# Fully-Developed Flow in a Pipe:
# A CFD Solution

Gerald Recktenwald*

January 22, 2020

**Abstract**

A CFD model of fully-developed laminar flow in a pipe is derived and implemented. This well-known problem is used to introduce the basic concepts of CFD including: the finite-volume mesh, the discrete nature of the numerical solution, and the dependence of the result on the mesh refinement. A MATLAB implementation of the numerical model is provided. Numerical results are presented for a sequence of finer meshes, and the dependency of the truncation error on mesh size is verified.

## One-Dimensional Fully-Developed Flow

The left side of Figure 1 shows the geometry of a simple round pipe of radius $R$. The governing equation for fully-developed flow in a pipe is

$$\frac{\mu}{r}\frac{d}{dr}\left(r\frac{du}{dr}\right) - \frac{dp}{dx} = 0 \tag{1}$$

where $u$ is the velocity component along the pipe axis ($x$ direction), $\mu$ is the dynamic viscosity, and $p$ is the pressure. The pressure gradient is a specified constant. The velocity is a function of $r$ alone. The boundary conditions are

$$\left.\frac{du}{dr}\right|_{r=0} = 0 \qquad \text{(symmetry)} \tag{2}$$

$$u(R) = 0 \qquad \text{(no slip)} \tag{3}$$

*Mechanical Engineering Department, Portland State University, Portland, OR, 97201, gerry@me.pdx.edu

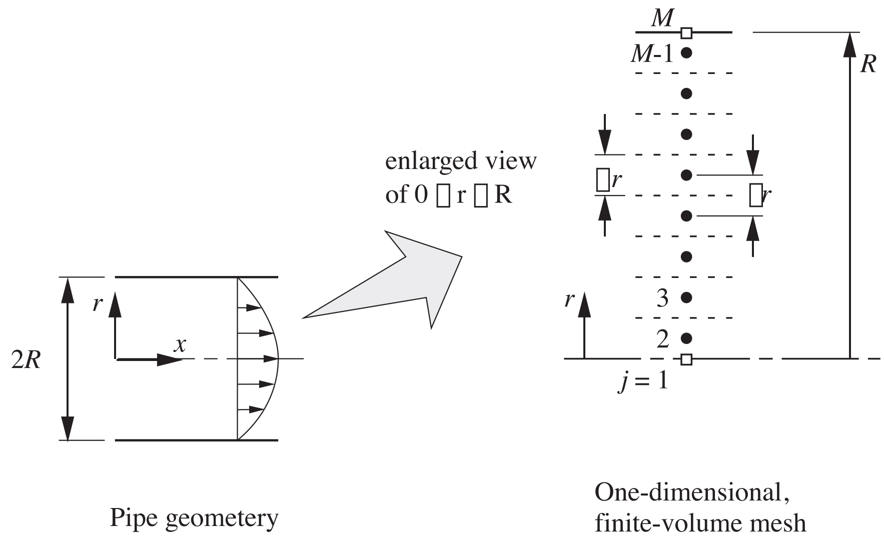


Figure 1: Geometry of fully-developed flow in a pipe.

**Analytical Solution**

The exact solution to Equation (1) subject to the boundary conditions is

$$u(r) = \frac{R^2}{4\mu}\left(-\frac{\partial p}{\partial x}\right)\left[1 - \left(\frac{r}{R}\right)^2\right] \tag{4}$$

The maximum velocity in the pipe is at the centerline. Evaluating the preceding formula for $r = 0$ gives

$$u_{\max} = \frac{R^2}{4\mu}\left(-\frac{\partial p}{\partial x}\right) \tag{5}$$

For flow in the positive $x$ direction, $dp/dx < 0$. Combining Equation (4) and (5) gives a more compact expression for the velocity profile.

$$u(r) = u_{\max}\left[1 - \left(\frac{r}{R}\right)^2\right] \tag{6}$$

The average velocity in the pipe is

$$u_{\text{ave}} = \frac{1}{A}\int_A u\,dA = \frac{1}{\pi R^2}\int_0^R u\,2\pi r\,dr \tag{7}$$

$$= \frac{u_{\max}}{R^2}\int_0^R \left[1 - \left(\frac{r}{R}\right)^2\right] r\,dr$$

$$= \frac{u_{\max}}{2} \tag{8}$$

The shear stress at the wall is

$$\tau_w = \mu \left| \left( \frac{du}{dr} \right)_{r=R} \right| \tag{9}$$

The absolute value sign is necessary because $du/dr$ is negative at the wall[1]. Using Equation (6) in Equation (9) gives

$$\tau_w = \mu \left| - \left( \frac{2u_{\text{max}}}{R} \right)_{r=R} \right| = \frac{4\mu u_{\text{ave}}}{R} \tag{10}$$

The definition of the Darcy Friction factor is

$$f = \frac{8\tau_w}{\rho u_{\text{ave}}^2} \tag{11}$$

Substituting Equation (10) into Equation (11) gives

$$f_{\text{pipe}} = \frac{32\mu}{\rho u_{\text{ave}} R} = \frac{64}{\text{Re}}$$

or

$$f_{\text{pipe}}\text{Re} = 64 \tag{12}$$

where

$$\text{Re} = \frac{\rho u_{\text{ave}} D}{\mu} \tag{13}$$

is the Reynolds number.

## Finite Volume Mesh

The exact solution to Equation (1) is continuous. In other words, the function for $u(r)$ in Equation (4) is defined at all points in $0 \leq r \leq R$. The numerical solution is said to be *discrete* because it is only obtained at a finite number of points called *nodes* or *vertices*. The nodes are represented by the solid dots and open squares in the right side of Figure 1. The solid dots are interior nodes. The open squares are boundary nodes, which are used for boundary conditions, as will be explained later. The radial location of the nodes is $r_j$ where $j = 1, \ldots, M$ is the node index, and $M$ is the total number of nodes (both boundary and interior).

In the finite-volume method, the nodes are usually located at the center of discrete volumes called *cells* or *control volumes*. For fully developed flow in

[1] To be more precise, the $x$-direction shear stress on the $+r$ face of the control volume is $\tau_{rx} = \mu \frac{du}{dr}$. This stress is positive when it acts in the positive $x$ direction. Evaluating this formula gives $\tau_{rx} < 0$ because the shear stress is in the negative direction on the face of the control volume adjacent to the wall. Thus, the shear stress *in the direction opposite to the main flow* is $\tau_w = -\mu \frac{du}{dr}$. Since a negative shear stress seems a bit contrived, at least without this explanation, I resorted to the even cheaper trick of using the absolute value. Aren't you glad you read this footnote?

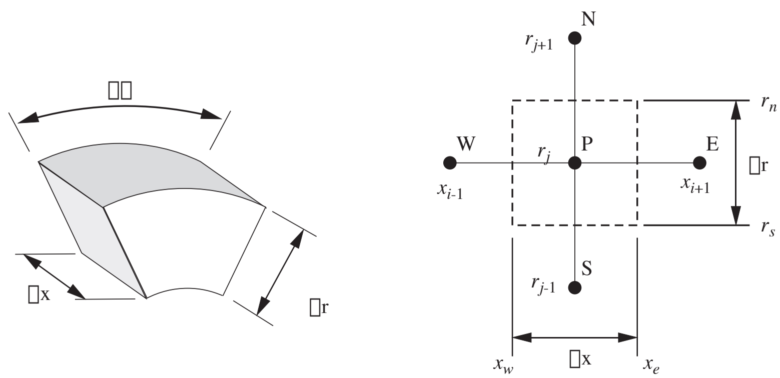


Figure 2: Control volume in axisymmetric coordinates.

a round pipe, the axial velocity is a function of $r$ only. The problem is one-dimensional, and the control volumes are radial slabs delineated by dashed lines in Figure 1. For simplicity, we will choose control volumes of uniform thickness $\Delta r$.

$$\Delta r = \frac{R}{M-2}. \tag{14}$$

Since the interior nodes are located in the center of the control volumes, and since the control volumes have uniform thickness, the interior nodes are uniformly spaced a distance $\delta r = \Delta r$ apart. There is no need to have a uniform mesh, but for this simple problem it is both convenient and preferred. Although the mesh is uniform, the distance between the boundary nodes and the nearest interior nodes is $(\delta r)/2$. This is a consequence of locating the nodes at the geometric center of the control volumes.

Figure 2 shows two views of a typical axisymmetric control volume. On the left is a three-dimensional representation. On the right is a two-dimensional view showing the extent of a control volumn in the $r$ and $x$ directions. In the two-dimensional view, there are nodes in the $x$-direction, but these nodes do not play a role in this simple model.

The two-dimensional view of the control volume identifies nodes with the so-called *compass point* notation, which aides in the development of the discrete model. The point in the center of the control volume is called "P", and is located at $r_j$. Node $N$ is at $r_{j+1}$, and node $S$ is at $r_{j-1}$. The labels "E", "W", "N", and "S" are mnemonic devices referring to the north, south, east, and west directions on a compass. The control volume faces in the $r$-direction are at $r_n$ and $r_s$, where the lower case "n" and "s" refers to the location of the north and south control volume *faces*, not the north and south neighbor nodes N and S.

## Finite Volume Approximation

The discrete model of the flow is obtained by integrating the governing equation over a typical control volume. Multiply Equation (1) by $r\,dr$ and integrate with respect to $r$.

$$\int_{r_s}^{r_n} \frac{\mu}{r}\frac{d}{dr}\left(r\frac{du}{dr}\right) r\,dr - \int_{r_s}^{r_n} \frac{dp}{dx} r\,dr = 0 \tag{15}$$

The limits of the integrals are $r_s$, and $r_n$, the location of the control volume faces.

Direct evaluation of the integral in the first term on the left side of Equation (15) gives

$$\int_{r_s}^{r_n} \frac{\mu}{r}\frac{d}{dr}\left(r\frac{du}{dr}\right) r\,dr = \mu\left[\left(r\frac{du}{dr}\right)_n - \left(r\frac{du}{dr}\right)_s\right] \tag{16}$$

Since $dp/dx$ is a constant, direct evaluation of the integral in the second term on the left hand side of Equation (15) gives

$$-\int_{r_s}^{r_n} \frac{dp}{dx} r\,dr = \frac{dp}{dx}\int_{r_s}^{r_n} r\,dr \approx -\frac{dp}{dx}\, r_P \Delta r \tag{17}$$

where $r_P = r_j$. In the last step the exact evaluation of the integral is avoided to make subsequent algebraic steps more convenient. Substituting Equations (16) and (17) into Equation (15) gives

$$\mu\left[\left(r\frac{du}{dr}\right)_n - \left(r\frac{du}{dr}\right)_s\right] - \frac{dp}{dx}\, r_P\, \Delta r = 0 \tag{18}$$

The derivative terms in Equation (18) are approximated by *finite differences*

$$\left(r\frac{du}{dr}\right)_n \approx r_n \frac{u_{j+1} - u_j}{r_{j+1} - r_j} = r_n \frac{u_{j+1} - u_j}{(\delta r)_j} \tag{19}$$

$$\left(r\frac{du}{dr}\right)_s \approx r_s \frac{u_j - u_{j-1}}{r_j - r_{j-1}} = r_s \frac{u_j - u_{j-1}}{(\delta r)_{j-1}} \tag{20}$$

where

$$(\delta r)_j = r_{j+1} - r_j \qquad\qquad (\delta r)_{j-1} = r_j - r_{j-1}$$

Substituting Equations (19) and (20) into Equation (18) and rearranging gives

$$-a_S u_{j-1} + a_P u_j - a_N u_{j+1} = -\frac{dp}{dx} \tag{21}$$

where

$$a_S = \frac{\mu r_s}{(\delta r)_{j-1} r_j \Delta r}, \qquad a_N = \frac{\mu r_n}{(\delta r)_j r_j \Delta r}, \qquad a_P = a_S + a_N. \tag{22}$$

Equation (21) applies to each interior node in the computational domain. The boundary nodes need special treatment as described below. Since there are $M-2$ interior nodes, there are $M-2$ versions of Equation (21) that must be simultaneously satisfied by the set of unknown $u_j$ values.

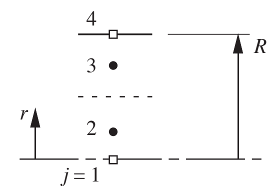


Figure 3: Finite volume mesh with two interior nodes and two boundary nodes.

**Boundary Conditions**

At $r = 0$, we use a finite-difference approximation to Equation (2)

$$\left.\frac{du}{dr}\right|_{r=0} \approx \frac{u_2 - u_1}{r_2 - r_1} = 0$$
$$\Longrightarrow \quad u_2 = u_1 \quad \text{or} \quad u_1 - u_2 = 0 \tag{23}$$

At $r = R$ the velocity is zero. Therefore, one boundary condition is

$$u_M = 0 \tag{24}$$

**System of Equations**

Consider the mesh with just two interior nodes as depicted in Figure 3. Writing the boundary condition equations and the two forms of Equation (21) for $u_2$ and $u_3$ gives

$$\begin{aligned}
u_1 - u_2 &= 0 && \text{(node 1)} \\
-a_{S,2}u_1 + a_{P,2}u_2 - a_{N,2}u_3 &= -\frac{dp}{dx} && \text{(node 2)} \\
-a_{S,3}u_2 + a_{P,3}u_3 - a_{N,3}u_4 &= -\frac{dp}{dx} && \text{(node 3)} \\
u_4 &= 0 && \text{(node 4)}
\end{aligned}$$

This is a system of four equations in four unknowns that can be written

$$\begin{bmatrix} 1 & -1 & 0 & 0 \\ -a_{S,2} & a_{P,2} & -a_{N,2} & 0 \\ 0 & -a_{S,3} & a_{P,3} & -a_{N,3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{dp}{dx} \\ -\frac{dp}{dx} \\ 0 \end{bmatrix} \tag{25}$$

The solution to this system gives the velocities at the boundary and interior nodes.

In general, for $M$ total nodes in the domain, the equations are

$$u_1 - u_2 = 0$$

$$-a_{S,2}u_1 + a_{P,2}u_2 - a_{N,2}u_3 = -\frac{dp}{dx}$$

$$-a_{S,3}u_2 + a_{P,3}u_3 - a_{N,3}u_4 = -\frac{dp}{dx}$$

$$\vdots$$

$$-a_{S,M-1}u_{M-2} + a_{P,M-1}u_{M-1} - a_{N,M-1}u_M = -\frac{dp}{dx}$$

$$u_M = 0$$

and the matrix form of the system of equations is

$$\begin{bmatrix} 1 & -1 & 0 & \cdots & & 0 \\ -a_{S,2} & a_{P,2} & -a_{N,2} & 0 & \cdots & 0 \\ 0 & -a_{S,3} & a_{P,3} & -a_{N,3} & 0 & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & & & -a_{S,M-1} & a_{P,M-1} & -a_{N,M-1} \\ 0 & 0 & & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{M-1} \\ u_M \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{dp}{dx} \\ -\frac{dp}{dx} \\ \vdots \\ -\frac{dp}{dx} \\ 0 \end{bmatrix} \tag{26}$$

Equation (26) is a tridiagonal system of equations. The solution to this equation gives the nodal values of $u_j$.

**Extracting Results from the Solution**

The solution to Equation (26) gives the velocity profile, $u_j = f(r_j)$ as a set of discrete $(r_j, u_j)$ pairs. This data is interesting insofar as it shows the shape of the velocity profile. Additional information can be obtained from the discrete $u_j$ data.

The shear stress at the wall can be computed from the discrete equivalent of Equation (9). Using a finite-difference approximation to $du/dr$ we get[2].

$$\tau_w \approx \mu \frac{u_{M-1} - u_m}{r_M - r_{M-1}} \tag{27}$$

The average velocity is computed by the discrete analog of Equation (7)

$$u_{\text{ave}} = \frac{1}{A}\int_A u\,dA \approx \frac{1}{\pi R^2}\sum_{i=2}^{M-1} u_j 2\pi r_j \Delta r = \frac{2}{R^2}\sum_{i=2}^{M-1} u_j r_j \Delta r \tag{28}$$

[2]Note that the sign of $\tau_w$ will be positive because $u_{M-1} > u_m$ and $r_M > r_{M-1}$

**Computational Procedure**

The following steps organize the preceding equations into a sequential computational procedure.

1. Define the mesh: $\Delta r$, $r_j$, and $(\delta r)_j$
2. Compute coefficients for each node using Equation (22)
3. Store coefficients in a matrix (or equivalent data structure).
4. Compute and store the right hand side vector in Equation (26)
5. Make any adjustments necessary to implement boundary conditions
6. Solve system of equations to obtain the $u_j$.
7. Compute $\tau_w$, $f$, $f\,\mathrm{Re}$, and any other engineering quantities from the $u_j$ values.

**Truncation Error**

The truncation error for the approximation leading to Equation (21) is $\mathcal{O}\left(\Delta r^2\right)$. For convenience let

$$h = \Delta r$$

If $u_{ex}(r_j)$ is the exact solution and $u_j(r_j)$ is the numerical solution, then at any point in the domain the error is

$$e_j \equiv u_j(r_j) - u_{ex}(r_j) \sim \mathcal{O}\left(h^2\right) \tag{29}$$

The largest error in the domain should also obey the order of magnitude estimate

$$\|e_j\|_\infty \sim \mathcal{O}\left(h^2\right) \tag{30}$$

This estimate of the truncation error can be used to check the correctness of any computer code that implements a control-volume finite-difference model of Equation (1). If the code is working correctly, doubling the number of control volumes should reduce the largest error by a factor of four.

## Matlab Implementation

The numerical model is implemented in the Matlab codes listed in Table 1. The `fullyDev1dr` function computes the finite-volume coefficients and stores them in a sparse tridiagonal matrix. The `demoPipe1d` and `refinePipe1d` functions are main programs that use `fullyDev1dr`. The `demoPipe1d` obtains the velocity profile and friction factor for a single set of parameters. The default values are $M = 4$, $\mu = 1$, $dp/dx = -1$. Running `demoPipe1d` for eight nodes (six internal and two boundary nodes) gives

| m-file function | Description |
|---|---|
| `fullyDev1dr` | Evaluates control-volume, finite-difference coefficients |
| `demoPipe1d` | Uses `fullyDev1dr` to solve the fully-developed flow problem and plot the velocity profile |
| `refinePipe1d` | Verifies that the numerical solution approaches the exact solution as the mesh is refined. |

Table 1: MATLAB functions used implement and test the finite-volume approximation to one-dimensional, fully-developed, laminar flow in a pipe.

```
>> demoPipe1d(8)
fRe =
   62.2703
umax = 0.250000    max(u) = 0.250000
```

and the velocity profile plot in Figure 4. The maximum velocity predicted by the numerical model is in exact agreement with the analytical solution. The $f$Re product predicted by the numerical model is in error by three percent. This error is due to the error in the velocity gradient at the wall obtained by the numerical model. Refining the mesh reduces this error.

The `refinePipe1d` function obtains the numerical model on a series of finer meshes. Running `refinePipe1d` gives the following output

```
>> refinePipe1d

   m      Delta r       fRe       max error     error ratio
   4      0.500000    51.2000     1.563e-002
   8      0.166667    62.2703     1.736e-003     9.000
  16      0.071429    63.6751     3.189e-004     5.444
  32      0.033333    63.9290     6.944e-005     4.592
  64      0.016129    63.9834     1.626e-005     4.271
 128      0.007937    63.9960     3.937e-006     4.130
 256      0.003937    63.9990     9.688e-007     4.064
 512      0.001961    63.9998     2.403e-007     4.032
```

The last column is the ratio of truncation errors on subsequent runs of the model. As $M$ increases, doubling the number of control volumes reduces the truncation error by a factor of four, as predicted by Equation (30).

Figure 5 shows the velocity profiles obtained with a sequence of finer meshes. As the control volume thickness is reduced, the velocity profile from the numerical model quickly approaches the exact velocity profile.

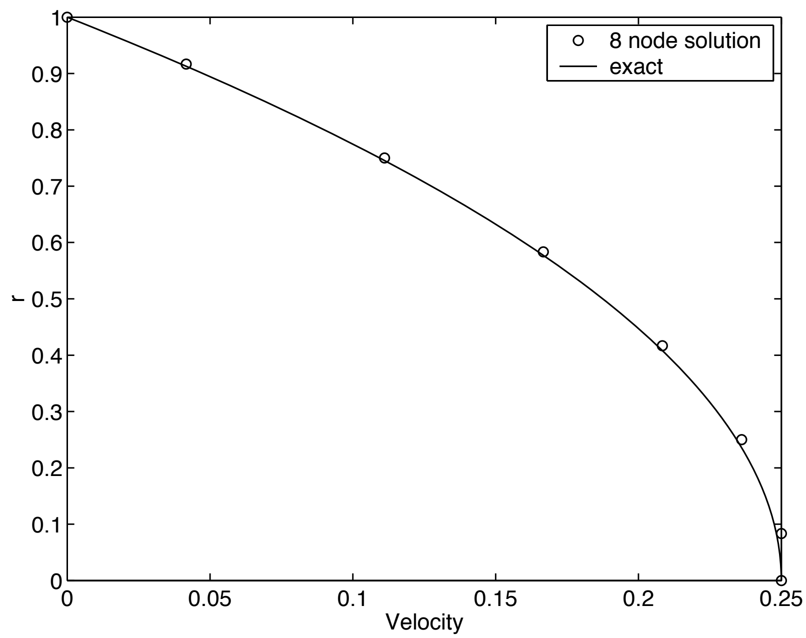


Figure 4: Velocity profile from the numerical model with six internal nodes and two boundary nodes.

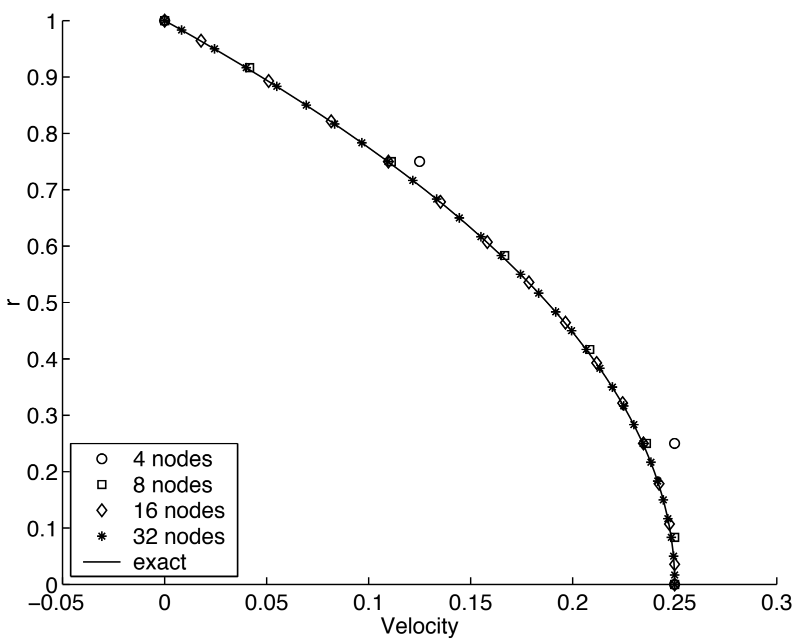


Figure 5: Velocity profiles for a series of finer meshes.

```
function [ap,an,as,b,r] = fullyDev1dr(m,rout,mu,dpdx)
% fullyDev1dr  CVFD coefficients for 1D fully developed flow in a pipe
%
% Synopsis:  [ap,an,as,b,r] = fullyDev1dr
%            [ap,an,as,b,r] = fullyDev1dr(m)
%            [ap,an,as,b,r] = fullyDev1dr(m,rout)
%            [ap,an,as,b,r] = fullyDev1dr(m,rout,mu)
%            [ap,an,as,b,r] = fullyDev1dr(m,rout,mu,dpdx)
%
% Input: m = (optional) number of nodes in range 0 <= r <= rout;
%            m-2 interior nodes;  Default:  m = 4
%        rout = (optional) outer radiums;      Default:  rout = 1
%        mu   = (optional) dynamic viscosity;  Default:  mu = 1
%        dpdx = (optional) pressure gradient;  Default:  dpdx = -1
%        verbose = (optional) flag to control printing of coefficients;
%                  Default:  verbose = 0, no printing
%
% Output: ap,an,as = diagonals of the coefficient matrix for 3 point
%                    finite volume scheme assuming centerline is symmetry
%                    BC and wall is no slip
%         b = right hand side vector
%         r = vector of radial positions of cell centers

if nargin<1, m = 4;        end
if nargin<2, rout = 1;     end
if nargin<3, mu = 1;       end
if nargin<4, dpdx = -1;    end

% --- Mesh constants
Deltar = rout/(m-2);    %  height of control volume = rn - rs
dr2 = Deltar/2;         %  half-height of control volume
r = [0;  (dr2:Deltar:(rout-dr2))'; rout];   %  node locations, column vector
dr = [0; diff(r)];                          %  dr(j) = r(j) - r(j-1)
rn = [0; r(2:m-1)+dr2; rout];               %  position of north face

% --- Compute CVFD coefficients
an = zeros(m,1);  as = an;
for j=2:m-1
  as(j) = mu*rn(j-1)/(dr(j)*Deltar*r(j));
  an(j) = mu*rn(j)/(dr(j+1)*Deltar*r(j));
end
ap = an + as;

% --- Adjust boundary coefficients and store rhs vector
ap(1) = 1;   an(1) = 1;       %  symmetry boundary condition
as(m) = 0;   ap(m) =  1;      %  zero slip boundary condition
b = [0; -dpdx*ones(m-2,1); 0];
```

Listing 1: The `fullyDev1dr` function computes the finite volume coefficients for one-dimensional, fully-developed, laminar flow in a pipe.

```
function demoPipe1d(m,rout,mu,dpdx)
% demoPipe1d  Test finite volume solution to 1D fully-developed pipe flow
%
% Synopsis:  demoPipe1d
%            demoPipe1d(m)
%            demoPipe1d(m,rout)
%            demoPipe1d(m,rout,mu)
%            demoPipe1d(m,rout,mu,dpdx)
%
% Input:  m = (optional) total number of nodes (including boundary nodes)
%             in the model.  There are m-2 control volumes.  Default:  m = 4
%         rout = (optional) outer radius of the pipe.  Default:  rout = 1
%         mu = (optional) viscosity.  Default:  mu = 1
%         dpdx = (optional) pressure gradient.  Default:  dpdx = -1
%
% Output:  Plot of velocity profile, print out out f*Re, maximum velocity and
%          elapsed time to solve the problem.

if nargin<1, m = 4;        end
if nargin<2, rout = 1;     end
if nargin<3, mu = 1;       end
if nargin<4, dpdx = -1;    end

% --- Get CVFD coefficients and solve the system
[ap,an,as,b,r] = fullyDev1dr(m,rout,mu,dpdx);
u = tridiagSolve(ap,-an,-as,b);   %  Note change of sign for an and as

% --- Compute overal results from numerical solution
m = length(r);
tauw = mu * (u(m-1) - u(m))/(r(m) - r(m-1));
Deltar = rout/(m-2);
uave = sum(u.*r*Deltar)*2/rout^2;
fRe = 16*tauw/(mu*uave)

% --- Evaluate exact solution
umax = rout^2/(4*mu)*(-dpdx);
fprintf('umax = %f    max(u) = %f\n',umax,max(u));
fprintf('Elapsed time for %d node solution is %f seconds\n',m,et);

re = linspace(0,rout);
ue = umax*(1- (re/rout).^2);
plot(u,r,'o',ue,re,'-');
legend(sprintf('%d node solution',m),'exact');
xlabel('Velocity');   ylabel('r');
```

Listing 2: The `demoPipe1d` function solves the finite volume model for one-dimensional, fully-developed, laminar flow in a pipe.

```
function refinePipe1d(rout,mu,dpdx)
% refinePipe1d  Mesh refinement study for 1D fully-developed pipe flow

if nargin<1, rout = 1;     end
if nargin<2, mu = 1;       end
if nargin<3, dpdx = -1;    end

% --- Prepare for refinement study
umax = rout^2/(4*mu)*(-dpdx);       %  exact value of maxium velocity
mm = [4 8 16 32 64 128 256 512];    %  sequence of finer meshes
Deltar = rout./(mm-2);              %  CV sizes
symbols = ['ro';'bs';'cd';'m*';'y^';'kh';'gv';'r>';'b<'];  % used in plots
fRe = zeros(size(mm));  err = fRe;
legstr = cell(1,length(mm)+1);    nplots = 0;

for i = 1:length(mm)

  % --- Get CVFD coefficients and solve the system
  m = mm(i);
  [ap,an,as,b,r] = fullyDev1dr(m,rout,mu,dpdx);
  u = tridiagSolve(ap,-an,-as,b);   %  Note change of sign for an and as

  % --- Compute overal results from numerical solution
  tauw = mu * (u(m-1) - u(m))/(r(m) - r(m-1));   %  wall shear stress
  uave = sum(u.*r*Deltar(i))*2/rout^2;           %  average velocity
  fRe(i) = 16*tauw/(mu*uave);
  uex = umax*(1 - (r/rout).^2);    %  exact solution on this grid
  err(i) = norm(u-uex,inf);        %  maximum error

  if m<=32   %  keep plot from getting too crowded
     hold on;  plot(u,r,symbols(i,:));
     nplots = nplots + 1;
     legstr{i} = sprintf('%d nodes',m);
  end

end

% --- Evaluate exact solution and add it to the plot
re = linspace(0,rout);   ue = umax*(1- (re/rout).^2);
plot(ue,re,'-');  xlabel('Velocity');   ylabel('r');
legstr{nplots+1} = 'Exact';   %  Add last legend entry
% legstr{1:(nplots+1)} only includes non-empty legend strings
legend(legstr{1:(nplots+1)},'Location','northeast')
hold off

fprintf('\n   m      Delta r       fRe       max error   error ratio\n');
for i=1:length(fRe)
  fprintf('%5d    %9.6f   %8.4f   %12.3e',mm(i),Deltar(i),fRe(i),err(i));
  if i>1,  fprintf('  %8.3f\n',err(i-1)/err(i));
  else,    fprintf('\n');
  end
end
```

Listing 3: The `refinePipe1d` function solves the finite volume model for a series of finer meshes.