HW 3 Solutions

Answer the problems below. Please follow these guidelines when preparing your solutions:

- when deriving or calculating answers, place boxes around the final answers.
- make sure all answers list the appropriate units.
- clearly label all axes on any plots.

[redacted]

1. [redacted] (Exercise by S. Santana) Use an iterative technique to solve the nonlinear equation $u^3 - u^4 = -11$. Show your derivation of the linearized equation as well as the results for your iterations with four initial guesses. There are two solutions. Indicate how your initial guess affects the final solution.

Solution: We start with

$$u^3 - u^4 = -11\,. \tag{1}$$

now we linearize u around our guess u_g:

$$u = u_g + (u - u_g) = u_g + \Delta u\,, \tag{2}$$

leading to

$$u^4 = u_g^4 + 4u_g^3\Delta u + 6u_g^2\Delta u^2 + 4u_g\Delta u^3 + \Delta u^4\,. \tag{3}$$

and

$$u^3 = u_g^3 + 3{u_g}^2\Delta u + 3u_g\Delta u^2 + \Delta u^3\,. \tag{4}$$

We omit terms that are 2nd order or higher in Δu, giving

$$u^4 \simeq u_g^4 + 4u_g^3\Delta u \tag{5}$$

and

$$u^3 \simeq u_g^3 + 3{u_g}^2\Delta u\,. \tag{6}$$

Now we substitute $\Delta u = u - u_g$ back in for Δu to obtain

$$-2u_g^3 + 3{u_g}^2 u + 3u_g^4 - 4u_g^3 u = -11\,, \tag{7}$$

which can be rearranged to give

$$u = \frac{-11 + 2u_g^3 - 3u_g^4}{3u_g^2 - 4u_g^3}\,. \tag{8}$$

$\boxed{\text{Repeated application of Eq. (8) will make the guesses proceed toward the correct answer. Values above approximately 0.75 will lead to the } u \simeq 2.13 \text{ solution, whereas values below that number will converge to the } u \simeq -1.61 \text{ solution.}}$

2. [redacted] Assume that you know the values of $u(x)$ at four gridpoints (u_i, u_{i+1}, u_{i+2}, and u_{i+3}), where u_i is at $x = x_i$, u_{i+1} is at $x = x_i + \Delta x$, u_{i+2} is at $x = x_i + 2\Delta x$, and u_{i+3} is at $x = x_i + 3\Delta x$.

   (a) Write u_{i+1}, u_{i+2}, and u_{i+3} as Taylor expansions of u around x_i.

   (b) Combine and rearrange these equations to give an expression for $\left.\frac{d^3u}{dx^3}\right|_i$ in terms of u_i, u_{i+1}, u_{i+2}, and u_{i+3}.

   (c) Show that this approximation is first-order, i.e., show that the truncation error is $O(\Delta x)$.

Solution:

Taylor expansions. We write Taylor expansions for the three nearby gridpoints, first u_{i+1}:

$$u_{i+1} = u_i + \Delta x \left.\frac{du}{dx}\right|_i + \frac{1}{2}\Delta x^2 \left.\frac{d^2u}{dx^2}\right|_i + \frac{1}{6}\Delta x^3 \left.\frac{d^3u}{dx^3}\right|_i + \frac{1}{24}\Delta x^4 \left.\frac{d^4u}{dx^4}\right|_i + \frac{1}{120}\Delta x^5 \left.\frac{d^5u}{dx^5}\right|_i + \dots, \tag{9}$$

then u_{i+2}:

$$u_{i+2} = u_i + 2\Delta x \left.\frac{du}{dx}\right|_i + 2\Delta x^2 \left.\frac{d^2u}{dx^2}\right|_i + \frac{4}{3}\Delta x^3 \left.\frac{d^3u}{dx^3}\right|_i + \frac{2}{3}\Delta x^4 \left.\frac{d^4u}{dx^4}\right|_i + \frac{4}{15}\Delta x^5 \left.\frac{d^5u}{dx^5}\right|_i + \dots, \tag{10}$$

then u_{i+3}:

$$u_{i+3} = u_i + 3\Delta x \left.\frac{du}{dx}\right|_i + \frac{9}{2}\Delta x^2 \left.\frac{d^2u}{dx^2}\right|_i + \frac{9}{2}\Delta x^3 \left.\frac{d^3u}{dx^3}\right|_i + \frac{27}{8}\Delta x^4 \left.\frac{d^4u}{dx^4}\right|_i + \frac{81}{40}\Delta x^5 \left.\frac{d^5u}{dx^5}\right|_i + \dots. \tag{11}$$

Combine and rearrange. We need all terms lower order than $\left.\frac{d^3u}{dx^3}\right|_i$ to be zero. Each combination of u_i, u_{i+1}, u_{i+2}, and u_{i+3} can be used to eliminate one of these terms. We pick the multipliers for the velocities to do so, and then rearrange the sum to solve for $\left.\frac{d^3u}{dx^3}\right|_i$:

$$\left.\frac{d^3u}{dx^3}\right|_i = \frac{u_{i+3} - 3u_{i+2} + 3u_{i+1} - u_i}{\Delta x^3} - \frac{3}{2}\Delta x \left.\frac{d^4u}{dx^4}\right|_i + \dots. \tag{12}$$

The u's are known, so our approximation is

$$\left.\frac{d^3u}{dx^3}\right|_i = \frac{u_{i+3} - 3u_{i+2} + 3u_{i+1} - u_i}{\Delta x^3}. \tag{13}$$

Order of accuracy.

The truncation error is $-\frac{3}{2}\Delta x \left.\frac{d^4u}{dx^4}\right|_i + \dots$, which is $O(\Delta x)$.

~~4. For air, rank the following from largest to smallest:~~

~~(a) a^*/a_0,~~

~~(b) p^*/p_0,~~

~~(c) ρ^*/ρ_0,~~

~~(d) T^*/T_0.~~

~~**Solution:** *a* then *T* then ρ then *p*. ($a^*/a_0 = 0.91$, $T^*/T_0 = 0.83$, $\rho^*/\rho_0 = 0.63$, $p^*/p_0 = 0.52$).~~

~~5. Consider a normal shock wave between conditions 1 (pre-shock) and 2 (post-shock). Rank the following from largest to smallest:~~

~~(a) T_2/T_1,~~

~~(b) ρ_2/ρ_1,~~

~~(c) u_2/u_1,~~

~~(d) p_2/p_1.~~

2

Answer the problems below. Please follow these guidelines when preparing your solutions:

- when deriving or calculating answers, place boxes around the final answers.
- make sure all answers list the appropriate units.
- clearly label all axes on any plots.

1. ~~Reading: read Chapters 6 and 7. Review Appendices B and C.~~
2. For one-dimensional flow of fluid between two plates oriented normal to the y axis, the Navier–Stokes equations simplify to

$$0 = -\frac{dp}{dx} + \mu\frac{\partial^2 u}{\partial y^2} \tag{1}$$

For simplicity, consider $\frac{dp}{dx} = -1$ (corresponding to flow from left to right) and $\mu = 1$. Assume that the plates are located at $y = 1$ and $y = -1$, at which we apply the boundary condition that $u = 0$.

(a) Solve this equation analytically by treating $\frac{dp}{dx}$ as a constant and integrating the $\frac{\partial^2 u}{\partial y^2}$ term.

(b) Solve this equation numerically on 5-point and 9-point grids by applying the finite-difference method to this equation to get a linearized difference equation at grid point i away from the boundary. Note that a second-order difference approximation for the second-derivative is

$$\left(\frac{d^2u}{dx^2}\right)_i = \frac{u_{i-1} - 2u_i + u_{i+1}}{\Delta x^2} + O\left(\Delta x^2\right). \tag{2}$$

i. Assemble the discrete system of equations for the grid into a matrix system of the form

$$Au = b, \tag{3}$$

where, for example, for the 5-point grid:

$$u = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix}. \tag{4}$$

and solve this system using MATLAB. You may solve this system using direct inversion of the matrix, or using an iterative technique. If you use an iterative technique, use $u = 0$ for your initial guess.

ii. Plot the finite-difference solution obtained on the 5- and 9-point grids and compare it with the exact solution. Plot u on the abscissa and y on the ordinate.

(c) Now solve the equation

$$0 = -\frac{dp}{dx} + \mu\frac{\partial^2 u}{\partial y^2} - 0.4u^2 \tag{5}$$

On the same domain, with same $\frac{dp}{dx}$ and same μ and same boundary conditions and same grids. In this case, the nonlinear term will require that you use an iterative technique. Use $u = 0$ for your initial guess. Physically, the $-0.4u^2$ term corresponds to a retardation body force per unit volume proportional to the kinetic energy per unit volume of the fluid. This is not common with normal fluids, but can be observed with exotic fluids like ferrofluids that respond to magnetic fields. This solution does not allow for trivially simple integration like the first equation, but a numerical solution is still possible. Solve the equation, plot the results and residual, and compare to the solution without the retardation force. What is the effect of the retardation on the velocity profile?

Why does the solution without the retardation force give zero residual after one iteration while the solution with the retardation force has a finite residual? (Stated another way, why is only one matrix inversion required to solve the first problem?) Also, what does this exercise show you about the relative merits of analytical solution of the differential equations vs. numerical solution of the differential equations?

Solution:

solve the equation analytically... this equation can be integrated directly because μ and $\frac{dp}{dx}$ are uniform. Thus

$$0 = -\frac{dp}{dx} + \mu\frac{\partial^2 u}{\partial y^2} \tag{6}$$

$$\frac{1}{\mu}\frac{dp}{dx} = \frac{\partial^2 u}{\partial y^2} \tag{7}$$

$$\left(\frac{1}{\mu}\frac{dp}{dx}\right)y + c_1 = \frac{\partial u}{\partial y} \tag{8}$$

$$\left(\frac{1}{2\mu}\frac{dp}{dx}\right)y^2 + c_1 y + c_2 = u \tag{9}$$

setting $u = 0$ at $y = \pm 1$, find $c_1 = 0$ and $c_2 = -\frac{1}{2\mu}\frac{dp}{dx}$. Thus

$$u = -\frac{1}{2\mu}\frac{dp}{dx}\left(1 - y^2\right) \tag{10}$$

or, for the specified values,

$$\boxed{u = \frac{1}{2}\left(1 - y^2\right) \tag{11}}$$

Apply the finite-difference method... The equation is simple, so all that is needed is the approximation for the second derivative:

$$\mu\left(\frac{u_{i-1} - 2u_i + u_{i+1}}{\Delta x^2}\right) = \frac{dp}{dx} + O\left(\Delta x^2\right) \tag{12}$$

Where $O\left(\Delta x^2\right)$ indicates "of the order of Δx^2", which tells us how fast the error will go away as we make the grid more and more refined.

Assemble the discrete system of equations... The above difference equation is applied at $i = 2, 3, 4$. Boundary conditions are applied at $i = 1, 5$. Plugging in the specified values for μ and $\frac{dp}{dx}$, the resulting simultaneous equations take the form

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ -\Delta x^2 \\ -\Delta x^2 \\ -\Delta x^2 \\ 0 \end{bmatrix} \tag{13}$$

plot the finite-difference solution... The solutions for the 5-point and 9-point grids are plotted in Fig. 1. The results are, in fact, *exactly* equal to the analytical solution.

The residuals for the 5- and 9-point grid are plotted in Fig. 2. The residual goes to zero identically in one iteration.

The MATLAB code is included in Fig. 3.

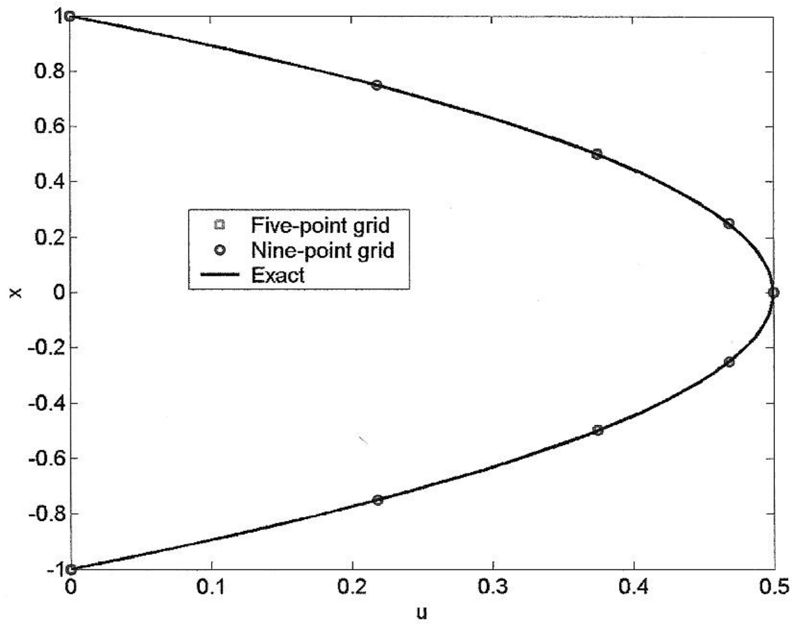

Figure 1: Numerical solutions for u vs x for the 5-point and 9-point grids compared with the exact solution.

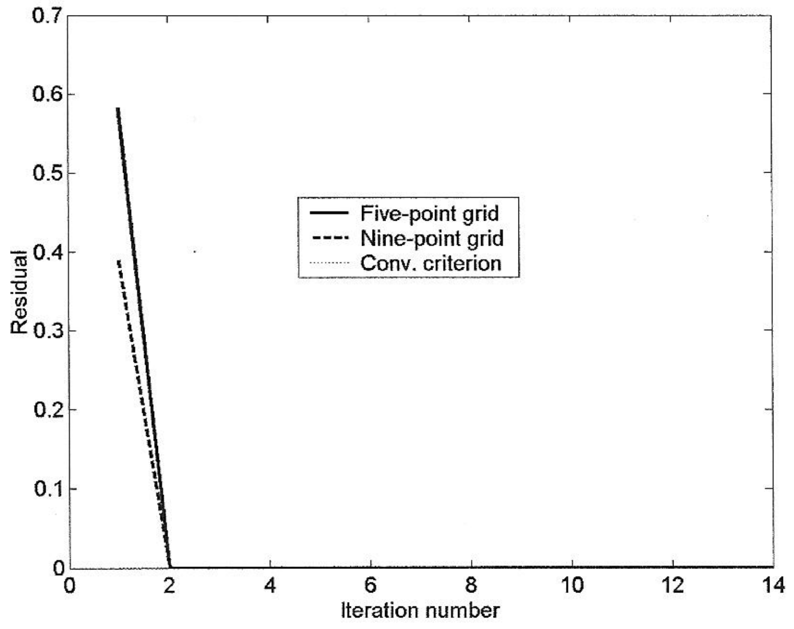

Figure 2: Residuals as a function of time for the 5-point and 9-point and 9-point grids. The convergence tolerance value of 10^{-6} is hard to see because this is a linear plot. Note that the residual goes identically to zero after one iteration

```
%this code solves the differential equation
% mu * u''= dpdx;
% where u(1)=0 and u(-1)=0;
% this matlab code uses x as the independent variable whereas problem statement uses y.  this is just notation.
% soln by B Kirby based on code by R Bhaskaran
L = 2; dpdx=-1; visc=1; %specifies the length of the domain, from x=-1 to x=1 plus dpdx and mu
niter = 14; %No. of iterations.  more iterations will reduce the residual.  here, 7 is picked because it is
           %good enough to get residuals below 1e-6
for Nj=1:2  %solve two times, first for 5-pt and then for 9-pt
 N = 4*Nj+1; %sets N to 5 or 9
 dx = L/(N-1); % divides the domain equally into grid points separated by dx
 x = linspace(-1,1,N)'; % makes a matrix of x values evenly spaced from 0 to L
 ug = linspace(0,0,N)'; %sets the initial guess for u, which we use to steadily improve the guess
 A = zeros(N,N); %starts by setting the A matrix to all zeros; we will fill in with other stuff later
 b = zeros(N,1); %sets the b matrix to all zeros; we will fill in with other stuff later
 A(1,1) = 1; %at x=0 the equation is u=1, so set A(1,1)=1 and b(1)=1
 b(1) = 0; %
 A(N,N) = 1; %at x=L the equation is u=1/9, so set A(1,1)=1 and b(1)=1/9
 b(N) = 0; %
% now consider interior points.  when we discretize the equations, the approximation for the 2nd deriv leads
% to terms like u(i-1) and i(i+1).  In the A matrix, this means the A values to left and right of the diagonal are 1
 for i=2:N-1  % note that these parts of the A matrix never change, since they are the boundary conditions
   A(i,i-1) = 1;
   A(i,i+1) = 1;
 end;
 for iter = 1:niter % this loop repeats the process of updating the guess several times (in this case, 7 times)
   for i=2:N-1
     A(i,i) = -2; % here we are putting the values on the diagonal in A
     b(i) = dpdx/visc*(dx^2);      % here we are putting the values in the b matrix
   end
 u = A\b; %the matrix equation is Au=b which means A-1*A*u=A-1*b which means u=A-1 b
          %this line effectively sets u=A-1 b   (A-1 means A inverse)  ug was the first guess.  we just calculated
          % our next guess u
 res(iter,Nj) = sqrt((ug-u)'*(ug-u))/sum(abs(u)); %calculates residual by summing differences between old and new u
 ug = u%; %replaces ug with the new guess
 end % end of the loop that repeats for each guess
% here we
 if(Nj == 1)
   x_grid1 = x;  %stores the 5-point result here
   u_grid1 = u;
 elseif(Nj == 2)
   x_grid2 = x; % stores the 9-point result here
   u_grid2 = u;
 end
end
%Plot of solution
figure(1); clf;
set(gca,'Box','on','LineWidth',2,'FontName','Helvetica',...
   'FontSize',14);
h=plot(u_grid1,x_grid1,'sb',u_grid2,x_grid2,'or'); hold on;
set(h,'LineWidth',2);
x_exact = linspace(-L/2,L/2,51);
u_exact =1./2.*(1-x_exact.*x_exact);
h=plot(u_exact,x_exact,'-k');
set(h,'LineWidth',2); xlabel('u'); ylabel('x');
legend('Five-point grid','Nine-point grid','Exact');
%Plot of residual
figure(2); clf;
set(gca,'Box','on','LineWidth',2,'FontName','Helvetica',...
   'FontSize',14);
h=plot(1:niter,res(:,1),'-k',1:niter,res(:,2),'--k'); hold on;
set(h,'LineWidth',2);
h = plot(linspace(1,niter,2),linspace(1e-6,1e-6,2),':k');
xlabel('Iteration number');
ylabel('Residual');
legend('Five-point grid','Nine-point grid','Conv. criterion',3);
```

Figure 3: MATLAB solution for pressure-driven flow between two infinite plates.

Now solve the equation... We now have a nonlinear source term that is a function of u^2. We must linearize this to put the solution in matrix form. We begin by deriving a linear approximation for u^2. Defining

$$\Delta u \equiv u - u_g \tag{14}$$

where u_g is the "guess value," we can write

$$u^2 = \left(u_g + \Delta u\right)^2 \tag{15}$$

now multiply out:

$$u^2 = u_g^2 + 2u_g\Delta u + \Delta u^2 \tag{16}$$

now, we assume that Δu is small (in particular, Δu is the difference between the current guess and the next guess. When we start getting close to the answer, Δu will approach zero). Given that Δu is small, we neglect the higher-order terms in Δu. Thus the Δu^2 term we assume approaches zero. so we have

$$u^2 = u_g^2 + 2u_g\Delta u \tag{17}$$

Note that this is now an *approximation* which will get better and better as we iterate toward the final solution. Now, replace Δu with $u - u_g$ and solve for our approximation for u^2 as a function of u_g:

$$u^2 = u_g^2 + 2u_g\Delta u \tag{18}$$

$$u^2 = u_g^2 + 2u_g(u - u_g) \tag{19}$$

$$u^2 = u_g^2 + 2u_g u - 2u_g^2 \tag{20}$$

$$u^2 = -u_g^2 + 2u_g u \tag{21}$$

and thus

$$-0.4u^2 = 0.4u_g^2 - 0.8u_g u \tag{22}$$

Substituting this and the approximation for the second derivative into the differential equation yields

$$\boxed{\mu\frac{u_{i-1} - 2u_i + u_{i+1}}{\delta x^2} - 0.8u_{gi}u_i = \frac{dp}{dx} - 0.4u_{g,i}^2 + O\left(\delta x^2\right)} \tag{23}$$

which, in matrix form is

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -2-0.8u_{g,2}\delta x^2/\mu & 1 & 0 & 0 \\ 0 & 1 & -2-0.8u_{g,3}\delta x^2/\mu & 1 & 0 \\ 0 & 0 & 1 & -2-0.8u_{g,4}\delta x^2/\mu & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ \left(\delta x^2/\mu\right)\left(\frac{dp}{dx} - 0.4u_{g,2}{}^2\right) \\ \left(\delta x^2/\mu\right)\left(\frac{dp}{dx} - 0.4u_{g,3}{}^2\right) \\ \left(\delta x^2/\mu\right)\left(\frac{dp}{dx} - 0.4u_{g,4}{}^2\right) \\ 0 \end{bmatrix} \tag{24}$$

or, plugging in values for μ and $\frac{dp}{dx}$:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -2-0.8u_{g,2}\delta x^2 & 1 & 0 & 0 \\ 0 & 1 & -2-0.8u_{g,3}\delta x^2 & 1 & 0 \\ 0 & 0 & 1 & -2-0.8u_{g,4}\delta x^2 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ \delta x^2\left(-1 - 0.4u_{g,2}{}^2\right) \\ \delta x^2\left(-1 - 0.4u_{g,3}{}^2\right) \\ \delta x^2\left(-1 - 0.4u_{g,4}{}^2\right) \\ 0 \end{bmatrix} \tag{25}$$

The solution for the 5-point and 9-point grid is plotted in Fig. 4.

The residuals for the 5- and 9-point grid are plotted in Fig. 5. Note that, now that there is a linearized u^2 term, the residual does not immediately go to zero.

The MATLAB code is included in Fig. 6.

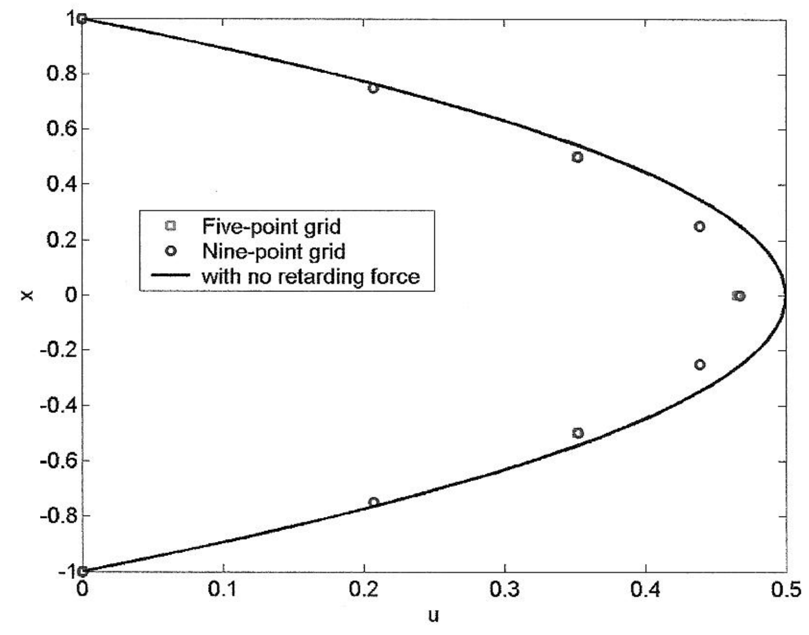

Figure 4: Numerical solution for u vs x for the 5-point and 9-point grids with a retarding force proportional to $0.4u^2$.

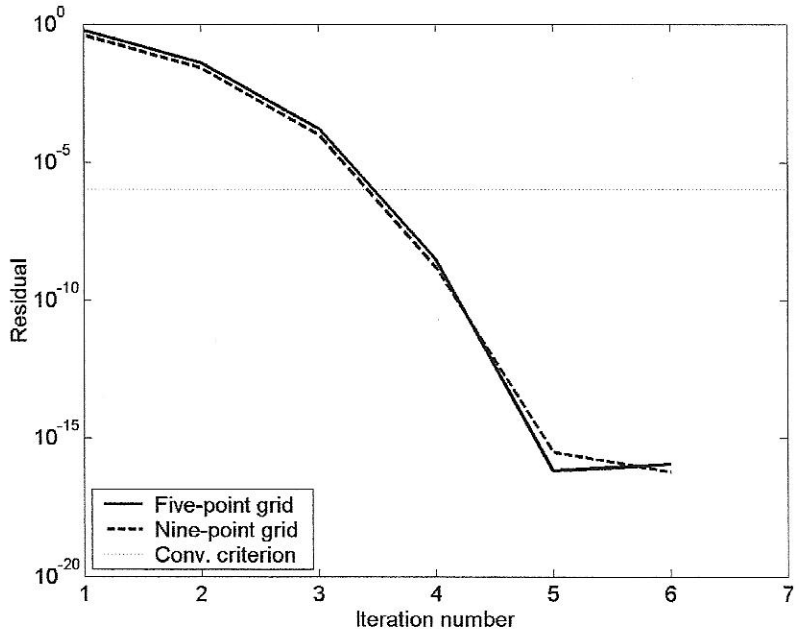

Figure 5: Residuals as a function of time for the 5-point and 9-point grids.

```
%this code solves the differential equation
% mu * u''= dpdx-.4u^2;
% where u(1)=0 and u(-1)=0;
% this matlab code uses x as the independent variable whereas problem statement uses y.  this is just notation.
% soln by B Kirby based on code by R Bhaskaran
clear all;L = 2; dpdx=-1; visc=1; %specifies the length of the domain, from x=-1 to x=1 plus dpdx and mu
niter = 7; %No. of iterations.  more iterations will reduce the residual.  here, 7 is picked because it is
           %good enough to get residuals below 1e-6
for Nj=1:2  %solve two times, first for 5-pt and then for 9-pt
 N = 4*Nj+1; %sets N to 5 or 9
 dx = L/(N-1); % divides the domain equally into grid points separated by dx
 x = linspace(-1,1,N)'; % makes a matrix of x values evenly spaced from 0 to L
 ug = linspace(0,0,N)'; %sets the initial guess for u, which we use to steadily improve the guess
 A = zeros(N,N); %starts by setting the A matrix to all zeros; we will fill in with other stuff later
 b = zeros(N,1); %sets the b matrix to all zeros; we will fill in with other stuff later
 A(1,1) = 1; %at x=-1 the equation is u=0, so set A(1,1)=1 and b(1)=0
 b(1) = 0; %
 A(N,N) = 1; %at x=L the equation is u=0, so set A(1,1)=1 and b(1)=0
 b(N) = 0; %
% now consider interior points.  when we discretize the equations, the approximation for the 2nd deriv leads
% to terms like u(i-1) and i(i+1).  In the A matrix, this means the A values to left and right of the diagonal are 1
 for i=2:N-1  % note that these parts of the A matrix never change, since they are the boundary conditions
   A(i,i-1) = 1;
   A(i,i+1) = 1;
 end;
 for iter = 1:niter % this loop repeats the process of updating the guess several times (in this case, 7 times)
   for i=2:N-1
     A(i,i) = -2-.8*dx^2/visc*ug(i); % here we are putting the values on the diagonal in A
     b(i) = (dx^2)/visc*(dpdx-.4*ug(i)^2);      % here we are putting the values in the b matrix
   end
 u = A\b; %the matrix equation is Au=b which means A-1*A*u=A-1*b which means u=A-1 b
          %this line effectively sets u=A-1 b   (A-1 means A inverse)  ug was the first guess.  we just calculated
          % our next guess u
 res(iter,Nj) = sqrt((ug-u)'*(ug-u))/sum(abs(u)); %calculates residual by summing differences between old and new u
 ug = u%; %replaces ug with the new guess
 end % end of the loop that repeats for each guess
% here we
 if(Nj == 1)
   x_grid1 = x;  %stores the 5-point result here
   u_grid1 = u;
 elseif(Nj == 2)
   x_grid2 = x; % stores the 9-point result here
   u_grid2 = u;
 end
end
%Plot of solution
figure(1); clf;
set(gca,'Box','on','LineWidth',2,'FontName','Helvetica',...
    'FontSize',14);
h=plot(u_grid1,x_grid1,'sb',u_grid2,x_grid2,'or'); hold on;
set(h,'LineWidth',2);
x_exact = linspace(-L/2,L/2,51);
u_exact =1./2.*(1-x_exact.*x_exact);
h=plot(u_exact,x_exact,'-k');
set(h,'LineWidth',2); xlabel('u'); ylabel('x');
legend('Five-point grid','Nine-point grid','with no retarding force');
%Plot of residual
figure(2); clf;
set(gca,'Box','on','LineWidth',2,'FontName','Helvetica',...
    'FontSize',14);
h=semilogy(1:niter,res(:,1),'-k',1:niter,res(:,2),'--k'); hold on;
set(h,'LineWidth',2);
h = semilogy(linspace(1,niter,2),linspace(1e-6,1e-6,2),':k');
xlabel('Iteration number');
ylabel('Residual');
legend('Five-point grid','Nine-point grid','Conv. criterion',3);
```

Figure 6: MATLAB solution for pressure-driven flow between two infinite plates with a nonlinear ($4u^2$) retarding force.