assginment_1a

January 26, 2020

1 A aquifer storage and recovery system to reduce extraction from river during summer

$Zhang\,Fengbo$

$(Student\,number:\ 1068520;\ E-mail:\ fzh001@un-ihe.org;\ Locker:\ 333)$

1.1 Introduction

A water company extracts water from a small river to treat and distribute it as drinking water for the population of a small town nearby. However, the extraction method is not suitable in the summer when the little river has a dry season, which the discharge of the river is at its lowest, due to rising demand for drinking water and increasing environmental concern. Thus, the environmental agency has recently even forbidden to extract water from the river during the summer months further.

To solve the problem that this causes for the drinking water supply, the drinking water company has suggested an "Aquifer Storage and Recovery" system (these so-called ASR systems are becoming more and more popular). It can take in more river water during winter and inject it through a well (or wells) at some distance from the river into the local water-table aquifer so that this water can be extracted during the next summer. This way, no water-intake will be necessary during the summer months.

This paper constructs the numerical model of the ASR system and finds the minimum distance where no stored water flow back to the river during the winter and no infiltration flow from the river to the well during the summer because of the extraction. Addtionally, the suggestion of the water company is discussed that whether the well is located at a distance of 500m away from the river is suitable.

1.2 Loading required modules & defining default graph

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.special import exp1
```

[3]:
```
def newfig(title='forgot title?', xlabel='forgot xalbel?', ylabel='forgot␣
↪ylabel?', xlim=None, ylim=None,
           xscale='linear', yscale='linear', size_inches=(12, 6)):
```

```
    fig, ax = plt.subplots()
    fig.set_size_inches(size_inches)
    ax.set_title(title)
    ax.set_xlabel(xlabel)
    ax.set_ylabel(ylabel)
    ax.set_xscale(xscale)
    ax.set_yscale(yscale)
    if not xlim is None: ax.set_xlim(xlim)
    if not ylim is None: ax.set_ylim(ylim)
    ax.grid()
    return ax
```

1.3 Study Area

A well is to be constructed next to a river that is in direct contact whith the aquifer next to it, as shown in Fig. 1.

Fig 1: Situation sketch

According to the requirment, the aquifer properties are confirmed that Transmissivity and storage coefficient are given as

$$kD = 900\, m^2/d$$

$$S = 0.2\, [-]$$

Meanwhile, the drinking water demand, Q, is considered constant year-round at 150 L/d per inhabitant for the 10000 inhabitants of the town, and the water company well extracts during 3 summer months (June, July, Aug) its full demand. It compensates that during 6 winter months (October-March) by injecting half its daily capacity. There is neither injection nor extraction during the months of April, May, and September. Thus, Q is shown in table 1 with varying monthly as follows (extraction is negative and injection is positive), but the time step is one day in the model to decrease numerical differentiation errors.

[4]:
```
kD = 900 # m2/d
S = 0.2 # [-]

ayear = 365.25 #Convenence parameter, to convert between days and years
```

Table 1 the time-variation of the discharge Q in the well

- n - the n th month
- Qfac - the fraction of drinking water demand Q ((October-March) Qfac = 0.5, (June-Augest) Qfac = -1)
- dQ - the change in the discharge
- tch - the dayscale from the strating month to n th month

Note: the extraction is negative and the injection is positive

[5]:
```
welldata = pd.read_excel('time_variation_dischargeQ.xlsx', sheet_name='dQtch',␣
↪parse_dates=True,dayfirst=True,
```

```
                         index_col=0)

fig, ax = plt.subplots()
fig.patch.set_visible(False)
ax.axis('off')
ax.axis('tight')
data=[]
datalabels=['n','Qfac','dQ', 'tch']
for iw in range(len(welldata)):
    Qfac, dQ, tch = welldata.iloc[iw][['Qfac','dQ', 'tch']]
    n = iw
    data.append([n, Qfac, dQ, tch])

the_table = ax.table(cellText=data, colLabels= datalabels, loc='center')
```

n	Qfac	dQ	tch
0	0.5	750	0
1	0.5	0	31
2	0.5	0	59
3	0.0	-750	90
4	0.0	0	120
5	-1.0	-1500	151
6	-1.0	0	181
7	-1.0	0	212
8	0.0	1500	243
9	0.5	750	273
10	0.5	0	304
11	0.5	0	334
12	0.5	0	365
13	0.5	0	396
14	0.5	0	425
15	0.0	-750	456
16	0.0	0	486
17	-1.0	-1500	517
18	-1.0	0	547
19	-1.0	0	578
20	0.0	1500	609
21	0.5	750	639
22	0.5	0	670
23	0.5	0	700
24	0.5	0	731
25	0.5	0	762
26	0.5	0	790
27	0.0	-750	821
28	0.0	0	851
29	-1.0	-1500	882
30	-1.0	0	912
31	-1.0	0	943
32	0.0	1500	974
33	0.5	750	1004
34	0.5	0	1035
35	0.5	0	1065
36	0.5	0	1096
37	0.5	0	1127
38	0.5	0	1155
39	0.0	-750	1186
40	0.0	0	1216
41	-1.0	-1500	1247
42	-1.0	0	1277
43	-1.0	0	1308
44	0.0	1500	1339
45	0.5	750	1369
46	0.5	0	1400
47	0.5	0	1430
48	0.5	0	1461
49	0.5	0	1492
50	0.5	0	1520
51	0.0	-750	1551
52	0.0	0	1581
53	-1.0	-1500	1612
54	-1.0	0	1642
55	-1.0	0	1673
56	0.0	1500	1704
57	0.5	750	1734
58	0.5	0	1765
59	0.5	0	1795

1.4 Methods

1.4.1 Conceptual hydrogeological model

- The aquifer is homogeneous and isotropic, which means Transmissivity is constant;
- The aqufier is bounded by a straight constant head boundaty on the left, as shown in Fig.2.1;
- A pumping well is located near the boundary with the changed pumping rate, as shown in table 1.

Fig 2.1: conceptual Sketch(left)

Fig 2.2: conceptual Sketch(right)

Considering the situation that the well is not near the river is be to show how the drawdown varies by time.

- (1)By using Theis' well function, drawdowns s(u,tch) can be computed for a change in the discharge dQ which starting at t time.

$$s(u,tch) = \frac{Q}{4\pi kD} W(u), \quad W(u) = \int_u^\infty \frac{e^{-y}}{y} dy, \quad u = \frac{r^2 S}{4kDt}$$

$$tch = 0,\ 31,\ 59,\ ....\ 1795.$$

>+ Take a point (x_p y_p) along the river and consider as well at (x_w, y_w), which get the distance from the well to the point

$$r = \sqrt{(x_w - x_p)^2 + (y_w - y_p)^2}$$

>+ (2)the total drawdown is the sum of the each drawdowns, as shown in fig.4.

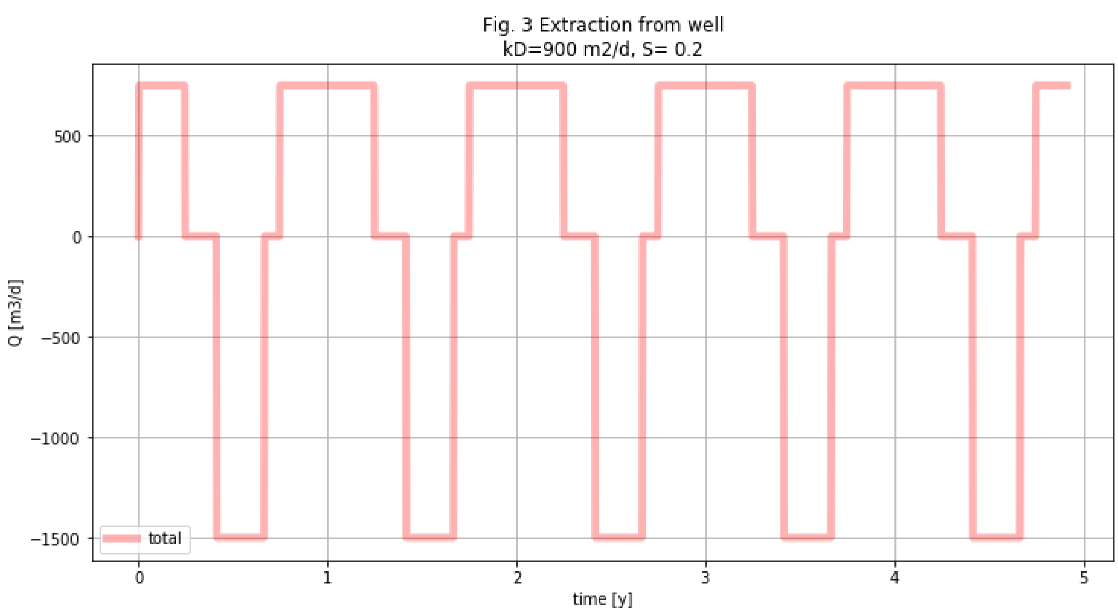
$$s(u) = s(u,0) + s(u,31) + s(u,59)... + s(u,1795)$$

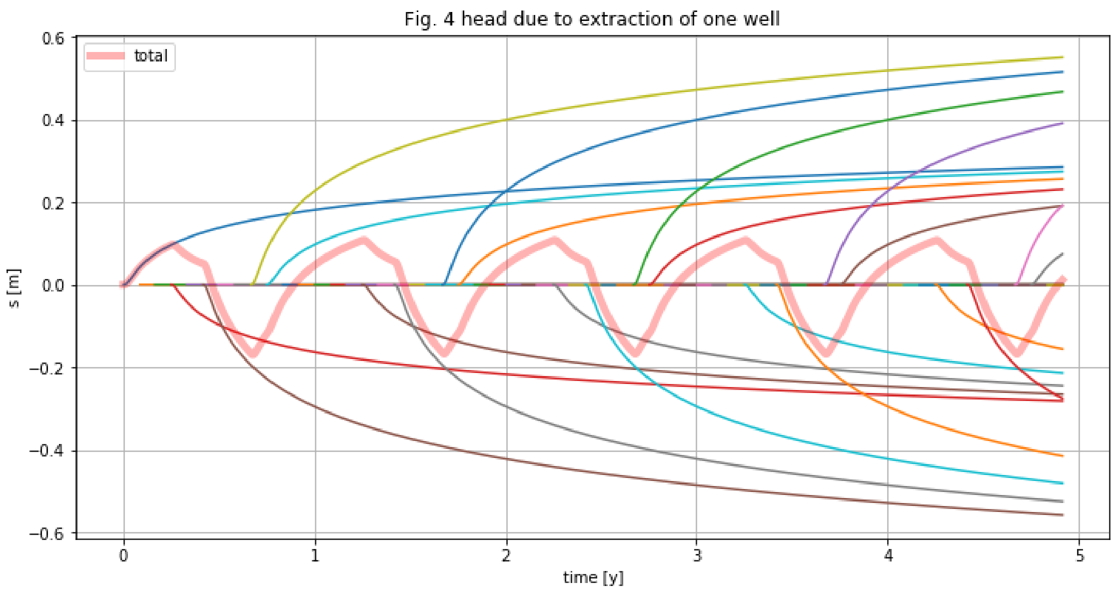
Note: In this example,the location(x_p y_p) of the drawdown is at (0,0) when the well x_w, y_w ia located at (-500,0)

[6]:
```
subttl = f'\nkD={kD:.0f} m2/d, S={S:4g}'

# Generate two axes for our visualization:
ax0 = newfig(title='Fig. 3 Extraction from well' + subttl, xlabel='time [y]',␣
↪ylabel='Q [m3/d]')
ax1 = newfig(title='Fig. 4 head due to extraction of one well',
             xlabel='time [y]', ylabel='s [m]')

xw, yw = -500., 0. # The well coordinates
xp, yp = 0., 0.    # The coordinates of the observation point
```

```
r = np.sqrt((xw - xp) ** 2 + (yw - yp) ** 2) # Distance between well and␣
↪observation point
time = np.linspace(0, 1795, 1796)
s = np.zeros_like(time) # Initialize the head to all zeros
Q = np.zeros_like(time) # Same for the discharge

# Loop over all wells in the well data DataFrame
for iw in range(len(welldata)):  # iw is the line number in the DataFrame
    dQ, tch = welldata.iloc[iw][['dQ', 'tch']]  # From this line select flow␣
↪change dQ and change time tch
    u = r ** 2 * S / (4 * kD * (time[time > tch] - tch))
    ds = dQ/(4 * np.pi * kD) * exp1(u) # head change due to this single dQ
    Q[time > tch] += dQ  # Add this dQ to all times > tch
    ax1.plot(time[time > tch] / ayear, ds) # plot ds versus time for t > tch
    s[time > tch] += ds # Add this contribution to the total hean change
ax0.plot(time / ayear, Q, 'r', lw=5, alpha=0.3, label='total') # plot total␣
↪Discharge
ax1.plot(time / ayear, s, 'r', lw=5, alpha=0.3, label='total') # Plot overall␣
↪head change
ax0.legend()
ax1.legend()
```

[6]: <matplotlib.legend.Legend at 0x2ae84313a48>





Building the model of a well near the river

the method of the model

- considering the constant head boundary as a mirror
- putting an imaginary injecting well at the image location of the pumping well, i.e. at point (x=-p, y=0), as shown in Fig.2.2;
- giving the injecting rate equal to the pumping rate(when the well injects, the mirror well extracts.);
- using the principle of superposition to find the solution.

the function of the model

- (1)the drawdown s(u, tch) is the sum of the drawdowns which each well influence this piont

$$s(u_1, tch) = \frac{Q}{4\pi kD} W(u_1), \quad W(u_1) = \int_u^\infty \frac{e^{-y}}{y} dy, \quad u = \frac{r_1^2 S}{4kDt}$$

  - (drawdown caused by real well)

$$s(u_2, tch) = \frac{Q}{4\pi kD} W(u_2), \quad W(u_2) = \int_u^\infty \frac{e^{-y}}{y} dy, \quad u = \frac{r_2^2 S}{4kDt}$$

>>+ (drawdown caused by mirror well)

$$s(u, tch) = s(u_1, tch) + s(u_2, tch)$$

>>+ (Total drawdown)

$$tch = 0,\ 31,\ 59,\ ....\ 1795.$$

$$r_1 = \sqrt{(x_w - x_p)^2 + (y_w - y_p)^2}$$

$$r_2 = \sqrt{(x_2 - x_p)^2 + (y_2 - y_p)^2}$$

- (2)the total drawdown varied by time is the sum of the each drawdowns, as shown in fig.6.

$$s(u) = s(u, 0) + s(u, 31) + s(u, 59)... + s(u, 1795)$$

Note: (1)the location of mirror well is at (x_2,y_2),which equals to ($-x_w$,y_w);(2)In this example,the location(x_p y_p) of the drawdown is at (100,100) when the well x_w, y_w ia located at (-500,0)

[7]:
```
def ASRQ(xw=None,yw=None,xp=None,yp=None):

    welldata = pd.read_excel('time_variation_dischargeQ.xlsx',␣
 ↪sheet_name='dQtch', parse_dates=True,dayfirst=True,
                              index_col=0)

    x2, y2 = -xw, 0
    r1 = np.sqrt((xw - xp) ** 2 + (yw - yp) ** 2)
    r2 = np.sqrt((x2 - xp) ** 2 + (y2 - yp) ** 2)
    time = np.linspace(0, 1795, 1796)
    s = np.zeros_like(time)
    Q = np.zeros_like(time)
    for iw in range(len(welldata)):
        dQ, tch = welldata.iloc[iw][['dQ', 'tch']]
        n = np.sum(time > tch)
        u1 = r1 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        u2 = r2 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        Q[time > tch]  += dQ
        ds = -dQ/(4 * np.pi * kD) * (exp1(u1) - exp1(u2))
        s[time > tch] += ds
    return Q
```

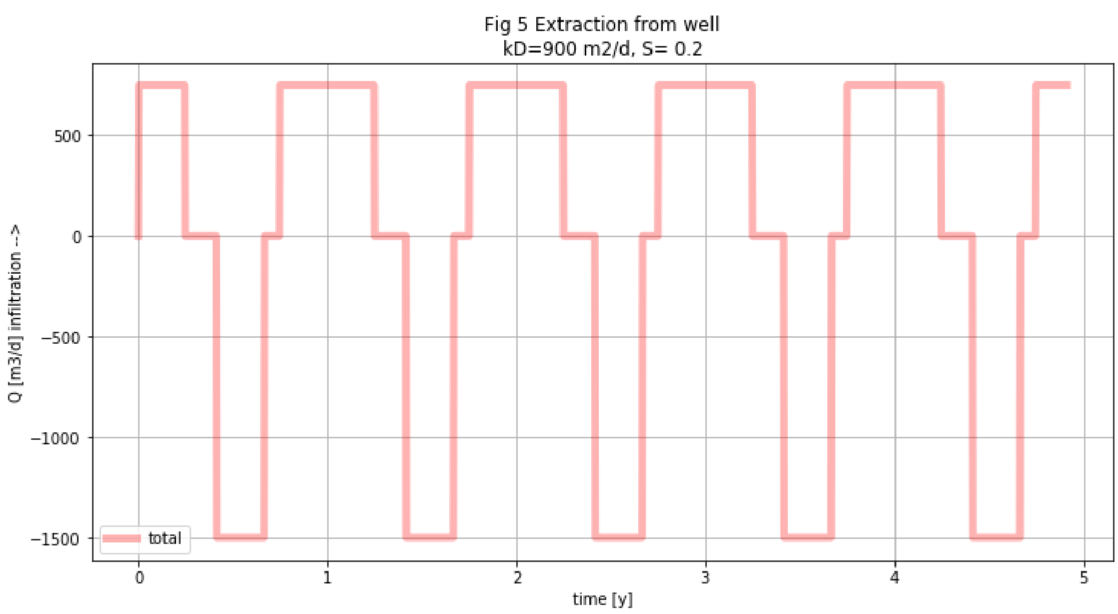
[8]:
```
def ASRs(xw=None,yw=None,xp=None,yp=None):

    welldata = pd.read_excel('time_variation_dischargeQ.xlsx',␣
 ↪sheet_name='dQtch', parse_dates=True,dayfirst=True,
                              index_col=0)
    x2, y2 = -xw, 0
    r1 = np.sqrt((xw - xp) ** 2 + (yw - yp) ** 2)
    r2 = np.sqrt((x2 - xp) ** 2 + (y2 - yp) ** 2)
    time = np.linspace(0, 1795, 1796)
    s = np.zeros_like(time)
    Q = np.zeros_like(time)
    for iw in range(len(welldata)):
```

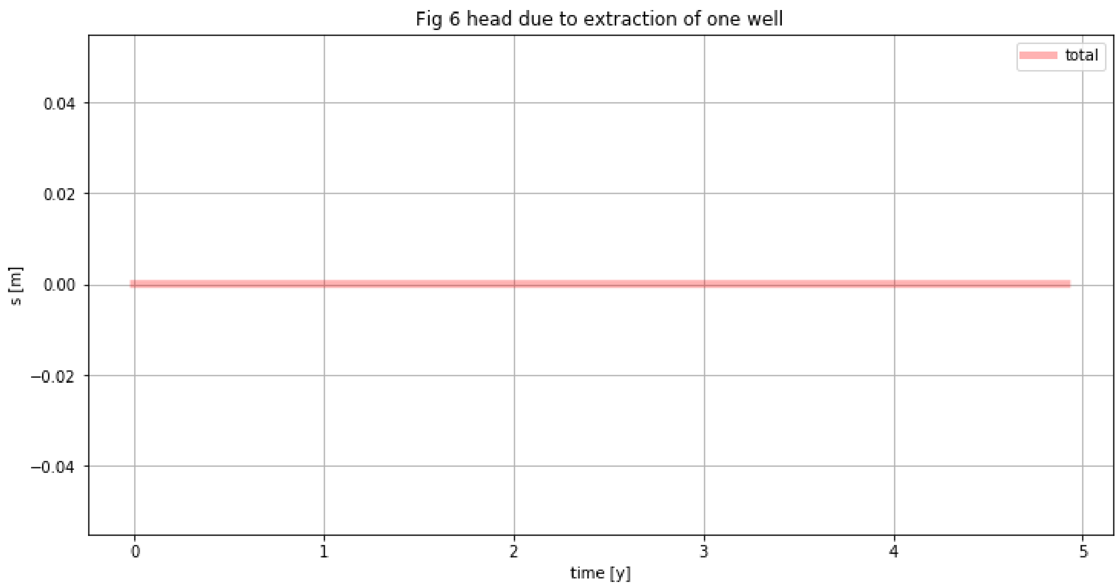
```python
        dQ, tch = welldata.iloc[iw][['dQ', 'tch']]
        n = np.sum(time > tch)
        u1 = r1 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        u2 = r2 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        Q[time > tch]  += dQ
        ds = -dQ/(4 * np.pi * kD) * (exp1(u1) - exp1(u2))
        s[time > tch] += ds
    return s
```

[9]:

```python
ax0 = newfig(title='Fig 5 Extraction from well' + subttl, xlabel='time [y]',
             ylabel='Q [m3/d] infiltration -->')
ax1 = newfig(title='Fig 6 head due to extraction of one well',
             xlabel='time [y]', ylabel='s [m]')

Q = ASRQ(xw=-500,yw=0,xp=0,yp=0)
s = ASRs(xw=-500,yw=0,xp=0,yp=0)

ax0.plot(time / ayear, Q, 'r', lw=5, alpha=0.3, label='total')
ax1.plot(time / ayear, s, 'r', lw=5, alpha=0.3, label='total')
ax0.legend()
ax1.legend()
```

[9]: <matplotlib.legend.Legend at 0x2ae8446ab08>





The figure 6 represent that the drawdown at the point which along the river is constant.

Confirm the specific discharge of any point along the river Last point is how much water will be extracted from the river?

The discharge towards a Theis well is obtained from the solution

$$s = \frac{Q}{4\pi kD} \int_u^\infty \frac{e^{-y}}{y} dy$$

$$Q_r = -kD(2\pi r)\frac{\partial s}{\partial r}$$

$$Q_r = -kD(2\pi r)\frac{Q}{4\pi kD}\frac{-e^{-u}}{u}\frac{du}{dr}$$

$$\frac{du}{dr} = \frac{2rS}{4kDt} = \frac{2u}{r}$$

$$Q_r = Qe^{-u}$$

$$q_r = \frac{Q}{2\pi r}e^{-u}$$

Then the flow perpendicular to the shore is

$$q_p = \frac{Q_r}{2\pi r}\cos(\alpha) = \frac{x_w - x_p}{r^2}\frac{Qe^{-u}}{2\pi}$$

Thus, the flow which is influenced by the real well and the mirror well is

$$Qin = \frac{x_w - x_p}{r_1^2}\frac{Qe^{-u_1}}{2\pi} + \frac{x_2 - x_p}{r_2^2}\frac{Qe^{-u_2}}{2\pi}$$

$$r_1 = \sqrt{(x_w - x_p)^2 + (y_w - y_p)^2}$$

$$r_2 = \sqrt{(x_2 - x_p)^2 + (y_2 - y_p)^2}$$

Note: (1)the location of mirror well is at (x_2,y_2),which equals to ($-x_w$,y_w);(2)In this example,the location(x_p y_p) of the drawdown is at (0,0) when the well (x_w, y_w) ia located at (-500,0)

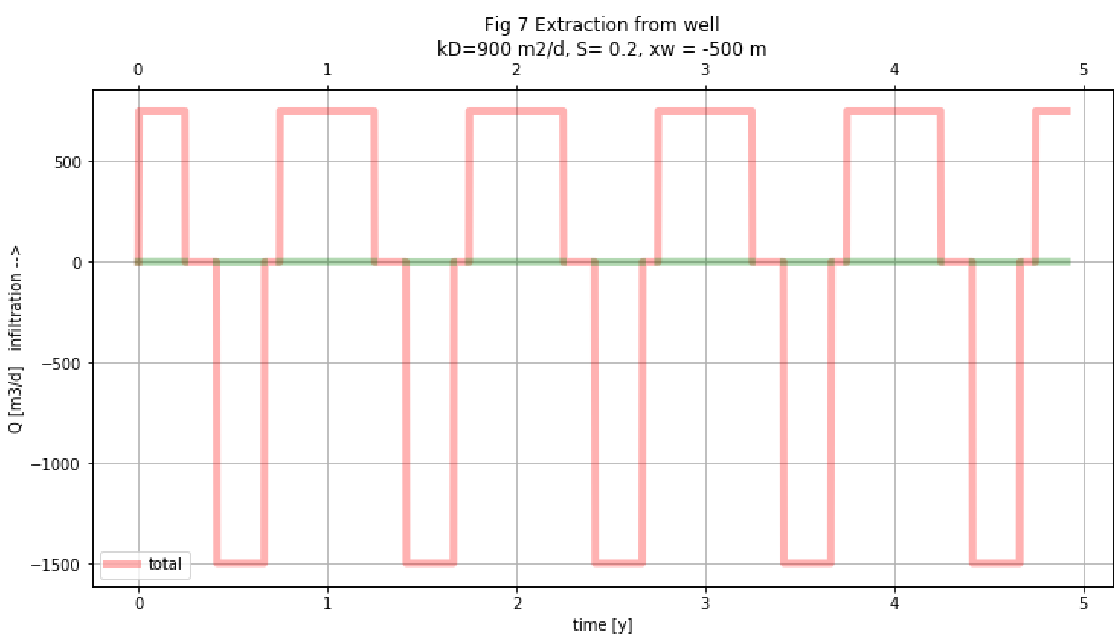
[10]:
```
def ASRQin(xw=None,yw=None,xp=None,yp=None):

    welldata = pd.read_excel('time_variation_dischargeQ.xlsx',␣
↪sheet_name='dQtch', parse_dates=True,dayfirst=True,
                                index_col=0)


    x2, y2 = -xw, 0
    r1 = np.sqrt((xw - xp) ** 2 + (yw - yp) ** 2)
    r2 = np.sqrt((x2 - xp) ** 2 + (y2 - yp) ** 2)
    time = np.linspace(0, 1795, 1796)
    Qin = np.zeros_like(time)
    Q = np.zeros_like(time)
    for iw in range(len(welldata)):
        dQ, tch = welldata.iloc[iw][['dQ', 'tch']]
        n = np.sum(time > tch)
        u1 = r1 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        u2 = r2 ** 2 * S / (4 * kD * (time[time > tch] - tch))
        Q[time > tch] += dQ
        dQin = -dQ * (((xw-xp)/r1) * np.exp(-u1)/ (2 * np.pi * r1) + ((xp-x2)/
↪r2) * np.exp(-u2) / (2 * np.pi * r2))
        Qin[time > tch] += dQin

    return Qin
```

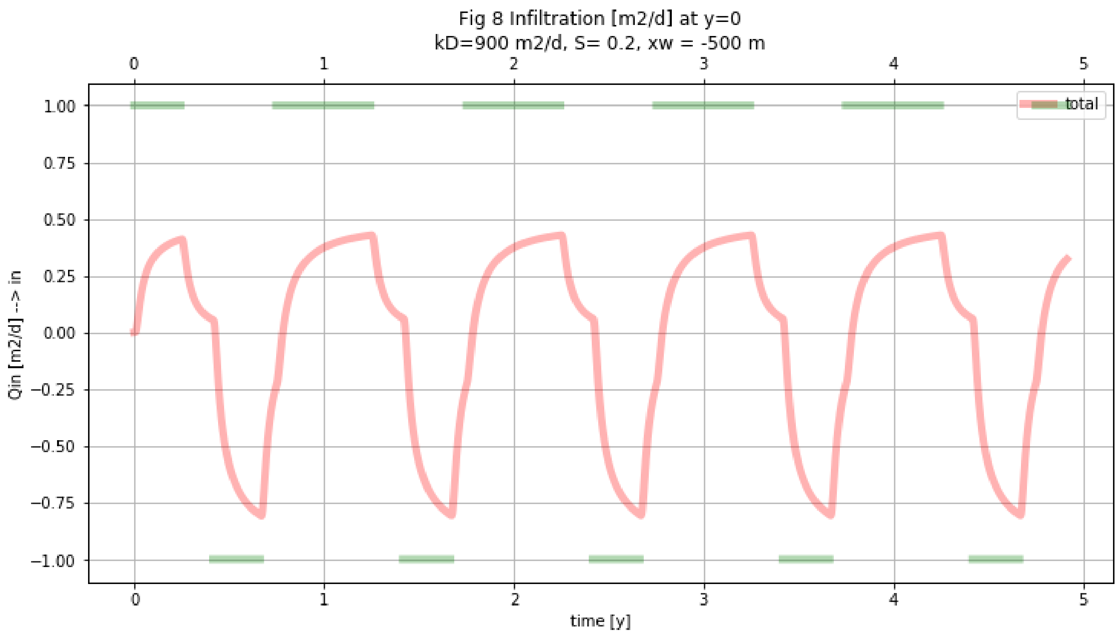
[11]:
```
xw, yw = -500,0
xp, yp = 0,0


Q = ASRQ(xw=xw,yw=yw,xp=xp,yp=yp)

Qin = ASRQin(xw=xw,yw=yw,xp=xp,yp=yp)


subttl = f'\nkD={kD:.0f} m2/d, S={S:4g}, xw = {xw:.0f} m'
```

```
ax0 = newfig(title='Fig 7 Extraction from well' + subttl, xlabel='time [y]',
             ylabel='Q [m3/d]    infiltration -->')
ax1 = newfig(title='Fig 8 Infiltration [m2/d] at y=0' + subttl, xlabel='time␣
↪[y]',
             ylabel='Qin [m2/d] --> in')

ax0.plot(time / ayear, Q, 'r', lw=5, alpha=0.3, label='total')
ax1.plot(time / ayear, Qin, 'r', lw=5, alpha=0.3, label='total')

ax2 = ax1.twiny()
ax2.plot(time / ayear, Q/np.abs(Q), 'g', lw=5, alpha=0.3, label='season')
ax3 = ax0.twiny()
ax3.plot(time / ayear, Q/np.abs(Q), 'g', lw=5, alpha=0.3, label='season')
ax0.legend()
ax1.legend()
```

```
C:\Users\fzh001\AppData\Local\Continuum\anaconda3\lib\site-
packages\ipykernel_launcher.py:21: RuntimeWarning: invalid value encountered in
true_divide
C:\Users\fzh001\AppData\Local\Continuum\anaconda3\lib\site-
packages\ipykernel_launcher.py:23: RuntimeWarning: invalid value encountered in
true_divide
```

[11]: <matplotlib.legend.Legend at 0x2ae855eff88>





1.5 Results

The model of the ASR system has been coded. Meanwhile, the changed discharge in the well, the drawdown at the shore, and the specific discharge of any point along the river can be calculated by the functions(ASRQ(),ASRs(),ASRqQin()).

To infer whether stored the water in winter and extracted it during summer without substantially affect the already low summer-discharge of the river, the whole specific discharge of the shore should be displayed, and the maximum specific discharge along the river should be found.

The results are obtained by numerical differentiation. Meanwhile, the specific discharge for a large number of points between $-ax_w < yp < ax_w$, where a is sufficiently large(a=10,Np=500).

Note: Np, which equals 500, is too large to calculate too much time by the computer. Thus, redefining Np equals to 50.

1.5.1 the specific discharge at whole point along the river

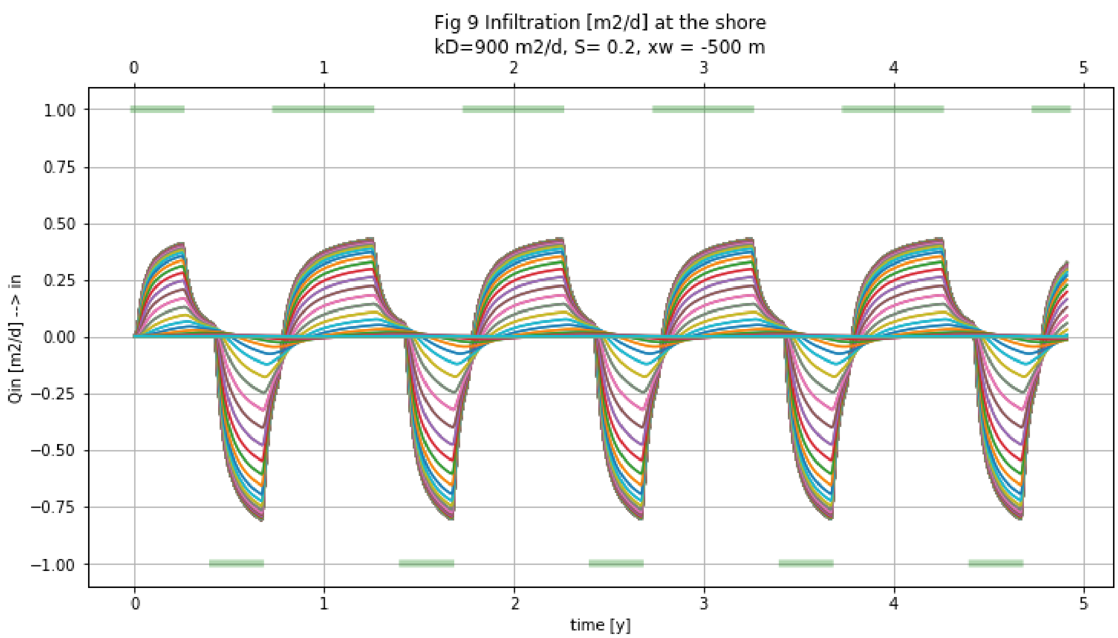
[12]:
```
xw, yw = -500,0
xp = 0
a = 10
Np = 50
y = np.hstack((-np.logspace(0, np.log10(a * -xw), Np)[::-1], np.logspace(0, np.
↪log10(a * -xw), Np)))

subttl = f'\nkD={kD:.0f} m2/d, S={S:4g}, xw = {xw:.0f} m'
ax1 = newfig(title='Fig 9 Infiltration [m2/d] at the shore' + subttl,␣
↪xlabel='time [y]',
```

```
            ylabel='Qin [m2/d] --> in')

for yp in y:

    Qin = ASRQin(xw=xw,yw=yw,xp=xp,yp=yp)
    ax1.plot(time / ayear, Qin)

Q = ASRQ(xw=xw,yw=yw,xp=xp,yp=yp)
ax2 = ax1.twiny()
ax2.plot(time / ayear, Q/np.abs(Q), 'g', lw=5, alpha=0.3, label='season')
```

```
C:\Users\fzh001\AppData\Local\Continuum\anaconda3\lib\site-
packages\ipykernel_launcher.py:18: RuntimeWarning: invalid value encountered in
true_divide
```

[12]:

```
[<matplotlib.lines.Line2D at 0x2ae85cda788>]
```



1.5.2 Obtain the point along the river which has maximum specific discharge and its location

[38]:

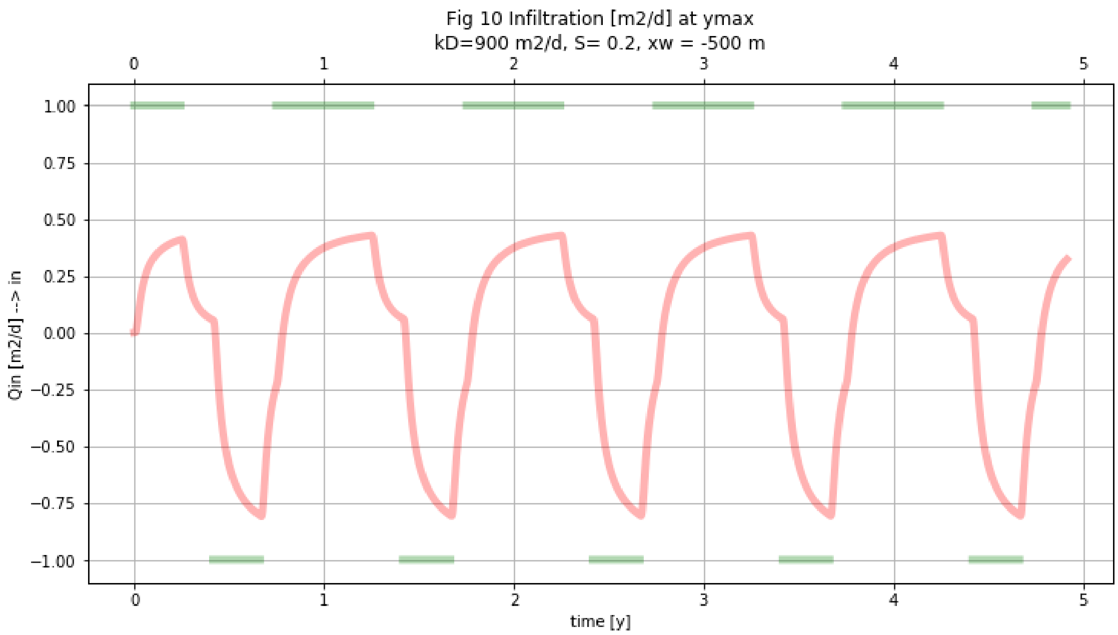
```
# the maximum specific dicharge at the shore is found, when the well is located
↪at (-500,0)
xw, yw = -500,0
xp = 0.
a = 10
Np = 50
```

```python
y = np.hstack((-np.logspace(0, np.log10(a * -xw), Np)[::-1], np.logspace(0, np.
↪log10(a * -xw), Np)))

Qmax = 0

# infer the maximum specific discharge at the whole Spatiotemporal distribution␣
↪and find its location
for yp in y:

    Qin = ASRQin(xw=xw,yw=yw,xp=xp,yp=yp)
    Qn= max(abs(Qin))

    if Qmax < Qn:
        Qmax = Qn
        ymax = yp
print('the location of the maximum specific discharge along the river is at␣
↪(',xp,',',ymax,')')


# the maximum specifc discharge varies by time
Qmax = ASRQin(xw=xw,yw=yw,xp=xp,yp=ymax)
Q = ASRQ(xw=xw,yw=yw,xp=xp,yp=yp)

Q2 = min(Qmax)
print('the maximum infiltration from the river into the aquifer equals to',Q2)

subttl = f'\nkD={kD:.0f} m2/d, S={S:4g}, xw = {xw:.0f} m'
ax1 = newfig(title='Fig 10 Infiltration [m2/d] at ymax' + subttl, xlabel='time␣
↪[y]',
             ylabel='Qin [m2/d] --> in')
ax1.plot(time / ayear, Qmax, 'r', lw=5, alpha=0.3, label='total')
ax2 = ax1.twiny()
ax2.plot(time / ayear, Q/np.abs(Q), 'g', lw=5, alpha=0.3, label='season')
```

```
the location of the maximum specific discharge along the river is at ( 0.0 ,
-1.0 )
the maximum infiltration from the river into the aquifer equals to
-0.8082307798505579
```

```
C:\Users\fzh001\AppData\Local\Continuum\anaconda3\lib\site-
packages\ipykernel_launcher.py:34: RuntimeWarning: invalid value encountered in
true_divide
```

[38]:
```
[<matplotlib.lines.Line2D at 0x2ae872c89c8>]
```



1.5.3 Confirm the suitable distance of seting up the well

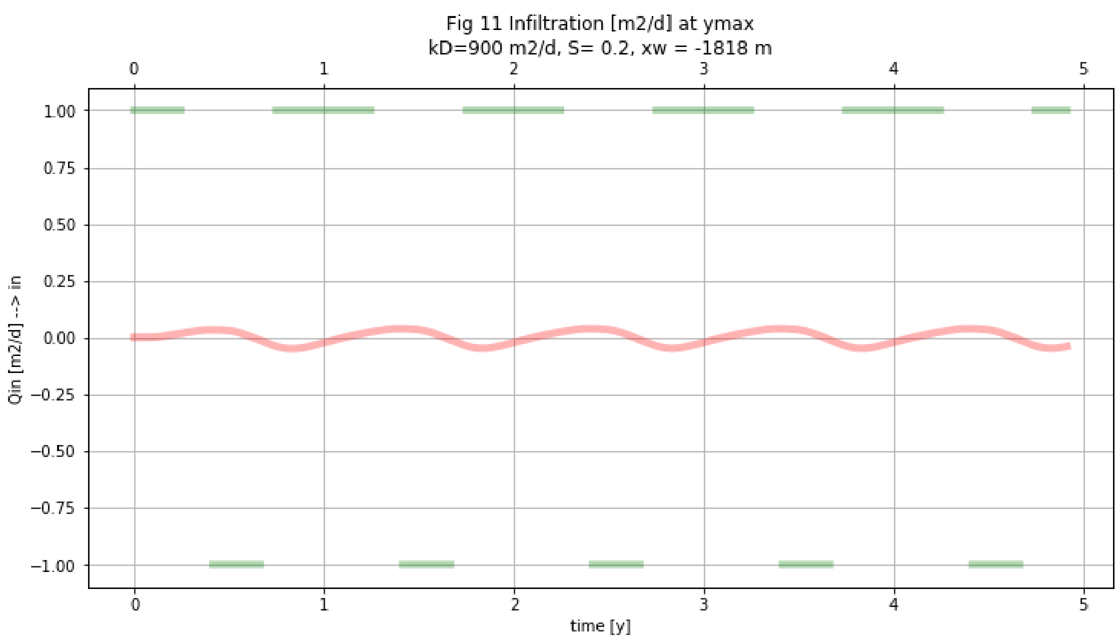
[39]:
```
yw = 0.
xp, yp= 0.,-1.0
a = 10
Np = 50

x = np.hstack(np.linspace(-1000, -2000, 100))

Qmax = 0

# infer the maximum specific discharge at the differnce distance of the well␣
↪and find its location
for xw in x:
    Qin = ASRQin(xw=xw,yw=yw,xp=xp,yp=yp)
    Qmax= max(abs(Qin))
    if Qmax < 0.05:
        xs=xw
        break
print('the location of the maxinum specific discharge along the river is at␣
↪(',xp,',',yp,')')
print('the suitiable distance of constructing the well is at (',xs,',',yw,')')

# the maximum specifc discharge varies by time
Qin = ASRQin(xw=xs,yw=yw,xp=xp,yp=ymax)
Q = ASRQ(xw=xw,yw=yw,xp=xp,yp=yp)
```

```
Q1 = max(Qin)
Q2 = min(Qin)
print('the maximum stored water flow back to the river equals to',Q1)
print('the maximum infiltration from the river into the aquifer equals to',Q2)

subttl = f'\nkD={kD:.0f} m2/d, S={S:4g}, xw = {xs:.0f} m'
ax1 = newfig(title='Fig 11 Infiltration [m2/d] at ymax' + subttl, xlabel='time␣
↪[y]',
             ylabel='Qin [m2/d] --> in')
ax1.plot(time / ayear, Qin, 'r', lw=5, alpha=0.3, label='total')
ax2 = ax1.twiny()
ax2.plot(time / ayear, Q/np.abs(Q), 'g', lw=5, alpha=0.3, label='season')
```

```
the location of the maxinum specific discharge along the river is at ( 0.0 ,
-1.0 )
the suitiable distance of constructing the well is at ( -1818.181818181818 , 0.0
)
the maximum stored water flow back to the river equals to 0.037586407827617435
the maximum infiltration from the river into the aquifer equals to
-0.04967994074612393
```

```
C:\Users\fzh001\AppData\Local\Continuum\anaconda3\lib\site-
packages\ipykernel_launcher.py:33: RuntimeWarning: invalid value encountered in
true_divide
```

[39]: [<matplotlib.lines.Line2D at 0x2ae88727b08>]



1.6 Discussion

According to the model, the suitable distance to set up the well is 1818 m, and it makes that the maximum infiltration from the river to aquifer during the summer is smaller than $0.05\,m^2/d$, which is negligible compared with the injection and extraction.

As shown in Fig 11, the stored water flows back to the river during the winter, while the extraction induces the infiltration from the river to the aquifer. However, the maximum discharge and infiltration equal to $0.037\,m^2/d$ and $0.05\,m^2/d$ respectively, which means the well hardly affects the river during the summer.

The water company suggested a distance of 500 m from the river. Figure 10 represents the changed infiltration and discharge, where the well set up at a distance of 500 m. It clearly shows that the maximum infiltration equal to $0.8\,m^2/d$, which means the well affect already low summer-discharge of the river. Thus, the environmental agency should disagree.

On the other hand, the model still has a drawback. The point along the river, which has the maximum specific discharge, may change when the distance from the river change, and it takes an extensive number of loop calculations. However, the Author's computer cannot undertake this vast calculation.