGE_THREE_MONTHS)):
         # Extract the current month
         m = DISCHARGE_THREE_MONTHS.startMonth[i]
         # Extract the current year
         y = DISCHARGE_THREE_MONTHS.year[i]
         DISCHARGE_THREE_MONTHS.loc[DISCHARGE_THREE_MONTHS.eval('startMonth==@m &␣
      ↪year==@y'),'rank_average']  = DISCHARGE_THREE_MONTHS.
      ↪query('startMonth==@m')['discharge'].rank()
         DISCHARGE_THREE_MONTHS.loc[DISCHARGE_THREE_MONTHS.eval('startMonth==@m &␣
      ↪year==@y'),'non_missing']  = DISCHARGE_THREE_MONTHS.
      ↪query('startMonth==@m')["discharge"].notnull().sum()
         DISCHARGE_THREE_MONTHS.loc[DISCHARGE_THREE_MONTHS.eval('startMonth==@m &␣
      ↪year==@y'),'average_percentage'] = (DISCHARGE_THREE_MONTHS['discharge'][i] -␣
      ↪DISCHARGE_AVERAGE_THREE_MONTH.query('startMonth == @m')["discharge"].item())␣
      ↪/ DISCHARGE_AVERAGE_THREE_MONTH.query('startMonth == @m')["discharge"].item()
```

print results

```
[ ]: HTML(DISCHARGE_THREE_MONTHS.head(6).to_html())
```

```
[ ]: <IPython.core.display.HTML object>
```

1.2.2 Calculate the percentile using Weibull formula

Calculate the percentile using this formula

$$percentile = \frac{i}{N+1}$$

where i is the rank of the current month and N is the number of months in the period of record

```
[ ]: DISCHARGE_THREE_MONTHS['percentile'] = DISCHARGE_THREE_MONTHS['rank_average']/
     ↪(DISCHARGE_THREE_MONTHS['non_missing']+1)
```

1.2.3 Assign the percentile to a category

Category	Percentile Range
High flow	0.870000 - 1.000000
Above normal	0.720000 - 0.869999
Normal range	0.280000 - 0.719999
Below normal	0.130000 - 0.279999
Low flow	0.000000 - 0.129999

```
[ ]: criteria_three_months = [DISCHARGE_THREE_MONTHS['percentile'].between(0.87,1.
     ↪00),
                 DISCHARGE_THREE_MONTHS['percentile'].between(0.72,0.87),
                 DISCHARGE_THREE_MONTHS['percentile'].between(0.28,0.72),
                 DISCHARGE_THREE_MONTHS['percentile'].between(0.13,0.28),
                 DISCHARGE_THREE_MONTHS['percentile'].between(0.00,0.13)]

     values_three_months = ['High flow','Above normal','Normal range','Below␣
     ↪normal','Low flow']

     DISCHARGE_THREE_MONTHS['percentile_range'] = np.
     ↪select(criteria_three_months,values_three_months,None)
```

In this section we rename the `startMonth` column into a new column called `period` which describe the three month period of calculation.

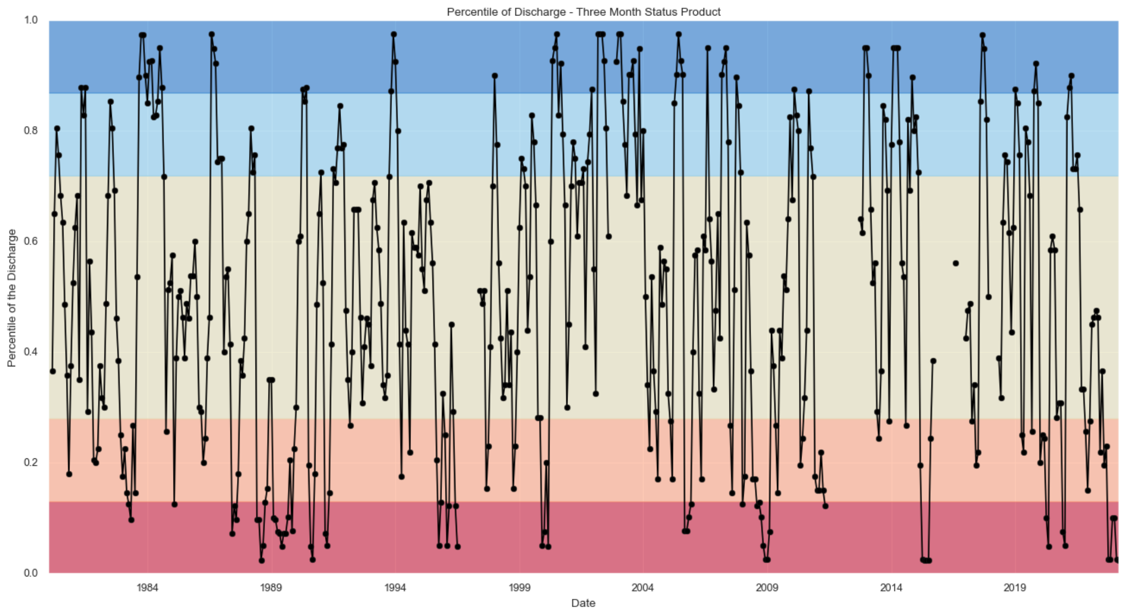
```
[ ]: row_labels = {1:'JFM',
                  2:'FMA',
                  3:'MAM',
                  4:'AMJ',
                  5:'MJJ',
                  6:'JJA',
                  7:'JAS',
                  8:'ASO',
                  9:'SON',
                  10:'OND',
                  11:'NDE',
                  12:'DEF'}
     DISCHARGE_THREE_MONTHS['period'] = DISCHARGE_THREE_MONTHS['startMonth'].
     ↪replace(row_labels)
```

print results

```
[ ]: HTML(DISCHARGE_THREE_MONTHS.tail(6).to_html())
[ ]: <IPython.core.display.HTML object>
```

Make a plot of the Percentile and visualize the percentile category

```
[ ]: ax2 = DISCHARGE_THREE_MONTHS['percentile'].plot(figsize=(20,␣
       ↪10),color='black',linestyle='-', marker='o')
     ax2.axhspan(0.0, 0.13, color='#CD233F',alpha=0.6)
     ax2.axhspan(0.13, 0.28, color='#FFA885',alpha=0.6)
     ax2.axhspan(0.28, 0.72, color='#E7E2BC',alpha=0.6)
     ax2.axhspan(0.72, 0.87, color='#8ECEEE',alpha=0.6)
     ax2.axhspan(0.87, 1.00, color='#2C7DCD',alpha=0.6)
     ax2.set_xlabel('Date')
     ax2.set_ylabel('Percentile of the Discharge')
     ax2.set_title('Percentile of Discharge - Three Month Status Product');
```



1.3 Export to CSV files

```
[ ]: DISCHARGE_MONTHLY[['discharge','month','year','average_percentage','percentile','percentile_ra
       ↪to_csv(f'../output/{station_name}_one-month.csv',float_format='%.3f')
```

```
[ ]: DISCHARGE_THREE_MONTHS[['discharge','startMonth','endMonth','year','average_percentage','perce
       ↪to_csv(f'../output/{station_name}_three-month.csv',float_format='%.3f')
```