|  |
| --- |
|  |
| ME 811: HW 6 |
|  |
|  |
| **Michael Crawley** |
|  |

|  |
| --- |
|  |

Figure 1 shows the pressure contours and velocity vectors, as well as the residuals at each iteration, for a Reynolds number flow of 100, with a mesh size of 40 by 200 nodes, computed via the SIMPLE algorithm on a collocated mesh. Similarly, Figure 2 shows the pressure contours, velocity contours, and residuals for a Reynolds number flow of 200 with an identical mesh. As one would expect the flow over the backward facing step results in a recirculation region just downstream; increasing the Reynolds number (increasing the inlet velocity) results in the center of the recirculation region moving farther downstream and a slight enlargement in the size of the recirculation region. Additionally, the regions where the pressure is not constant moves downstream with increasing Reynolds number, meaning that for higher Reynolds number it would be necessary to increasing the size of the domain so that the boundary condition of constant pressure would still be valid. Oddly, the higher Reynolds number case required significantly less iterations than the lower Reynolds number case in order to converge to the same accuracy; it is not immediately clear why the higher Reynolds number case converges faster than the lower Reynolds number case.

|  |  |
| --- | --- |
| Z:\My Documents\Matlab\Coursework\ME 811\HW 6\40x200 Re100\40x200 mesh, Re = 100 pressure.png | Z:\My Documents\Matlab\Coursework\ME 811\HW 6\40x200 Re100\40x200 mesh, Re = 100 Residuals.png |
| Z:\My Documents\Matlab\Coursework\ME 811\HW 6\40x200 Re100\40x200 mesh, Re = 100 u velocity.png | Z:\My Documents\Matlab\Coursework\ME 811\HW 6\40x200 Re100\40x200 mesh, Re = 100 v velocity.png |

Figure 1: Pressure and velocity contours, and residuals, for 40 by 200 mesh with Reynolds number of 100.

|  |  |
| --- | --- |
|  |  |
|  |  |

Figure 2: Pressure and velocity contours, and residuals, for 40 by 200 mesh with Reynolds number of 200.

## Appendix

function [u v p Resx Resy Resp] = SIMPLEcol(Uin,M)

tic;

Rtol = 10E-10;

itrmax = 2E3;

N = 5\*M; %force uniform dx, dy

rho = ones(M,N);

gamma = 2E-5;

H = 0.01; %m

omega.u = 0.5;

omega.p = 0.2;

alpha = 0.2;

dx = 10\*H/N;

dy = 2\*H/M;

x = dx/2:dx:10\*H-dx/2;

y = dy/2:dy:2\*H-dy/2;

xstepi = find(x <= 2\*H,1,'last'); %locate x-indices inside step

ystepi = find(y <= H,1,'last'); %lcoate y-indices inside step

pout = 0;

u = Uin\*ones(M,N); u(1:ystepi,1:xstepi) = 0;

v = zeros(M,N);

p = v; p(:,end) = pout;

Resx = zeros(1,itrmax+1);Resx(1) = 1;

counter = 1;

Resy = Resx; Resp = Resx;

Ax = [];

[uf vf] = calcFaceVelocities(u,v,p,Uin,xstepi,ystepi,Ax,dx,dy,counter);

while (Resx(counter) >= Rtol || Resy(counter) >= Rtol || Resp(counter) >= Rtol) && counter <= itrmax

counter = counter+1;

[Ax Ay] = calcMomcoefs(uf,vf,rho,p,gamma,Uin,dx,dy,xstepi,ystepi);

[u Resx(counter)] = solveMomEQ(u,Ax,alpha);

[v Resy(counter)] = solveMomEQ(v,Ay,alpha);

[uf vf] = calcFaceVelocities(u,v,p,Uin,xstepi,ystepi,Ax.O,dx,dy,counter);

[Ap] = calcPresscoefs(uf,vf,rho,dx,dy,xstepi,ystepi,Ax.O);

[pp Resp(counter)] = solvePressEQ(Ap);

[u,uf,v,vf,p] = updatevalues(u,uf,v,vf,p,pp,dx,dy,xstepi,ystepi,omega,Ax.O);

fprintf('counter: %i Resx: %1.2e Resy: %1.2e Resp: %1.2e \n',[counter-1 Resx(counter) Resy(counter) Resp(counter)]);

end

Re=round(mean(mean(rho))\*Uin\*2\*H/gamma);

foldername = [num2str(M),'x',num2str(N),' Re',num2str(Re)];

filename = [num2str(M),'x',num2str(N),' mesh, Re = ',num2str(Re)];

if exist(foldername,'file') ~= 7

mkdir(pwd,foldername);

end

cd(foldername);

compute\_time = toc;

save data.mat;

h(1) = figure;

contourf(x,y,p,50);colormap(flipud(gray));colorbar;hold on;

quiver(x,y,u,v,3);xlabel('x');ylabel('y');xlim([x(1) x(end)]); ylim([y(1) y(end)]);plot([0 0.02 0.02],[0.01 0.01 0],'k');hold off;

xlabel('x (m)'); ylabel('y (m)'); title(['Velocity Vectors and Pressure Contours for ',filename]);

h(2) = figure;

semilogy(1:counter,Resx(1:counter),'k',1:counter,Resy(1:counter),'k--',1:counter,Resp(1:counter),'k-.');

legend('u Residual','v Residual','p Residual');xlabel('iteration');ylabel('Residual');title(['Outer Iteration Residuals for ',filename]);

h(3) = figure;

contourf(x,y,u);hold on;plot([0 0.02 0.02],[0.01 0.01 0],'k');hold off;colormap(flipud(gray));colorbar;xlabel('x');ylabel('y');title(['U velocity Contours for ',filename]);

h(4) = figure;

contourf(x,y,v);hold on;plot([0 0.02 0.02],[0.01 0.01 0],'k');hold off;colormap(flipud(gray));colorbar;xlabel('x');ylabel('y');title(['V velocity Contours for ',filename]);

%Save figures

saveas(h(1),[filename,' pressure'],'fig');saveas(h(1),[filename,' pressure'],'png');

saveas(h(2),[filename,' Residuals'],'fig');saveas(h(2),[filename,' Residuals'],'png');

saveas(h(3),[filename,' u velocity'],'fig');saveas(h(3),[filename,' u velocity'],'png');

saveas(h(4),[filename,' v velocity'],'fig');saveas(h(4),[filename,' v velocity'],'png');

close(h);

cd ..

end

function [Ax Ay] = calcMomcoefs(uf,vf,rho,P,gamma,Uin,dx,dy,xstepi,ystepi)

[M N] = size(P);

%Compute Face pressures

p.e = [0.5\*(P(:,1:end-1)+P(:,2:end)) zeros(M,1)]; %set outlet boundary condition

p.w = [P(:,1) 0.5\*(P(:,1:end-1)+P(:,2:end))];

p.w(1:ystepi,xstepi+1) = P(1:ystepi,xstepi+1); %fix for step

p.n = [0.5\*(P(1:end-1,:)+P(2:end,:)); P(end,:)];

p.s = [P(1,:); 0.5\*(P(1:end-1,:)+P(2:end,:))];

p.s(ystepi+1,1:xstepi) = P(ystepi+1,1:xstepi); %fix for step

%%Calculate X-momentum link coefficients

ce = rho.\*uf.e; %need to modify rho for weakly compressible flows

cw = rho.\*uf.w;

cn = rho.\*vf.n;

cs = rho.\*vf.s;

dep = 0.5\*(abs(ce)+ce);

dem = 0.5\*(abs(ce)-ce);

dwp = 0.5\*(abs(cw)+cw);

dwm = 0.5\*(abs(cw)-cw);

dnp = 0.5\*(abs(cn)+cn);

dnm = 0.5\*(abs(cn)-cn);

dsp = 0.5\*(abs(cs)+cs);

dsm = 0.5\*(abs(cs)-cs);

%Interior nodes

Ax.O = (dep+dwm+2\*gamma/dx)\*dy+(dnp+dsm+2\*gamma/dy)\*dx;

Ax.E = -(dem+gamma/dx)\*dy;

Ax.W = -(dwp+gamma/dx)\*dy;

Ax.N = -(dnm+gamma/dy)\*dx;

Ax.S = -(dsp+gamma/dy)\*dx;

Ax.P = -(p.e-p.w)\*dy;

%Inlet boundary

Ax.O(ystepi+1:end,1) = Ax.O(ystepi+1:end,1)+2\*gamma\*dy/dx;

Ax.E(ystepi+1:end,1) = Ax.E(ystepi+1:end,1)-gamma\*dy/3/dx;

Ax.W(ystepi+1:end,1) = 0;

Ax.P(ystepi+1:end,1) = Ax.P(ystepi+1:end,1)+rho(ystepi+1:end,1)\*(Uin^2)\*dy+8\*gamma\*Uin/3/dx\*dy;

%Ceiling boundary

Ax.O(end,:) = Ax.O(end,:) - dnp(end,:)\*dx+2\*gamma\*dx/dy;

Ax.N(end,:) = 0;

Ax.S(end,:) = Ax.S(end,:) - gamma\*dx/3/dy;

%Floor boundary

Ax.O(1,:) = Ax.O(1,:)-dsm(1,:)\*dx+2\*gamma\*dx/dy;

Ax.N(1,:) = Ax.N(1,:) -gamma\*dx/3/dy;

Ax.S(1,:) = 0;

%Outlet boundary

Ax.O(:,end) = Ax.O(:,end) -dep(:,end)\*dy-gamma\*dy/dx+rho(:,end).\*uf.e(:,end)\*dy;

Ax.E(:,end) = 0;

%Step floor boundary

Ax.O(ystepi+1,1:xstepi) = Ax.O(ystepi+1,1:xstepi)-dsm(ystepi+1,1:xstepi)\*dx+2\*gamma\*dx/dy;

Ax.N(ystepi+1,1:xstepi) = Ax.N(ystepi+1,1:xstepi) -gamma\*dx/3/dy;

Ax.S(ystepi+1,1:xstepi) = 0;

%Step wall boundary

Ax.O(1:ystepi,xstepi+1) = Ax.O(1:ystepi,xstepi+1)-dwm(1:ystepi,xstepi+1)\*dy+2\*gamma\*dy/dx;

Ax.E(1:ystepi,xstepi+1) = Ax.E(1:ystepi,xstepi+1)-gamma\*dy/3/dx;

Ax.W(1:ystepi,xstepi+1) = 0;

%In step nodes

Ax.O(1:ystepi,1:xstepi) = 1;

Ax.E(1:ystepi,1:xstepi) = 0;

Ax.W(1:ystepi,1:xstepi) = 0;

Ax.N(1:ystepi,1:xstepi) = 0;

Ax.S(1:ystepi,1:xstepi) = 0;

Ax.P(1:ystepi,1:xstepi) = 0;

%%Calculate Y-momentum link coefficients

Ay.O = Ax.O;

Ay.E = Ax.E;

Ay.W = Ax.W;

Ay.S = Ax.S;

Ay.N = Ax.N;

Ay.P = -(p.n-p.s)\*dx;

%In step nodes

Ay.P(1:ystepi,1:xstepi) = 0;

end

function [Ap] = calcPresscoefs(uf,vf,rhof,dx,dy,xstepi,ystepi,Ao)

[M N] = size(Ao);

mimb = -(rhof.\*uf.e-rhof.\*uf.w)\*dy-(rhof.\*vf.n-rhof.\*vf.s)\*dx;

Ap.O = zeros(M,N);

Ap.E = Ap.O;

Ap.W = Ap.O;

Ap.N = Ap.O;

Ap.S = Ap.O;

Ap.P = -mimb;

%interior nodes

Ap.E(:,1:end-1) = 0.5\*rhof(:,1:end-1)\*dy\*dy.\*(1./Ao(:,1:end-1)+1./Ao(:,2:end));

Ap.W(:,2:end) = 0.5\*rhof(:,2:end)\*dy\*dy.\*(1./Ao(:,2:end)+1./Ao(:,1:end-1));

Ap.N(1:end-1,:) = 0.5\*rhof(1:end-1,:)\*dx\*dx.\*(1./Ao(1:end-1,:)+1./Ao(2:end,:));

Ap.S(2:end,:) = 0.5\*rhof(2:end,:)\*dx\*dx.\*(1./Ao(2:end,:)+1./Ao(1:end-1,:));

%step wall nodes

Ap.W(1:ystepi,xstepi+1) = 0;

Ap.S(ystepi+1,1:xstepi) = 0;

Ap.O = -(Ap.E+Ap.W+Ap.N+Ap.S);

%outlet nodes

Ap.O(:,end) = Ap.O(:,end)-rhof(:,end)\*dy\*dy.\*(1./Ao(:,end));

%In step nodes

Ap.O(1:ystepi,1:xstepi) = 1;

Ap.E(1:ystepi,1:xstepi) = 0;

Ap.W(1:ystepi,1:xstepi) = 0;

Ap.N(1:ystepi,1:xstepi) = 0;

Ap.S(1:ystepi,1:xstepi) = 0;

Ap.P(1:ystepi,1:xstepi) = 0;

end

function [Ri Res] = calcRes(A,phi)

[M N] = size(phi);

% Calculate the residual

Ri = A.P-A.S .\* [zeros(1,N); phi(1:M-1,:)] ...

- A.W .\* [zeros(M,1), phi(:,1:N-1)] ...

- A.O.\* phi ...

- A.E .\* [phi(:,2:N), zeros(M,1)] ...

- A.N .\* [phi(2:M,:); zeros(1,N)];

Res = norm(Ri);

end

function [uf vf] = calcFaceVelocities(u,v,p,Uin,xstepi,ystepi,A,dx,dy,counter)

[M N] = size(u);

uf.e = zeros(M,N);

uf.w = uf.e; vf.n = uf.e; vf.s = uf.e;

if counter == 1 %DWIM

uf.e = [0.5\*(u(:,1:end-1)+u(:,2:end)) u(:,end)];

uf.w = [Uin\*ones(M,1) 0.5\*(u(:,1:end-1)+u(:,2:end))];

vf.n = [0.5\*(v(1:end-1,:)+v(2:end,:)); zeros(1,N)];

vf.s = [zeros(1,N); 0.5\*(v(1:end-1,:)+v(2:end,:))];

else %PWIM

Ox = 3:N-2; Oy = 3:M-2;

%interior nodes

uf.e(:,Ox) = 0.5\*(u(:,Ox)+u(:,Ox+1))+0.5\*dx\*dy\*(1./A(:,Ox).\*(p(:,Ox+1)-p(:,Ox-1))/2/dx+1./A(:,Ox+1).\*(p(:,Ox+2)-p(:,Ox))/2/dx-(1./A(:,Ox)+1./A(:,Ox+1)).\*(p(:,Ox+1)-p(:,Ox))/dx);

uf.w(:,Ox) = 0.5\*(u(:,Ox)+u(:,Ox-1))+0.5\*dx\*dy\*(1./A(:,Ox).\*(p(:,Ox+1)-p(:,Ox-1))/2/dx+1./A(:,Ox-1).\*(p(:,Ox)-p(:,Ox-2))/2/dx-(1./A(:,Ox)+1./A(:,Ox-1)).\*(p(:,Ox)-p(:,Ox-1))/dx);

vf.n(Oy,:) = 0.5\*(v(Oy,:)+v(Oy+1,:))+0.5\*dx\*dy\*(1./A(Oy,:).\*(p(Oy+1,:)-p(Oy-1,:))/2/dy+1./A(Oy+1,:).\*(p(Oy+2,:)-p(Oy,:))/2/dy-(1./A(Oy,:)+1./A(Oy+1,:)).\*(p(Oy+1,:)-p(Oy,:))/dy);

vf.s(Oy,:) = 0.5\*(v(Oy,:)+v(Oy-1,:))+0.5\*dx\*dy\*(1./A(Oy,:).\*(p(Oy+1,:)-p(Oy-1,:))/2/dy+1./A(Oy-1,:).\*(p(Oy,:)-p(Oy-2,:))/2/dy-(1./A(Oy,:)+1./A(Oy-1,:)).\*(p(Oy,:)-p(Oy-1,:))/dy);

%Near Outlet nodes

uf.w(:,N-1) = uf.e(:,N-2);

uf.e(:,N-1) = 0.5\*(u(:,N-1)+u(:,N))+0.5\*dx\*dy\*(1./A(:,N-1).\*(p(:,N)-p(:,N-2))/2/dx+1./A(:,N).\*(p(:,N)-p(:,N-1))/dx-(1./A(:,N-1)+1./A(:,N)).\*(p(:,N)-p(:,N-1))/dx);

%Outlet nodes

uf.e(:,N) = u(:,N);

uf.w(:,N) = uf.e(:,N-1);

%Near inlet nodes

uf.e(ystepi+1:end,2) = uf.w(ystepi+1:end,3);

uf.w(ystepi+1:end,2) = 0.5\*(u(ystepi+1:end,2)+u(ystepi+1:end,1))+0.5\*dx\*dy\*(1./A(ystepi+1:end,2).\*(p(ystepi+1:end,2+1)-p(ystepi+1:end,2-1))/2/dx+1./A(ystepi+1:end,2-1).\*(p(ystepi+1:end,2)-p(ystepi+1:end,1))/dx-(1./A(ystepi+1:end,2)+1./A(ystepi+1:end,2-1)).\*(p(ystepi+1:end,2)-p(ystepi+1:end,2-1))/dx);

%Inlet nodes

uf.w(ystepi+1:end,1) = Uin;

uf.e(ystepi+1:end,1) = uf.w(ystepi+1:end,2);

%Near step wall nodes

uf.e(1:ystepi,xstepi+2) = uf.w(1:ystepi,xstepi+3);

uf.w(1:ystepi,xstepi+2) = 0.5\*(u(1:ystepi,xstepi+2)+u(1:ystepi,xstepi+2-1))+0.5\*dx\*dy\*(1./A(1:ystepi,xstepi+2).\*(p(1:ystepi,xstepi+2+1)-p(1:ystepi,xstepi+2-1))/2/dx+1./A(1:ystepi,xstepi+2-1).\*(p(1:ystepi,xstepi+2)-p(1:ystepi,xstepi+2-1))/dx-(1./A(1:ystepi,xstepi+2)+1./A(1:ystepi,xstepi+2-1)).\*(p(1:ystepi,xstepi+2)-p(1:ystepi,xstepi+2-1))/dx);

%Step wall nodes

uf.w(1:ystepi,xstepi+1) = 0;

uf.e(1:ystepi,xstepi+1) = uf.w(1:ystepi,xstepi+2);

%Near ceiling nodes

vf.s(end-1,:) = vf.n(end-2,:);

vf.n(end-1,:) = 0.5\*(v(end-1,:)+v(end-1+1,:))+0.5\*dx\*dy\*(1./A(end-1,:).\*(p(end-1+1,:)-p(end-1-1,:))/2/dy+1./A(end-1+1,:).\*(p(end-1+1,:)-p(end-1,:))/dy-(1./A(end-1,:)+1./A(end-1+1,:)).\*(p(end-1+1,:)-p(end-1,:))/dy);

%Ceiling nodes

vf.n(end,:) = 0;

vf.s(end,:) = vf.n(end-1,:);

%Near step floor nodes

vf.n(ystepi+2,1:xstepi) = vf.s(ystepi+3,1:xstepi);

vf.s(ystepi+2,1:xstepi) = 0.5\*(v(ystepi+2,1:xstepi)+v(ystepi+2-1,1:xstepi))+0.5\*dx\*dy\*(1./A(ystepi+2,1:xstepi).\*(p(ystepi+2+1,1:xstepi)-p(ystepi+2-1,1:xstepi))/2/dy+1./A(ystepi+2-1,1:xstepi).\*(p(ystepi+2,1:xstepi)-p(ystepi+2-1,1:xstepi))/dy-(1./A(ystepi+2,1:xstepi)+1./A(ystepi+2-1,1:xstepi)).\*(p(ystepi+2,1:xstepi)-p(ystepi+2-1,1:xstepi))/dy);

%Step floor nodes

vf.s(ystepi+1,1:xstepi) = 0;

vf.n(ystepi+1,1:xstepi) = vf.s(ystepi+2,1:xstepi);

%Near floor nodes

vf.n(2,xstepi+1:end) = vf.s(3,xstepi+1:end);

vf.s(2,xstepi+1:end) = 0.5\*(v(2,xstepi+1:end)+v(2-1,xstepi+1:end))+0.5\*dx\*dy\*(1./A(2,xstepi+1:end).\*(p(2+1,xstepi+1:end)-p(2-1,xstepi+1:end))/2/dy+1./A(2-1,xstepi+1:end).\*(p(2,xstepi+1:end)-p(2-1,xstepi+1:end))/dy-(1./A(2,xstepi+1:end)+1./A(2-1,xstepi+1:end)).\*(p(2,xstepi+1:end)-p(2-1,xstepi+1:end))/dy);

%Floor nodes

vf.s(1,xstepi+1:end) = 0;

vf.n(1,xstepi+1:end) = vf.s(2,xstepi+1:end);

end

%modifications for step

uf.w(1:ystepi,1:xstepi+1) = 0;

uf.e(1:ystepi,1:xstepi) = 0;

vf.s(1:ystepi+1,1:xstepi) = 0;

vf.n(1:ystepi,1:xstepi) = 0;

end

function [uh Resx] = solveMomEQ(u,A,alpha)

Rtol = 1E-20;

itrmax = 2;

[Ri Resx] = calcRes(A,u);

A.O = (1+alpha)\*A.O;

A.P = Ri;

[up ~] = ADIpm(A,Rtol,itrmax);

uh = u+up;

end

function [pp Resp] = solvePressEQ(Ap)

Rtol = 1E-20;

itrmax = 20;

[pp ~] = ADIpm(Ap,Rtol,itrmax);

[~, Resp] = calcRes(Ap,pp);

end

function [phi R] = ADIpm(A,Rtol,itrmax)

[M N] = size(A.O);

if ~exist('Rtol','var')

Rtol = 1E-5;

end

if ~exist('itrmax','var')

itrmax = 1E4;

end

phi = zeros(M,N);

R = zeros(1,itrmax);

[~,R(1)] = calcRes(A,phi);

counter = 0;

while (R(counter+1) >= Rtol) && (counter < itrmax)

%Row sweep

S = A.P(1,:) - A.N(1,:).\*phi(2,:);

phi(1,:) = TDMAsolver(A.W(1,:),A.O(1,:),A.E(1,:),S);

for m=2:M-1

S = A.P(m,:) - A.S(m,:).\*phi(m-1,:) - A.N(m,:).\*phi(m+1,:);

phi(m,:) = TDMAsolver(A.W(m,:),A.O(m,:),A.E(m,:),S);

end

S = A.P(M,:) - phi(M-1,:).\*A.S(M,:);

phi(M,:) = TDMAsolver(A.W(M,:),A.O(M,:),A.E(M,:),S);

%Column sweep

S = A.P(:,1) - A.E(:,1).\*phi(:,2);

phi(:,1) = TDMAsolver(A.S(:,1),A.O(:,1),A.N(:,1),S);

for n=2:N-1

S = A.P(:,n) - A.W(:,n).\*phi(:,n-1) - A.E(:,n).\*phi(:,n+1);

phi(:,n) = TDMAsolver(A.S(:,n),A.O(:,n),A.N(:,n),S);

end

S = A.P(:,N) - A.W(:,N).\*phi(:,N-1);

phi(:,N) = TDMAsolver(A.S(:,N),A.O(:,N),A.N(:,N),S);

counter = counter + 1;

[~,R(counter+1)] = calcRes(A,phi);

end

R = R(1:counter+1);

end

function [u,uf,v,vf,p] = updatevalues(u,uf,v,vf,p,pp,dx,dy,xstepi,ystepi,omega,Ao)

[M N] = size(u);

%update cell center pressure

p = p+omega.p\*pp;

ppx = [pp(:,1) pp zeros(M,1)];

ppy = [pp(1,:); pp; pp(end,:)];

ppx(1:ystepi,xstepi+1) = ppx(1:ystepi,xstepi+2);

ppy(ystepi+1,1:xstepi) = ppy(ystepi+2,1:xstepi);

%update cell center velocities

up = (ppx(:,1:end-2)-ppx(:,3:end))\*dy/2./Ao;

vp = (ppy(1:end-2,:)-ppy(3:end,:))\*dx/2./Ao;

up(1:ystepi,1:xstepi) = 0;

vp(1:ystepi,1:xstepi) = 0;

u = u+omega.u\*up;

v = v+omega.u\*vp;

%update cell face velocities

uf.ep = zeros(size(p));

uf.wp = uf.ep;

vf.np = uf.ep;

vf.sp = uf.ep;

uf.ep(:,1:end-1) = 0.5\*dy\*(1./Ao(:,1:end-1)+1./Ao(:,2:end)).\*(pp(:,1:end-1)-pp(:,2:end));

uf.wp(:,2:end) = 0.5\*dy\*(1./Ao(:,2:end)+1./Ao(:,1:end-1)).\*(pp(:,1:end-1)-pp(:,2:end));

uf.ep(1:ystepi,xstepi) = 0;

uf.wp(1:ystepi,xstepi+1) = 0;

vf.np(1:end-1,:) = 0.5\*dx\*(1./Ao(1:end-1,:)+1./Ao(2:end,:)).\*(pp(1:end-1,:)-pp(2:end,:));

vf.sp(2:end,:) = 0.5\*dx\*(1./Ao(2:end,:)+1./Ao(1:end-1,:)).\*(pp(1:end-1,:)-pp(2:end,:));

vf.np(ystepi,1:xstepi) = 0;

vf.sp(ystepi+1,1:xstepi) = 0;

uf.e = uf.e+omega.u\*uf.ep;

uf.w = uf.w+omega.u\*uf.wp;

vf.n = vf.n+omega.u\*vf.np;

vf.s = vf.s+omega.u\*vf.sp;

end