CE 5364 Groundwater Transport Phenemona
Fall 2025 Exercise Set 5

LAST NAME, FIRST NAME

R00000000

Purpose :

Apply selected analytical models for reactive transport

Assessment Criteria :

Completion, results plausible, format correct, example calculations shown.

Problem 1 (Problem 6-9, pg. 589)

An unintentional discharge from a point source introduced $10\ kg$ of contaminant mass to an aquifer. The seepage velocity is $0.1\ \frac{ft}{day}$ in the $+x$ direction. The longitudinal dispersion coefficient is $D_x = 0.01\frac{ft^2}{day}$ the transverse dispersion coefficients are $D_y = D_z = 0.001\frac{ft^2}{day}$.

Determine:

1. Calculate the maximum concentration at $x = 100\ ft$ and $t = 5\ years$.
2. Calculate the concentration at $(x, y, z, t) = (200\ ft, 5\ ft, 2\ ft, 5\ years)$

governing principles

point source spill; use Equation 6.28 in book

The corollary equation in 3-D from a point source was derived by Baetsle (1969) and can sometimes be used to represent the sudden release from a single source or tank in the subsurface. Baetsle's model gives the following equation:

$$C(x,y,z,t) = \frac{C_0 V_0}{8(\pi t)^{3/2}\left(D_x D_y D_z\right)^{1/2}} \exp\left[-\frac{(x-vt)^2}{4D_x t} - \frac{y^2}{4D_y t} - \frac{z^2}{4D_z t} - \lambda t\right] \qquad (6.28)$$

where C_0 is the original concentration; V_0 is the original volume so that C_0V_0 is the mass involved in the spill; D_x, D_y, D_z are the coefficients of hydrodynamic dispersion; v is the velocity of the contaminant; λ is the first order decay constant for a radioactive substance. For a nonradioactive substance, the term λt is ignored.

solution details (e.g. step-by-step computations)

1. Create a prototype function

In [18]:
```
def c3addinst(x,y,z,t,m,dx,dy,dz,v,lm):
# Baetsle 1969 model
    import math
    term0 = math.exp(-1.0*lm*t)
    term1 = 8.0*math.sqrt(math.pi*dx*t*math.pi*dy*t*math.pi*dz*t)
    term2 = math.exp(-((x-v*t)**2)/(4.0*dx*t) -((y)**2)/(4.0*dy*t) -((z)**2)/(4.0*d
    c3addinst = term0*(mass/term1)*term2
    return(c3addinst)
```

2. Build input data manager, report intermediate computations

In [19]:
```
mass         = 10.0 #kg
velocity     = 0.1 #ft/day
disp_x = 0.01  #ft^2/day
disp_y = 0.001 #ft^2/day
disp_z = 0.001 #ft^2/day

print("         Mass : ",round(mass,3)," kg/m^3")
print("Pore velocity : ",round(velocity,3)," ft/day")
print(" Dispersion x : ",round(disp_x,3)," ft^2/day")
print(" Dispersion y : ",round(disp_y,3)," ft^2/day")
print(" Dispersion z : ",round(disp_z,3)," ft^2/day")
```

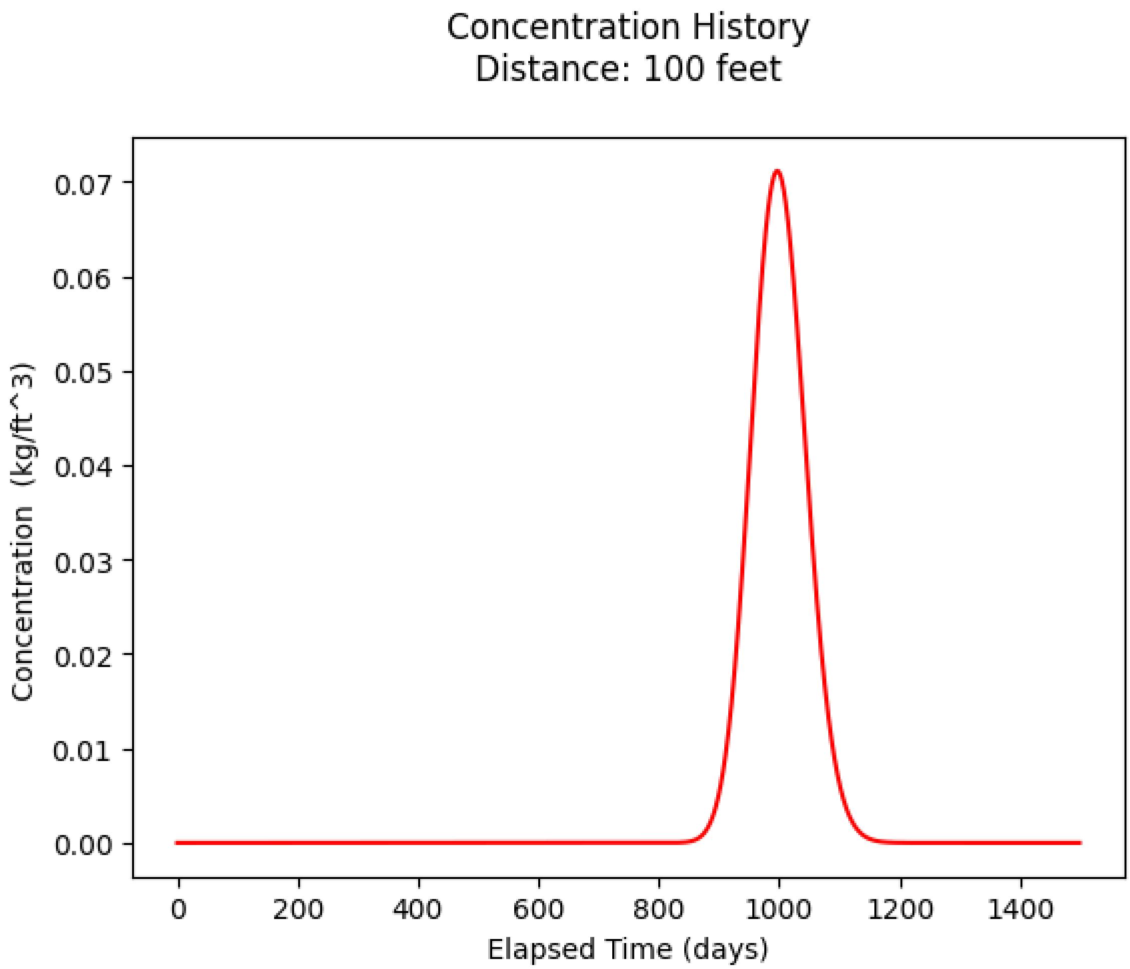
```
         Mass :  10.0  kg/m^3
Pore velocity :  0.1  ft/day
 Dispersion x :  0.01  ft^2/day
 Dispersion y :  0.001  ft^2/day
 Dispersion z :  0.001  ft^2/day
```

3. Calculate the maximum concentration at $x = 100\ ft$

- use a history plot

In [20]:
```
deltat       = (1.0) #days
howmany =      1500/deltat
```

```python
howmany = int(howmany)

t = [] #days
for i in range(howmany):
    t.append(float(i)*deltat)
    if t[i] == 0: # trap zero time to prevent divide by zero
        t[i]= 0.00000001

x         = 100  #ft
y         = 0
z         = 0

c = [0 for i in range(howmany)] #concentration

for i in range(howmany):
    c[i]=c3addinst(x,y,z,t[i],mass,disp_x,disp_y,disp_z,velocity,0)



#
# Import graphics routines for picture making
#
from matplotlib import pyplot as plt
#
# Build and Render the Plot
#
plt.plot(t,c, color='red', linestyle = 'solid')  # make the plot object
plt.title(" Concentration History \n Distance: " + repr(x) + " feet \n" ) # caption
plt.xlabel(" Elapsed Time (days) ") # label x-axis
plt.ylabel(" Concentration  (kg/ft^3)      ") # label y-axis
plt.show() # plot to stdio -- has to be last call as it kills prior objects
plt.close('all') # needed when plt.show call not invoked, optional here
```



In [21]:
```
print("Maximum concentration : ",max(c)," kg/ft^3")
print("          Observed at : ",t[c.index(max(c))]," days")
```

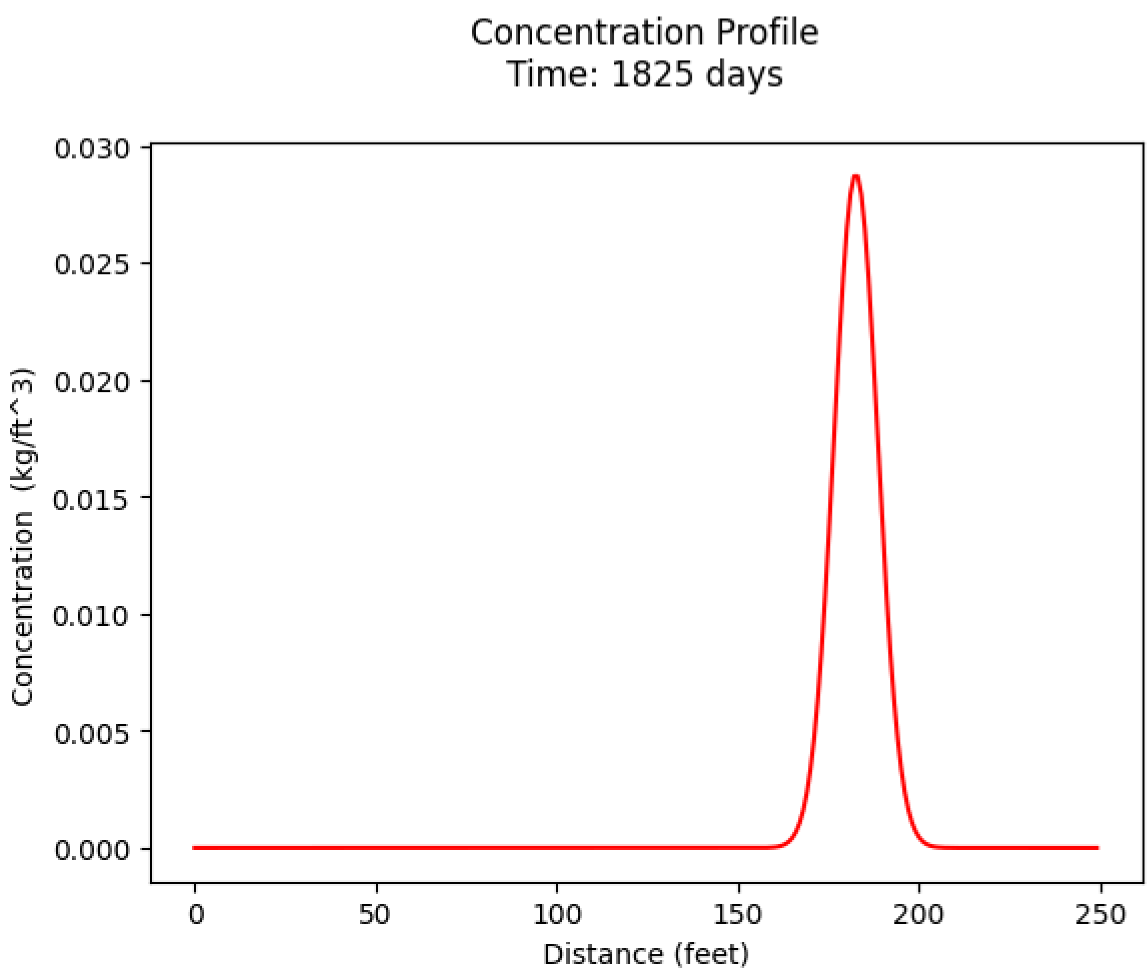
```
Maximum concentration :  0.07114794548155023  kg/ft^3
          Observed at :  997.0  days
```

4. Calculate the maximum concentration at $t = 5\ years$

- use a profile plot

In [22]:
```
deltax      = (1.0) #days
howmany =     250/deltax
howmany = int(howmany)

x = [] #feet
for i in range(howmany):
    x.append(float(i)*deltax)


t         = 5*365  #days
y         = 0
z         = 0

c = [0 for i in range(howmany)] #concentration
```

```python
for i in range(howmany):
    c[i]=c3addinst(x[i],y,z,t,mass,disp_x,disp_y,disp_z,velocity,0)


#
# Import graphics routines for picture making
#
from matplotlib import pyplot as plt
#
# Build and Render the Plot
#
plt.plot(x,c, color='red', linestyle = 'solid')  # make the plot object
plt.title(" Concentration Profile \n Time: " + repr(t) + " days \n" ) # caption the
plt.xlabel(" Distance (feet) ") # label x-axis
plt.ylabel(" Concentration  (kg/ft^3)      ") # label y-axis
plt.show() # plot to stdio -- has to be last call as it kills prior objects
plt.close('all') # needed when plt.show call not invoked, optional here
```



In [23]:
```python
print("Maximum concentration : ",max(c)," kg/ft^3")
print("          Observed at : ",x[c.index(max(c))]," feet")
```

```
Maximum concentration :  0.02869482568927895  kg/ft^3
          Observed at :  182.0  feet
```

5. Calculate the concentration at $(x, y, z, t) = (200\ ft, 5\ ft, 2\ ft, 5\ years)$

In [24]:
```
conc = c3addinst(200,5,2,5*365,mass,disp_x,disp_y,disp_z,velocity,0)
print("C(200,5,2,5) : ",round(conc,7)," kg/ft^3")
```

```
C(200,5,2,5) :  8.2e-06  kg/ft^3
```

Discussion

Direct application of impulse model. Observe weird units - probably would convert cubic feet to cubic meter equivalents - so output would be kg/m^3

Problem 2 (Problem 6-10, pg. 589)

Apply the Domenico and Schwartz (1998) planar source model (pg. XXX) to a case of a continuous source that has been leaking contaminant into an aquifer for 15 years. The source had a width $Y = 6\ m$ and depth $Z = 6\ m$. The source concentration is $10\ \frac{mg}{l}$. The seepage velocity is $0.057\ \frac{m}{day}$. The longitudinal, transverse, and vertical dispervities are $1\ m$, $0.1\ m$, and $0.01\ m$ respectively.

Determine:

1. The contaminant concentration history at a location $x = 200\ m$ from the source using 1-year increments for 30 years.

sketch(s)

Then:

- list known quantities
- list unknown quantities
- governing principles: Using the planar source model (Eqn 6.31)

$$\frac{C(x,y,z,t)}{C_0} = \left(\frac{1}{8}\right) \mathrm{erfc}\left[\frac{(x-vt)}{2(\alpha_x vt)^{1/2}}\right] \left\{\mathrm{erf}\left[\frac{\left(y+\frac{Y}{2}\right)}{2(\alpha_y x)^{1/2}}\right] - \mathrm{erf}\left[\frac{\left(y-\frac{Y}{2}\right)}{2(\alpha_y x)^{1/2}}\right]\right\} \left\{\mathrm{erf}\left[\frac{(z+Z)}{2(\alpha_z x)^{1/2}}\right] - \mathrm{erf}\left[\frac{(z-Z)}{2(\alpha_z x)^{1/2}}\right]\right\} \qquad (6.31)$$

solution details (e.g. step-by-step computations)

1. Usual procedure, first a prototype function - unlike prior cases will use dispersivities rather than dispersion coefficients:

In [25]:
```python
def c3dad(conc0, distx, disty, distz, lenX, lenY, lenZ, dispx, dispy, dispz, veloci
    import math
    from scipy.special import erf, erfc # scipy needs to already be loaded into the
    # Constant of integration
    term1 = conc0 / 8.0

    # Centerline axis solution
    arg1 = (distx - velocity*etime) / (2*math.sqrt(dispx*velocity*etime)) #dispx is
    term2 = erfc(arg1)

    # Off-axis solution, Y direction
#    arg2 = 2.0 * math.sqrt(dispy*distx / velocity)
    arg2 = 2.0 * math.sqrt(dispy*distx) #dispy is dispersivity
    arg3 = disty + 0.5*lenY
    arg4 = disty - 0.5*lenY
    term3 = erf(arg3 / arg2) - erf(arg4 / arg2)

    # Off-axis solution, Z direction
#    arg5 = 2.0 * math.sqrt(dispz*distx / velocity)
    arg5 = 2.0 * math.sqrt(dispz*distx) #dispz is dispersivity
    arg6 = distz + 0.5*lenZ
    arg7 = distz - 0.5*lenZ
    term4 = erf(arg6 / arg5) - erf(arg7 / arg5)

    # Convolve the solutions
    c3dad = term1 * term2 * term3 * term4
    return c3dad
```

2. Now an input manager section

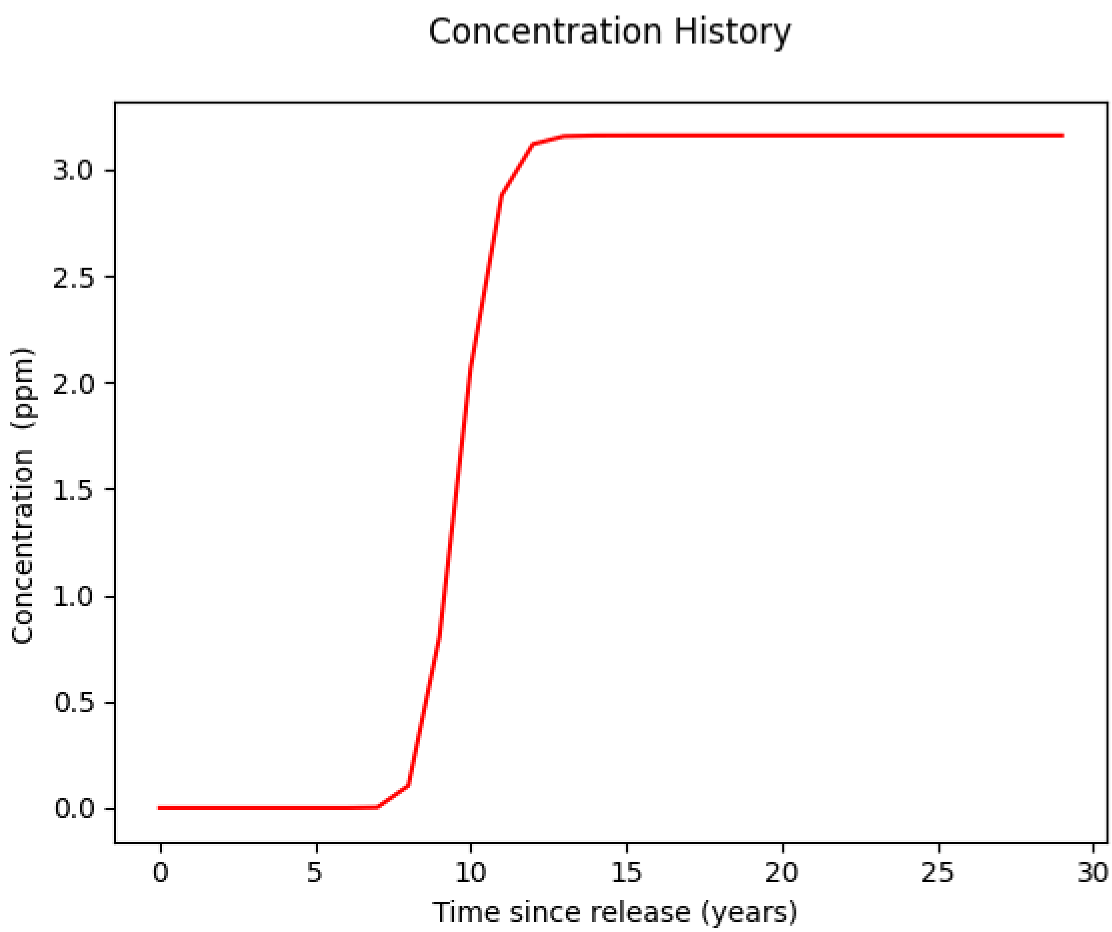
In [26]:
```python
# inputs
conco = 10.0
velocity = 0.057
dispersivity_x = 1.0
dispersivity_y = 0.1
dispersivity_z = 0.01
width_y = 6.0
width_z = 6.0
xloc = 200.0
yloc = 0.0 # not explicit in problem statement
zloc = 0.0
time = 30*365 #years as days
# echo inputs
print("Source Concentration : ",round(conco,3)," ppm ")
print("            Velocity : ",round(velocity,3)," m/sec ")
print("     Dispersivity_x  : ",round(dispersivity_x,3)," m ")
print("     Dispersivity_y  : ",round(dispersivity_x,3)," m ")
print("     Dispersivity_z  : ",round(dispersivity_x,3)," m ")
print("             Width Y : ",round(width_y,3)," m ")
print("             Width Z : ",round(width_z,3)," m ")
```

```
Source Concentration :  10.0  ppm
            Velocity :  0.057  m/sec
     Dispersivity_x  :  1.0  m
     Dispersivity_y  :  1.0  m
     Dispersivity_z  :  1.0  m
             Width Y :  6.0  m
             Width Z :  6.0  m
```

3. Now build script for concentration history (time is the variable)

In [27]:

```
#
# forward define and initialize vectors for a profile plot
#
how_many_points = 30
deltat = time/how_many_points
t = [i*deltat for i in range(how_many_points)] # constructor notation
c = [0.0 for i in range(how_many_points)]      # constructor notation

t[0]=1e-5 #cannot have zero time, so use really small value first position in list
#
# build the profile predictions
#
for i in range(0,how_many_points,1):
    c[i] = c3dad(conco, xloc, yloc, zloc, 0, width_y, width_z, dispersivity_x, disp
for i in range(0,how_many_points,1):
    t[i]=t[i]/365 # days as years
#
# Import graphics routines for picture making
#
from matplotlib import pyplot as plt
#
# Build and Render the Plot
#
plt.plot(t,c, color='red', linestyle = 'solid')  # make the plot object
plt.title(" Concentration History \n  ") # caption the plot object
plt.xlabel(" Time since release (years)") # label x-axis
plt.ylabel(" Concentration  (ppm)       ") # label y-axis
#plt.savefig("ogatabanksplot.png") # optional generates just a plot for embedding i
plt.show() # plot to stdio -- has to be last call as it kills prior objects
plt.close('all') # needed when plt.show call not invoked, optional here
#sys.exit() # used to elegant exit for CGI-BIN use
```



discussion

The equilibrium concentration can be found from the plot either by finding the maximum in the list or just taking the last element in the list.

In [28]:
```
print("Equilibrium concentration @ x=200,y=0,z=0,t->big : ",round(max(c),3)," ppm "
```

```
Equilibrium concentration @ x=200,y=0,z=0,t->big :  3.16  ppm
```