Transversal Exercise Project

30-Dec-2019

Shaoxu Zheng

Abstract: When rainfall occurs somewhere, researchers want to get runoff data at various points. This report will use python to process and analyze all the data, which uses a tank model (a storage-type hydrological model). Throughout the code, the entire project will be classified using objects, divided into three categories (elevation, precipitation and discharge).

Contents

1. INTRODUCTION

A hydrological cycle elaborates on a water cycle process, balance and budget among a drainage basin or on a global scale. In hydrological models, the tank models only include a few storage tanks lined up horizontally, representing a zonal structure of water in the target basin. [1] This report mainly focuses on the rainfall runoff relationship and then utilizes the tank model to analyze the flow processes belonging to one part of the hydrological cycle. What's more, this report applies conditionals, loops, file manipulation and functions, likewise matrix manipulation.

2. METHODS

2.1 LUMPED MODEL

Lumped means that the catchment is seen as one unit. Since the rule of large numbers concept from probability and statistics promotes us make massive macro simplification when we check the combination of vast amounts of underlying random macro events.

2.2 DISTRIBUTED MODEL

Suppose we separate the catchment into many units, regarded as water tanks:

In time t, each cell (or tank) gets heavy precipitation (P).

The upstream tanks' discharge Q is input to the downstream tank.

2.3 TANK MODEL

The tank model is a conceptual representation of hydrological processes in a watershed per unit area. It simulates the operation of a river flowing from one unit to the next.

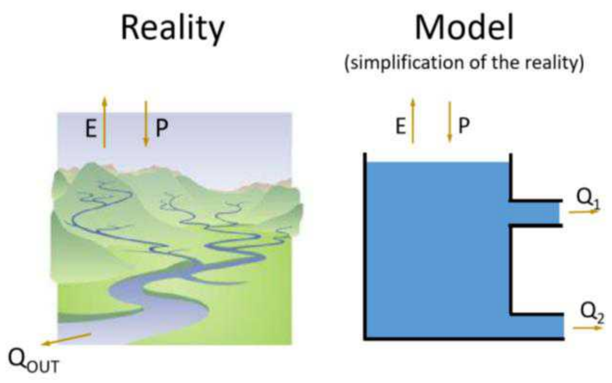

Figure 1. The tank model

In this experiment, we assume that 25 tank models represent 25 units.

2.4 FLOW DIRECTION

An eight-direction (D8) flow model is obtaining surface hydrologic characteristics is a capability to decide the flow direction for each cell in the grid. Eight valid output directions relate to the eight neighboring cells that the stream may enter. At last, the flow direction is the direction with the most significant gradient. [2]

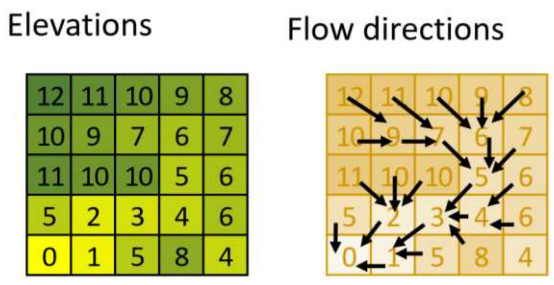

Figure 2. The direction of flow

This experiment has given the flow direction.

3. CODE DESIGN IDEA

In the code, we use three objects (elevation, precipitation and discharge). Next, we will introduce their functions in turn.

Note：

In total, I wrote two different versions, one for the original version and the other for class. In the following, I will use the release of the class as an example.

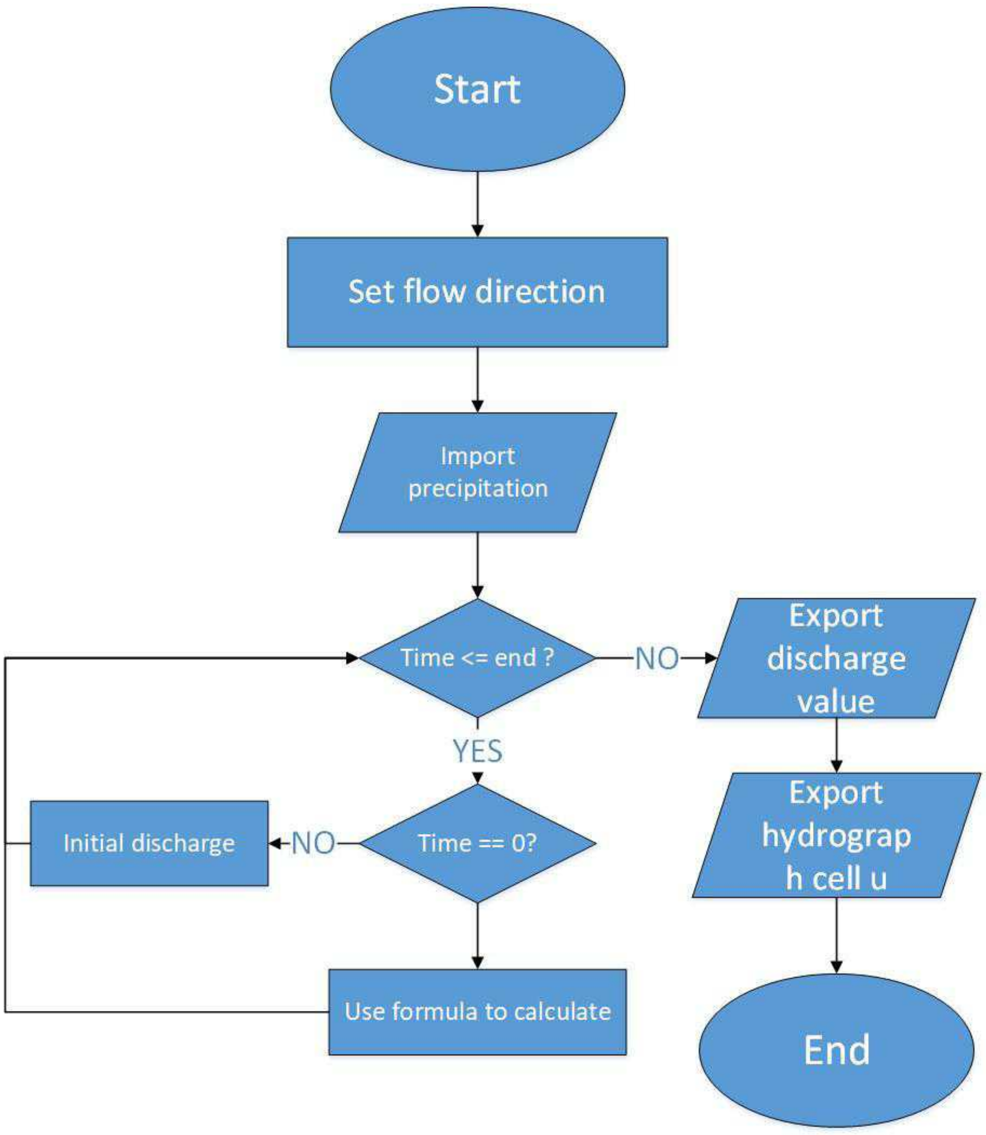

Figure 3. The flowchart

3.1 ELEVATION

This object aims to store the flow direction and provide values to each cell.

The variable ‘flow direction’ adopts dictionary way to express that water flows from one unit to another.

The function ‘get_value (self)’ is to construct cells of the discretized basin.

```
transversal_class.py
```

```python
# ==============================================================================
# class elevation
#==============================================================================
class elevation:
    def __init__(self,i_len,j_len,direction):
        self.i_len = i_len
        self.j_len = j_len
        self.direction = direction
        self.get_value()


    # construct the cells of the discretized basin
    def get_value(self):
        ele = 1
        value = np.zeros((self.i_len,self.j_len))
        for i in range(self.i_len):
            for j in range(self.j_len):
                value[i,j] = ele
                ele += 1
        self.value = value
```

```python
transversal.py
# ==============================================================================
# elevation
#==============================================================================
direction = {7:[1,6],8:[2,7],9:[3,4,5],14:[8,9,10],17:[11,12,13]\
             ,18:[14,19,24],19:[15,20],21:[16,17,22],22:[18,23]}
elevation = tc.elevation(5,5,direction)
value = elevation.value
```

3.2 PRECIPITATION

This object aims to get the data from a text file. The precipitation data of the basin is stored in a text file. Read P from the given text file using a loop and store the data in a 3D array called Precip.

In the function ‘get_precip (self)’, the data is sliced by discriminating whether the first character is ‘P.’

```python
transversal_class.py
# ==============================================================================
# class precipitation
# ==============================================================================
class precipitation:
    def __init__(self,filename):
        self.filename = filename
        self.shape_pre()
        self.get_precip()


    # get the shape of array precip
    def shape_pre(self):
        t_len = 0
        # get the total length of t
        with open(self.filename,'r') as fname:
            # read all data
            data_all = fname.readlines()
            #find the starting line of Precip data
            point = -1
            for d_initial in data_all:
                point += 1
                if d_initial[0] == 'P':
                    break
            self.point = point
            # choose the data for the corresponding date
            data = data_all[self.point:]
            while data:
                d = data.pop()
                if d[0] =='P':
                    t_len += 1
            i_len = 0
            data = data_all[self.point:]
            while data:
                d = data.pop()
                if d[0] == 'P':
                    break
                i_len +=1

            data = data_all[self.point:]
            strings = data[1].split( )
            j_len = len(strings)

        self.i_len = i_len
        self.j_len = j_len
        self.t_len = t_len


    # get the full data of precip
    def get_precip(self):
        precip = np.zeros((self.t_len,self.i_len,self.j_len))
        i = 5
        t = 16
        with open(self.filename,'r') as fname:
            # read all data
            data_all = fname.readlines()
            # choose the data for the corresponding date
            data = data_all[self.point:]
            while data:
                d = data.pop()
                # separate data based on spaces
                line = d.split( )
                # combine all data into one list
                if line[0] =='P':
                    t -= 1
                else:
                    i -= 1
                    j = -1
                    for string in line:
                        j += 1
                        precip[t,i,j] = float(string)
                    if i == 0:
                        i = 5
        self.precip = precip
```

```python
transversal.py
# ==============================================================================
# precipitation
# ==============================================================================
filename = 'Distributed_P.txt'
precipitation = tc.precipitation(filename)
t_len = precipitation.t_len
```

```
i_len = precipitation.i_len
j_len = precipitation.j_len
precip = precipitation.precip
```

3.3 DISCHARGE

In this object, the code has three main functions. The first is 'calculation' to calculate the resulting time series of Q in the basin's outlet when the precipitation event stored in matrix Precip [t, i, j] occurs.

For the first time step (t= 1), calculate Q for each cell using equation$Q_t = Q_{t-1} k e^{-k\Delta t} + P_t (1 - k e^{-k\Delta t})$, and store the calculated discharge at this time step

For the next time step, calculate the discharge for each cell taking into account the contribution of upstream cells (according to the given flow direction)

The second function is 'save_file (self)' to save the result into a text file. The code use loop operation to slice 3D data. At the same time, it adds time marks to distinguish each period.

The third function is 'hydrograph (self, i, j)' to provide hydrograph for every point.

```
# ============================================================================
# class discharge
# ============================================================================

class discharge:
    def __init__(self,t_len,i_len,j_len,initial_discharge,k):
        self.t_len = t_len
        self.i_len = i_len
        self.j_len = j_len
        self.initial_discharge =initial_discharge
        self.k = k

    # calculate the resulting time series of Q in the basin's outlet
    def calculation(self,value,direction,precip):
        discharge = np.zeros((self.t_len,self.i_len,self.j_len))

        # for the initial step(t=0)
        for i in range(self.i_len):
            for j in range(self.j_len):
                discharge[0,i,j] = self.initial_discharge

        # for the next time step
        for t in range(1,self.t_len):
            value_number = 0
            for i in range(self.i_len):
                for j in range(self.j_len):
                    discharge[t,i,j] = \
                    discharge[t-1,i,j] * self.k * np.exp(-self.k * 1) \
                    + precip[t,i,j] * (1 - self.k * np.exp(-self.k * 1))
                    value_number += 1
                    # make sure whether water is flowing into this cell
                    try:
                        given_index = direction[value_number]
                    except:
                        # if not, skip the loop and continue to run next cell
                        continue
                    for number in given_index:
                        [[m,n]] = np.argwhere(value == number)
                        discharge[t,i,j] += discharge[t-1,m,n]
```

```python
        self.discharge = np.round(discharge,2)
        self.save_file()
        return discharge


    # save the result of discharge
    def save_file(self):
        filename = 'Result_'+str(self.initial_discharge)+'_'+str(self.k)+'.txt'
        result = []
        # write the explanation
        name = 'Shaoxu Zheng'
        locker = 'Locker number: 387'
        first_line =  name + '\t' + locker +'\n'
        second_line = '*This file contains information of discharge per cell and per time.\n'
        third_line = '*For each time, 25 values are provided, corresponding to the 25 cells of the discretized basin.\n'
        forth_line = '* 1  2  3  4  5\n* 6  7  8  9 10\n*11 12 13 14 15\n*16 17 18 19 20\n*21 22 23 24 25\n\n******************\n'
        result.append(first_line)
        result.append(second_line)
        result.append(third_line)
        result.append(forth_line)
        # splice the data
        for t in range(self.t_len):
            unit_title = 'P\tt=\t'+str(t)+'\n'
            result.append(unit_title)
            for i in range(self.i_len):
                unit_num = ''
                for j in range(self.j_len):
                    unit_num += str(self.discharge[t,i,j]) + '\t'
                unit_num += '\n'
                result.append(unit_num)
        with open(filename,'w') as f:
            f.writelines(result)

    # plot the hydrograph of point (4,0) and save it
    def hydrograph(self,i,j):
        Q = self.discharge[:,i,j]
        x = range(self.t_len)
        plt.plot(x,Q)
        plt.xlabel('time (h)',size=12)
        plt.ylabel('discharge (mm/h)',size=12)
        plt.title('the hydrograph of point ('+str(i)+','+str(j)+')_'+str(self.initial_discharge)+'_'+str(self.k),size=20)
        plt.grid()
        plt.savefig('the hydrograph of point ('+str(i)+','+str(j)+')_'+str(self.initial_discharge)+'_'+str(self.k)+'.jpg')
        plt.show()
```

```python
# ==============================================================================
# discharge
# ==============================================================================
initial_discharge = 10
k = 1.5
discharge = tc.discharge(t_len,i_len,j_len,initial_discharge,k)
discharge_total = discharge.calculation(value,direction,precip)
discharge.hydrograph(4,0)

# change initial_discharge to 0
initial_discharge_ch = 0
discharge_ch = tc.discharge(t_len,i_len,j_len,initial_discharge_ch,k)
discharge_total_ch = discharge_ch.calculation(value,direction,precip)
```

```
discharge_ch.hydrograph(4,0)
```

4. EXAMPLE

In the code, we give an example of the calculation of the resulting discharge produced by cell h for the first three-time steps (1, 2 and 3). Use the same variable names you use in your code.

4.1 CODING PROCESSION

```python
# for the initial time step(t=0)
for i in range(i_len):
    for j in range(j_len):
        discharge[0,i,j] = initial_discharge

# for the next time step
for t in range(10,3):
    value_number = 0
    for i in range(i_len):
        for j in range(j_len):
            discharge[t,i,j] = discharge[t-1,i,j] * k * np.exp(-k * 1) +\
            Precip[t,i,j] * (1 - k * np.exp(-k * 1))
            value_number += 1
            # make sure whether water is flowing into this cell
            try:
                given_index = direction[value_number]
            except:
                # if not, skip the loop and continue to run next cell
                continue
            for number in given_index:
                [[m,n]] = np.argwhere(value == number)
                discharge[t,i,j] += discharge[t-1,m,n]

# the first time step
h_1 = discharge[1,1,2]
# the second time step
h_2 = discharge[2,1,2]
# the third time step
h_3 = discharge[3,1,2]
```

4.2 CALCULATION PROCESSION

discharge[0,1,2] = initial_discharge = 10(mm/h)

The first time step

discharge[1,1,2] = discharge[0,1,2] * k * np.exp(-k * 1) + Precip[1,1,2] * (1 - k * np.exp(-k * 1))+discharge[0,0,1] + discharge[0,1,1]

= 10 * 1.5 * exp(-1.5) + 10 + 10 = 23.35(mm/h)

The second time step

discharge[2,1,2] = discharge[1,1,2] * k * np.exp(-k * 1) + Precip[2,1,2] * (1 - k * np.exp(-k * 1))+discharge[1,0,1] + discharge[1,1,1]

= 23.35 * 1.5 * exp(-1.5) + 3.35 + 23.35 = 34.51(mm/h)

The third time step

discharge[3,1,2] = discharge[2,1,2] * k * np.exp(-k * 1) + Precip[3,1,2] * (1 - k * np.exp(-k * 1))+discharge[2,0,1] + discharge[2,1,1]

= 34.51 * 1.5 * exp(-1.5) + 1.12 + 14.51 = 27.18(mm/h)

5. THE GRAPH t vs Q AT THE OULET (cell u)

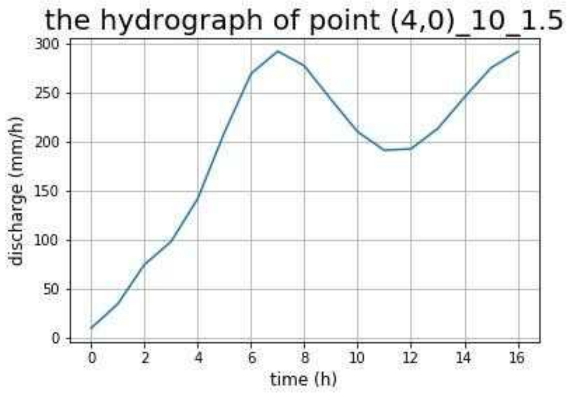

Figure 4. The hydrograph of cell u

6. QUESTIONS

6.1 Q_1

You have read the precipitation from the text file Distributed_P.txt and stored it in the variable Precip. What is the Python expression to get the precipitation value of cell i=3, j=4 at time step 5? What is this value?

Precip[5,3,4] = 1.0 mm/h

6.2 Q_2

How the graph t vs Q would look like if all tanks are empty at the beginning, (i.e., if Qt-1 = 0, instead of 10)

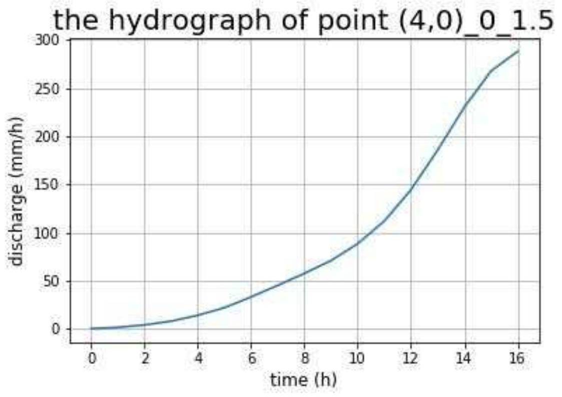

Figure 5. The hydrograph of cell u (Q_{t-1}=0)

REFERENCE

1. Phuong, H. T., et al. (2018). "A Hydrological Tank Model Assessing Historical Runoff Variation in the Hieu River Basin." Asian Journal of Water, Environment and Pollution 15(1): 75-86.

2. https://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/flow-direction.htm
3. https://docs.python.org/3/tutorial/classes.html
4. https://www.programiz.com/python-programming/methods/list/pop