Problem: *Water Balance for the Flood Control Reservoir*

Statement:

A flood control reservoir has been releasing flows in anticipation of a forecast flood event. The reservoir storage is 490,000 m^3 at 10 am. The latest forecast inflow (Q_i) and the proposed releases (Q_o) are provided below:

t (h)	Q_i (m^3/s)	Q_o (m^3/s)
10 am	57	85
12 noon	74	79
2 pm	122	57
4 pm	164	34
6 pm	136	25
8 pm	102	23

Do the following:

a. Plot the inflow and outflow (in m^3/s) versus time for the forecast flood event.
b. Compute change in reservoir storage (in m^3) for each 2-h time step
c. Compute the reservoir storage (in m^3) at each time (e.g., the times shown in the table above)

Note: The inflows and outflows are instantaneous rates at the time shown. You will need to use these instantaneous rates to estimate inflow and outflow flow **volumes** (in m^3) for **each 2-h time step** (e.g., from 10 am to 12 noon, then from 12 noon to 2 pm, and so on). Ignore all other fluxes (e.g., precipitation, evaporation, etc) into and out of the reservoir.

Solution:

Problem-04

Gregory Ewing

February 11, 2019

1 Water Balance for the Flood Control Reservoir

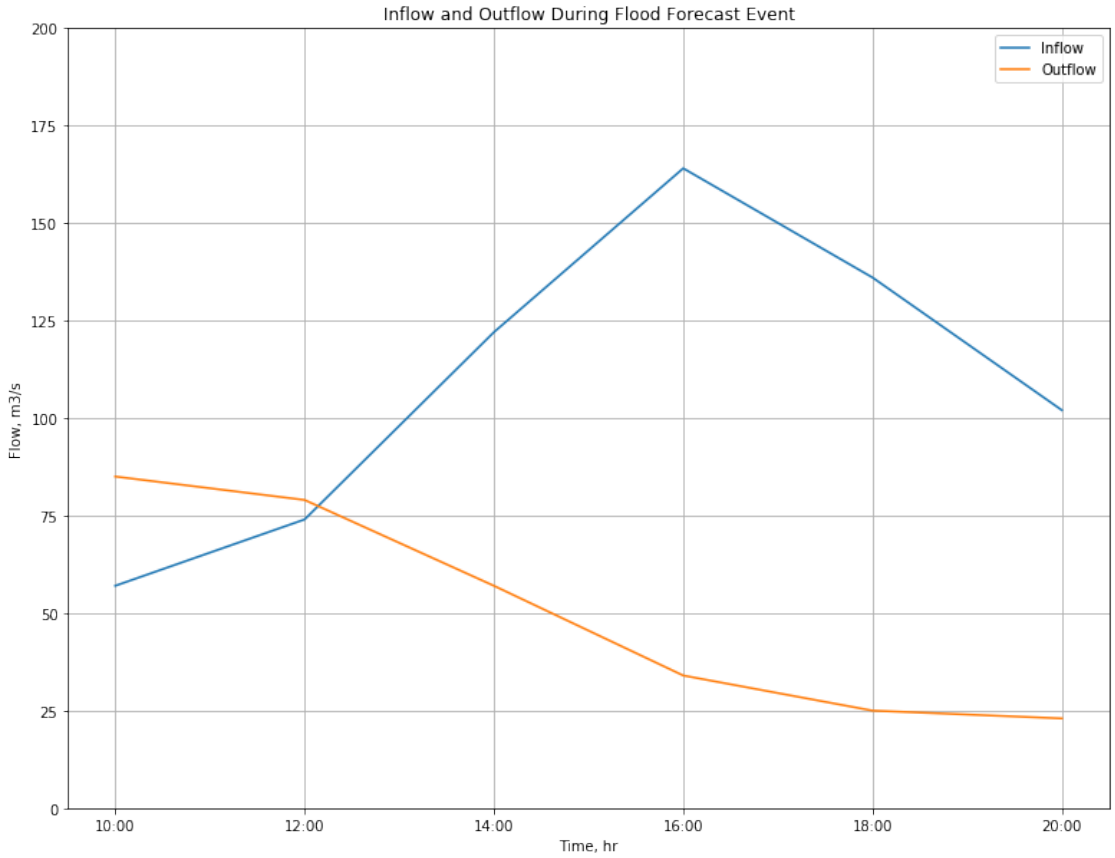
A flood control reservoir has been releasing flows in anticipation of a forecast flood event. The reservoir storage is 490,000 m^3 at 10 am. The latest forecast in (Q_i) and the proposed releases (Q_o) are provide below:

t [hr]	Q_i [m^3/s]	Q_o [m^3/s]
10 am	57	85
12 noon	74	79
2 pm	122	57
4 pm	164	34
6 pm	136	25
8 pm	102	23

a) Plot the inflow and outflow (in m^3/s) versus time for the forecast flood event

```
In [5]: t = ['10:00','12:00','14:00','16:00','18:00','20:00']
        Qi = [57,74,122,164,136,102]
        Qo = [85,79,57,34,25,23]

In [20]: data = pd.DataFrame(
             data = np.transpose(np.array([Qi,Qo])),
             columns = [
                 'Q_i',
                 'Q_o'
             ],
             index = t,
         )
```



b) Compute change in reservoir storage (in m^3) for each 2-h time step.
The change in storage can be calculated using the average flow rate for each 2-hr time step.

```
In [113]: Qi_interp = np.zeros(len(Qi)-1)
          Qo_interp = np.zeros(len(Qi)-1)
          index = []


          for i in np.arange(len(Qi)-1):
              Qi_interp[i] = (1/2)*(Qi[i] + Qi[i+1])
              Qo_interp[i] = (1/2)*(Qo[i] + Qo[i+1])

              index.append(t[i] + ' - ' + t[i+1])

          data_interp = pd.DataFrame(
              data = np.transpose(np.array([Qi_interp,Qo_interp])),
              columns = [
                  'Q_i_avg',
                  'Q_o_avg'
              ],
              index = index
```

```
)

data_interp['DeltaS m3/s'] = data_interp.Q_i_avg - data_interp.Q_o_avg
data_interp['DeltaS m3'] = data_interp['DeltaS m3/s'] * 60 * 60 * 2

# print(tabulate(data_interp,headers=data_interp.columns.values,tablefmt='pipe'))
```

interval	Q_i_avg	Q_o_avg	DeltaS m3/s	DeltaS m3
10:00 - 12:00	65.5	82	-16.5	-118,800
12:00 - 14:00	98	68	30	216,000
14:00 - 16:00	143	45.5	97.5	702,000
16:00 - 18:00	150	29.5	120.5	867,600
18:00 - 20:00	119	24	95	684,000

c) Compute the reservoir storage (in m^3) at each time (e.g., the time shown in the table above)

```
In [109]: V = [490000]
          for i in data_interp['DeltaS m3']:
              V.append(V[-1]+i)

          data['Reservoir Storage m3'] = V

          # print(tabulate(data.round(),headers=data.columns.values,tablefmt='pipe'))
```

t [hr]	Q_i [m3/s]	Q_o [m3/s]	Reservoir Storage m3
10:00	57	85	490,000
12:00	74	79	371,200
14:00	122	57	587,200
16:00	164	34	1,289,200
18:00	136	25	2,156,800
20:00	102	23	2,840,800