

# Pipe Flow Analysis with MATLAB

Gerald Recktenwald\*

January 28, 2007

This document describes a collection of MATLAB programs for pipe flow analysis. Using these functions it is relatively easy to perform head loss calculations, solve flow rate problems, generate system curves, and find the design point for a system and pump.

## 1 Governing Equations

Figure 1 shows a single pipe flow system. For steady flow between station 1 and station 2 mass conservation (continuity) requires

$$\dot{m}_1 = \dot{m}_2 \quad \text{or} \quad \rho_1 A_1 V_1 = \rho_2 A_2 V_2 \quad (1)$$

where  $\dot{m}$  is the mass flow rate,  $\rho$  is the fluid density,  $A$  is the cross-section area, and  $V$  is the *average* velocity. In most applications the flow is incompressible, so  $\rho_1 = \rho_2$  and Equation (1) simplifies to

$$A_1 V_1 = A_2 V_2 \quad \text{or} \quad Q_1 = Q_2 \quad (2)$$

where  $Q$  is the volumetric flow rate.

Energy conservation between two stations requires [1]

$$\left[ \frac{p}{\gamma} + \frac{V^2}{2g} + z \right]_{\text{out}} = \left[ \frac{p}{\gamma} + \frac{V^2}{2g} + z \right]_{\text{in}} + h_s - h_L \quad (3)$$

where the subscript “out” indicates the *downstream* station, and “in” indicates the *upstream* station,  $\gamma = \rho g$  is the specific weight of the fluid,  $h_s$  is the shaft work done *on* the fluid, and  $h_L$  is the head loss due to friction. For the system depicted in Figure 1, station 1 is “in” and station 2 is “out”.

All terms in Equation (3) have units of head. To convert  $h_s$  and  $h_L$  to units of power use

$$\dot{W}_s = \dot{m}gh_s \quad \dot{W}_L = \dot{m}gh_L. \quad (4)$$

---

\* Associate Professor, Mechanical and Materials Engineering Department Portland State University, Portland, Oregon, [gerry@me.pdx.edu](mailto:gerry@me.pdx.edu)

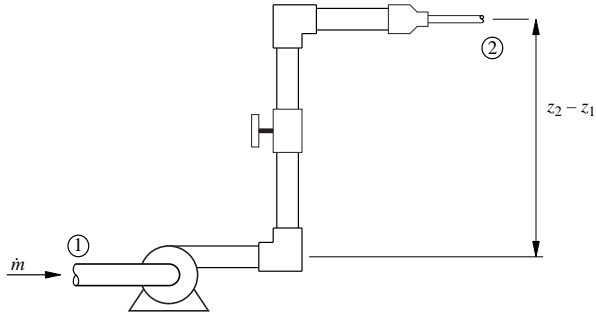


Figure 1: Energy must be supplied to the flowing fluid to overcome elevation changes, viscous losses in straight sections of pipe, and minor losses due to elbows, valves and changes of flow area.

### 1.1 Empirical Head Loss Data

The head loss due to friction is given by the Darcy-Weisbach equation

$$h_L = f \frac{L}{D} \frac{V^2}{2g} \quad (5)$$

where  $f$  is the Darcy Friction Factor. For fully-developed, incompressible laminar flow in a round pipe

$$f_{\text{lam}} = \frac{64}{\text{Re}}. \quad (6)$$

The friction factor for turbulent flow in smooth and rough pipes is correlated with the Colebrook equation

$$\frac{1}{\sqrt{f}} = -2 \log_{10} \left( \frac{\varepsilon/D}{3.7} + \frac{2.51}{\text{Re}\sqrt{f}} \right) \quad (7)$$

where  $\varepsilon$  is the roughness of the pipe wall, and  $\text{Re}$  is the Reynolds number

$$\text{Re} = \frac{\rho V D}{\mu} = \frac{V D}{\nu}.$$

If the pipe is not round the same formulas may be applied if the *hydraulic diameter*,  $D_h$ , is substituted for  $D$  in the definition of  $\text{Re}$ , and in the  $\varepsilon/D$  term in the Colebrook equation. For a pipe with any cross-section area  $A$  and perimeter  $P$  the hydraulic diameter is

$$D_h = \frac{4 \times \text{cross sectional area}}{\text{wetted perimeter}} = \frac{4A}{P}.$$

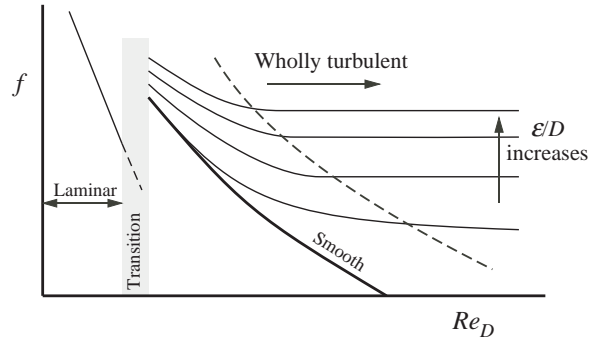


Figure 2: Moody Diagram.

## 1.2 Minor Losses

Minor losses are caused by elbows, valves, and other components that disrupt the fully-developed flow through the pipe. “Minor Loss” is a misnomer because in many practical systems elbows, valves, and other components can account for a significant fraction of the head loss.

Minor losses must be included in the head loss term in the energy equation. For most pipe runs there will be multiple minor loss elements connected by sections of straight pipe. The total loss can be represented by

$$h_{L,total} = \sum h_{L,pipe} + \sum h_{L,minor} \quad (8)$$

where  $h_{L,pipe}$  is the viscous loss in a straight section of pipe and  $h_{L,minor}$  is a minor loss due to a fitting or other element. Note that the  $h_{L,pipe}$  contributions are usually computed by assuming that the flow in the pipe section is fully developed. The empirical model for an individual minor loss is

$$h_{L,minor} = K_L \frac{V^2}{2g} \quad (9)$$

where  $K_L$  is a loss coefficient.

## 2 Solving the Colebrook Equation with MATLAB

Equation (7) cannot be solved analytically for  $f$  when  $\epsilon/D$  and  $Re$  are given. However, if Equation (7) is rearranged as

$$\mathcal{F}(f) = \frac{1}{\sqrt{f}} + 2 \log_{10} \left( \frac{\epsilon/D}{3.7} + \frac{2.51}{Re\sqrt{f}} \right) \quad (10)$$

a numerical root-finding procedure can be used to find the  $f$  that makes  $\mathcal{F}(f) = 0$  when  $\epsilon/D$  and  $Re$  are known [2].

```

function f = moody(ed,Re,verbose)
% moody Find friction factor by solving the Colebrook equation (Moody Chart)
%
% Synopsis: f = moody(ed,Re)
%
% Input:    ed = relative roughness = epsilon/diameter
%           Re = Reynolds number
%
% Output:   f = friction factor
%
% Note:     Laminar and turbulent flow are correctly accounted for

if Re<0
    error(sprintf('Reynolds number = %f cannot be negative',Re));
elseif Re<2000
    f = 64/Re; return % laminar flow
end
if ed>0.05
    warning(sprintf('epsilon/diameter ratio = %f is not on Moody chart',ed));
end

if Re<4000, warning('Re = %f in transition range',Re); end

% --- Use fzero to find f from Colebrook equation.
% coleFun is an inline function object to evaluate F(f,e/d,Re)
% fzero returns the value of f such that F(f,e/d/Re) = 0 (approximately)
% fi = initial guess from Haaland equation, see White, equation 6.64a
% Iterations of fzero are terminated when f is known to within +/- dfTol
coleFun = inline('1.0/sqrt(f) + 2.0*log10( ed/3.7 + 2.51/( Re*sqrt(f)) )',...
    'f','ed','Re');
fi = 1/(1.8*log10(6.9/Re + (ed/3.7)^1.11))^2; % initial guess at f
dfTol = 5e-6;
f = fzero(coleFun,fi,optimset('TolX',dfTol,'Display','off'),ed,Re);
% --- sanity check:
if f<0, error(sprintf('Friction factor = %f, but cannot be negative',f)); end

```

Listing 1: The moody function finds the friction factor from the formulas used to create the Moody chart.

The moody function in Listing 1 uses MATLAB's built-in `fzero` function as a root-finder to solve  $\mathcal{F}(f)$ . The initial guess at the  $f$  for the root finder is the explicit formula of Haaland given by White [3].

$$f = \left[ 1.8 \log_{10} \left( \frac{6.9}{Re} + \left( \frac{ed}{3.7} \right)^{1.11} \right) \right]^{-2} \quad (11)$$

The moody function can be called from the command line like this

```

>> moody(0.002,50e3)
ans =
    0.0265

```

The moody function also uses Equation (6) for laminar flow when  $Re < 2000$ . In the range  $2000 \leq Re \leq 4000$ , the moody function uses the Colebrook equation

```

function myMoody
% myMoody Make a simple Moody chart

% --- Generate a log-spaced vector of Re values in the range 2500 <= Re < 10^8
Re = logspace(log10(2500),8,50);
ed = [0 0.00005 0.0002 0.005 0.001 0.005 0.02];
f = zeros(size(Re));

% --- Plot f(Re) curves for one value of epsilon/D at a time
% Temporarily turn warnings off to avoid lots of messages when
% 2000 < Re < 4000
warning('off')
figm = figure; hold('on');
for i=1:length(ed)
    for j=1:length(Re)
        f(j) = moody(ed(i),Re(j));
    end
    loglog(Re,f,'k-');
end
ReLam = [100 2000];
fLam = 64./ReLam;
loglog(ReLam,fLam,'r-');
xlabel('Re'); ylabel('f','Rotation',0)
axis([100 1e9 5e-3 2e-1]); hold('off'); grid('on')
warning('on');

% --- MATLAB resets loglog scale? Set it back
set(gca,'Xscale','log','Yscale','log');

% --- Add text labels like "epsilon/D = ..." at right end of the plot
Remax = max(Re); ReLabel = 10^( floor(log10(Remax)) - 1);
for i=2:length(ed)
    fLabel = moody(ed(i),Remax);
    text(ReLabel,1.1*fLabel,sprintf('\epsilon/D = %5.4f',ed(i)),'FontSize',14);
end

```

Listing 2: The myMoody function uses the moody function to create a simplified version of the Moody chart.

to find  $f$ . Inspection of Figure 2 shows that the Colebrook equation for smooth pipes gives a more conservative estimate of  $f$  (higher  $f$  for a given  $Re$ ) than Equation (6). For example

```

>> Re = 3000; ed = 0;
>> f = moody(ed,Re)
Warning: Re = 3000.000000 in transition range
> In moody at 22
f =
    0.0435
>> flam = 64/Re
flam =
    0.0213

```

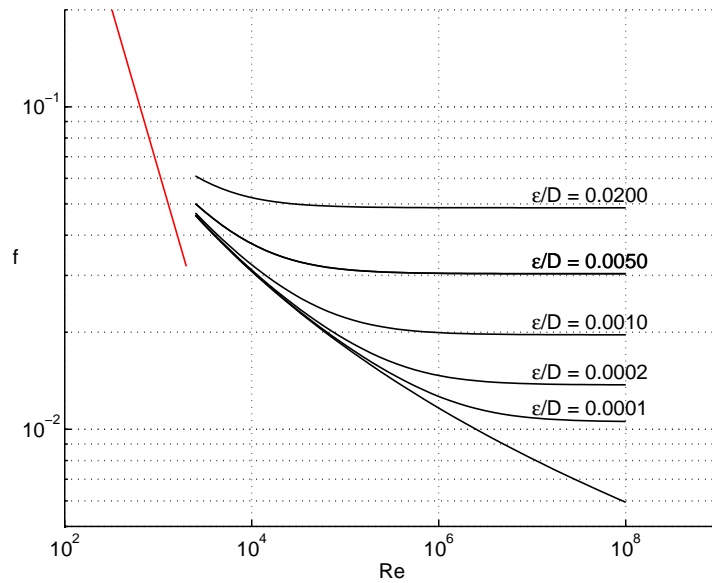


Figure 3: Moody Diagram created with the myMoody function.

### Example 1 Creating a Custom Moody Chart

The moody function accepts only scalar values of  $\varepsilon/D$  and  $Re$ . To generate a vector of friction factor values, an explicit loop is necessary. For example, the following code snippet prints a table of friction factors for  $\varepsilon/D = 0.002$  and for five Reynolds numbers in the range  $5000 \leq Re \leq 5 \times 10^6$ .

```

Re = logspace(log10(5000),6,5);
ed = 0.002;
for i=1:length(Re)
    f(i) = moody(ed,Re(i));
    fprintf(' %12.2e  %8.6f\n',Re(i),f(i))
end

5.00e+03  0.039567
1.88e+04  0.030085
7.07e+04  0.025713
2.66e+05  0.024099
1.00e+06  0.023607

```

The preceding code suggests that the moody function can be used to create a Moody chart. The myMoody function in Listing 2 makes the plot in Figure 3.

□

## 3 Head Loss Computation

The simplest pipe flow problem involves computing the head loss when the pipe diameter and the flow rate are known.

```

function MYOEx85
% MYOEx85 Head loss problem of Example 8.5 in Munson, Young and Okiishi

% --- Define constants for the system and its components
L = 0.1;           % Pipe length (m)
D = 4e-3;         % Pipe diameter (m)
A = 0.25*pi*D.^2; % Cross sectional area
e = 0.0015e-3;    % Roughness for drawn tubing
rho = 1.23;       % Density of air (kg/m^3) at 20 degrees C
g = 9.81;        % Acceleration of gravity (m^2/s)
mu = 1.79e-5;    % Dynamic viscosity (kg/m/s) at 20 degrees C
nu = mu/rho;     % Kinematic viscosity (m^2/s)
V = 50;          % Air velocity (m/s)

% --- Laminar solution
Re = V*D/nu;
flam = 64/Re;
dplam = flam * 0.5*(L/D)*rho*V^2;

% --- Turbulent solution
f = moody(e/D,Re);
dp = f*(L/D)*0.5*rho*V^2;

% --- Print summary of losses
fprintf('\nMYO Example 8.5: Re = %12.3e\n\n',Re);
fprintf('\tLaminar flow: ');
fprintf('flam = %8.5f; Dp = %7.0f (Pa)\n',flam,dplam)
fprintf('\tTurbulent flow: ');
fprintf('f = %8.5f; Dp = %7.0f (Pa)\n',f,dp);

```

Listing 3: The MYOEx85 function obtains the solution to Example 8.5 in the textbook by Munson, Young and Okiishi [1].

### Example 2 MYO Example 8.5: Head Loss in a Horizontal Pipe

The MYOEx85 function in Listing 3 obtains the solution for Example 8.5 in the textbook by Munson, Young, and Okiishi [1]. Air flows in a horizontal pipe of diameter 4.0 mm. The problem involves computing the pressure drop for a 0.1 m long section of the pipe if the flow is assumed to be laminar and if the flow is assumed to be turbulent. Running MYOEx85 produces the following output.

```

>> MYOEx85

MYO Example 8.5: Re = 1.374e+04

Laminar flow: flam = 0.00466; Dp = 179 (Pa)
Turbulent flow: f = 0.02910; Dp = 1119 (Pa)

```

\_\_\_\_\_ □

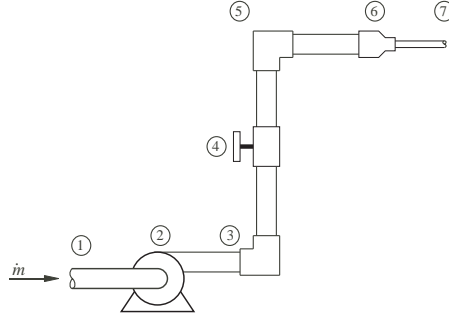


Figure 4: Pipe system used to develop a generic calculation procedure.

### 3.1 Head Loss in Single-Pipe Systems

Head loss problems are slightly more complicated when multiple pipe segments and multiple minor loss elements are connected in series. The calculations are more tedious than difficult, and a computer tool can relieve the tedium.

A general procedure can be developed by considering the system in Figure 4. The system consists of six pipe segments with lengths  $L_{1,2}$ ,  $L_{2,3}$ ,  $L_{3,4}$ ,  $L_{4,5}$ ,  $L_{5,6}$ ,  $L_{6,7}$  and diameters  $D_{1,2}$ ,  $D_{2,3}$ ,  $D_{3,4}$ ,  $D_{4,5}$ ,  $D_{5,6}$ ,  $D_{6,7}$ . In general, each pipe segment could be made of a different material, so each segment could potentially have a different wall roughness.

The system in Figure 4 also has four minor loss elements: elbows at stations 3 and 5, a valve at station 4, and a reducer at station 6.

Applying Equation (8) to the system in Figure 4 gives the total head loss from station 2 to station 7 is

$$\begin{aligned}
 h_{L,2-7} = & f_{2,3} \frac{L_{2,3}}{D_{2,3}} \frac{Q^2}{2gA_{2,3}^2} + f_{3,4} \frac{L_{3,4}}{D_{3,4}} \frac{Q^2}{2gA_{3,4}^2} + f_{4,5} \frac{L_{4,5}}{D_{4,5}} \frac{Q^2}{2gA_{4,5}^2} \\
 & + f_{5,6} \frac{L_{5,6}}{D_{5,6}} \frac{Q^2}{2gA_{5,6}^2} + f_{6,7} \frac{L_{6,7}}{D_{6,7}} \frac{Q^2}{2gA_{6,7}^2} \\
 & + K_{L,3} \frac{Q^2}{2gA_3^2} + K_{L,4} \frac{Q^2}{2gA_4^2} + K_{L,5} \frac{Q^2}{2gA_5^2} + K_{L,6} \frac{Q^2}{2gA_6^2}
 \end{aligned} \tag{12}$$

Notice that the flow rate,  $Q$  is common to all terms. Also notice that the viscous terms involve the product of  $Q^2$  with  $f$ , which is also a function of  $Q$ , because  $Q$  determines the Reynolds number.

#### 3.1.1 Head Loss Worksheets

Figure 5 and Figure 6 contain worksheets to list the terms on the right hand side of Equation (8). Each pipe section is specified by its length,  $L$ , hydraulic diameter,  $D_h$ , cross-sectional area,  $A$ , and roughness,  $\varepsilon$ . Values of these parameters are entered for each pipe section in the first four columns of the worksheet in Figure 5. The cross-sectional area is specified independently of the hydraulic



diameter so that non-circular ducts can be properly treated. For a round pipe,  $A_i = (\pi/4)D_i^2$ , where  $i$  is the index of the pipe section (row in Figure 5).

Each minor loss element is specified by its loss coefficient,  $K_L$ , and the cross-sectional area,  $A_m$  that defines the flow velocity for the element. For most minor loss elements, e.g., valves and elbows,  $A_m$  is the cross-sectional area of the pipe. For flow through a sudden contraction,  $A_m$  is the downstream pipe area. For flow through a sudden expansion,  $A_m$  is the upstream pipe area.

### **3.2 Generating a System Curve**

## **4 Solving Flow-Rate Problems**

## **5 Solving Pipe-Sizing Problems**

## **6 Generating a Pump Curve from Tabulated Performance Data**

## **7 System Balance Point**

$$Q = \boxed{\frac{m^3}{s}} \quad \nu = \boxed{\frac{kg}{m \cdot s}}$$

$L$	$D_h$	$A$	$\varepsilon$	$V = \frac{Q}{A}$	$Re_{D_h} = \frac{VD_h}{\nu}$	$f = F\left[\frac{\varepsilon}{D_h}, Re_{D_h}\right]$	$h_L = f \frac{L}{D_h} \frac{V^2}{2g}$

Total: \_\_\_\_\_

Figure 5: Worksheet for calculating viscous losses in straight pipes.

$$Q = \boxed{\frac{m^3}{s}}$$

$K_L$	$A_m$	$V = \frac{Q}{A}$	$h_m = K_L \frac{V^2}{2g}$

Total: \_\_\_\_\_

Figure 6: Worksheet for calculating minor losses.

```

function [out1,out2,out3] = pipeLoss(Q,L,A,Dh,e,nu,KL,Am)
% pipeLoss Viscous and minor head loss for a single pipe
%
% Synopsis: hL = pipeLoss(Q,L,A,Dh,e,nu)
%           hL = pipeLoss(Q,L,A,Dh,e,nu,KL)
%           hL = pipeLoss(Q,L,A,Dh,e,nu,KL,Am)
%           [hL,f] = pipeLoss(...)
%           [hv,f,hm] = pipeLoss(...)
%
% Input: Q = flow rate through the system (m^3/s)
%        L = vector of pipe lengths
%        A = vector of cross-sectional areas of ducts. A(1) is area of
%           pipe with length L(1) and hydraulic diameter Dh(1)
%        Dh = vector of pipe diameters
%        e = vector of pipe roughnesses
%        nu = kinematic viscosity of the fluid
%        KL = minor loss coefficients. Default: KL = [], no minor losses
%        Am = areas associated with minor loss coefficients.
%           For a flow rate, Q, through minor loss element 1, the area,
%           Am(1) gives the appropriate velocity from  $V = Q/Am(1)$ . For
%           example, the characteristic velocity of a sudden expansion
%           is the upstream velocity, so Am for that element is the
%           area of the upstream duct.
%
% Output: hL = (scalar) total head loss
%        hv = (optional,vector) head losses in straight sections of pipe
%        hm = (optional, vector) minor losses
%        f = (optional, vector) friction factors for straight sections

if nargin<7, KL = []; Am = []; end
if nargin<8, Am = A(1)*ones(size(KL)); end
if size(Am) ~= size(KL)
    error('size(Am) = [%d,%d] not equal size(KL) = [%d,%d]',size(Am),size(KL));
end

% --- Viscous losses in straight sections
g = 9.81; % acceleration of gravity, SI units
if isempty(L)
    hv = 0; f = []; % no straight pipe sections
else
    V = Q./A; % velocity in each straight section of pipe
    f = zeros(size(L)); % initialize friction factor vector
    for k=1:length(f)
        f(k) = moody(e(k)/Dh(k),V(k)*Dh(k)/nu); % friction factors
    end
    hv = f.*(L./Dh).*(V.^2)/(2*g); % viscous losses in straight sections
end

% --- minor losses
if isempty(KL)
    hm = 0; % no minor losses
else
    hm = KL.*((Q./Am).^2)/(2*g); % minor losses
end

% --- optional return variables
if nargin==1
    out1 = sum(hv) + sum(hm); % return hL = total head loss
elseif nargin==2
    out1 = hv; out2 = f; % return viscous losses and friction factors
elseif nargin==3
    11
    out1 = hv; out2 = f; % return viscous losses, friction factors
    out3 = hm; % and minor losses
else
    error('Only 1, 2 or 3 return arguments are allowed');
end

```

Listing 4: The pipeLoss function computes the total head for a single pipe.

```

function MYOEx88
% MYOEx88 Flow rate problem in Example 8.8 in Munson, Young and Okiishi

% --- Define system components
L = [15 10 5 10 10 10]*0.3048; % Pipe lengths (m)
D = 0.0625*0.3048*ones(size(L)); % True pipe diameters, all equal
% at 0.0625*0.3048 (m)
A = 0.25*pi*D.^2; % Cross sectional area
e = 0.0015e-3*ones(size(L)); % Roughness for drawn tubing
KL = [1.5 1.5 1.5 1.5 10 2]; % Minor loss coefficients
Am = A(1)*ones(size(KL)); % Areas of pipe used with minor loss
% coefficients are same as pipe area

rho = 998.2; % Density of water (kg/m^3) at 20 degrees C
gam = rho*9.81; % specific weight of fluid (N/m^3)
mu = 1.002e-3; % Dynamic viscosity (kg/m/s) at 20 degrees C
nu = mu/rho; % Kinematic viscosity (m^2/s)
dz = -20*0.3048; % Inlet is 20ft below outlet
Q = 12*6.309e-5; % 12 gpm flow rate converted to m^3/s

% --- The pipeLoss function uses root-finding to find head loss at given Q
[hv,f,hm] = pipeLoss(Q,L,A,D,e,nu,KL,Am);

% --- Print summary of losses
fprintf('\nViscous losses in straight pipe sections\n');
fprintf(' section Re f hL\n');
for k=1:length(L)
    Re = (Q/A(k))*D(k)/nu;
    fprintf(' %3d %12.2e %8.4f %6.3f\n',k,Re,f(k),hv(k));
end

fprintf('\nMinor losses in straight pipe sections\n');
fprintf(' fitting KL hm\n');
for k=1:length(KL)
    fprintf(' %3d %8.2f %6.3f\n',k,KL(k),hm(k));
end

% --- Total head loss is sum of viscous (major) and minor losses
hLtot = sum(hv) + sum(hm);
fprintf('\nTotal head loss');
fprintf(' = %6.1f (m H2O), %6.1f (ft H2O)\n',hLtot,hLtot/0.3048);

% --- Total static pressure difference
p1 = gam*(hLtot-dz);
fprintf('\np1 = %8.0f (Pa) = %8.1f (psi)\n',p1,p1*14.696/101325);
fprintf('\nFlow rate = %11.3e (m^3/s) = %6.1f (gpm)\n',Q,Q/6.309e-5);
Wp = gam*Q*hLtot; % pump power (eta = 1) (W)
fprintf('Pump power = %6.1f (W) = %6.1f hp\n',Wp,Wp*1.341e-3);

```

Listing 5: The MYOEx88 function obtains the solution to Example 8.8 in the textbook by Munson, Young, and Okiishi [1].

```

function demoSystemCurve
% demoSystemCurveGenerate the system curve for a single pipe system

% --- data for straight pipe sections
dsteel = 0.127;      % diameter of commercial steel pipe used (m)
diron  = 0.102;      % diameter of cast iron pipe used (m)
epsSteel = 0.045e-3; % roughness of steel pipe (m)
epsIron  = 0.26e-3;  % roughness of cast iron pipe (m)

L = [50; 10; 20; 25; 20];          % pipe lengths (m)
D = [dsteel; dsteel; dsteel; diron; dsteel]; % pipe diameters (m)
A = 0.25*pi*D.^2;                  % X-sectional areas (m^2)
e = [epsSteel; epsSteel; epsSteel; epsIron; epsSteel]; % roughness

% --- data for minor losses
kelbow = 0.3;
kvalve = 10;
Asteel = 0.25*pi*dsteel^2;
Airon  = 0.25*pi*diron^2;
kcontract = contractionLoss(Asteel,Airon);
kexpand  = (1 - Airon/Asteel)^2;
KL = [kelbow; kelbow; kvalve; kcontract; kexpand; kvalve];
Am = [Asteel; Asteel; Asteel; Airon; Airon; Asteel];

% --- overall elevation change
dz = L(2);          % outlet is above inlet by this amount

% --- fluid properties
rho = 999.7;        % density of water (kg/m^3) at 10 degrees C
mu  = 1.307e-3;     % dynamic viscosity (kg/m/s) at 10 degrees C
nu  = mu/rho;       % kinematic viscosity (m^2/s)

% --- Create system curve for a range of flow rates
Q = linspace(0,0.075,10);
hL = zeros(size(Q));
for k = 2:length(Q) % loop starts at 2 to avoid Q=0 calculation
    hL(k) = pipeLoss(Q(k),L,A,D,e,KL,Am,nu);
end
hsys1 = hL + dz;

% --- Repeat analysis with gate valves instead of globe valves
KL(3) = 0.15; KL(6) = 0.15; % replace loss coefficients for valves
for k = 2:length(Q) % loop starts at 2 to avoid Q=0 calculation
    hL(k) = pipeLoss(Q(k),L,A,D,e,KL,Am,nu);
end
hsys2 = hL + dz;

% --- Plot both system curves and annotate
plot(Q,hsys1,'o-',Q,hsys2,'s-');
xlabel('Flow rate (m^3/s)'); ylabel('Head (m)');
legend('globe valves','gate valves',2);

```

Listing 6: The `testSystemCurve` function generates and plots the system curve for a single pipe.

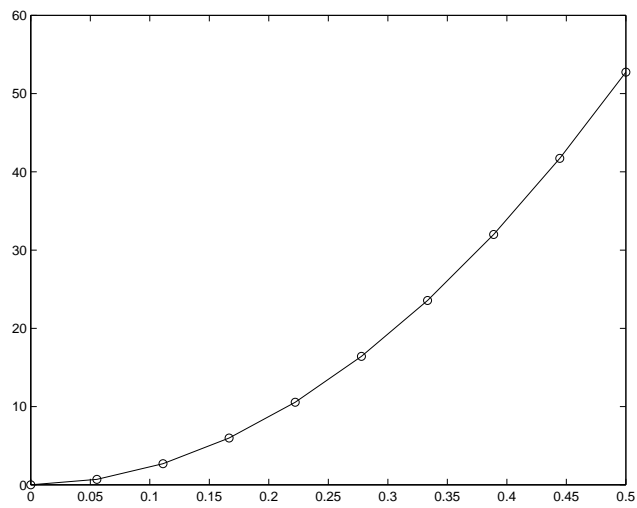


Figure 7: System curve.

```

function h = pipeHeadBal(Q,L,A,Dh,e,KL,Am,nu,dz,pc)
% pipeHeadBal Head imbalance in energy equation for a single pipe with a pump
% This equations is of the form f(Q) = 0 for use with a
% standard root-finding routine for flow rate problems or
% system balance point problems
%
% Synopsis: h = pipeHeadBal(Q,L,A,Dh,e,KL,Am,nu,dz,pc)
%
% Input: Q = flow rate through the system (m^3/s)
% L = vector of pipe lengths
% A = vector of cross-sectional areas of ducts. A(1) is area of
% pipe with length L(1) and hydraulic diameter Dh(1)
% Dh = vector of pipe diameters
% e = vector of pipe roughnesses
% KL = vector of minor loss coefficients
% Am = vector of areas associated with minor loss coefficients.
% For a flow rate, Q, through minor loss element 1, the area,
% Am(1) gives the appropriate velocity from V = Q/Am(1).
% Thus, different minor loss elements will have different
% velocities for the same flow rate. For a sudden expansion
% the characteristic velocity is the upstream velocity, so
% Am for that element is the area of the upstream duct.
% nu = kinematic viscosity of the fluid
% dz = change in elevation, z_out - z_in; dz>0 if outlet is above
% the inlet; dz = 0 for closed loop
% pc = vector of polynomial coefficients defining the pump curve.
% Note that the polynomial is defined in decreasing power of Q:
% hs(Q) = pc(1)Q^n + pc(2)*Q^(n-1) + ... + pc(n)*Q + pc(n+1)
%
% Output: h = imbalance in head in energy equation; h = 0 when
% energy equation is satisfied

if isempty(pc)
    hs = 0; % no pump curve data -> no pump
else
    hs = polyval(pc,Q); % evaluate polynomial model of pump curve at Q
end

h = hs - dz - pipeLoss(Q,L,A,Dh,e,nu,KL,Am);

```

Listing 7: The pipeHeadBal function evaluates the energy equation and returns the head imbalance. When the correct flow rate and pipe diameter are supplied, the head imbalance is zero.

```

function c = makePumpCurve(Q,h,n);
% makePumpCurve Find polynomial model for pump curve
%
% Synopsis: c = makePumpCurve(Q,h,n)
%
% Input: Q = pump flow rate; (Both Q and h must be given)
%        h = pump head
%        n = (optional) degree of polynomial curve fit; Default: n=3

if nargin<3; n = 3; end

c = polyfit(Q,h,n);
fprintf('\nCoefficients of the polynomial fit to the pump curve:\n');
fprintf('      (in DECREASING powers of Q)\n');
for k=1:length(c)
    fprintf(' Q^%d   %18.10e\n',n+1-k,c(k));
end

% --- evaluate the fit and plot along with original data
Qfit = linspace(min(Q),max(Q));
hfit = polyval(c,Qfit);
plot(Q,h,'o',Qfit,hfit,'-');
xlabel('Flow rate'); ylabel('pump head');

```

Listing 8: The `makePumpCurve` function uses the least squares method to obtain the coefficients of a polynomial curve fit to a pump curve. If no data is supplied, the data from the pump curve given in Example 12.4 by Munson et. al [1] is used.



```

function MYOEx124
% MYOEx124 Find balance point for single pipe system in Example 12.4
%           of Munson, Young and Okiishi

L = 200*0.3048;      % pipe length (m)
D = (6/12)*0.3048;  % true pipe diameter (m)
A = 0.25*pi*D^2;    % cross sectional area
e = 0.26e-3;        % roughness for cast iron
KL = [0.5; 1.5; 1]; % minor loss for entrance, elbow, and exit
Ah = [ A;  A;  A];  % areas of pipe associated with minor loss
                        % coefficients are same as pipe area
rho = 998.2;        % density of water (kg/m^3) at 20 degrees C
mu = 1.002e-3;     % dynamic viscosity (kg/m/s) at 20 degrees C
nu = mu/rho;        % kinematic viscosity (m^2/s)
dz = 10*0.3048;    % outlet is 10 ft above inlet

% --- Data for pump curve
Q = [ 0 400 800 1400 1800 2200 2300]*6.309e-5; % m^3/s
h = [88 86 81 70 60 46 40]*0.3048; % m
n = 3;
pc = makePumpCurve(Q,h,n); % polynomial curve fit of degree n
eta = 0.84; % pump efficiency

% --- evaluate pump curve for balance point plot
Qfit = linspace(min(Q),max(Q));
hfit = polyval(pc,Qfit);

% --- generate system curve
Qs = linspace(min(Q),max(Q),10);
hL = zeros(size(Qs));
for k = 2:length(Qs) % loop starts at 2 to avoid Q=0 calculation
    hL(k) = pipeLoss(Qs(k),L,A,D,e,nu,KL,Ah);
end

% --- superimpose system curve and pump curve in new window
% to give graphical solution to balance point
f = figure;
plot(Q,h,'o',Qfit,hfit,'-',Qs,hL+dz,'v-');
xlabel('Flow rate (m^3/s)'); ylabel('head (m)');
legend('pump data','pump curve (fit)','system curve',2);

Q0 = [0.01 1.2]; % initial guess brackets the flow rate (m^3/s)
[Q,hL,hs] = pipeFlowSolve(Q0,L,A,D,e,KL,Ah,nu,dz,pc); % find balance point
Wp = rho*9.81*Q*hs/eta; % pump power (W) with efficiency of eta
hold on;
plot([Q Q],[0 hL+dz], '--'); % vertical line at balance pt.
hold off;

% --- convert to ancient system of units and print
fprintf('\nFlow rate = %6.1f gpm\n',Q/6.309e-5);
fprintf('Pump head = %6.1f ft\n',hs/0.3048);
fprintf('Pump power = %6.1f hp\n',Wp*1.341e-3);

```

Listing 9: The MYOEx124 function solves the flow rate problem given as Example 12.4 by Munson et. al [1].

## 8 References

### References

- [1] Bruce R. Munson, Donald F. Young, and Theodore H. Okiishi. *Fundamentals of Fluid Mechanics*. Wiley, New York, fifth edition, 2006.
- [2] Gerald W. Recktenwald. *Numerical Methods with MATLAB: Implementations and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 2000.
- [3] Frank M. White. *Fluid Mechanics*. McGraw-Hill, New York, fourth edition, 1999.