Henry's Case

The Henry's Case has been in the following exercise implemented in Flopy. The files were compared with PMWIN to make sure the results match.

MODFLOW model Creation

In [1]:

```
import flopy
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
print(os.getcwd())
os.chdir("../Henry's Case")
print(os.getcwd())
```

```
C:\Applied_GW_models\Henry's Case
C:\Applied_GW_models\Henry's Case
```

In [2]:

```
L = 10000
D = 150
delx = 10
delz = 10
dely = 1

nrow = 1

ncol = 100

nlay = 50

stress_periods = 1

time_length = 7300

Initial_C = 19000

bouyancy = 0.025

K = 10 #m/d

vka = 1

ne = 0.35

salinity = 35 #kg/m3 salinity of the sea

alpha_L = 10
alpha_T = 0

recharge = 360/365

Recharge_C = 0
```

In [3]:

```
# Create the basic MODFLOW model structure
modelname = 'henry'
workspace = '..'
swt = flopy.seawat.Seawat(modelname, exe_name=os.path.join('..','swt_v4x64'), model_w

henry_top = 500

henry_botm = np.linspace(henry_top - delz, 0, 50)

# Add DIS package to the MODFLOW model
dis = flopy.modflow.ModflowDis(swt, nlay = nlay, nrow = nrow, ncol = ncol, nper=1, del
                               delc=dely, laycbd=0, top=henry_top, steady = False,
                               botm=henry_botm, perlen=7300, nstp=730)


# Variables for the BAS package
ibound = np.ones((nlay, nrow, ncol), dtype=np.int32)
ibound[:,:,-1] = -1
bas = flopy.modflow.ModflowBas(swt, ibound, 1)
```

In [4]:

```
#LPF package:
ipakcb = 53

lpf = flopy.modflow.ModflowLpf(swt, hk=K, vka=K, ss = 0.0001, laytyp=0, ipakcb = ipakc
```

In [5]:

```
#Wel package:
dat = []
for i in range(0,nlay):
    dat.append((i,0,0,1))
stress_period_data = {}
stress_period_data[0] = dat
wel = flopy.modflow.ModflowWel(swt,stress_period_data = stress_period_data)
```

In [6]:

```
pcg = flopy.modflow.ModflowPcg(swt, hclose=1.0e-8)
```

In [7]:

```
oc = flopy.modflow.ModflowOc(
    swt,
    stress_period_data={(0, 0): ["save head", "save budget"]},
    compact=True,
)
```

MT3DMS and SEAWAT input files creation:

In [8]:

```
#Create MT3DMS model structure
timprs = np.linspace(365,7300,20)
initial_c = np.zeros((nlay,nrow,ncol))
initial_c[:,:,-1] = 35
btn = flopy.mt3d.Mt3dBtn(
    swt,
    nprs=20,
    timprs = timprs,
    prsity=0.35,
    #icbund = ibound,
    sconc=initial_c,
    ifmtcn=0,
    ncomp=1
)
adv = flopy.mt3d.Mt3dAdv(swt, mixelm=-1)

dsp = flopy.mt3d.Mt3dDsp(swt, al=alpha_L, trpt=0.1, trpv=0.1, dmcoef=5)

gcg = flopy.mt3d.Mt3dGcg(swt, iter1=5000, mxiter=1, isolve=1, cclose=1e-7)

from flopy import mt3d
itype = mt3d.Mt3dSsm.itype_dict()
print(itype)
ssm_data = {}
dat = []
for i in range(0,nlay):
    dat.append((i, 0, 0, 0, itype['WEL']))
    dat.append((i,0,ncol-1, 35, itype['BAS6']))

ssm_data[0] = dat

ssm = flopy.mt3d.Mt3dSsm(swt, stress_period_data = ssm_data)
```

```
{'CHD': 1, 'BAS6': 1, 'PBC': 1, 'WEL': 2, 'DRN': 3, 'RIV': 4, 'GHB': 5, 'MAS': 15, 'C
C': -1}
```

In [9]:

```
vdf = flopy.seawat.SeawatVdf(
    swt,
    mt3drhoflg = -1,
    mfnadvfd = 1,
    nswtcpl = 0,
    iwtable=0,
    denseref = 1000,
    drhodprhd = 0,
    prhdref =0,
    nsrhoeos = 1,
    mtrhospec = 1,
    drhodc = 0.7143,
    crhoref = 0,
    dt0 = 0.01
)
```

In [10]:

```
swt.check()
```

```
henry MODEL DATA VALIDATION SUMMARY:
  No errors or warnings encountered.

  Checks that passed:
    Unit number conflicts
    DIS package: zero or negative thickness
    DIS package: thin cells (less than checker threshold of 1.0)
    DIS package: nan values in top array
    DIS package: nan values in bottom array
    BAS6 package: isolated cells in ibound array
    BAS6 package: Not a number
    LPF package: zero or negative horizontal hydraulic conductivity values
    LPF package: zero or negative vertical hydraulic conductivity values
    LPF package: negative horizontal anisotropy values
    LPF package: horizontal hydraulic conductivity values below checker threshold of 1
e-11
    LPF package: horizontal hydraulic conductivity values above checker threshold of 1
00000.0
    LPF package: vertical hydraulic conductivity values below checker threshold of 1e-
11
    LPF package: vertical hydraulic conductivity values above checker threshold of 100
000.0
    LPF package: zero or negative specific storage values
    LPF package: specific storage values below checker threshold of 1e-06
    LPF package: specific storage values above checker threshold of 0.01
    LPF package: zero or negative specific yield values
    LPF package: specific yield values below checker threshold of 0.01
    LPF package: specific yield values above checker threshold of 0.5
    WEL package: BC indices valid
    WEL package: not a number (Nan) entries
    WEL package: BC in inactive cells
    SSM package: BC indices valid
    SSM package: not a number (Nan) entries
    SSM package: BC in inactive cells
```

Out[10]:

```
<flopy.utils.check.check at 0x172d7880d30>
```

Run Model:

In [11]:

```
swt.write_input()
```

In [12]:

```
success, buff = swt.run_model(silent=True, report=True)
if not success:
    raise Exception("SEAWAT did not terminate normally.")
```

In [13]:

```
import flopy.utils.binaryfile as bf
ucnobj = bf.UcnFile("MT3D001.UCN", model=swt)
times = ucnobj.get_times()
print(len(times))
times_2 = times
len(times_2)
concentration = ucnobj.get_data(totim=times[-1])
```

```
20
```

In [14]:

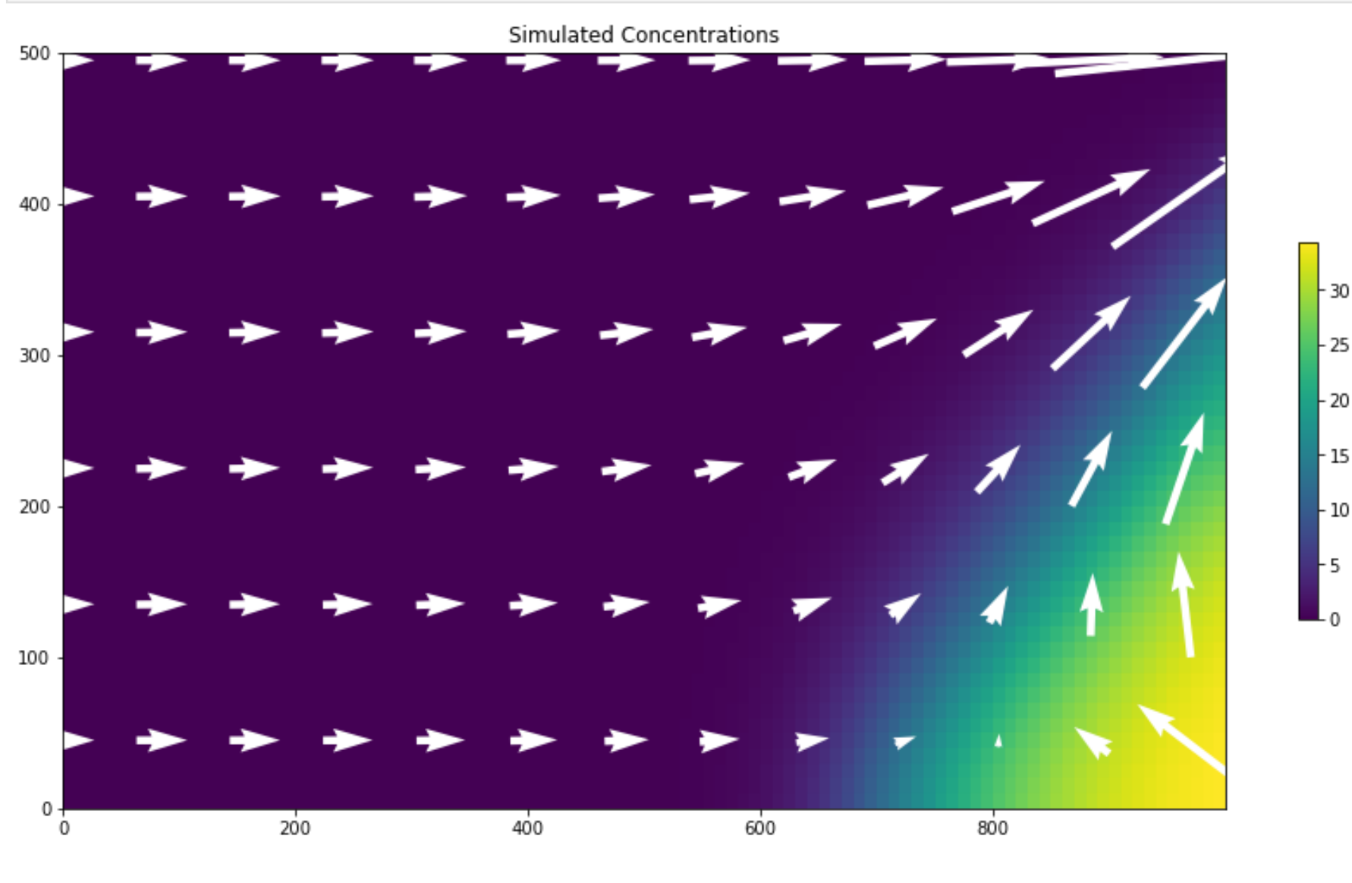
```
cbbobj = bf.CellBudgetFile("henry.cbc")
times = cbbobj.get_times()
print(times)
qx = cbbobj.get_data(text="flow right face", totim=times[-1])[0]
qy = np.zeros((nlay, nrow, int(ncol)), dtype=np.float32)
qz = cbbobj.get_data(text="flow lower face", totim=times[-1])[0]
```

```
[10.0]
```

Plotting the Concentration Results - Last Time Step

In [15]:

```
fig = plt.figure(figsize=(15,8))
ax = fig.add_subplot(1, 1, 1)
pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax, line={"row": 0})
arr = pmv.plot_array(concentration)
pmv.plot_vector(qx, qy, -qz, color="white", kstep=9, hstep=8)
plt.colorbar(arr, shrink=0.5, ax=ax)
ax.set_title("Simulated Concentrations");
```

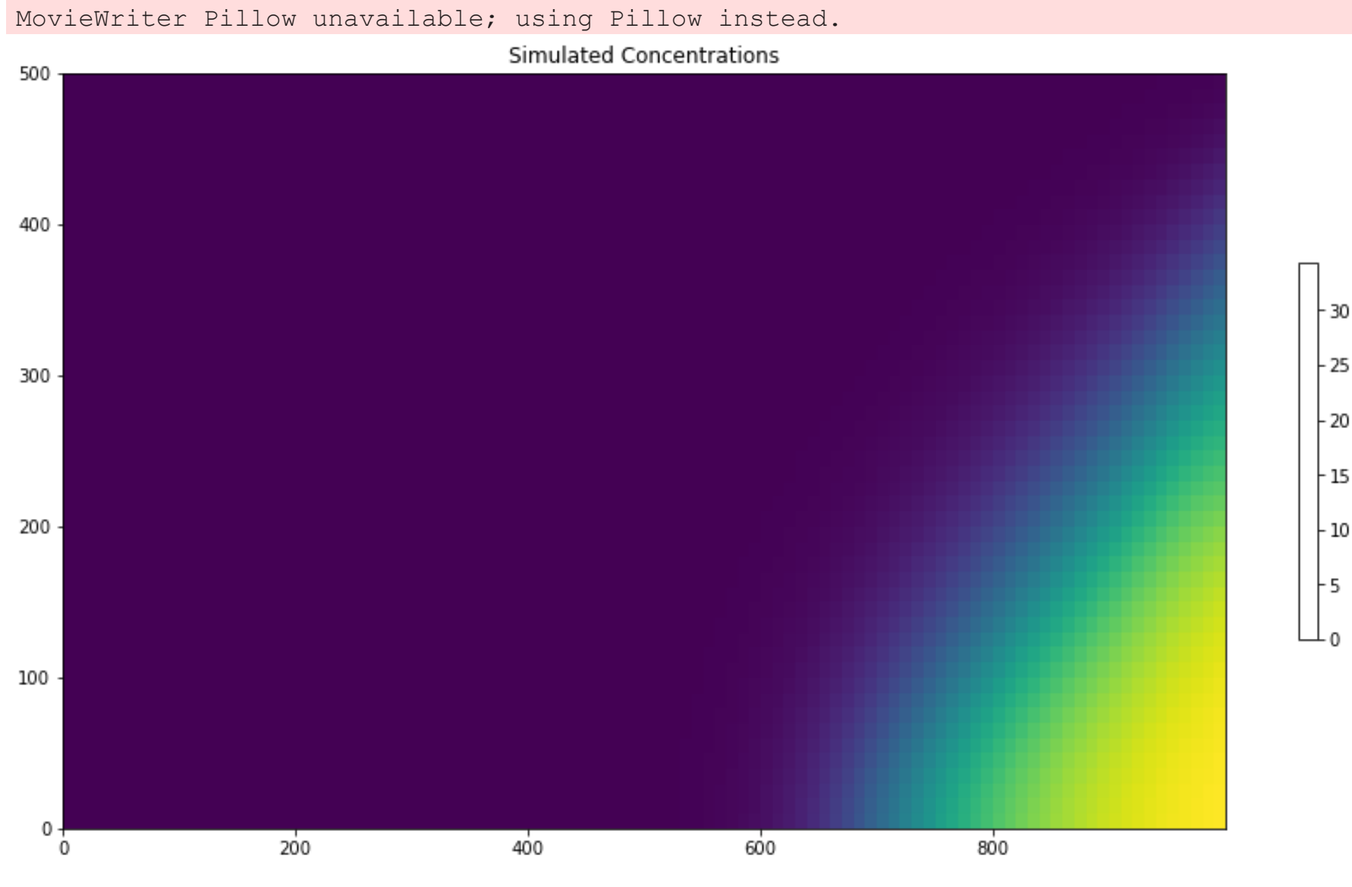


Animation of Concentration

In [16]:

```
fig = plt.figure(figsize=(15,8))
ax = fig.add_subplot(1, 1, 1)
pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax, line={"row": 0})
conc= ucnobj.get_data(totim=times_2[-1])
arr = pmv.plot_array(conc)
#arr = pmv.plot_array(concentration)
plt.colorbar(arr, shrink=0.5, ax=ax)
ax.set_title("Simulated Concentrations")

from celluloid import Camera
camera = Camera(fig)


# funciton takes frame as an input
for frame in range(1,len(times_2)):

    conc= ucnobj.get_data(totim=times_2[frame])
    #pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax, line={"row": 0})
    arr = pmv.plot_array(conc)
    #plt.colorbar(arr, shrink=0.5, ax=ax)
    #ax.set_title("Simulated Concentrations - "+ str(times[frame]/365) + "years")
    camera.snap()


animation = camera.animate()


animation.save('simulated_concentrations_henry.gif', writer = 'Pillow')

plt.close()
```

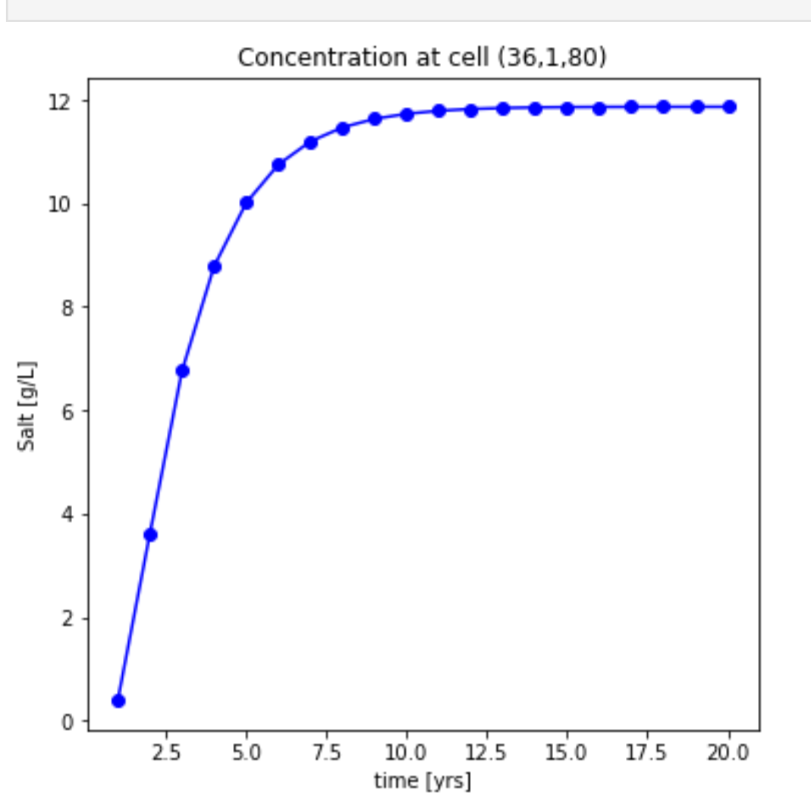
```
MovieWriter Pillow unavailable; using Pillow instead.
```

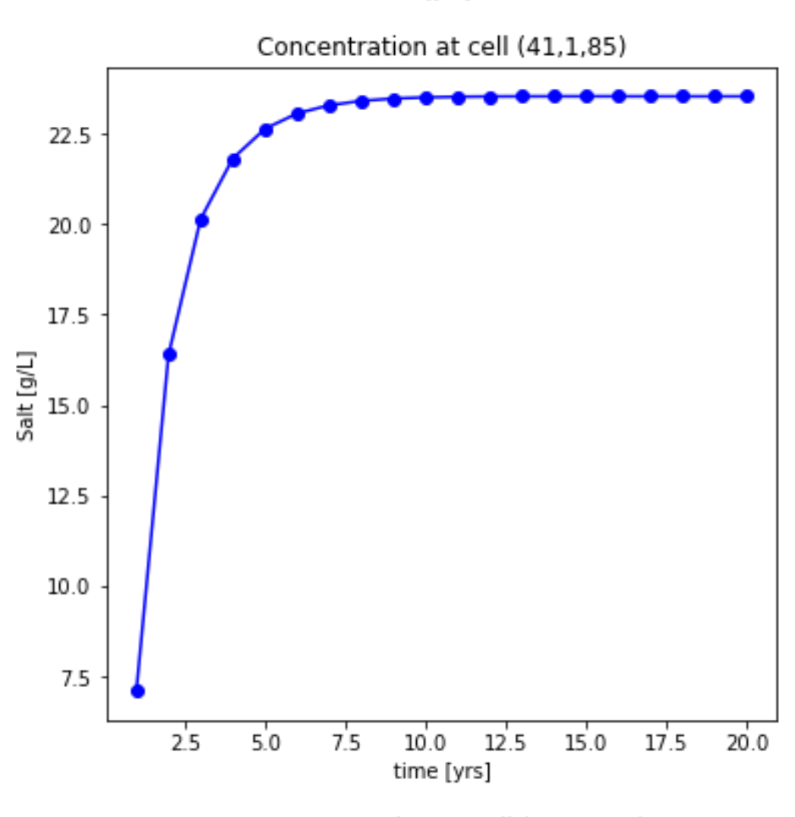


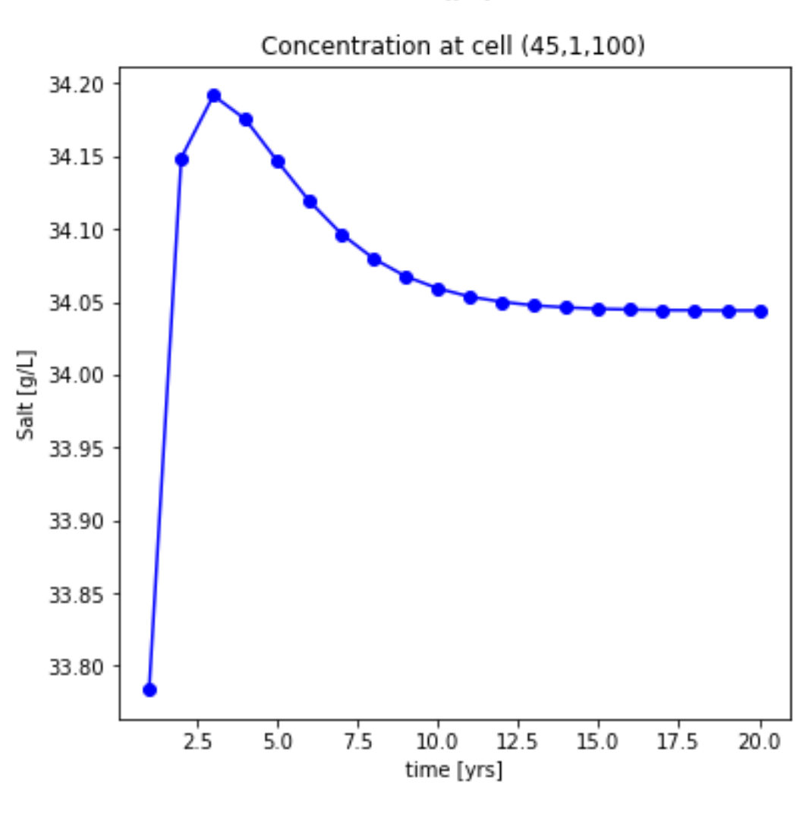
GIF image

In [17]:

```
points_coord = [(795,0.99,35),(845,0.99,40),(995,0.99,44)]
points = []
for x,y,lay in points_coord:
    r, c = dis.get_rc_from_node_coordinates(x, y)
    point = (lay, r, c)
    points.append(point)

for point in points:
    ts = ucnobj.get_ts(point)
    fig = plt.figure(figsize=(6, 6))
    ax = fig.add_subplot(1, 1, 1)
    ttl = "Concentration at cell ({0},{1},{2})".format(point[0] + 1, point[1] + 1, po
    ax.set_title(ttl)
    ax.set_xlabel("time [yrs]")
    ax.set_ylabel("Salt [g/L]")
    ax.plot(ts[:, 0]/365, ts[:, 1], "bo-")
```



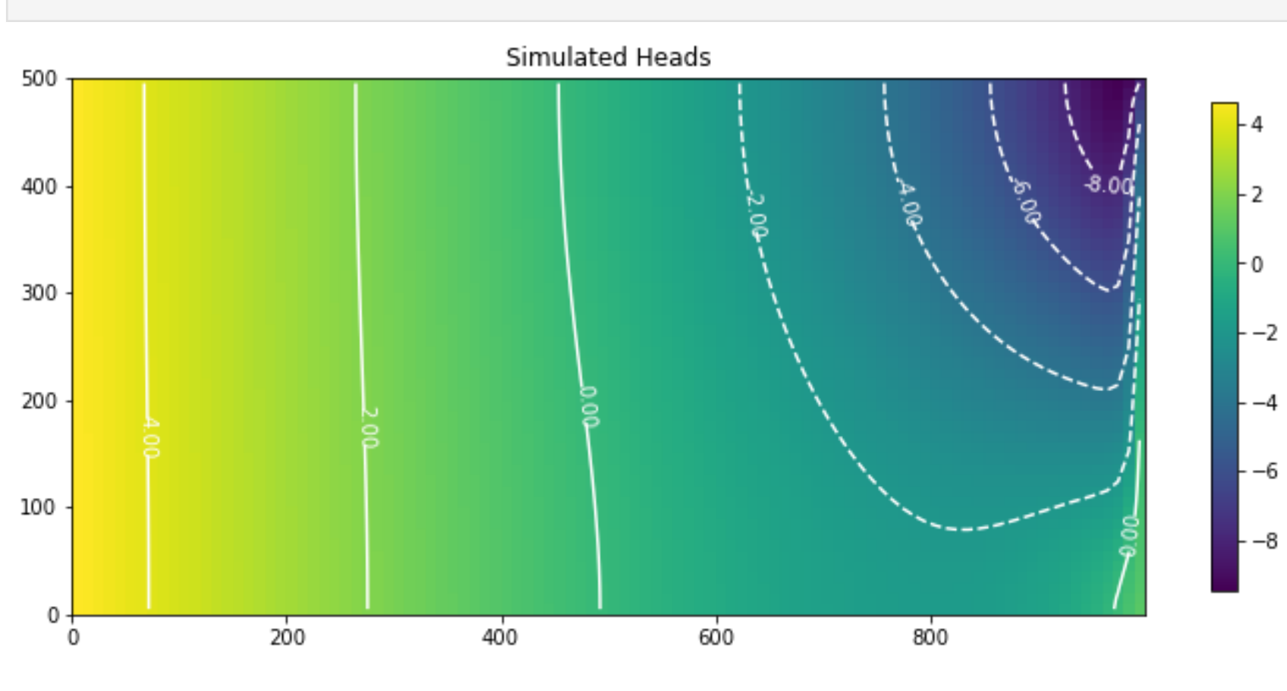




Simulated Heads Results

In [18]:

```
headobj = bf.HeadFile("henry.hds")
times = headobj.get_times()
head = headobj.get_data(totim=times[-1])

fig = plt.figure(figsize=(12, 9))
ax = fig.add_subplot(1, 1, 1, aspect="equal")
pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax, line={"row": 0})
arr = pmv.plot_array(head)
contours = pmv.contour_array(head, colors="white")
ax.clabel(contours, fmt="%2.2f")
plt.colorbar(arr, shrink=0.5, ax=ax)
ax.set_title("Simulated Heads");
```



Adding Abstraction Well

- For the abstraction well exercise, we will add a well 250 m from shore and with 50 m depth.
- The abstraction rate will be of 10 m3/d/m'
- We are considering the same time-steps and boundary conditions from the current model

In [19]:

```
well_col = ncol - (25+1) #Location of 0-indexed col of well 250 m from shore
well_row = 0 # Only row in the model
well_layer = 4 # Well located 50 m from ground elevation (layer 4 in 0-indexed form)
well_pumping_rate = -10
well_data_pumping = [(well_layer, well_row, well_col, well_pumping_rate)] # Wells pla
# and pumping rate of 5 m3/d

stress_period_data[0] = stress_period_data[0] + well_data_pumping

## Updating the wel package to the model:

wel = flopy.modflow.ModflowWel(swt, stress_period_data = stress_period_data)
#Create MT3DMS model structure
btn = flopy.mt3d.Mt3dBtn(
    swt,
    nprs=20,
    timprs = timprs,
    prsity=0.35,
    #icbund = ibound,
    sconc=initial_c,
    ifmtcn=0,
    ncomp=1
)
adv = flopy.mt3d.Mt3dAdv(swt, mixelm=-1)

dsp = flopy.mt3d.Mt3dDsp(swt, al=alpha_L, trpt=0.1, trpv=0.1, dmcoef=5)

gcg = flopy.mt3d.Mt3dGcg(swt, iter1=5000, mxiter=1, isolve=1, cclose=1e-7)

ssm = flopy.mt3d.Mt3dSsm(swt, stress_period_data = ssm_data)


#swt.check()
```

In [20]:

```
swt.write_input()

success, buff = swt.run_model(silent=True, report=True)
if not success:
    raise Exception("SEAWAT did not terminate normally.")

ucnobj = bf.UcnFile("MT3D001.UCN", model=swt)
times = ucnobj.get_times()
```

In [ ]:

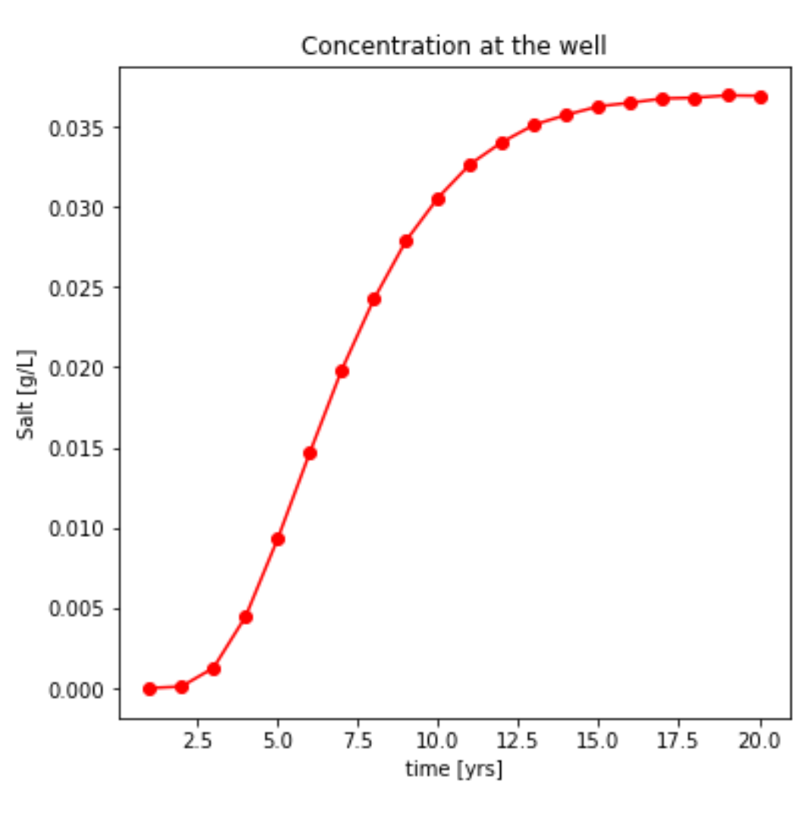
Plotting the concentration in the well with time:

In [21]:

```
point = (well_layer, well_row, well_col)
ts = ucnobj.get_ts(point)
fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(1, 1, 1)
ttl = "Concentration at the well"
ax.set_title(ttl)
ax.set_xlabel("time [yrs]")
ax.set_ylabel("Salt [g/L]")
ax.plot(ts[:, 0]/365, ts[:, 1], "ro-")
```

Out[21]:

```
[<matplotlib.lines.Line2D at 0x172db377eb0>]
```



- Concentration in the well increases with the years. This means that the salt-water plume is rising towards the well. Concentration is stabilized aroudn 0.035 g/L of salt.
- This value is adequate for drinking water and could be considered a supply value.
- This value is of concern for people with blood pressure conditions, that should avoid water with salt in concentrations above 20 mg/l.

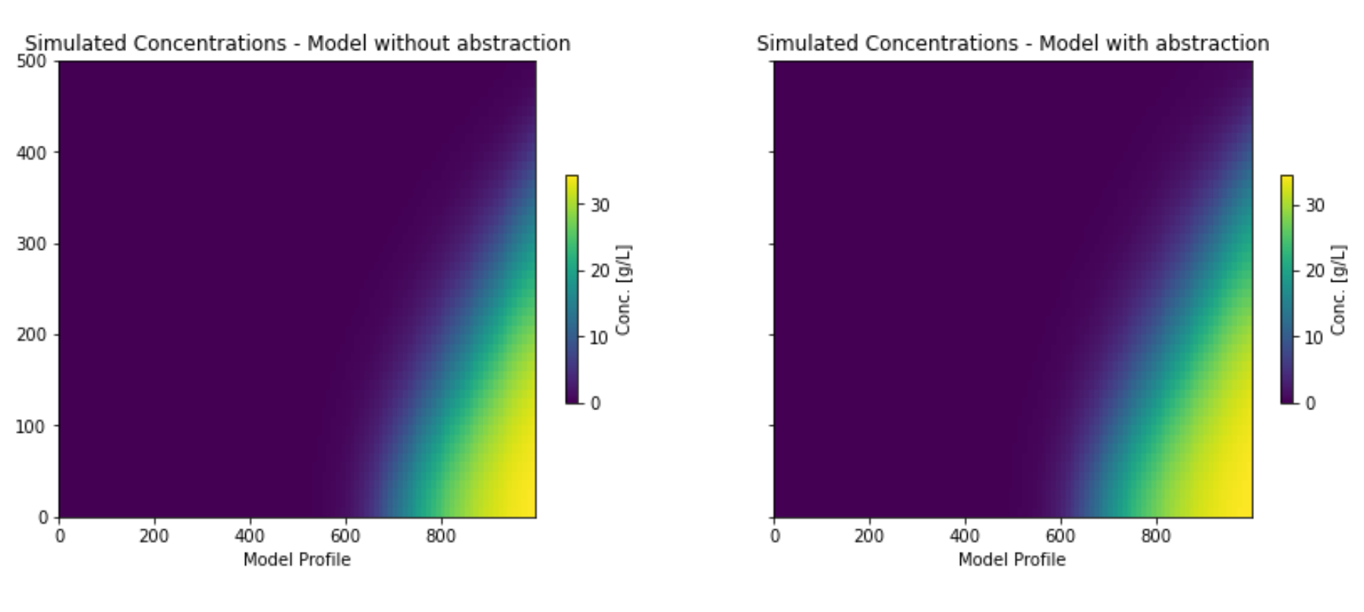
Concentrations last time step with and without well:

In [23]:

```
fig, ax = plt.subplots(1, 2, sharey = True, sharex = True, figsize = (14,5))
pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax[0], line={"row": 0})
arr = pmv.plot_array(concentration)
fig.colorbar(arr, shrink=0.5, ax=ax[0], label = "Conc. [g/L]")
ax[0].set_xlabel("Model Profile")
ax[0].set_title("Simulated Concentrations - Model without abstraction")

concentration_2 = ucnobj.get_data(totim=times[-1])

pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax[1], line={"row": 0})
arr = pmv.plot_array(concentration_2)
plt.colorbar(arr, shrink=0.5, ax=ax[1], label = "Conc. [g/L]")
ax[1].set_xlabel("Model Profile")
ax[1].set_title("Simulated Concentrations - Model with abstraction")
```

Out[23]:

```
Text(0.5, 1.0, 'Simulated Concentrations - Model with abstraction')
```



- We can see the comparison between the model without abstraction and with abstraction in the figure above. The salt-water plume is significantly bigger and higher in the scenario with pumping.

Testing higher pumping rate:

In [24]:

```
well_col = ncol - (25+1) #Location of 0-indexed col of well 250 m from shore
well_row = 0 # Only row in the model
well_layer = 4 # Well located 50 m from ground elevation (layer 4 in 0-indexed form)
well_pumping_rate = -20
well_data_pumping = [(well_layer, well_row, well_col, well_pumping_rate)] # Wells pla
# and pumping rate of 5 m3/d

stress_period_data[0] = stress_period_data[0] + well_data_pumping

## Updating the wel package to the model:

wel = flopy.modflow.ModflowWel(swt, stress_period_data = stress_period_data)
#Create MT3DMS model structure
btn = flopy.mt3d.Mt3dBtn(
    swt,
    nprs=20,
    timprs = timprs,
    prsity=0.35,
    #icbund = ibound,
    sconc=initial_c,
    ifmtcn=0,
    ncomp=1
)
adv = flopy.mt3d.Mt3dAdv(swt, mixelm=-1)

dsp = flopy.mt3d.Mt3dDsp(swt, al=alpha_L, trpt=0.1, trpv=0.1, dmcoef=5)

gcg = flopy.mt3d.Mt3dGcg(swt, iter1=5000, mxiter=1, isolve=1, cclose=1e-7)

ssm = flopy.mt3d.Mt3dSsm(swt, stress_period_data = ssm_data)


#swt.check()
```

In [25]:

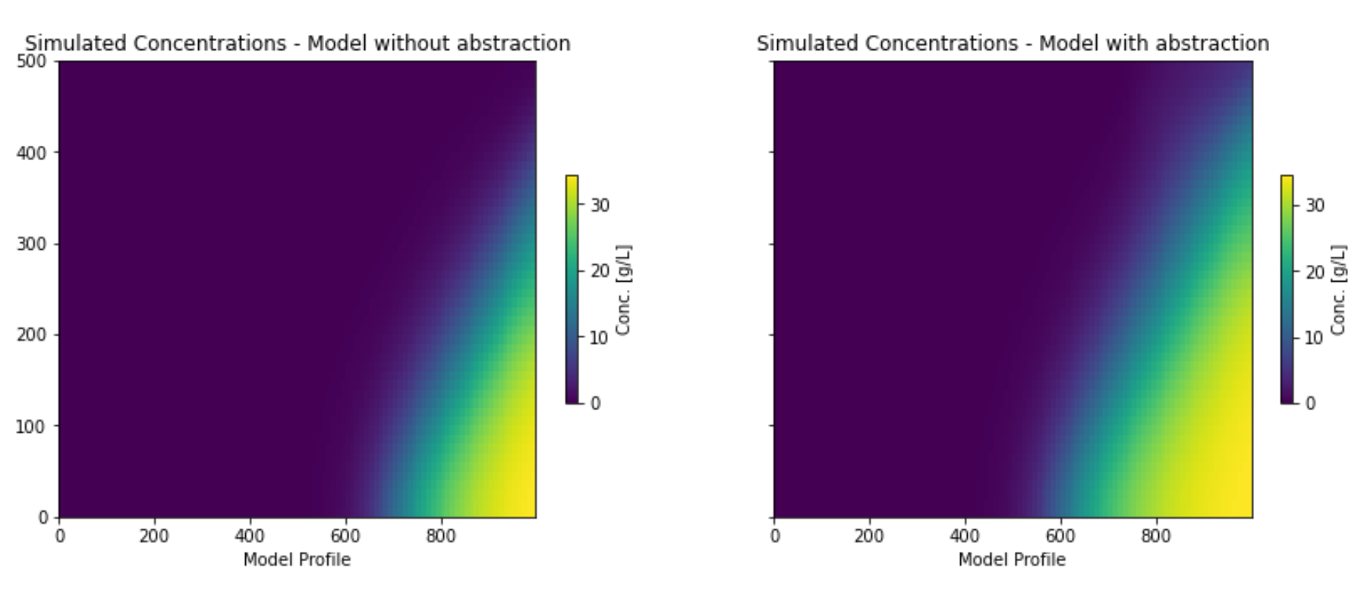
```
swt.write_input()

success, buff = swt.run_model(silent=True, report=True)
if not success:
    raise Exception("SEAWAT did not terminate normally.")

ucnobj = bf.UcnFile("MT3D001.UCN", model=swt)
times = ucnobj.get_times()
```

In [26]:

```
fig, ax = plt.subplots(1, 2, sharey = True, sharex = True, figsize = (14,5))
pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax[0], line={"row": 0})
arr = pmv.plot_array(concentration)
fig.colorbar(arr, shrink=0.5, ax=ax[0], label = "Conc. [g/L]")
ax[0].set_xlabel("Model Profile")
ax[0].set_title("Simulated Concentrations - Model without abstraction")

concentration_2 = ucnobj.get_data(totim=times[-1])

pmv = flopy.plot.PlotCrossSection(model=swt, ax=ax[1], line={"row": 0})
arr = pmv.plot_array(concentration_2)
plt.colorbar(arr, shrink=0.5, ax=ax[1], label = "Conc. [g/L]")
ax[1].set_xlabel("Model Profile")
ax[1].set_title("Simulated Concentrations - Model with abstraction")
```

Out[26]:

```
Text(0.5, 1.0, 'Simulated Concentrations - Model with abstraction')
```



- In the Figure above, we see that the model with abstraction has now higher salinity in the upper left part, with coloring indicating that the salinity is near 10 g/L in the surface. This indicates that the new abstraction rates are producing too much upconing of saltwater.