Balanco

May 14, 2019

1 Programa de Pós-graduação em Recursos Hídricos e Sanemento - PPGRHS

1.1 Hidrologia

Balanço Hídrico

Clebson Farias

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
time = ["01/08/13 10:00 PM", "01/08/13 11:00 PM", "01/09/13 12:00 AM",
        "01/09/13 01:00 AM", "01/09/13 02:00 AM", "01/09/13 03:00 AM",
        "01/09/13 04:00 AM", "01/09/13 05:00 AM", "01/09/13 06:00 AM",
        "01/09/13 07:00 AM", "01/09/13 08:00 AM", "01/09/13 09:00 AM",
        "01/09/13 10:00 AM", "01/09/13 11:00 AM", "01/09/13 12:00 PM",
        "01/09/13 01:00 PM", "01/09/13 02:00 PM", "01/09/13 03:00 PM",
        "01/09/13 04:00 PM", "01/09/13 05:00 PM", "01/09/13 06:00 PM",
        "01/09/13 07:00 PM", "01/09/13 08:00 PM", "01/09/13 09:00 PM",
        "01/09/13 10:00 PM", "01/09/13 11:00 PM", "01/10/13 12:00 AM",
        "01/10/13 01:00 AM", "01/10/13 02:00 AM", "01/10/13 03:00 AM",
        "01/10/13 04:00 AM", "01/10/13 05:00 AM", "01/10/13 06:00 AM",
        "01/10/13 07:00 AM", "01/10/13 08:00 AM", "01/10/13 09:00 AM",
        "01/10/13 10:00 AM", "01/10/13 11:00 AM", "01/10/13 12:00 PM"]
len(time)
```

[2]:
```
39
```

[3]:
```
precipitacao = [0.3, 0.0, 1.8, 7.9, 8.1, 7.1, 5.6, 2.8, 1.8, 4.1, 2.5, 0.8, 0.
 ↪8, 0.3, 0.5, 0.3, 0.0,
                0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
 ↪0, 0.0, 0.0, 0.0, 0.0,
                0.0, 0.0, 0.0, 0.0, 0.0]
len(precipitacao)
```

[3]:
```
39
```

[4]:
```
vazao = [0.048, 0.095, 0.109, 0.616, 0.700, 12.292, 10.528, 8.960, 5.880, 5.068,
        4.396, 5.572, 3.808, 2.940, 2.772, 2.128, 1.456, 1.260, 1.008, 0.868,␣
↪0.756,
        0.644, 0.560, 0.504, 0.448, 0.392, 0.336, 0.336, 0.308, 0.269, 0.241,␣
↪0.216,
        0.193, 0.174, 0.160, 0.148, 0.129, 0.129, 0.115]
len(vazao)
```

[4]:

```
39
```

1.1.1 Dados de precipitação e vazão de uma estação telemétrica para um evento de chuva são mostrado na tabela abaixo.

[8]:

```
area = 31.82
dados = {"P": precipitacao, "Q": vazao, "Qmm": [i*(3.6/area) for i in vazao]}
dados = pd.DataFrame(dados, index=time)
dados.head()
```

[8]:

```
                      P      Q       Qmm
01/08/13 10:00 PM  0.3  0.048  0.005431
01/08/13 11:00 PM  0.0  0.095  0.010748
01/09/13 12:00 AM  1.8  0.109  0.012332
01/09/13 01:00 AM  7.9  0.616  0.069692
01/09/13 02:00 AM  8.1  0.700  0.079195
```

Mostre como a distribuição de armazenamento se dará assumindo que a bacia passou por um longo período de estiagem antes deste evento(ou seja, armazenamento inicial é zero).

$$\frac{\Delta S}{t} = P - Q \tag{1}$$

$$S = S_o + P - Q \tag{2}$$

Variação de armazenamento

[9]:

```
def armazenamento(dados, so, area):
    fator = 3.6
    s = pd.DataFrame(so + dados.P - ((dados.Q/area) * 3.6), columns=["S"])

    return dados.combine_first(s)

dados = armazenamento(dados, 0, area)
dados
```

[9]:

```
                      P      Q       Qmm         S
01/08/13 10:00 PM  0.3    0.048  0.005431  0.294569
01/08/13 11:00 PM  0.0    0.095  0.010748 -0.010748
01/09/13 12:00 AM  1.8    0.109  0.012332  1.787668
01/09/13 01:00 AM  7.9    0.616  0.069692  7.830308
01/09/13 02:00 AM  8.1    0.700  0.079195  8.020805
```

```
01/09/13 03:00 AM  7.1  12.292  1.390673  5.709327
01/09/13 04:00 AM  5.6  10.528  1.191100  4.408900
01/09/13 05:00 AM  2.8   8.960  1.013702  1.786298
01/09/13 06:00 AM  1.8   5.880  0.665242  1.134758
01/09/13 07:00 AM  4.1   5.068  0.573375  3.526625
01/09/13 08:00 AM  2.5   4.396  0.497348  2.002652
01/09/13 09:00 AM  0.8   5.572  0.630396  0.169604
01/09/13 10:00 AM  0.8   3.808  0.430823  0.369177
01/09/13 11:00 AM  0.3   2.940  0.332621 -0.032621
01/09/13 12:00 PM  0.5   2.772  0.313614  0.186386
01/09/13 01:00 PM  0.3   2.128  0.240754  0.059246
01/09/13 02:00 PM  0.0   1.456  0.164727 -0.164727
01/09/13 03:00 PM  0.0   1.260  0.142552 -0.142552
01/09/13 04:00 PM  0.0   1.008  0.114041 -0.114041
01/09/13 05:00 PM  0.0   0.868  0.098202 -0.098202
01/09/13 06:00 PM  0.0   0.756  0.085531 -0.085531
01/09/13 07:00 PM  0.0   0.644  0.072860 -0.072860
01/09/13 08:00 PM  0.0   0.560  0.063356 -0.063356
01/09/13 09:00 PM  0.0   0.504  0.057021 -0.057021
01/09/13 10:00 PM  0.0   0.448  0.050685 -0.050685
01/09/13 11:00 PM  0.0   0.392  0.044349 -0.044349
01/10/13 12:00 AM  0.0   0.336  0.038014 -0.038014
01/10/13 01:00 AM  0.0   0.336  0.038014 -0.038014
01/10/13 02:00 AM  0.0   0.308  0.034846 -0.034846
01/10/13 03:00 AM  0.0   0.269  0.030434 -0.030434
01/10/13 04:00 AM  0.0   0.241  0.027266 -0.027266
01/10/13 05:00 AM  0.0   0.216  0.024437 -0.024437
01/10/13 06:00 AM  0.0   0.193  0.021835 -0.021835
01/10/13 07:00 AM  0.0   0.174  0.019686 -0.019686
01/10/13 08:00 AM  0.0   0.160  0.018102 -0.018102
01/10/13 09:00 AM  0.0   0.148  0.016744 -0.016744
01/10/13 10:00 AM  0.0   0.129  0.014595 -0.014595
01/10/13 11:00 AM  0.0   0.129  0.014595 -0.014595
01/10/13 12:00 PM  0.0   0.115  0.013011 -0.013011
```

Armazenamento Acumulado

[10]:

```
def armazenamento_acumulado(dados):
    s_a = pd.Series(dados.S.cumsum(), name="S_A")

    return dados.combine_first(pd.DataFrame(s_a))

dados = armazenamento_acumulado(dados)
dados
```

[10]:

```
                     P       Q       Qmm         S       S_A
01/08/13 10:00 PM  0.3   0.048  0.005431  0.294569  0.294569
01/08/13 11:00 PM  0.0   0.095  0.010748 -0.010748  0.283821
```

```
01/09/13 12:00 AM  1.8    0.109  0.012332  1.787668   2.071490
01/09/13 01:00 AM  7.9    0.616  0.069692  7.830308   9.901798
01/09/13 02:00 AM  8.1    0.700  0.079195  8.020805  17.922602
01/09/13 03:00 AM  7.1   12.292  1.390673  5.709327  23.631930
01/09/13 04:00 AM  5.6   10.528  1.191100  4.408900  28.040830
01/09/13 05:00 AM  2.8    8.960  1.013702  1.786298  29.827128
01/09/13 06:00 AM  1.8    5.880  0.665242  1.134758  30.961886
01/09/13 07:00 AM  4.1    5.068  0.573375  3.526625  34.488510
01/09/13 08:00 AM  2.5    4.396  0.497348  2.002652  36.491163
01/09/13 09:00 AM  0.8    5.572  0.630396  0.169604  36.660767
01/09/13 10:00 AM  0.8    3.808  0.430823  0.369177  37.029943
01/09/13 11:00 AM  0.3    2.940  0.332621 -0.032621  36.997322
01/09/13 12:00 PM  0.5    2.772  0.313614  0.186386  37.183708
01/09/13 01:00 PM  0.3    2.128  0.240754  0.059246  37.242954
01/09/13 02:00 PM  0.0    1.456  0.164727 -0.164727  37.078228
01/09/13 03:00 PM  0.0    1.260  0.142552 -0.142552  36.935676
01/09/13 04:00 PM  0.0    1.008  0.114041 -0.114041  36.821634
01/09/13 05:00 PM  0.0    0.868  0.098202 -0.098202  36.723432
01/09/13 06:00 PM  0.0    0.756  0.085531 -0.085531  36.637901
01/09/13 07:00 PM  0.0    0.644  0.072860 -0.072860  36.565041
01/09/13 08:00 PM  0.0    0.560  0.063356 -0.063356  36.501684
01/09/13 09:00 PM  0.0    0.504  0.057021 -0.057021  36.444664
01/09/13 10:00 PM  0.0    0.448  0.050685 -0.050685  36.393979
01/09/13 11:00 PM  0.0    0.392  0.044349 -0.044349  36.349629
01/10/13 12:00 AM  0.0    0.336  0.038014 -0.038014  36.311615
01/10/13 01:00 AM  0.0    0.336  0.038014 -0.038014  36.273602
01/10/13 02:00 AM  0.0    0.308  0.034846 -0.034846  36.238755
01/10/13 03:00 AM  0.0    0.269  0.030434 -0.030434  36.208322
01/10/13 04:00 AM  0.0    0.241  0.027266 -0.027266  36.181056
01/10/13 05:00 AM  0.0    0.216  0.024437 -0.024437  36.156618
01/10/13 06:00 AM  0.0    0.193  0.021835 -0.021835  36.134783
01/10/13 07:00 AM  0.0    0.174  0.019686 -0.019686  36.115097
01/10/13 08:00 AM  0.0    0.160  0.018102 -0.018102  36.096996
01/10/13 09:00 AM  0.0    0.148  0.016744 -0.016744  36.080251
01/10/13 10:00 AM  0.0    0.129  0.014595 -0.014595  36.065657
01/10/13 11:00 AM  0.0    0.129  0.014595 -0.014595  36.051062
01/10/13 12:00 PM  0.0    0.115  0.013011 -0.013011  36.038052
```

Total Precipitado e escoado!

[33]:

```
total_P = dados.P.sum()
total_Q = dados.Qmm.sum()
print("Total Precipitado:", total_P)
print("Total Escoado:", round(total_Q,4))
```

```
Total Precipitado: 44.7
Total Escoado: 8.6619
```

Total de água que ficou retido na bacia até o final do evento?

[32]:
```
print("Total Armazenado:", round((total_P - total_Q), 4))
```

```
Total Armazenado: 36.0381
```

Qual o percentual de precipitação que foi escoado?

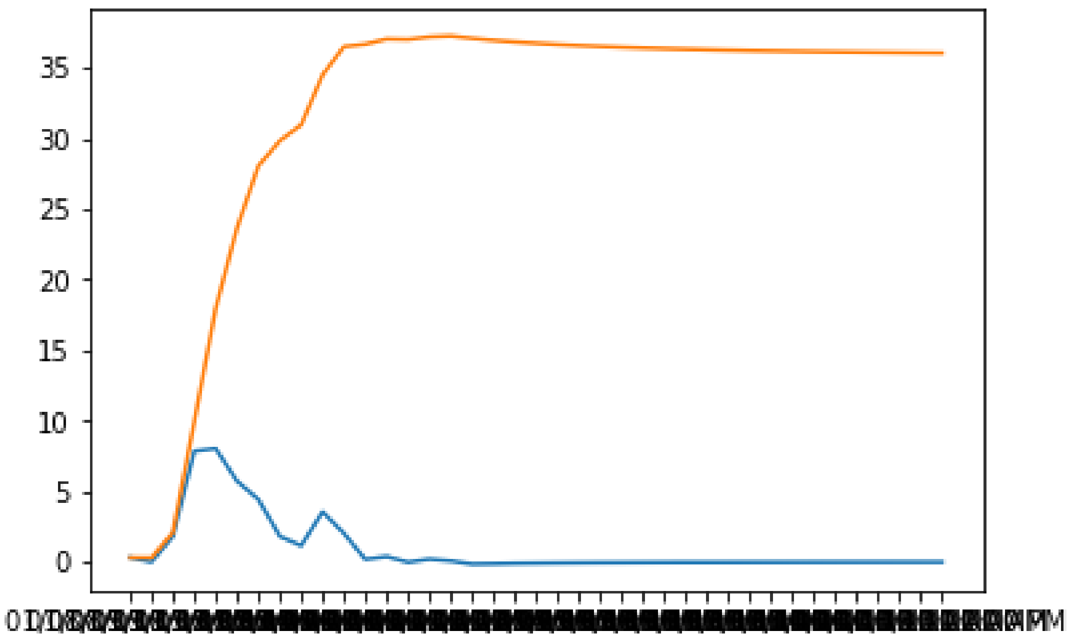
[34]:
```
print("Precipitação escoada(%): ", round((total_Q/total_P), 2))
```

```
Precipitação escoada(%):  0.19
```

1.1.2 Gráfico variação de armazenamento e armazenamento acumulado:

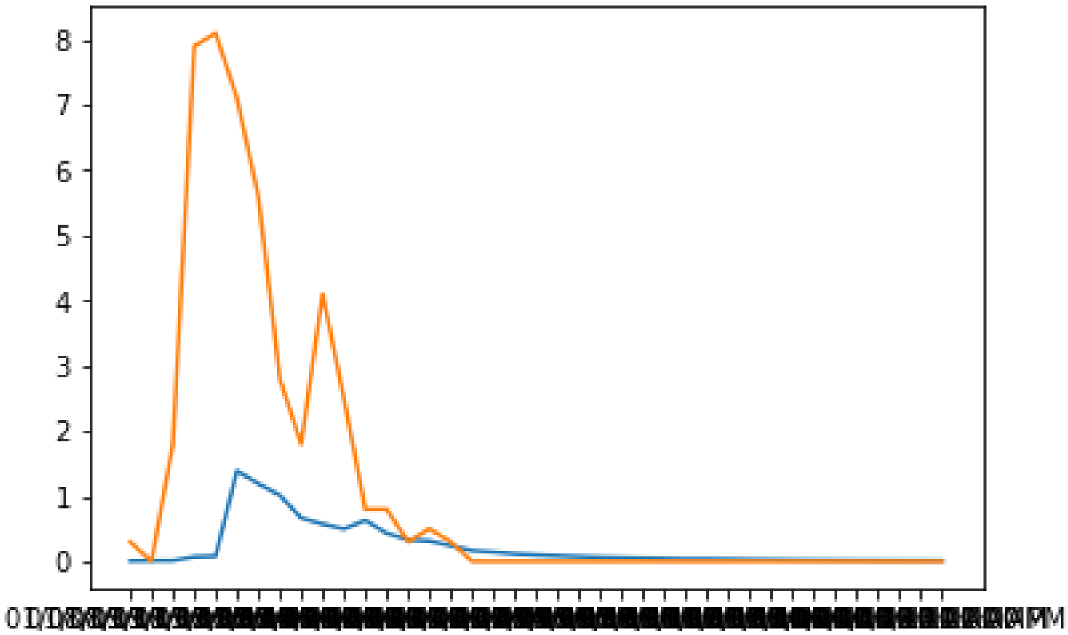
[12]:
```
t = dados.index
data1 = dados.S
data2 = dados.S_A

plt.plot(t, data1, data2)
plt.show()
```



[13]:
```
t = dados.index
data_Q = dados.Qmm
data_P = dados.P

plt.plot(t, data_Q, data_P)
plt.show()
```


8
7
6
5
4
3
2
1
0

[ ]: