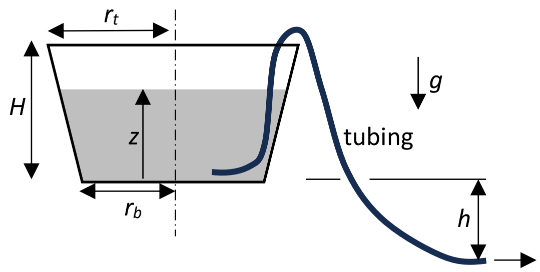
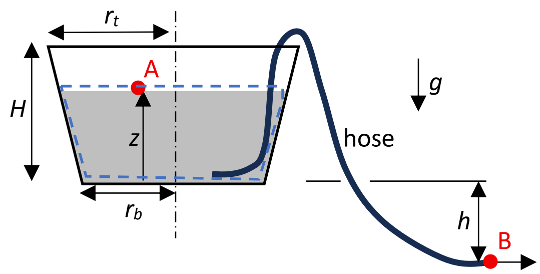
We wish to estimate the time it will take to drain a bucket. The bucket has a circular cross-section with a radius that varies linearly with the height (i.e., a conical bucket). The bucket is initially filled to a depth z_0. Plastic tubing is used to drain the bucket and has a total length L and internal diameter d. The tubing discharges into the atmosphere a distance h below the bottom of the bucket. Assume the water is at 60 °F when draining.

How long will it take for the water in the bucket to reach a final height of z_f?



$r_t = 5\,{}^5/_8$ in.
$r_b = 5$ in.
$H = 14$ in.
$z_0 = 10$ in.
$z_f = 2$ in.
$h = 8\,½$ in.
$L = 46\,{}^5/_8$ in.
$d = {}^1/_4$ in. (inner diameter)
$g = 32.2$ ft/s^2

SOLUTION:



As the bucket drains, the water free surface height decreases. Use Conservation of Mass applied to a control volume surrounding all of the water in the bucket and tubing to determine this height,

$$\frac{d}{dt}\int_{CV}\rho dV + \int_{CS}\rho \boldsymbol{u}_{rel}\cdot d\boldsymbol{A} = 0, \tag{1}$$

where, assuming water is incompressible,

$$\frac{d}{dt}\int_{CV}\rho dV = \frac{d}{dt}(\rho V) = \rho\frac{dV}{dt} \quad \text{(assume the mass of water in the tubing << mass of water in bucket)}, \tag{2}$$

$$\int_{CS}\rho \boldsymbol{u}_{rel}\cdot d\boldsymbol{A} = \rho\bar{V}_B\frac{\pi d^2}{4}. \tag{3}$$

Substitute and simplify,

$$\rho\frac{dV}{dt} + \rho\bar{V}_B\frac{\pi d^2}{4} = 0, \tag{4}$$

$$\frac{dV}{dt} = -\bar{V}_B\frac{\pi d^2}{4}. \tag{5}$$

To find the volume of the conical bucket as a function of fill height z,

$$V = \int_{z=0}^{z=z}\pi r^2 dz, \tag{6}$$

where,

$$r = r_b + \left(\frac{r_t - r_b}{H}\right)z. \tag{7}$$

Substituting, integrating, and simplifying,

$$V = \int_0^z \pi\left[r_b + \left(\frac{r_t-r_b}{H}\right)z\right]^2 dz = \int_0^z \pi\left[r_b^2 + 2r_b\left(\frac{r_t-r_b}{H}\right)z + \left(\frac{r_t-r_b}{H}\right)^2 z^2\right]dz \tag{8}$$

$$V = \pi\left[r_b^2 z + r_b\left(\frac{r_t-r_b}{H}\right)z^2 + \frac{1}{3}\left(\frac{r_t-r_b}{H}\right)^2 z^3\right]. \tag{9}$$

Now differentiate with respect to time in order to substitute into Eq. (5),

$$\frac{dV}{dt} = \pi\left[r_b^2\frac{dz}{dt} + r_b\left(\frac{r_t-r_b}{H}\right)\left(2z\frac{dz}{dt}\right) + \frac{1}{3}\left(\frac{r_t-r_b}{H}\right)^2\left(3z^2\frac{dz}{dt}\right)\right], \tag{10}$$

$$\frac{dV}{dt} = \pi \frac{dz}{dt}\left[r_b^2 + 2zr_b\left(\frac{r_t - r_b}{H}\right) + z^2\left(\frac{r_t - r_b}{H}\right)^2\right]. \quad (11)$$

Substitute Eq. (11) into Eq. (5) and simplify,

$$\pi \frac{dz}{dt}\left[r_b^2 + 2zr_b\left(\frac{r_t - r_b}{H}\right) + z^2\left(\frac{r_t - r_b}{H}\right)^2\right] = -\bar{V}_B \frac{\pi d^2}{4}, \quad (12)$$

$$\frac{dz}{dt} = -\bar{V}_B\left[\frac{d^2/4}{r_b^2 + 2zr_b\left(\frac{r_t - r_b}{H}\right) + z^2\left(\frac{r_t - r_b}{H}\right)^2}\right], \quad (13)$$

To find the water speed in the tubing, apply the Extended Bernoulli Equation from point A to point B in the figure,

$$\left(\frac{p}{\rho g} + \alpha\frac{\bar{V}^2}{2g} + z\right)_B = \left(\frac{p}{\rho g} + \alpha\frac{\bar{V}^2}{2g} + z\right)_A - H_L + H_S, \quad (14)$$

where,

$$p_A = p_B = p_{\text{atm}}, \quad (15)$$

$$\bar{V}_A \ll \bar{V}_B, \quad (16)$$

$$z_A = z \text{ and } z_B = -h, \quad (17)$$

$$H_L = \left(K_{\text{inlet}} + K_{\text{major}}\right)\frac{\bar{V}_B^2}{2g}, \quad (18)$$

$$H_S = 0. \quad (19)$$

Substitute and simplify,

$$\alpha_B\frac{\bar{V}_B^2}{2g} - h = z - \left(K_{\text{inlet}} + K_{\text{major}}\right)\frac{\bar{V}_B^2}{2g}, \quad (20)$$

$$\bar{V}_B^2 = \frac{2g(z+h)}{\alpha_B + K_{\text{inlet}} + K_{\text{major}}}, \quad (21)$$

$$\bar{V}_B^2 = \frac{2g(z+h)}{\alpha_B + K_{\text{inlet}} + f(L/d)}. \quad (22)$$

Model the entrance of the hose as a re-entrant inlet,

$$K_{\text{inlet}} = 0.8 \text{ (from minor loss table).} \quad (23)$$

Determine the friction factor numerically using the Colebrook formula with the Haaland formula serving as the initial seed for the iterative scheme,

$$\sqrt{\frac{1}{f}} = -1.8\log_{10}\left[\frac{6.9}{\text{Re}_d} + \left(\frac{e/d}{3.7}\right)^{1.11}\right] \text{ (Haaland formula),} \quad (24)$$

$$\sqrt{\frac{1}{f}} = -2.0\log_{10}\left(\frac{e/d}{3.7} + \frac{2.51}{\text{Re}_d\sqrt{f}}\right) \text{ (Colebrook formula).} \quad (25)$$

where,

$$\text{Re}_d = \frac{\bar{V}_B d}{\nu}. \quad (26)$$

Note that the Colebrook and Haaland formulas are reasonably accurate for $\text{Re}_d > 4000$. When calculating the Reynolds number, use the kinematic viscosity of water at 60 °F, i.e., $\nu = 1.21*10^{-5}$ ft^2/s. Also assume the flow to remain turbulent in the tubing during the entire discharge process, which means $\alpha_B \approx 1$. Finally, since the tubing is plastic, assume the absolute roughness is *e* = 0, i.e., the "smooth" curve is used in the Moody plot.

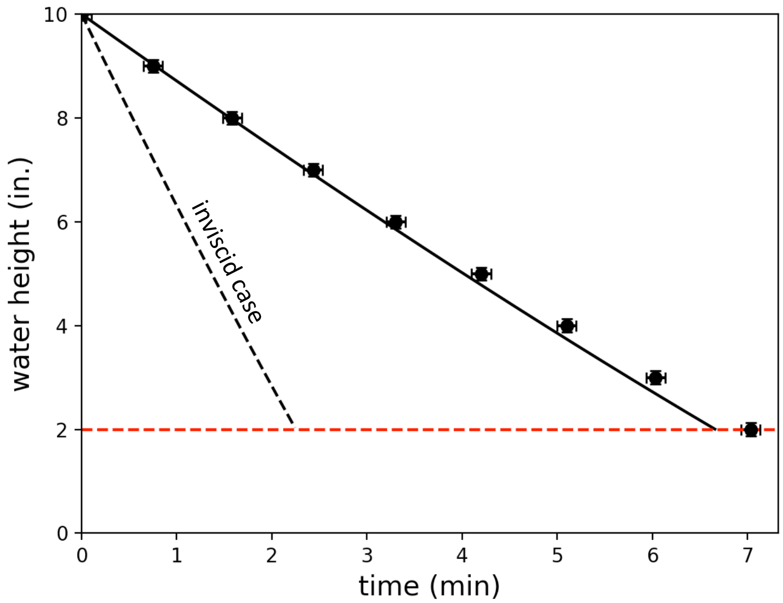
To determine the water height as a function of time, solve the ordinary differential Eq. (13) making use of Eq. (22) (and supporting equations). This equation must be solved numerically. Using the Python code given at the end of this solution, the time required to drain the bucket to the given final height is,

T = 399 s = 6.66 min.

The following plot shows the predicted water free surface height as a function of time (solid black line). The red dashed line in the plot is the final height z_f = 2 in. The symbols in the plot are experimental measurements corresponding to the problem statement, including uncertainty in the measurements. The black dashed line is the prediction if the inviscid Bernoulli's equation is used to determine the velocity at the tubing exit ($V_B = \sqrt{2g(z+h)}$). It's clear that the viscous model gives a good prediction of the actual system, although the error increases with increasing time. The model's relative error at the final time is,

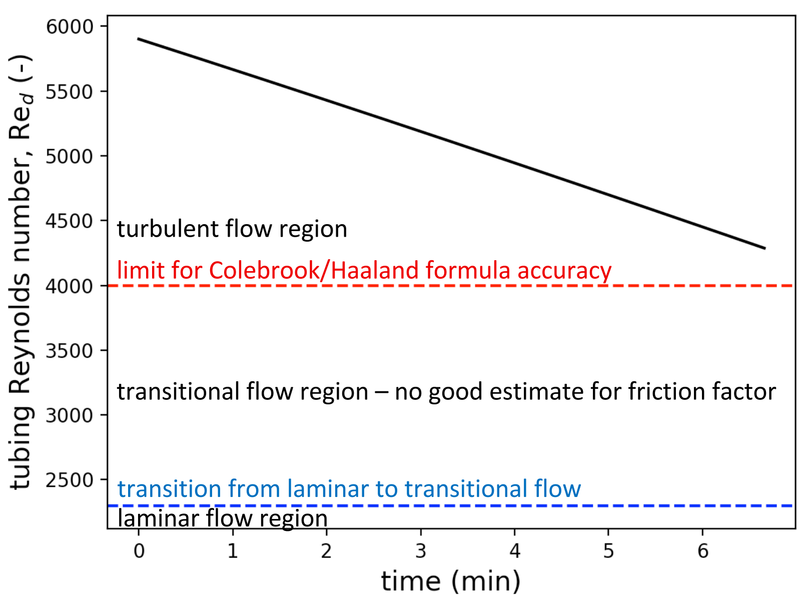
$$\text{relative error from exp} = \left(\frac{T_{\text{model}} - T_{\text{exp}}}{T_{\text{exp}}}\right) * 100\% = \text{-5.4\%}, \tag{27}$$

i.e., the model underpredicts the actual discharge time by 5.4%. The inviscid model clearly does a poor job at predicting the discharge rate.



Apparatus used to make the experimental measurements.

Note that the flow remains turbulent throughout the entire time period as shown in the following figure. The region between the red and blue dashed lines is the transition zone on the Moody plot. The Colebrook formula is valid for Reynolds numbers larger than 4000.



Following is the Python code used for the calculations and plots.

```
# pipe_70.py

import numpy as np
from scipy.optimize import root
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt

# Experimental measurements:  [z_exp (in.), t_exp (s)]
data = np.array([[10, 43.12],
                 [9, 88.10],
                 [8, 138.10],
                 [7, 189.13],
                 [6, 241.11],
                 [5, 295.14],
                 [4, 349.12],
                 [3, 405.15],
                 [2, 465.16]])

def alpha_fcn(V):  # kinetic energy correction factor
    Re_d = Re_d_fcn(V)
    if (Re_d <= 2300): # laminar
        return(2)
    else:  # turbulent
        return(1)

def Re_d_fcn(V):  # Reynolds number
    return(V*d/nu)

def f_fcn(V):  # friction factor
    Re_d = Re_d_fcn(V)
    if (Re_d > 4000):  # turbulent flow
        f_Haaland = 1/(-1.8*np.log10(6.9/Re_d + (e_d/3.7)**1.11))**2
        eqn = lambda x:  -2.0*np.log10(e_d/3.7 + 2.51/Re_d/np.sqrt(x)) - np.sqrt(1/x)
        result = root(eqn, f_Haaland)  # solve the implicit equation
        f = result.x
    elif (Re_d > 2300):  # transition region
        # Interpolate between the laminar and turbulent friction factors.
        # Note:  This is not a very accurate estimate.
        f_Haaland = 1/(-1.8*np.log10(6.9/Re_d + (e_d/3.7)**1.11))**2
        eqn = lambda x:  -2.0*np.log10(e_d/3.7 + 2.51/Re_d/np.sqrt(x)) - np.sqrt(1/x)
        result = root(eqn, f_Haaland)
        f_laminar = 64/Re_d
        f = f_laminar + (result.x - f_laminar)/(4000-2300)*(Re_d-2300)
    else:  # assume laminar flow
        f = 64/Re_d
    return(f)

def dzdt_fcn(t, z):  # differential eqn for the water height
    return( -V_fcn(z)*(d**2/4/(rb**2+2*z*rb*(rt-rb)/H+z**2*((rt-rb)/H)**2)) )

def stop_at_height(t, z):  # function to stop the explicit integration
    return(z[0] - z_final)

def V_fcn(z):  # velocity function from the EBE
    if (inviscid_flag == True):
        return( np.sqrt(2*g*(z+h)) )
    else:
        V_initial = np.sqrt(2*g*(z+h)) # use the inviscid speed as the first guess
        eqn = lambda x:  2*g*(z+h)/(alpha_fcn(x)+K_inlet+f_fcn(x)*L/d) - x**2
        result = root(eqn, V_initial)
        return(result.x)

# Initialize parameters.
g = 32.2  # ft/s^2, gravitational acceleration
rb = 5  # in., bucket bottom radius
rt = 5+5/8  # in., bucket top radius
H = 14  # in., bucket height
```

```
z_initial = 10 # in., initial water level
z_final = 2  # in., final water level
h = 8+1/2  # in., discharge elevation
L = 46+5/8  # in., tubing length
d = 1/4  # in., tubing inner diameter
e = 0  # in., tubing absolute roughness
K_inlet = 0.8  # -, re-entrant inlet
nu = 1.21e-5  # ft^2/s, water kinematic viscosity @ 60 degF

rb = rb/12  # convert from in. to ft
rt = rt/12  # convert from in. to ft
H = H/12  # convert from in. to ft
z_initial = z_initial/12  # convert from in. to ft
z_final = z_final/12  # convert from in. to ft
h = h/12  # convert from in. to ft
L = L/12  # convert from in. to ft
d = d/12  # convert from in. to ft
e = e/12  # convert from in. to ft

e_d = e/d  # relative roughness

# Parameter to stop the explicit integration when z = 0.
stop_at_height.terminal = True

tmax = 10*60  # s, maximum time for explicit time integration
t_span = (0, tmax)  # span of times for explicit time integration
t_range = np.linspace(0, tmax, 1000) # actual time points at which to solve the differential equation
# Integrate forward in time.  Stop when z = 0.
inviscid_flag = False
sol = solve_ivp(dzdt_fcn, t_span, [z_initial], t_eval=t_range, events=stop_at_height)
t = sol.t  # times used in the time integration
z = sol.y[0]  # water heights from the time integration
T = t[-1]  # the total time of discharge
print('T = %.3f s = %.3f min' % (T, T/60))

# Start the experimental time measurements from the 10 in. mark.
z_exp = data[:,0]
t_exp = data[:,1] - data[:,1][0]
z_error = 1/8  # in., error in height measurement
t_error = 0.1 # s, error in time measurement

# Plot the water height as a function of time.
plt.plot(t/60, z*12, 'k-')  # prediction
# Plot the experimental data with uncertainty bars.
plt.errorbar(t_exp/60, z_exp, yerr=z_error, xerr=t_error, fmt='ko', ecolor='black', capsize=3)

# Now plot the inviscid case.
inviscid_flag = True
sol = solve_ivp(dzdt_fcn, t_span, [z_initial], t_eval=t_range, events=stop_at_height)
plt.plot(sol.t/60, sol.y[0]*12, 'k--')  # prediction

plt.xlabel('time (min)', fontsize=14)
plt.ylabel('water height (in.)', fontsize=14)
plt.xlim(0, 1.1*T/60)
plt.ylim(0, z_initial*12)
# Draw a line at the minimum empty height.
plt.axhline(y=z_final*12, linestyle='--', color='red')
plt.show()

# Calculate and plot various quantities of interest.
V = np.array([])  # average water speed in tube
Re_d = np.array([])  # tube Reynolds number
K_major = np.array([])  # tube major loss coefficient
for z_iter in z:  # cycle through the heights
    V_iter = V_fcn(z_iter)
    V = np.append(V, V_iter)
    Re_d = np.append(Re_d, Re_d_fcn(V_iter))
    K_major = np.append(K_major, f_fcn(V_iter)*L/d)
```

```
# Plot the average water speed in the tube as a function of time.
plt.plot(t/60, V, 'k-')
plt.xlabel('time (min)', fontsize=14)
plt.ylabel(r'average water speed in the tube, $\overline{V}$ (ft/s)', fontsize=14)
plt.show()

# Plot the major loss coefficient in the tube as a function of time.
plt.plot(t/60, K_major, 'k-')
plt.xlabel('time (min)', fontsize=14)
plt.ylabel(r'tubing major loss coefficient, $K_{major}$ (-)', fontsize=14)
plt.show()

# Plot the tube Reynolds number as a function of time.
plt.plot(t/60, Re_d, 'k-')
plt.xlabel('time (min)', fontsize=14)
plt.ylabel(r'tubing Reynolds number, Re$_d$ (-)', fontsize=14)
# Show the Reynolds number limit for the Colebrook/Haaland formulas.
plt.axhline(y=4000, color='red', linestyle='--')
# Show the Reynolds number limit for laminar flow.
plt.axhline(y=2300, color='blue', linestyle='--')
plt.show()

# Calculate the model's relative error for the total discharge time.
rel_err = (T - t_exp[-1])/t_exp[-1]*100
print('rel err = %.1f perc.' % rel_err)
```