Kinematics: Reference Frames, Flow Fields, and Reynolds Transport Theorem (RTT)

AA 507 (Incompressible Fluid Mechanics) Winter 2026, UW

Assignment: Homework 2

(Ref file: HW2.pdf)

Author: Emmanuel Rodriguez

emmanueljrodriguez.com

Date: 24JAN2026

- 29JAN2026

Location: Renton, Seattle, WA

Table of Contents


```
clear % Clears workspace
clc % Clears command window
close all % Closes all figures
```

Problem 1. Data Collapse for Laminar Pipe Flow (Buckingham-Π)

In practice, fluid flow in pipes is commonly encountered in systems. For example, the flow of liquid propellants through the network of ducts and lines found in a rocket engine.

"In fully developed laminar flow, each fluid particle moves at a constant axial velocity along a streamline and the velocity profile *u(r)* remains unchanged in the flow direction." [1] See Fig. 8-11.

```
imshow('Fig8-11_LaminarFlow.png')
```

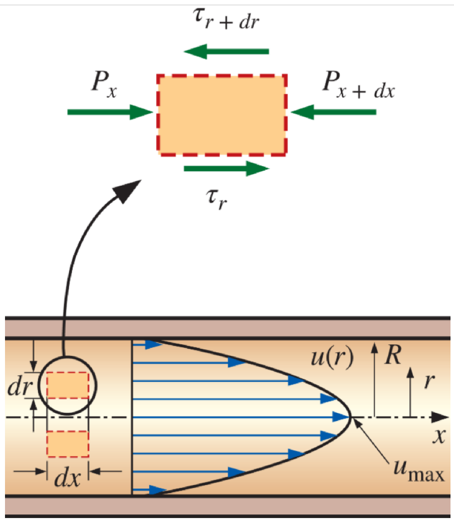


FIGURE 8–11 Free-body diagram of a ring-shaped differential fluid element of radius *r*, thickness *dr*, and length *dx* oriented coaxially with a horizontal pipe in fully developed laminar flow. (The size of the fluid element is greatly exaggerated for clarity.)

- [*Newtonian fluid* - viscosity is constant; shear stress is proportional to shear rate.]

Why aerospace engineers care about the Newtonian assumption:

Area	Newtonian assumption works	Non-Newtonian risk
Aerodynamics	✔ External flow	✖ Icing fluids, contaminants
Propulsion	✔ Gas dynamics	✖ Slurries, gels
Thermal	✔ Convective models	✖ Phase-changing coatings
CFD	✔ Standard solvers	✖ Needs constitutive modeling

1. Raw plot

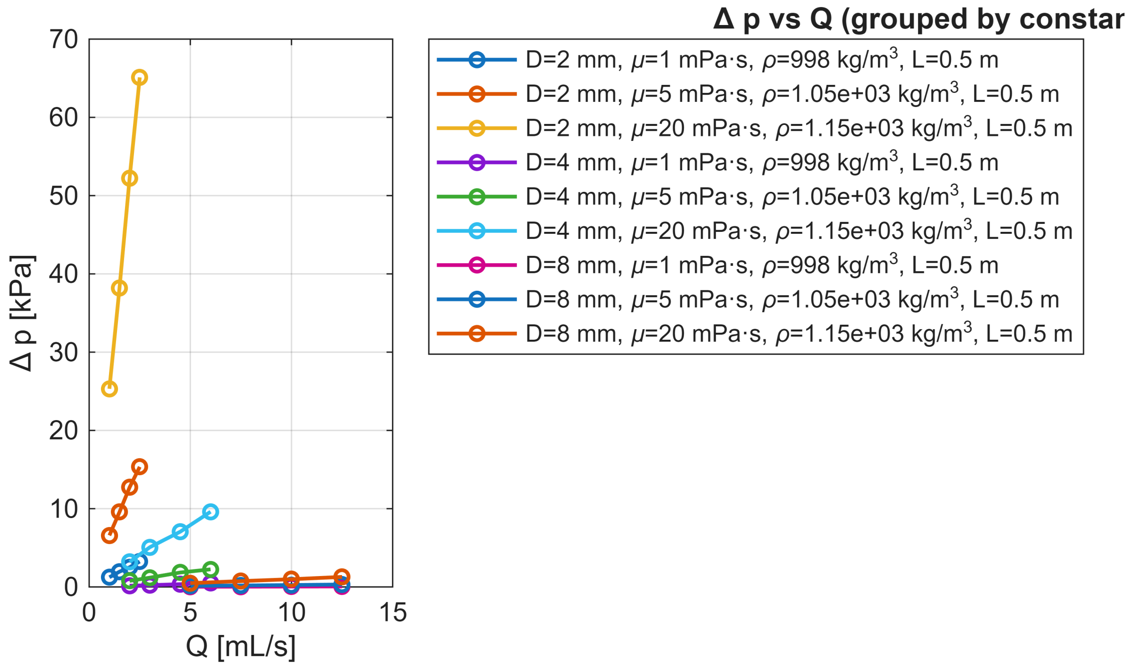
```
%% HW2 Problem 1 — Raw plot: Δp vs Q grouped by constant (D, mu, rho, L)
% 1) Reads pipe_lab.csv
% 2) Plots one curve per unique set of (D, mu, rho, L)

clear; clc; close all;

%% ---- 1) Read the raw data file ----
% Assumes pipe_lab.csv is in the current folder. Otherwise, set full path here:
fname = "pipe_lab.csv";
T = readtable(fname);

% Expected columns (per HW PDF):
```

```
% D_mm, mu_mPa_s, rho_kg_m3, L_m, Q_mL_s, dp_kPa
% If your file has different headers, adjust these names accordingly.
D   = T{:,2};  % [mm]
mu  = T{:,3};  % [mPa*s]
rho = T{:,4};  % [kg/m^3]
L   = T{:,5};  % [m]
Q   = T{:,6};  % [mL/s]
dp  = T{:,7};  % [kPa]

%% ---- 2) Group by constant parameters and plot Q vs Δp ----
% If any of these are floating with tiny roundoff, rounding makes grouping robust.
Dg   = round(D,   6);
mug  = round(mu,  6);
rhog = round(rho, 6);
Lg   = round(L,   6);
```

```
G = findgroups(Dg, mug, rhog, Lg); % Assign group numbers
nG = max(G); % Total number of unique groups
```

```
%fig = figure('Color','w','Visible', 'off');
fig = figure('Color','w');
fig.Theme = "light";
hold on; grid on; box on;

% Plot each group as its own curve
for k = 1:nG
    idx = (G == k);

    % Sort by Q so curves don't zig-zag
    [Qk, order] = sort(Q(idx));
    dpk = dp(idx);
    dpk = dpk(order);

    plot(Qk, dpk, '-o', 'LineWidth', 1.4, 'MarkerSize', 5);
end

xlabel('Q [mL/s]');
ylabel('\Delta p [kPa]');
t = title('\Delta p vs Q (grouped by constant D, \mu, \rho, L)');
t.Units = 'normalized';
t.Position(2) = 1.00;
t.Position(1) = 3.00;

% Build a readable legend label for each curve
labels = strings(nG,1);
for k = 1:nG
    idx = (G == k);
    % pull representative values from first row in group
    i0 = find(idx, 1, 'first');
```

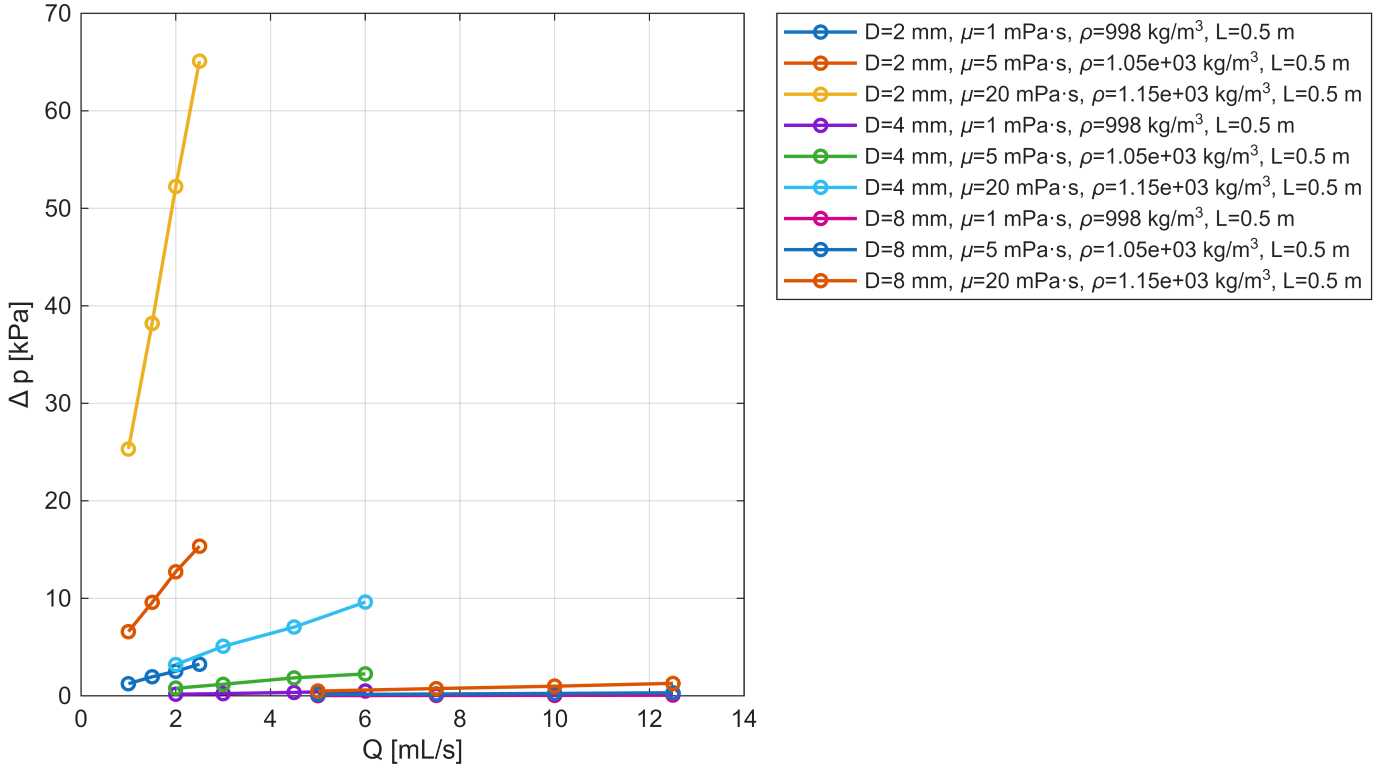
```
    labels(k) = sprintf('D=%.3g mm, \\mu=%.3g mPa·s, \\rho=%.3g kg/m^3, L=%.3g m',
...
        D(i0), mu(i0), rho(i0), L(i0));
end
legend(labels, 'Location', 'bestoutside');
```




```
%% Optional: if you want to also save the figure
%exportgraphics(gcf, "dp_vs_Q_grouped.png", "Resolution", 600);
```

```
%imshow('dp_vs_Q_grouped.png');
```

```
fig = gcf;
set(fig,'Units','inches');
set(fig,'Position',[0.5 0.5 8.5 4.8]);    % wide figure
```




```
set(fig,'PaperPositionMode','auto');     % keep on-screen size for printing/export
```

Discussion

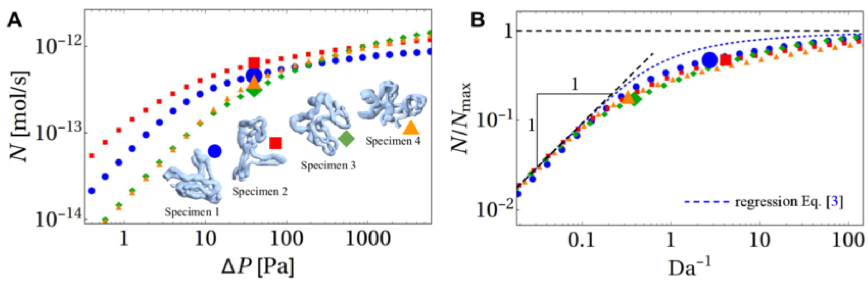
The curves do not collapse since the experiment's input parameters (independent variables - D, μ, ρ, and *L*) have not been properly *nondimensionalized.* In other words, it is not possible to gain insight about the relationships between the parameters from this raw plot.

- [Data collapse - is a way of establishing *scaling*; scaling may be accomplished through **nondimensionalization.** The law of dimensional homogeneity guarantees that every additive term in an equation has the same dimensions. [1]]

When properly nondimensionalized, the data from many experiments will produce a family of curves (or a single curve) that reveals the underlying physics or makes a universal law more apparent. As an illustration of this claim, the two plots shown in the figure below show "raw" computational data (A) and collapse when plotted using suitable dimensionless variables (B) - as presented by Erlich, et.al. in their work on *Physical and geometric determinants of transport in feto-placental microvascular networks*. [2]

```
figure; % Force a new figure window to open
imshow('Computational-data-A-collapse-B.png')
```

```
Warning: PNG library warning:
iCCP: profile 'icc': 0h: PCS illuminant is not D50
Warning: PNG library warning:
iCCP: profile 'icc': 0h: PCS illuminant is not D50
```



2. Buckingham Π analysis

Step 1:

Pressure drop:

$$\Delta p = f(Q, D, L, \mu, \rho) \rightarrow n = 6$$

Note: Only one dependent parameter is analyzed here, Δp; also, it is assumed that this dependent parameter is a function of volumetric flow rate, Q.

Step 2:

Parameter	Δp	Q	D	L	μ	ρ
Primary dimensions	$M^1L^{-1}T^{-2}$	L^3T^{-1}	L^1	L^1	$M^1L^{-1}T^{-1}$	M^1L^{-3}

Step 3:

As an initial guess, set the reduction j as the number of primary dimensions --

$$j = 3$$

(a) Dimensionless Groups Expected

If this initial guess is *correct*, the number of Π's, or dimensionless groups expected is k:

$$k = n - j = 6 - 3 = \mathbf{3}$$

Step 4:

Choose j repeating parameters --

- **Hint of the day:** A *wise* choice of repeating parameters for most fluid flow problems is a **length**, a **velocity**, and a **mass** or **density**.

```
% Call the 'buckinghamPi' sub-routine;
% it computes the Buckingham Pi groups (one set for 1 dependent variable)
% based off the input of repeating parameters.

params = ["dP", "Q", "D", "L", "mu", "rho"];
%depParams = ["dP", "Q"];
dimNames = ["M", "L", "T"];
```

```
% Dimensions:M  L  T    % Parameters:
dimMatrix = [1 -1 -2;   % dP
             0  3 -1;   % Q
             0  1  0;   % D
             0  1  0;   % L
             1 -1 -1;   % mu
             1 -3  0];  % rho
```

```
results = buckinghamPi(params, dimNames, dimMatrix);
```

```
--- Buckingham Pi Setup ---
n = 6 parameters
j = 3 primary dimensions: M, L, T
k = n - j = 3 expected Pi group(s)

Parameters and their dimension exponents (rows):
Param    M    L    T
dP    1    -1    -2
Q    0    3    -1
D    0    1    0
L    0    1    0
mu    1    -1    -1
rho    1    -3    0

Choose 3 repeating parameters.
Rule of thumb (fluids): pick ones that span M, L, T and are independent.
Available: dP, Q, D, L, mu, rho


Repeating parameters chosen: D, Q, rho
Non-repeating parameters:     dP, L, mu

--- Results ---
Computed 3 Pi group(s):

Pi_1 =
```

$$\frac{D^4\,dP}{Q^2\,\rho}$$

Exponents on repeating parameters [D, Q, rho]:

$$\begin{pmatrix} 4 & -2 & -1 \end{pmatrix}$$

Pi_2 =

$$\frac{L}{\mathrm{D}}$$

Exponents on repeating parameters [D, Q, rho]:

$$\begin{pmatrix} -1 & 0 & 0 \end{pmatrix}$$

Pi_3 =

$$\frac{\mathrm{D}\,\mu}{Q\,\rho}$$

Exponents on repeating parameters [D, Q, rho]:

$$\begin{pmatrix} 1 & -1 & -1 \end{pmatrix}$$

```
disp(results.exponents);
```

PiGroup	D	Q	rho
1	4	-2	-1
2	-1	0	0
3	1	-1	-1

```
disp(results.piGroups);
```

$$\left(\begin{array}{ccc} \frac{\mathrm{D}^4\,\mathrm{dP}}{Q^2\,\rho} & \frac{L}{\mathrm{D}} & \frac{\mathrm{D}\,\mu}{Q\,\rho} \end{array}\right)$$

(b) Π Groups

<u>Step 5</u>:

Construct k Π's, and manipulate as necessary.

```
clipboard('copy', latex(results.piGroups))
```

\left(\begin{array}{ccc} \frac{D^4\,\mathrm{dP}}{Q^2\,\rho } & \frac{L}{D} & \frac{D\,\mu }{Q\,\rho } \end{array}\right)

$$\Pi_1 = \left(\frac{D^4\,\Delta \mathrm{P}}{Q^2\,\rho}\right) \qquad (5\text{-}1)$$

$$\Pi_2 = \left(\frac{D\,\mu}{Q\,\rho}\right) \qquad (5\text{-}2)$$

$$\Pi_3 = \left(\frac{L}{D}\right) \qquad (5\text{-}3)$$

(c) Rewrite in Terms of Mean Speed

Solve for Q:

$$Q = \frac{\pi D^2 U}{4} \qquad \text{(5-4)}$$

Sub (5-4) into (5-1):

$$\Pi_{1,modified} = \frac{16}{\pi^2}\frac{\Delta P}{\rho U^2} \qquad \text{(5-5)}$$

$$\Pi_{2,modified} = \frac{4}{\pi}\frac{1}{Re} \qquad \text{(5-6)}$$

Step 6:

Write the functional relationship

$$\Delta P = f(f, AR)$$

$$\Pi_1 = \Phi(\Pi_2, \Pi_3)$$

$$\frac{\Delta P}{\rho U^2} = \Phi\left(Re, \frac{L}{D}\right) \qquad \text{(5-7)}$$

- Note: The pure dimensionless constants are omitted from the functional relationship; they're "absorbed" into the function Φ.

Discussion

This functional relationship unveils the physics -- the physical parameters -- that govern laminar pipe flow; pressure drop is a function of **dynamic pressure** ($\rho U^2/2$), the **Reynolds number**, and the geometry's aspect ratio (L/D).

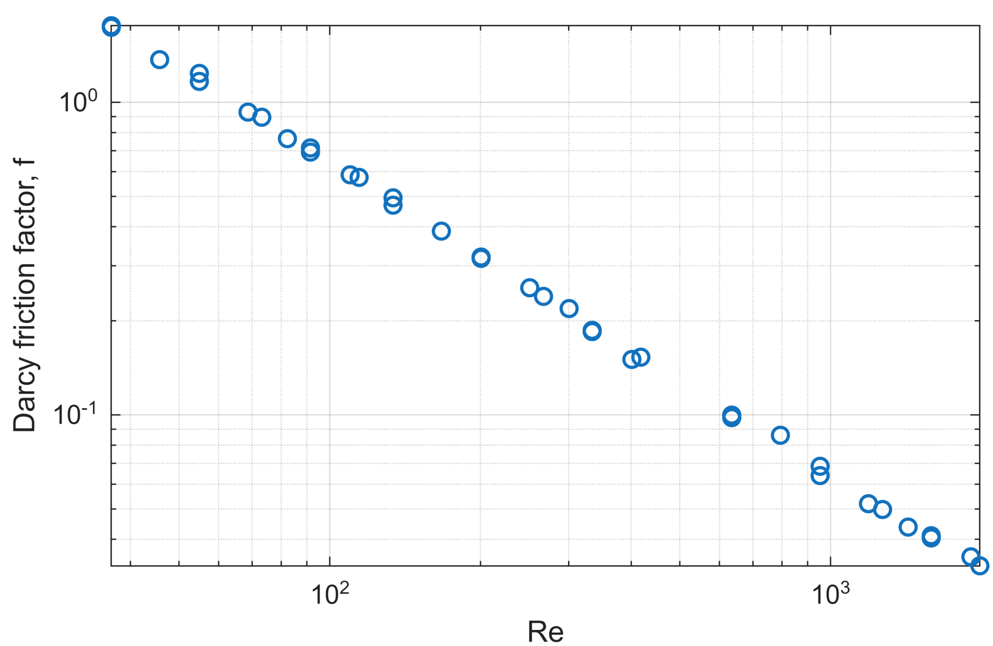
3. Variables that Collapse the Data

plot f- vs - Re on log-log axes

Plot *f* vs *Re* on log-log axes

```
%% HW1 Problem 1.3 — Compute (f, Re) for each run and plot f vs Re (log-log)

clear; clc; close all;

% Read data (pipe_lab.csv)
T = readtable("pipe_lab.csv");

% Columns:
```

```
D_mm     = T{:,2};   % [mm]
mu_mPas = T{:,3};   % [mPa*s]
rho      = T{:,4};   % [kg/m^3]
L        = T{:,5};   % [m]
Q_mLs    = T{:,6};   % [mL/s]
dp_kPa   = T{:,7};   % [kPa]

%% Unit conversions to SI (as required)
D  = D_mm * 1e-3;          % [m]
mu = mu_mPas * 1e-3;       % [Pa*s]
Q  = Q_mLs * 1e-6;         % [m^3/s]
dp = dp_kPa * 1e3;         % [Pa]

%% Mean speed
U = 4*Q ./ (pi*D.^2);      % [m/s]

%% Reynolds number and Darcy friction factor
Re = (rho .* U .* D) ./ mu;
f  = (2*dp .* D) ./ (rho .* U.^2 .* L);   % same as dp / ((L/D)*(rho U^2/2))

%% Plot collapse: f vs Re (log-log)
fig = figure('Color','w');
fig.Theme = "light";
loglog(Re, f, 'o', 'MarkerSize', 6, 'LineWidth', 1.2);
grid on; box on;
xlabel('Re');
ylabel('Darcy friction factor, f');
title('Data collapse: f vs Re');
```




```
% Optional: label some points or color by diameter/viscosity later if you want
```

Discussion

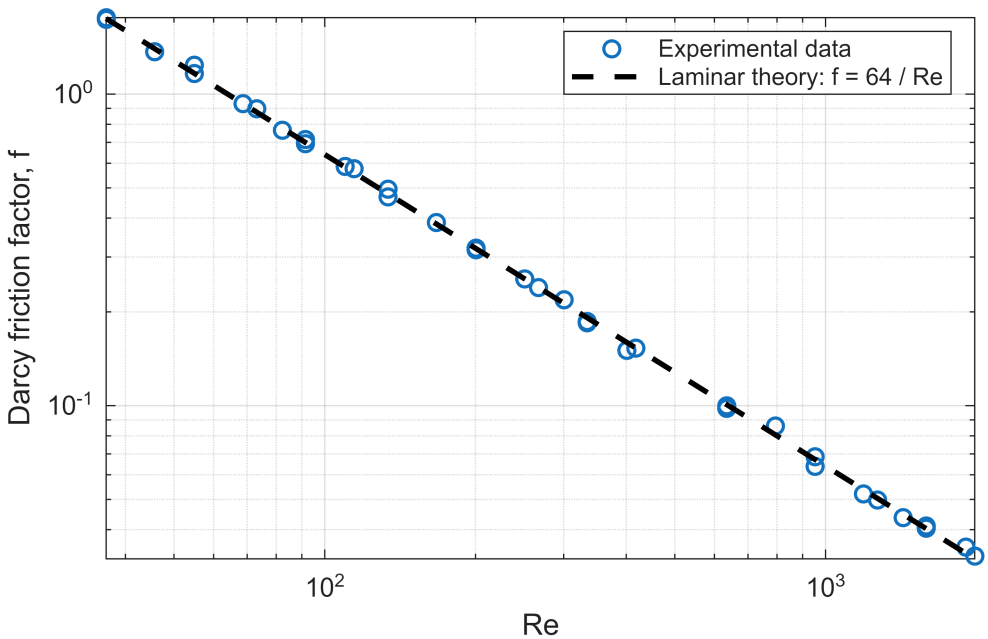
The f -vs- Re plot shows that data from different diameters, viscosities, and densities collapse onto a single curve, indicating the **Reynolds number** and the **Darcy friction factor** captures the relevant physics governing laminar pipe flow.

4. Compare to Laminar Theory

For fully developed laminar pipe flow, theory predicts:

$$f = \frac{64}{Re}$$

```
%% Overlay laminar theory: f = 64 / Re
hold on;

Re_th = logspace(log10(min(Re)), log10(max(Re)), 300);
f_th  = 64 ./ Re_th;

loglog(Re_th, f_th, 'k--', 'LineWidth', 2);

legend('Experimental data', 'Laminar theory: f = 64 / Re', ...
    'Location', 'best');
```



Discussion

The experimental data clearly shows **agreement** to the laminar theory ($f = 64/Re$). The experimental friction factor data collapse well onto the theoretical laminar prediction, indicating fully developed laminar flow and validating the nondimensionalization.

5. Inference - "Effective" Viscosity

For laminar flow [1]:

$$\Delta P = \frac{32\mu LU}{D^2} \rightarrow \mu_{fit} = \frac{\Delta P D^2}{32LU} \quad (1)$$

Or, equivalently, as in practice it is convenient to express the pressure loss for all types of fully developed internal flows as [1]:

$$\Delta P = f\frac{L}{D}\frac{\rho U^2}{2} \quad (2)$$

Substituting (2) into (1) gives:

$$\mu_{fit} = \frac{f\rho D}{64} \quad (3)$$

```
%% Problem 1.5 — Estimate mu_fit and compare to tabulated mu

% --- mu_fit per run (Poiseuille / fRe=64) ---
mu_fit = (dp .* D.^2) ./ (32 .* U .* L);    % [Pa*s]

% Tabulated mu is already in SI (Pa*s) from your conversion earlier: mu

% --- Percent error per run ---
pct_err = 100 * (mu_fit - mu) ./ mu;

% --- Put per-run results into a table (optional but helpful) ---
T_out = table(D, L, rho, mu, mu_fit, pct_err, Re, f, ...
    'VariableNames',
{'D_m','L_m','rho','mu_tab_Pa_s','mu_fit_Pa_s','pct_err','Re','f'});

disp("Per-run viscosity fit summary (all runs):");
```

```
Per-run viscosity fit summary (all runs):
```

```
disp(T_out(1:min(36,height(T_out)), :));
```

D_m	L_m	rho	mu_tab_Pa_s	mu_fit_Pa_s	pct_err	Re	f
0.002	0.5	998	0.001	0.0009731	-2.6903	635.35	0.098022
0.002	0.5	998	0.001	0.0010206	2.0575	953.02	0.068537
0.002	0.5	998	0.001	0.00098837	-1.1633	1270.7	0.04978
0.002	0.5	998	0.001	0.0010196	1.9593	1588.4	0.041082
0.002	0.5	1050	0.005	0.0051727	3.4535	133.69	0.49525
0.002	0.5	1050	0.005	0.005013	0.25999	200.54	0.31998
0.002	0.5	1050	0.005	0.0050002	0.004877	267.38	0.23937
0.002	0.5	1050	0.005	0.0048201	-3.5976	334.23	0.1846
0.002	0.5	1150	0.02	0.019887	-0.5674	36.606	1.7384
0.002	0.5	1150	0.02	0.02	-0.001121	54.908	1.1656
0.002	0.5	1150	0.02	0.020512	2.5603	73.211	0.89656
0.002	0.5	1150	0.02	0.020454	2.2704	91.514	0.71522
0.004	0.5	998	0.001	0.00099033	-0.96741	635.35	0.099758
0.004	0.5	998	0.001	0.00095151	-4.8494	953.02	0.063898

0.004	0.5	998	0.001	0.00097633	-2.3669	1429.5	0.04371
0.004	0.5	998	0.001	0.0010468	4.6763	1906	0.035148
0.004	0.5	1050	0.005	0.0048905	-2.1895	133.69	0.46824
0.004	0.5	1050	0.005	0.0049618	-0.76303	200.54	0.31671
0.004	0.5	1050	0.005	0.0051456	2.9122	300.8	0.21896
0.004	0.5	1050	0.005	0.0047099	-5.8012	401.07	0.15032
0.004	0.5	1150	0.02	0.020134	0.67022	36.606	1.7601
0.004	0.5	1150	0.02	0.021248	6.2412	54.908	1.2383
0.004	0.5	1150	0.02	0.019694	-1.5293	82.363	0.76517
0.004	0.5	1150	0.02	0.020124	0.62101	109.82	0.58641
0.008	0.5	998	0.001	0.0010671	6.7081	794.18	0.085992
0.008	0.5	998	0.001	0.00096813	-3.1868	1191.3	0.052012
0.008	0.5	998	0.001	0.0010019	0.18835	1588.4	0.040369
0.008	0.5	998	0.001	0.0010194	1.9436	1985.5	0.032861
0.008	0.5	1050	0.005	0.00505	1.0009	167.11	0.38681
0.008	0.5	1050	0.005	0.0049926	-0.14714	250.67	0.25494
0.008	0.5	1050	0.005	0.0048664	-2.6711	334.23	0.18637
0.008	0.5	1050	0.005	0.0049952	-0.096767	417.78	0.15304
0.008	0.5	1150	0.02	0.019568	-2.1615	45.757	1.3685
0.008	0.5	1150	0.02	0.019965	-0.17356	68.636	0.93084
0.008	0.5	1150	0.02	0.019815	-0.92325	91.514	0.69289
0.008	0.5	1150	0.02	0.020587	2.9356	114.39	0.5759

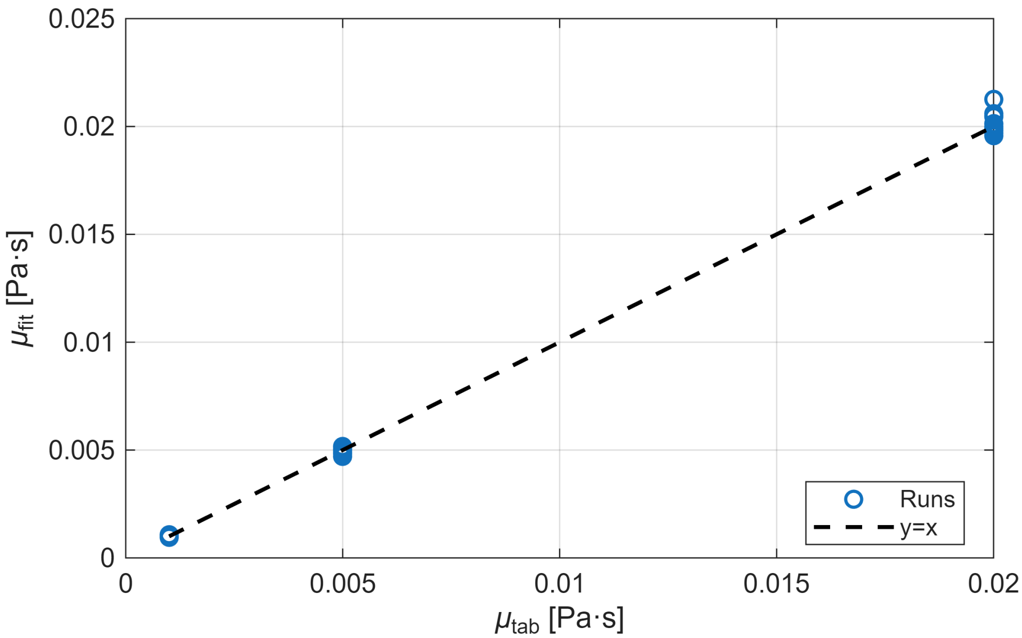
```
% --- Aggregate by "fluid condition"
% Usually viscosity condition is identified by (mu_tab, rho); you can add D and/or
L if needed.
G = findgroups(mu, rho);  % group by tabulated mu and density

mu_tab_group = splitapply(@mean, mu, G);
rho_group    = splitapply(@mean, rho, G);

mu_fit_mean  = splitapply(@mean, mu_fit, G);
mu_fit_std   = splitapply(@std,  mu_fit, G);

N_group      = splitapply(@numel, mu_fit, G);

pct_err_mean = 100 * (mu_fit_mean - mu_tab_group) ./ mu_tab_group;

T_group = table(mu_tab_group, rho_group, N_group, mu_fit_mean, mu_fit_std,
pct_err_mean, ...
    'VariableNames',
{'mu_tab_Pa_s','rho','N','mu_fit_mean_Pa_s','mu_fit_std_Pa_s','pct_err_mean'});

disp("Grouped by (mu_tab, rho):");
```

Grouped by (mu_tab, rho):

```
disp(T_group);
```

mu_tab_Pa_s	rho	N	mu_fit_mean_Pa_s	mu_fit_std_Pa_s	pct_err_mean
0.001	998	12	0.0010019	3.3993e-05	0.19242
0.005	1050	12	0.0049682	0.00013122	-0.63623
0.02	1150	12	0.020166	0.00046746	0.82855

```
% --- Quick visual compare (optional)
fig = figure('Color','w'); hold on; box on; grid on;
fig.Theme = "light";
plot(mu, mu_fit, 'o', 'LineWidth', 1.2);
xline = linspace(min(mu), max(mu), 100);
plot(xline, xline, 'k--', 'LineWidth', 1.5); % y=x perfect agreement
xlabel('\mu_{tab} [Pa·s]');
ylabel('\mu_{fit} [Pa·s]');
title('Viscosity: fitted vs tabulated');
legend('Runs','y=x','Location','best');
```



Discussion

The 'Per-run viscosity fit summary' table shows the effective viscosity to be within 0.001 up to 6.70 % of the tabulated viscosity, indicating **agreement** between to the dimensionalization process, and the fluid conditions of the experimental runs being in a fully-developed laminar flow regime - as assumed. :)

```
%% Color by viscosity
% figure('Color','w'); hold on; box on; grid on;
%
% muvals = unique(mu);
% cmap = lines(numel(muvals));
%
% for k = 1:numel(muvals)
%     idx = (mu == muvals(k));
%     loglog(Re(idx), f(idx), 'o', ...
%         'MarkerSize', 6, ...
%         'LineWidth', 1.2, ...
%         'Color', cmap(k,:));
```

```
% end
%
% xlabel('Re');
% ylabel('Darcy friction factor, f');
% title('Data collapse colored by viscosity');
%
% legend(arrayfun(@(x) sprintf('\\mu = %.2f mPa·s', x*1e3), ...
%     muvals, 'UniformOutput', false), ...
%     'Location','best');
```

Problem 2. Passive scalar transport from Reynolds Transport Theorem

- [A passive scalar is a scalar quantity that is transported by advection (the transfer of a substance, or matter, by bulk motion of a fluid) and diffusion in a flow but does not influence the velocity field or fluid dynamics.]

1. Time rate of change of dye mass

Fundamentals

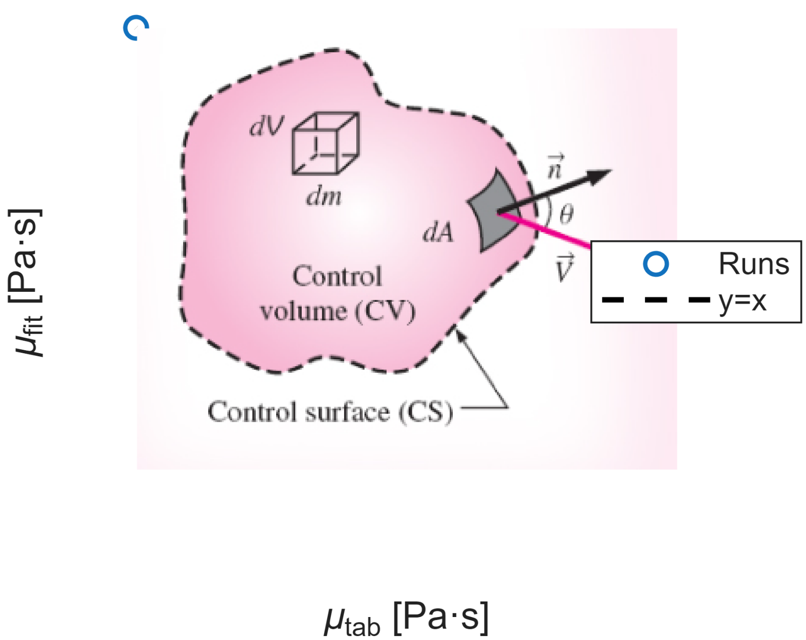
a. Conservation of Mass Principle

$m_{in} - m_{out} = \Delta_{CV}$

The net mass transfer to or from a control volume during a time interval Δt *is equal to the net change in the total mass within the control volume during* Δt. [1]

General Conservation of Mass

```
imshow('element for cons of mass.png')
```



$$\frac{d}{dt}\int_{\mathrm{CV}} \rho \, dV + \int_{\mathrm{CS}} \rho \, (\vec{V} \cdot \vec{n}) \, dA = 0 \qquad \text{(2-1)}$$

"... *the time rate of change of mass within the control volume plus the net mass flow rate through the control surface is equal to zero."*

b. Total Dye Mass:

Since:

$$c = \frac{M_{dye}}{M_{mixture}}$$

then,

$$M_{dye} = cM_{mixture} = c\rho_{mixture}V$$

Generally:

$$M_{dye}(t) = \int_V \rho(x,t)c(x,t) \, dV \qquad \text{(2-2)}$$

c. Time Rate of Change of Dye Mass:

The dye is conserved, so expanding on the *general conservation of mass* expression (2-1) above, gives:

$$\frac{d}{dt}\int_{\mathrm{V}} \rho c \, dV + \oint_{\partial V} (\rho c \vec{u} + \mathbf{j}) \cdot \vec{n} \, dA = 0 \qquad \text{(2-3)}$$

- The left integral represents the time rate of change of mass within the CV (i.e., change in a CV).
- The right integral represents the total flux through the boundary surface (i.e., change in a system).
- Where $(\rho c \vec{u})\vec{n}$ is the **advective/convective** dye flux and $\mathbf{j}\vec{n}$ is the **diffusive dye flux.**

2. Reynolds Transport Theorem and Divergence Theorem for the Differential Equation

d. Reynolds Transport Theorem

For the fixed CV, RTT reduces to:

$$\frac{d}{dt}\int_{\mathrm{V}} \rho c \, dV = \int_V \frac{\partial(\rho c)}{\partial t} dV$$

Since the CV is not moving or deforming with time, the time derivative can be moved inside since it is irrelevant whether we differentiate or integrate first. [1] It is expressed as a partial derivative since density and the dye mass ratio are space-dependent (functions of *x* and *t*).

e. Divergence Theorem on the Surface Flux

$$\oint_{\partial V} \mathbf{G} \cdot \mathbf{n}\, dA = \int_V \nabla \cdot \mathbf{G}\, dV$$

Let:

$$\mathbf{G} = (\rho c \overrightarrow{u} + \mathbf{j})$$

$$\oint_{\partial V} (\rho c \overrightarrow{u} + \mathbf{j}) \cdot \overrightarrow{n}\ dA = \int_V \nabla \cdot (\rho c \overrightarrow{u} + \mathbf{j})\, dV$$

f. Combine Them

$$\frac{\partial(\rho c)}{t} + \nabla \cdot (\rho c \overrightarrow{u} - \rho D \nabla c) = 0 \qquad \text{(2-4)}$$

3. Incompressible Flow and Constant Density and Diffusivity

Incompressible flow:

$\frac{\partial \rho}{\partial t} = 0$ (density does not change with time)

and

$$\nabla \cdot \overrightarrow{u} = 0$$

Expand $\nabla \cdot (c \overrightarrow{u})$

$$\nabla \cdot (c \overrightarrow{u}) = \overrightarrow{u} \cdot c + c(\nabla \cdot \overrightarrow{u}) = \overrightarrow{u} \cdot c$$

(2-4) becomes:

$$\frac{\partial c}{t} + \overrightarrow{u} \cdot \nabla c = D \nabla^2 c \qquad \text{(2-5)}$$

- This is the simplified passive-scalar transport equation.

Problem 3. Streamlines, Pathlines, Streaklines

Consider the *unsteady* 2D velocity field

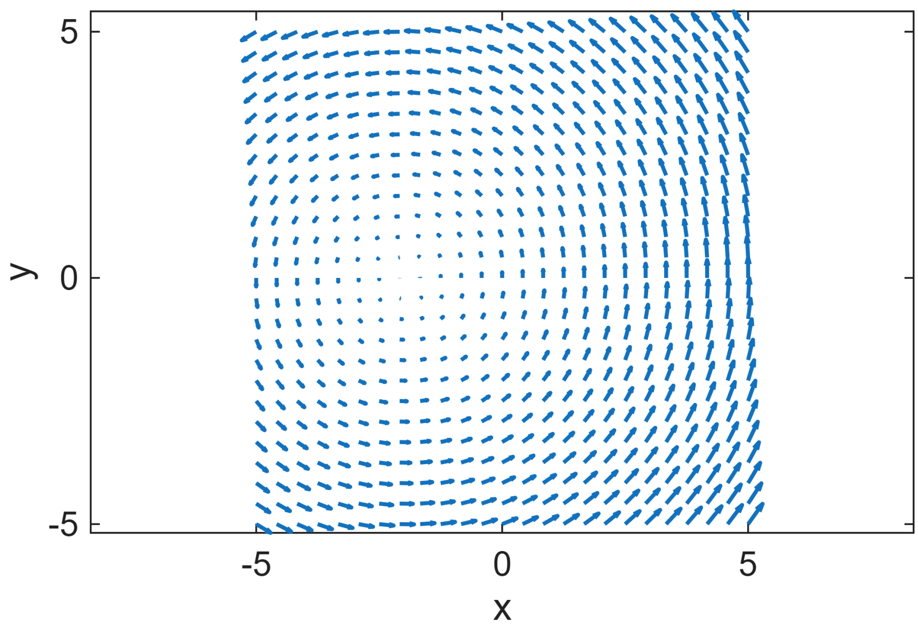
$$u(x, y, t) = -y, \quad v(x, y, t) = x + \beta t,$$

where β is a constant.

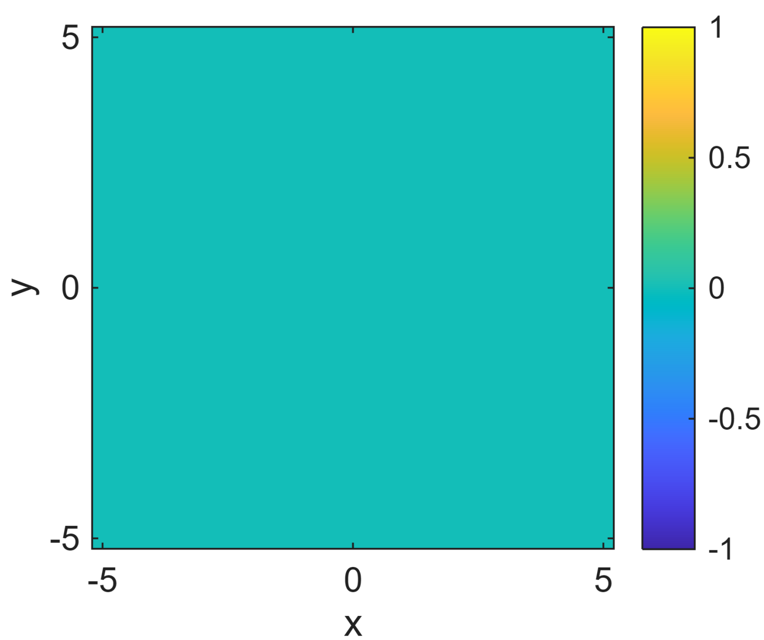
1. Show that the flow is incompressible

[Continue here... -ERODRIGUEZ, 12FEB2026 20:32]

```
%% HW1 Problem 3.1 — Visualize incompressibility (divergence = 0)
% Velocity field: u(x,y,t) = -y,  v(x,y,t) = x + beta*t
% We visualize the vector field and compute div(u) on a grid to confirm it's ~0.

clear; clc; close all;

%% Parameters
beta = 1;          % choose any constant
t0   = 2;          % choose a time to visualize the unsteady field

%% Grid
x = linspace(-5, 5, 25);
y = linspace(-5, 5, 25);
[X, Y] = meshgrid(x, y);

%% Velocity field at time t0
U = -Y;
V =  X + beta*t0;

%% Numerical divergence (finite-difference via gradient)
dx = x(2) - x(1);
dy = y(2) - y(1);

[dUdx, dUdy] = gradient(U, dx, dy);
[dVdx, dVdy] = gradient(V, dx, dy);

div = dUdx + dVdy;   % ∂u/∂x + ∂v/∂y

%% Plot 1: Vector field (quiver)
figure('Color','w'); box on; grid on;
quiver(X, Y, U, V, 'LineWidth', 1.1);
axis equal;
xlabel('x'); ylabel('y');
title(sprintf('Velocity field at t = %.2f (\\beta = %.2f)', t0, beta));
```




```
%% Plot 2: Divergence field (should be ~0 everywhere)
figure('Color','w'); box on;
imagesc(x, y, div);
set(gca, 'YDir', 'normal');
axis equal tight;
colorbar;
xlabel('x'); ylabel('y');
title('Numerical divergence: \nabla \cdot \mathbf{u} = \partial u/\partial x +
\partial v/\partial y');
```



```
Warning: Error in state of SceneNode.
String scalar or character vector must have valid interpreter syntax:
Numerical divergence: \nabla \cdot \mathbf{u} = \partial u/\partial x + \partial v/\partial y
```

```
%% Print a quick numerical check
fprintf('Max |div| on grid = %.3e\n', max(abs(div(:))));
```

```
Max |div| on grid = 0.000e+00
```

```
% NOTE:
% Analytically, ∂u/∂x = 0 and ∂v/∂y = 0, so divergence is exactly 0.
% Any tiny nonzero values printed are numerical roundoff.
```

Applications

[pending]

References

[1] Cengel, Y. (2018). *Fluid Mechanics: Fundamentals and Applications* (4th ed.). New York: McGraw-Hill.

[2] Erlich, Alexander & Pearce, Philip & Plitman, Romina & Jensen, Oliver & Chernyavsky, Igor. (2018). Physical and geometric determinants of transport in feto-placental microvascular networks. 10.48550/arXiv.1809.00749.