Computational Fluid Dynamics

Francisco Castillo
Homework 10

April 17, 2018

1 Introduction

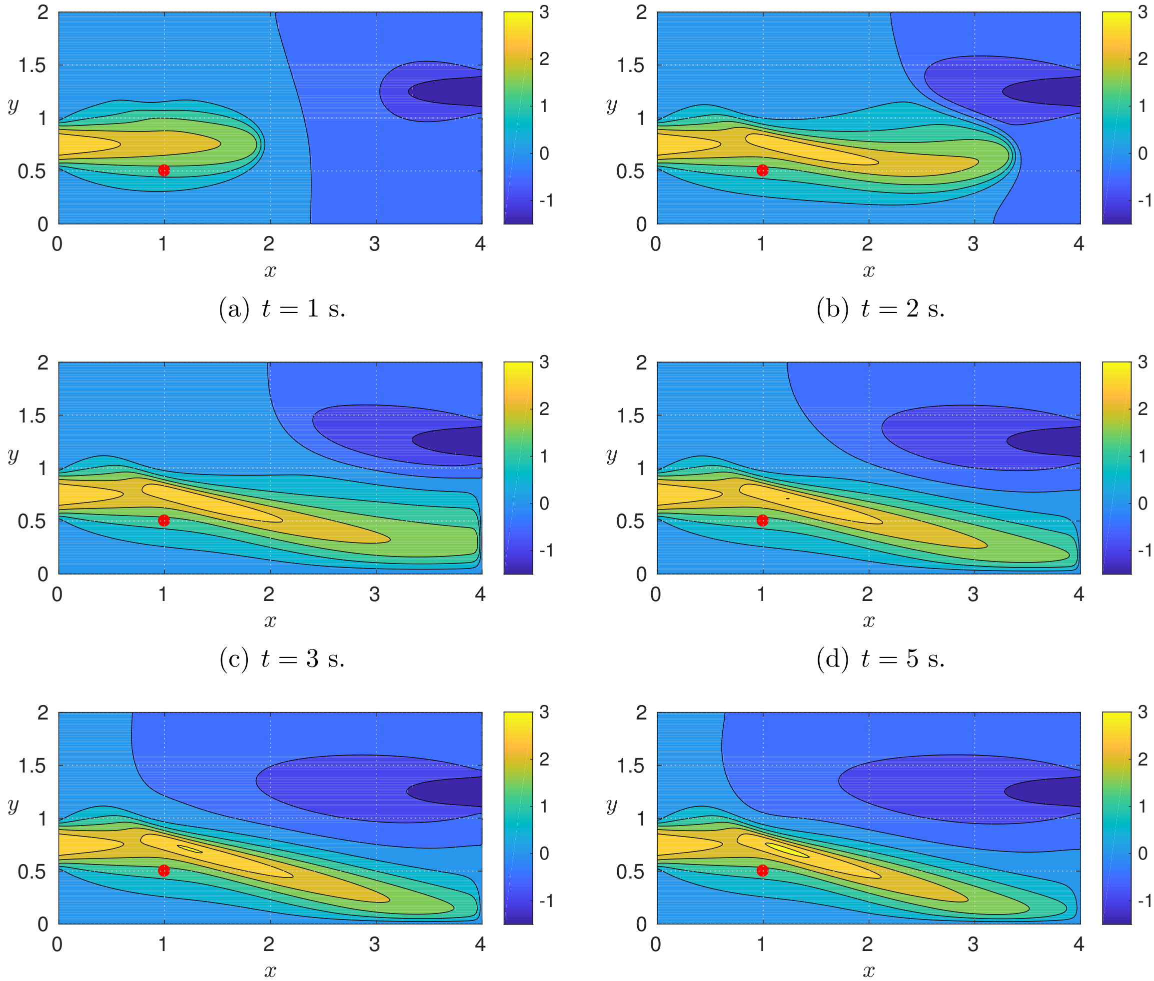
In this assignment we will study the solution to the following system

$$\begin{aligned}
\frac{\partial u}{\partial t}+\frac{\partial u^2}{\partial x}+\frac{\partial uv}{\partial y} &= \frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)\\
\frac{\partial v}{\partial t}+\frac{\partial uv}{\partial x}+\frac{\partial v^2}{\partial y} &= \frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)\\
\frac{\partial Y}{\partial t}+u\frac{\partial Y}{\partial x}+v\frac{\partial Y}{\partial y} &= \frac{1}{ReSc}\left(\frac{\partial^2 Y}{\partial x^2}+\frac{\partial^2 Y}{\partial y^2}\right)
\end{aligned}$$

and periodic boundary conditions. We will use an `Adams-Bashforth Crank-Nicolson` method for u and v on a staggered mesh and a `WENO-5 TVD-RK-3` method with Crank-Nicolson for Y on a cell centered mesh.

2 Results

In the figures below we can see the surface plots of the horizontal velocity u, vertical velocity v and mass fraction Y, for different values of t. They are all obtained using $M = 256$, $N = 128$ and $CFL = 0.8$. Below the contour plots, and using the same parameters, we have the curve plots of the probe measurements together with the performance parameter R *vs* time. To finish, we have the GCI-analysis of the quantities required by the problem. At the very end, the same GCI-analysis will be performed to the steady state solution.

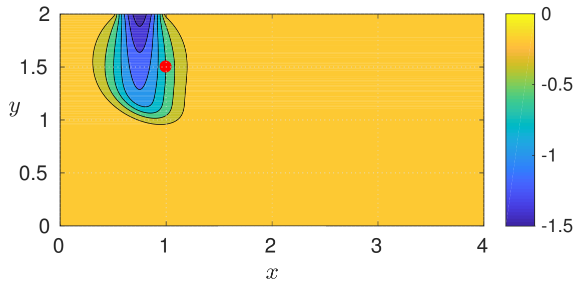


(a) $t = 1$ s. (b) $t = 2$ s.

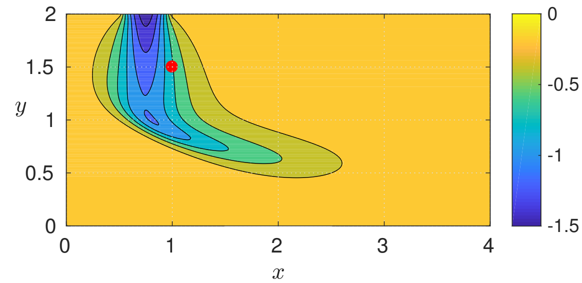
(c) $t = 3$ s. (d) $t = 5$ s.

(e) $t = 7$ s. (f) $t = 10$ s.

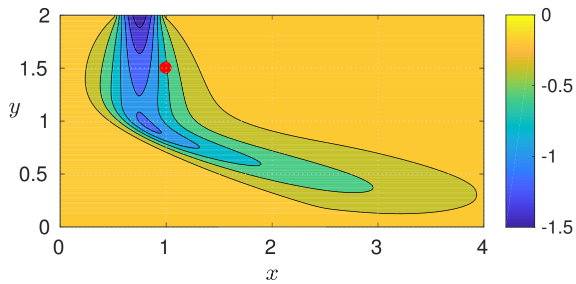
Figure 1: Profiles of horizontal velocity u for $M = 256$, $N = 128$ and $CFL = 0.8$.



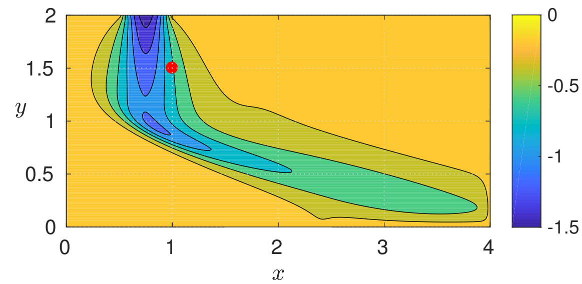
(a) $t = 1$ s.



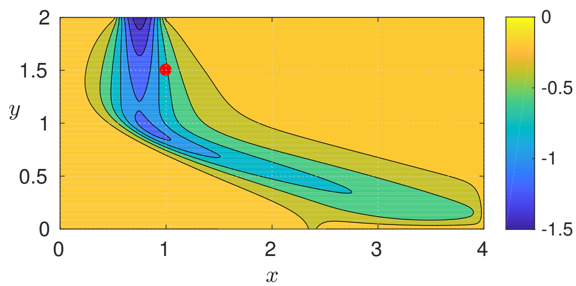
(b) $t = 2$ s.



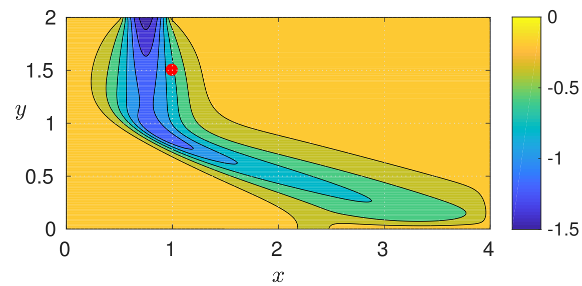
(c) $t = 3$ s.



(d) $t = 5$ s.

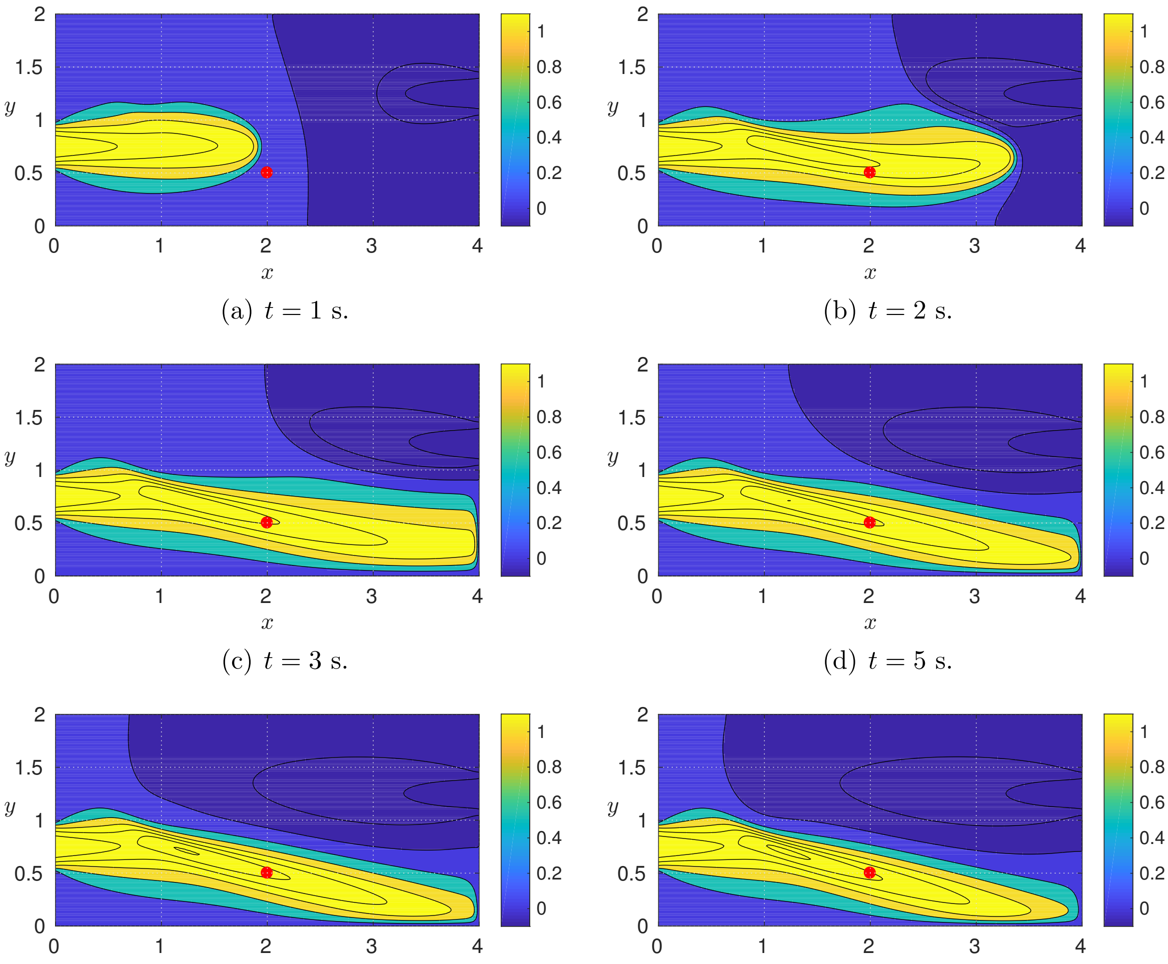


(e) $t = 7$ s.



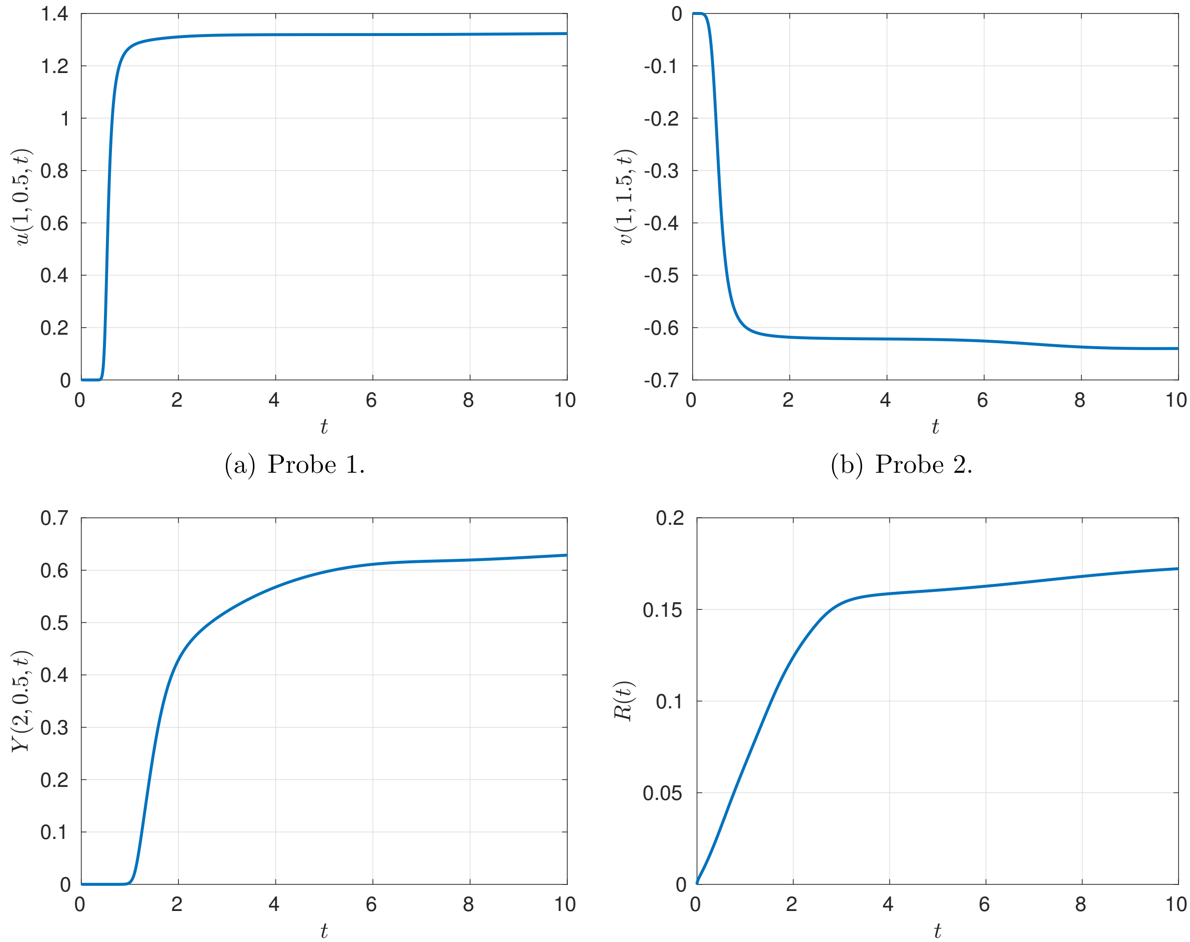
(f) $t = 10$ s.

Figure 2: Profiles of the vertical velocity v for $M = 256$, $N = 128$ and $CFL = 0.8$.



(a) $t = 1$ s. (b) $t = 2$ s. (c) $t = 3$ s. (d) $t = 5$ s. (e) $t = 7$ s. (f) $t = 10$ s.

Figure 3: Profiles of the mass fraction Y for $M = 256$, $N = 128$ and $CFL = 0.8$.



(a) Probe 1.

(b) Probe 2.

(c) Probe 3.

(d) Performance parameter.

Figure 4: Probe curves and performance parameter with time for $M = 256$, $N = 128$ and $CFL = 0.8$.

GCI analysis for u at t=5

The GCI analysis details are shown in the tables below. Note that

$$\beta = \frac{GCI_{12}}{GCI_{23}} r^p,$$

and $u_{h=0}$ is obtained by Richardson extrapolation. We can see that $\beta \in [0.95, 1.05]$ which implies that we are in the asymptotic range of convergence, and for the last mesh we have a GCI_{12} value less than 0.02%, the requested accuracy.

M	N	$u(1, 0.5)$
16	8	1.543690071443089
32	16	1.335416609160283
64	32	1.322019032777453
128	64	1.319745850662477
256	128	1.319205904441201
512	256	1.319067860936907

Table 1: GCI analysis data.

M	N	p	u_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	3.9584350799233	1.32109795907916	0.0870896783116511	1.34027931647915	1.01013417814015
128	64	2.5591868664554	1.31928134384948	0.0439958584412932	0.258854908675024	1.00172243929686
256	128	2.07382564275753	1.31903769772281	0.0159382547694397	0.0670728514988665	1.00040929639546
512	256	1.96769271899691	1.31902044645506	0.00449318068173795	0.0175728809198171	1.0001046523139

Figure 5: GCI analysis results for the probe 1 measurement.

GCI analysis for v at t=5

We can see that $\beta \in [0.95, 1.05]$ which implies that we are in the asymptotic range of convergence, and for the last mesh we have a GCI_{12} value less than 0.02%, the requested accuracy.

M	N	$v(1, 1.5)$
16	8	-0.757061737618705
32	16	-0.625718980721275
64	32	-0.622907528189143
128	64	-0.622804506934971
256	128	-0.622781023934629
512	256	-0.622773391772338

Table 2: GCI analysis data.

M	N	p	v_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	5.54587714377773	-0.622846031378078	0.0123406782470828	0.573929637687545	1.00451343482764
128	64	4.77030177533616	-0.622800588290663	0.000786491640724292	0.0214598262263602	1.00016541507425
256	128	2.13325335506559	-0.622774090776325	0.00139157224559648	0.00610467616774274	1.000037706673
512	256	1.62145299393937	-0.62276971689401	0.000737603431890711	0.00226946554749653	1.00001225512323

Figure 6: GCI analysis results for the probe 2 measurement.

GCI analysis for Y at t=5

We can see that $\beta \in [0.95, 1.05]$ which implies that we are in the asymptotic range of convergence, and for the last mesh we have a GCI_{12} value less than 0.4%, the requested accuracy.

M	N	$Y(2, 0.5)$
16	8	0.582656635075406
32	16	0.588540399303644
64	32	0.591682314985302
128	64	0.594958713346013
256	128	0.596429634107325
512	256	0.597084423776656

Table 3: GCI analysis data.

M	N	p	Y_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	0.90509497255469	0.595282672183354	0.76061872791951	1.431990243934	0.994689860416509
128	64	-0.0604663135207332	0.515135896804494	-16.7706629785703	-16.1713507912412	0.994493066010762
256	128	1.1553912471178	0.597627991753401	0.251152352588463	0.56081168901127	0.997533789944036
512	256	1.16761606493148	0.597609767701518	0.109981081389411	0.247332947322324	0.998903354964158

Figure 7: GCI analysis results for the probe 3 measurement.

GCI analysis for R at t=5

We can see that $\beta \in [0.95, 1.05]$ which implies that we are in the asymptotic range of convergence, and for the last mesh we have a GCI_{12} value less than 0.5%, the requested accuracy.

M	N	$v(1, 1.5)$
16	8	0.168352427538196
32	16	0.162648932748133
64	32	0.161109373859052
128	64	0.160690350976423
256	128	0.160521007689289
512	256	0.160443010433551

Table 4: GCI analysis data.

M	N	p	R_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	1.88932914252264	0.160540142816762	0.441649536472937	1.62066075053525	1.00955598580148
128	64	1.87741611637481	0.160533657951695	0.12189050538481	0.4466809515543	1.00260764184087
256	128	1.30707823223154	0.160406151892441	0.0894398484824594	0.221076656325998	1.00105496027943
512	256	1.11845552861879	0.16037641124134	0.0518869535290025	0.112599318019901	1.00048613682521

Figure 8: GCI analysis results for the parameter R.

Steady state results

I ran the simulation using $M = 256$ and $N = 128$ up to a $t = 30$ to reach the steady state solution of the problem. In the following tables are the results obtained, with the GCI analysis.

M	N	p	u_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	2.42025580560709	1.32325394491015	0.931314723997211	4.82842657100011	1.03242956496118
128	64	3.03635881705907	1.32718554467362	0.0688575707540411	0.562686755373776	1.00396849764547
256	128	2.18484741417542	1.32643123629202	0.0307875868946674	0.13986241874637	1.00087357609845

Figure 9: GCI analysis results for the probe 1 measurement.

M	N	p	v_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	10.0848745651173	-0.6357220573214	1.61430363000829e-05	0.0175345987016561	0.999859872011669
128	64	-2.99897524107977	-0.635620156381226	-0.159869715629154	-0.0200203163251724	0.998881025587533
256	128	2.03872828578847	-0.636663203035808	0.0109470301413326	0.0449917553978741	0.999727740193495

Figure 10: GCI analysis results for the probe 2 measurement.

M	N	p	Y_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	3.09041984891174	0.645196690168092	0.0851668530235064	0.72170703862935	1.0051218935378
128	64	7.97747394230764	0.645649757906483	1.01190659983891e-05	0.00255039941721193	0.999979678170884
256	128	-5.06035733276536	0.645636179483926	-0.0873197317099373	-0.00261871016576605	0.999322377632678

Figure 11: GCI analysis results for the probe 3 measurement.

M	N	p	R_h0	GCI12	GCI23	beta
16	8	NaN	NaN	NaN	NaN	NaN
32	16	NaN	NaN	NaN	NaN	NaN
64	32	1.0603013686232	0.175980285661211	1.13828471535011	2.35050986276912	1.00988365419262
128	64	1.42644495586338	0.176557559780812	0.27333458615699	0.731976915731664	1.00369075099285
256	128	1.25704037320375	0.176474670900958	0.139078136022136	0.331890297942506	1.00154660371752

Figure 12: GCI analysis results for the parameter R.

HOMEWORK 10 - FRANCISCO CASTILLO

Contents

Defined functions

```
function [u,v,Y]=initialization(M,N,hx,hy)
x=linspace(-hx/2,4+hx/2,M+2);
y=linspace(-hy/2,2+hy/2,N+2);
%% Initialize u
u = zeros(M+1,N+2);
% Boundary Conditions
u=applyBCs(u,M,N,hx,hy,'u');

%% Initialize v
v = zeros(M+2,N+1);
% Boundary Conditions
v=applyBCs(v,M,N,hx,hy,'v');

%% Initialize Y
x=linspace(-5*hx/2,4+5*hx/2,M+6);
y=linspace(-5*hy/2,2+5*hy/2,N+6);
Y = zeros(M+6,N+6);
% Boundary Conditions
Y=applyBCs(Y,M,N,hx,hy,'Y');
end


function phi = applyBCs(phi,M,N,hx,hy,opt)
switch opt
    case 'u'
        y=linspace(-hy/2,2+hy/2,N+2);
        %---------- Boundary Conditions for u----------%
        % Left and right
        for j=1:N+2
            if (y(j)>=0.5 && y(j)<=1) % Inlet 1
                phi(1,j)=-48*y(j)^2+72*y(j)-24;
            elseif (y(j)>=1 && y(j)<=1.5) % Inlet 2
                phi(M+1,j)=24*y(j)^2-60*y(j)+36;
            end
        end
        % Top and bottom
        phi(:,1)=-phi(:,2);
        phi(:,N+2)=-phi(:,N+1);
        case 'v'
        x=linspace(-hx/2,4+hx/2,M+2);
        %---------- Boundary Conditions ----------%
        % Top and bottom
        for i=1:M+2
            if (x(i)>=0.5 && x(i)<=1) % Inlet 3
                phi(i,N+1)=24*x(i)^2-36*x(i)+12;
            elseif (x(i)>=1.5 && x(i)<=2.5) % Outlet
                phi(i,1)=(4*phi(i,2)-phi(i,3))/3;
            end
        end
        % Left and right
        phi(1,:)=-phi(2,:);
        phi(M+2,:)=-phi(M+1,:);
    case 'Y'
        x=linspace(-5*hx/2,4+5*hx/2,M+6);
        y=linspace(-5*hy/2,2+5*hy/2,N+6);
        %---------- Boundary Conditions ----------%
        % Zero Neumann for walls and outlet
        %Left
        phi(3,4:N+3)=phi(4,4:N+3);
        phi(2,4:N+3)=phi(5,4:N+3);
        phi(1,4:N+3)=phi(6,4:N+3);
        %Right
```

```
        phi(M+4,4:N+3)=phi(M+3,4:N+3);
        phi(M+5,4:N+3)=phi(M+2,4:N+3);
        phi(M+6,4:N+3)=phi(M+1,4:N+3);
        %Bottom
        phi(4:M+3,3)=phi(4:M+3,4);
        phi(4:M+3,2)=phi(4:M+3,5);
        phi(4:M+3,1)=phi(4:M+3,6);
        %Top
        phi(4:M+3,N+4)=phi(4:M+3,N+3);
        phi(4:M+3,N+5)=phi(4:M+3,N+2);
        phi(4:M+3,N+6)=phi(4:M+3,N+1);
        % Dirichlet for the Inlets
        %Inlets 1 and 2
        for j=3:N+3
            if (y(j)>=0.5 && y(j)<=1) % Inlet 1
                phi(3,j)=2-phi(4,j);
                phi(2,j)=2-phi(5,j);
                phi(1,j)=2-phi(6,j);
            elseif (y(j)>=1 && y(j)<=1.5) % Inlet 2
                phi(M+4,j)=2*0.25-phi(M+3,j);
                phi(M+5,j)=2*0.25-phi(M+2,j);
                phi(M+6,j)=2*0.25-phi(M+1,j);
            end
        end
        %Inlet 3
        for i=3:M+3
            if (x(i)>=0.5 && x(i)<=1) % Inlet 3
                phi(i,N+4)=-phi(i,N+3);
                phi(i,N+5)=-phi(i,N+2);
                phi(i,N+6)=-phi(i,N+1);
            end
        end
end
end



function dt=calcTimeStep(u,v,hx,hy,CFL)
eps=1e-12;
dtu=CFL*min(hx,hy)/(eps+max(max(abs(2*u)))+max(max(abs(v))));
dtv=CFL*min(hx,hy)/(eps+max(max(abs(u)))+max(max(abs(2*v))));
dt=min(dtu,dtv);
end



function u=ADI_u(u,M,N,dt,hx,hy,Re,Q)
if nargin<8
    disp('Source term zero')
    Q=zeros(size(u));
end
%%%%%%%%%% STEP 1 %%%%%%%%%%
[a,b,c,d]=uvectors(u,M,N,dt,hx,hy,1/Re,1,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d); %Gaussian elimination
% Correspondance with u, obtain u(n+1/2)
for j=2:N+1
    for i=2:M
        u(i,j)=d((j-2)*(M-1)+i-1);
    end
end
% Update top and bottom boundaries (ghost cells), they depend on the
% interior, I don't call the BCs function because the for loops in it are
% unnecessary.
u(:,N+2)=-u(:,N+1);
u(:,1)=-u(:,2);
%%%%%%%%%% STEP 2 %%%%%%%%%%
[a,b,c,d]=uvectors(u,M,N,dt,hx,hy,1/Re,2,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d);
for i=2:M
    for j=2:N+1
        u(i,j)=d((i-2)*N+j-1);
    end
end
% Update top and bottom boundaries (ghost cells), they depend on the
% interior, I don't call the BCs function because the for loops in it are
% unnecessary.
u(:,N+2)=-u(:,N+1); % Top
u(:,1)=-u(:,2); % Bottom
end



function [a,b,c,d] = uvectors(u,M,N,dt,hx,hy,alpha,step,Q)
if nargin<9
    disp('nargin')
    Q=zeros(size(Y));
end
```

```
dx=alpha*dt/hx^2;
dy=alpha*dt/hy^2;
d1=dx/2;
d2=dy/2;
if step==1
    a = -d1*ones((M-1)*N,1);
    b = (1+2*d1)*ones((M-1)*N,1);
    c = -d1*ones((M-1)*N,1);
    d = zeros((M-1)*N,1);
    for j=2:N+1
        for i=2:M
            if i==2 % Left boundary BCs
                a((j-2)*(M-1)+1)=0;
                d((j-2)*(M-1)+1)=d2*u(2,j-1)+(1-2*d2)*u(2,j)+d2*u(2,j+1)...
                                    +(dt/2)*Q(i,j)+d1*u(1,j); %u(1,j) fixed at initialization
            elseif i==M % Right boundary BCs
                c((j-2)*(M-1)+M-1)=0;
                d((j-2)*(M-1)+M-1)=d2*u(i,j-1)+(1-2*d2)*u(i,j)+d2*u(i,j+1)...
                                    +(dt/2)*Q(i,j)+d1*u(M+1,j); %u(M+1,j) fixed at initialization
            else % The top and bottom boundaries are imposed by updating the ghost
                    % cells accordingly with them.
                d((j-2)*(M-1)+i-1)=d2*u(i,j-1)+(1-2*d2)*u(i,j)+d2*u(i,j+1)+(dt/2)*Q(i,j);
            end
        end
    end
elseif step==2
    a = -d2*ones((M-1)*N,1);
    b = (1+2*d2)*ones((M-1)*N,1);
    c = -d2*ones((M-1)*N,1);
    d = zeros((M-1)*N,1);
    for i=2:M
        for j=2:N+1
            if j==2 % Bottom boundary BCs
                a((i-2)*N+1)=0;
                b((i-2)*N+1)=1+3*d2;
                d((i-2)*N+1)=d1*u(i-1,2)+(1-2*d1)*u(i,2)+d1*u(i+1,2)+(dt/2)*Q(i,j);
            elseif j==N+1 % Top boundary BCs
                b((i-1)*N)=1+3*d2;
                c((i-1)*N)=0;
                d((i-1)*N)=d1*u(i-1,N+1)+(1-2*d1)*u(i,N+1)+d1*u(i+1,N+1)+(dt/2)*Q(i,j);
            else % The left and right boundaries are imposed by initialization,
                    % they do not change
                d((i-2)*N+j-1)=d1*u(i-1,j)+(1-2*d1)*u(i,j)+d1*u(i+1,j)+(dt/2)*Q(i,j);
            end
        end
    end
end
end



function v=ADI_v(v,M,N,dt,hx,hy,Re,Q)
if nargin<8
    disp('Source term zero')
    Q=zeros(size(u));
end
%%%%%%%%%% STEP 1 %%%%%%%%%%
[a,b,c,d]=vvectors(v,M,N,dt,hx,hy,1/Re,1,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d); %Gaussian elimination
% Correspondance with v, obtain v(n+1/2)
for j=2:N
    for i=2:M+1
        v(i,j)=d((j-2)*M+i-1);
    end
end
% Update left and right boundaries (ghost cells), as well as the outlet
% Neumann BC, they depend on the interior.
v=applyBCs(v,M,N,hx,hy,'v');

%%%%%%%%%% STEP 2 %%%%%%%%%%
[a,b,c,d]=vvectors(v,M,N,dt,hx,hy,1/Re,2,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d); %Gaussian elimination
% Correspondance with v, obtain v(n+1/2)
for i=2:M+1
    for j=2:N
        v(i,j)=d((i-2)*(N-1)+j-1);
    end
end
% Update left and right boundaries (ghost cells), as well as the outlet
% Neumann BC, they depend on the interior.
v=applyBCs(v,M,N,hx,hy,'v');
end



function [a,b,c,d] = vvectors(v,M,N,dt,hx,hy,alpha,step,Q)
xi=linspace(-hx/2,4+hx/2,M+2);
```

```matlab
% keyboard
dx=alpha*dt/hx^2;
dy=alpha*dt/hy^2;
d1=dx/2;
d2=dy/2;
if step==1
    a = -d1*ones(M*(N-1),1);
    b = (1+2*d1)*ones(M*(N-1),1);
    c = -d1*ones(M*(N-1),1);
    d = zeros(M*(N-1),1);
    for j=2:N
        for i=2:M+1
            if i==2
                a((j-2)*M+1)=0;
                b((j-2)*M+1)=1+3*d1;
                d((j-2)*M+1)=d2*v(2,j-1)+(1-2*d2)*v(2,j)+d2*v(2,j+1)+(dt/2)*Q(i,j);
            elseif i==M+1
                b((j-1)*M)=1+3*d1;
                c((j-1)*M)=0;
                d((j-1)*M)=d2*v(M+1,j-1)+(1-2*d2)*v(M+1,j)+d2*v(M+1,j+1)+(dt/2)*Q(i,j);
            elseif (j==2 && xi(i)>=1.5 && xi(i)<=2.5)
                d(i-1)=d2*(4*v(i,2)/3-v(i,3)/3)+(1-2*d2)*v(i,2)+d2*v(i,3)+(dt/2)*Q(i,j);
            else
                d((j-2)*M+i-1)=d2*v(i,j-1)+(1-2*d2)*v(i,j)+d2*v(i,j+1)+(dt/2)*Q(i,j);
            end
        end
    end
elseif step==2
    a = -d2*ones(M*(N-1),1);
    b = (1+2*d2)*ones(M*(N-1),1);
    c = -d2*ones(M*(N-1),1);
    d = zeros(M*(N-1),1);
    for i=2:M+1
        for j=2:N
            if j==2
                a((i-2)*(N-1)+1)=0;
                if (xi(i)>=1.5 && xi(i)<=2.5)
                    b((i-2)*(N-1)+1)=1+(2-4/3)*d2;
                    c((i-2)*(N-1)+1)=-(2/3)*d2;
                end
                d((i-2)*(N-1)+1)=d1*v(i-1,2)+(1-2*d1)*v(i,2)+d1*v(i+1,2)+(dt/2)*Q(i,j);
            elseif j==N
                c((i-1)*(N-1))=0;
                d((i-1)*(N-1))=d1*v(i-1,N)+(1-2*d1)*v(i,N)+d1*v(i+1,N)...
                                +(dt/2)*Q(i,j)+d2*v(i,N+1);
            else
                d((i-2)*(N-1)+j-1)=d1*v(i-1,j)+(1-2*d1)*v(i,j)+d1*v(i+1,j)+(dt/2)*Q(i,j);
            end
        end
    end
end
end


function Y=ADI_Y(Y,M,N,dt,hx,hy,Re,Sc,Q)
if nargin<9
    disp('Source term zero')
    Q=zeros(size(Y));
end
%%%%%%%%%% STEP 1 %%%%%%%%%%
[a,b,c,d]=Yvectors(Y,M,N,dt,hx,hy,1/(Re*Sc),1,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d); %Gaussian elimination
% Correspondance with Y, obtain Y(n+1/2)
for j=4:N+3
    for i=4:M+3
        Y(i,j)=d((j-4)*M+i-3);
    end
end
% Boundary Conditions
Y=applyBCs(Y,M,N,hx,hy,'Y');

%%%%%%%%%% STEP 2 %%%%%%%%%%
[a,b,c,d]=Yvectors(Y,M,N,dt,hx,hy,1/(Re*Sc),2,Q); %Obtain tridiagonal vectors
d=GaussTriSol(a,b,c,d); %Gaussian elimination
% Correspondance with Y, obtain Y(n+1)
for i=4:M+3
    for j=4:N+3
        Y(i,j)=d((i-4)*N+j-3);
    end
end
% Boundary Conditions
Y=applyBCs(Y,M,N,hx,hy,'Y');
end
```

```matlab
function [a,b,c,d] = Yvectors(Y,M,N,dt,hx,hy,alpha,step,Q)
Yinlet1=1;
Yinlet2=0.25;
if nargin<9
    disp('nargin')
    Q=zeros(size(Y));
end
dx=alpha*dt/hx^2;
dy=alpha*dt/hy^2;
d1=dx/2;
d2=dy/2;
x=linspace(-5*hx/2,4+5*hx/2,M+6);
y=linspace(-5*hy/2,2+5*hy/2,N+6);
if step==1
    a = -d1*ones(M*N,1);
    b = (1+2*d1)*ones(M*N,1);
    c = -d1*ones(M*N,1);
    d = zeros(M*N,1);
    for j=4:N+3
        for i=4:M+3
            if i==4 % Left boundary BCs
                a((j-4)*M+1)=0;
                if (y(j)>=0.5 && y(j)<=1)
                    b((j-4)*M+1)=1+3*d1;
                    d((j-4)*M+1)=d2*Y(4,j-1)+(1-2*d2)*Y(4,j)+d2*Y(4,j+1)...
                                    +d1*2*Yinlet1+(dt/2)*Q(i,j);
                else
                    b((j-4)*M+1)=1+d1;
                    d((j-4)*M+1)=d2*Y(4,j-1)+(1-2*d2)*Y(4,j)+d2*Y(4,j+1)...
                        +(dt/2)*Q(i,j);
                end
            elseif i==M+3% Right boundary BCs
                c((j-4)*M+M)=0;
                if (y(j)>=1 && y(j)<=1.5)
                    b((j-4)*M+M)=1+3*d1;
                    d((j-4)*M+M)=d2*Y(i,j-1)+(1-2*d2)*Y(i,j)+d2*Y(i,j+1)...
                                    +d1*2*Yinlet2+(dt/2)*Q(i,j);
                else
                    b((j-4)*M+M)=1+d1;
                    d((j-4)*M+M)=d2*Y(i,j-1)+(1-2*d2)*Y(i,j)+d2*Y(i,j+1)...
                                    +(dt/2)*Q(i,j);
                end
            else % The top and bottom boundaries are imposed by updating the ghost
                    % cells accordingly with them.
                d((j-4)*M+i-3)=d2*Y(i,j-1)+(1-2*d2)*Y(i,j)+d2*Y(i,j+1)+(dt/2)*Q(i,j);
            end
        end
    end
elseif step==2
    a = -d2*ones(M*N,1);
    b = (1+2*d2)*ones(M*N,1);
    c = -d2*ones(M*N,1);
    d = zeros(M*N,1);
    for i=4:M+3
        for j=4:N+3
            if j==4 % Bottom boundary BCs
                a((i-4)*N+1)=0;
                b((i-4)*N+1)=1+d2;
                d((i-4)*N+1)=d1*Y(i-1,4)+(1-2*d1)*Y(i,4)+d1*Y(i+1,4)...
                                +(dt/2)*Q(i,j);
            elseif j==N+3 % Top boundary BCs
                c((i-4)*N+N)=0;
                if (x(i)>=0.5 && x(i)<=1)
                    b((i-4)*N+N)=1+3*d2;
                else
                    b((i-4)*N+N)=1+d2;
                end
                d((i-4)*N+N)=d1*Y(i-1,N+3)+(1-2*d1)*Y(i,N+3)+d1*Y(i+1,N+3)...
                                +(dt/2)*Q(i,j);
            else % The left and right boundaries are imposed by initialization,
                    % they do not change
                d((i-4)*N+j-3)=d1*Y(i-1,j)+(1-2*d1)*Y(i,j)+d1*Y(i+1,j)...
                                +(dt/2)*Q(i,j);
            end
        end
    end
end
end


function HY=HyperbolicY(Y,M,N,hx,hy,dt,u,v)
Yconv = TVDRK3_2D(Y,M,N,hx,hy,dt,u,v);
HY=zeros(M+6,N+6);
HY(4:M+3,4:N+3)=(Yconv(4:M+3,4:N+3)-Y(4:M+3,4:N+3))/dt;
end
```

```matlab
function phi3 = TVDRK3_2D(phi0,M,N,hx,hy,dt,u,v)
% Constants and prealocation
a10=1;
a20=-3/4; a21=1/4;
a30=-1/12; a31=-1/12; a32=2/3;
phi1=zeros(M+6,N+6);
phi2=zeros(M+6,N+6);
phi3=zeros(M+6,N+6);

%%% STEP 1 %%%
for i=4:M+3
    for j=4:N+3
        uij=(u(i-3,j-2)+u(i-2,j-2))/2; % Values of u,v at cell centers
        vij=(v(i-2,j-3)+v(i-2,j-2))/2;
        [DphiDx0,DphiDy0]=WENO5_2D(phi0,uij,vij,i,j,hx,hy);
        phi1(i,j)=phi0(i,j)-a10*dt*(uij*DphiDx0+vij*DphiDy0);
    end
end
% Update ghost cells
phi1=applyBCs(phi1,M,N,hx,hy,'Y');
%%% STEP 2 %%%
for i=4:M+3
    for j=4:N+3
        uij=(u(i-3,j-2)+u(i-2,j-2))/2; % Values of u,v at cell centers
        vij=(v(i-2,j-3)+v(i-2,j-2))/2;
        [DphiDx0,DphiDy0]=WENO5_2D(phi0,uij,vij,i,j,hx,hy);
        [DphiDx1,DphiDy1]=WENO5_2D(phi1,uij,vij,i,j,hx,hy);
        phi2(i,j)=phi1(i,j)-a20*dt*(uij*DphiDx0+vij*DphiDy0)...
            -a21*dt*(uij*DphiDx1+vij*DphiDy1);
    end
end
% Update ghost cells
phi2=applyBCs(phi2,M,N,hx,hy,'Y');

%%% STEP 3 %%%
for i=4:M+3
    for j=4:N+3
        uij=(u(i-3,j-2)+u(i-2,j-2))/2; % Values of u,v at cell centers
        vij=(v(i-2,j-3)+v(i-2,j-2))/2;
        [DphiDx0,DphiDy0]=WENO5_2D(phi0,uij,vij,i,j,hx,hy);
        [DphiDx1,DphiDy1]=WENO5_2D(phi1,uij,vij,i,j,hx,hy);
        [DphiDx2,DphiDy2]=WENO5_2D(phi2,uij,vij,i,j,hx,hy);
        phi3(i,j)=phi2(i,j)-a30*dt*(uij*DphiDx0+vij*DphiDy0)...
            -a31*dt*(uij*DphiDx1+vij*DphiDy1)...
            -a32*dt*(uij*DphiDx2+vij*DphiDy2);
    end
end
% Update ghost cells
phi3=applyBCs(phi3,M,N,hx,hy,'Y');
end


function [DphiDx,DphiDy] = WENO5_2D(phi,uij,vij,i,j,hx,hy)
if uij>=0
    DphiDx = (1/(12*hx))*(phi(i-2,j)-8*phi(i-1,j)+8*phi(i+1,j)-phi(i+2,j))...
        -psiWENO((phi(i-1,j)-2*phi(i-2,j)+phi(i-3,j))/hx,...
        (phi(i,j)-2*phi(i-1,j)+phi(i-2,j))/hx,...
        (phi(i+1,j)-2*phi(i,j)+phi(i-1,j))/hx,...
        (phi(i+2,j)-2*phi(i+1,j)+phi(i,j))/hx);
elseif uij<0
    DphiDx = (1/(12*hx))*(phi(i-2,j)-8*phi(i-1,j)+8*phi(i+1,j)-phi(i+2,j))...
        +psiWENO((phi(i+3,j)-2*phi(i+2,j)+phi(i+1,j))/hx,...
        (phi(i+2,j)-2*phi(i+1,j)+phi(i,j))/hx,...
        (phi(i+1,j)-2*phi(i,j)+phi(i-1,j))/hx,...
        (phi(i,j)-2*phi(i-1,j)+phi(i-2,j))/hx);
end
if vij>=0
    DphiDy = (1/(12*hy))*(phi(i,j-2)-8*phi(i,j-1)+8*phi(i,j+1)-phi(i,j+2))...
        -psiWENO((phi(i,j-1)-2*phi(i,j-2)+phi(i,j-3))/hy,...
        (phi(i,j)-2*phi(i,j-1)+phi(i,j-2))/hy,...
        (phi(i,j+1)-2*phi(i,j)+phi(i,j-1))/hy,...
        (phi(i,j+2)-2*phi(i,j+1)+phi(i,j))/hy);
elseif vij<0
    DphiDy = (1/(12*hy))*(phi(i,j-2)-8*phi(i,j-1)+8*phi(i,j+1)-phi(i,j+2))...
        +psiWENO((phi(i,j+3)-2*phi(i,j+2)+phi(i,j+1))/hy,...
        (phi(i,j+2)-2*phi(i,j+1)+phi(i,j))/hy,...
        (phi(i,j+1)-2*phi(i,j)+phi(i,j-1))/hy,...
        (phi(i,j)-2*phi(i,j-1)+phi(i,j-2))/hy);
end

end
```

```
function psi = psiWENO(a,b,c,d)
eps=1e-6;
IS0=13*(a-b)^2+3*(a-3*b)^2;
IS1=13*(b-c)^2+3*(b+c)^2;
IS2=13*(c-d)^2+3*(3*c-d)^2;

a0=(eps+IS0)^(-2);
a1=6*(eps+IS1)^(-2);
a2=3*(eps+IS2)^(-2);

w0=a0/(a0+a1+a2);
w2=a2/(a0+a1+a2);

psi = (a-2*b+c)*w0/3+(w2-0.5)*(b-2*c+d)/6;
end



function [Hu,Hv]=HyperbolicBurgers2D(u,v,M,N,hx,hy)
Hu=zeros(size(u));
Hv=zeros(size(v));
for i=2:M
    for j=2:N+1
        Hu(i,j)=-(u(i+1,j)^2+2*u(i,j)*(u(i+1,j)-u(i-1,j))-u(i-1,j)^2)/(4*hx)...
            -((u(i,j)+u(i,j+1))*(v(i,j)+v(i+1,j))-(u(i,j-1)+u(i,j))*(v(i,j-1)+v(i+1,j-1)))/(4*hy);
    end
end
for i=2:M+1
    for j=2:N
        Hv(i,j)=-(v(i,j+1)^2+2*v(i,j)*(v(i,j+1)-v(i,j-1))-v(i,j-1)^2)/(4*hx)...
            -((u(i,j)+u(i,j+1))*(v(i,j)+v(i+1,j))-(u(i-1,j+1)+u(i-1,j))*(v(i,j)+v(i-1,j)))/(4*hy);
    end
end
end



function [u,v,Hu,Hv]=solveBurgers2D(u,v,Hu,Hv,M,N,hx,hy,dt,Re,time)
Huold=Hu;
Hvold=Hv;
[Hu,Hv]=HyperbolicBurgers2D(u,v,M,N,hx,hy);
if time==0 %FCTS for the first time step
    Huold=Hu;
    Hvold=Hv;
end
% % % % % for i=1:M+1
% % % % %     for j=1:N+2
% % % % %         HuoldVector((j-1)*(M+1)+i,1)=Huold(i,j);
% % % % %     end
% % % % % end
% % % % % for i=1:M+2
% % % % %     for j=1:N+1
% % % % %         HvoldVector((j-1)*(M+2)+i,1)=Hvold(i,j);
% % % % %     end
% % % % % end
u=ADI_u(u,M,N,dt,hx,hy,Re,1.5*Hu-0.5*Huold);
v=ADI_v(v,M,N,dt,hx,hy,Re,1.5*Hv-0.5*Hvold);

% % % % % for i=1:M+1
% % % % %         for j=1:N+2
% % % % %             uVector((j-1)*(M+1)+i,1)=u(i,j);
% % % % %         end
% % % % % end
% % % % % for i=1:M+2
% % % % %         for j=1:N+1
% % % % %             vVector((j-1)*(M+2)+i,1)=v(i,j);
% % % % %         end
% % % % % end
% % % % % keyboard
end



function R=performance(Y,M,N,hx,hy,L,H)
R=sum(sum(Y(4:M+3,4:N+3).*(1-Y(4:M+3,4:N+3))))*hx*hy/(H*L);
end



function d=GaussTriSol(a,b,c,d)
    N=length(a);
    for i=2:N
        b(i)=b(i)-c(i-1)*a(i)/b(i-1);
        d(i)=d(i)-d(i-1)*a(i)/b(i-1);
    end
    d(N)=d(N)/b(N);
    for i=N-1:-1:1
        d(i)=(d(i)-c(i)*d(i+1))/b(i);
```

```
    end
end


function ContourPlots(u,v,Y,M,N,hx,hy,n)
axisSize=14;
markersize=16;
linewidth=3.5;

% Define the points of the different meshes
xu=linspace(0,4,M+1);
yu=linspace(-hy/2,2+hy/2,N+2);
xv=linspace(-hx/2,4+hx/2,M+2);
yv=linspace(0,2,N+1);
xY=linspace(-5*hx/2,4+5*hx/2,M+6);
yY=linspace(-5*hx/2,2+5*hx/2,N+6);

% Plot u
figure(10+n)
contourf(xu,yu,u')
hold on
plot(1,0.5,'r.','markersize',markersize,'linewidth',linewidth)
axis([0 4 0 2])
caxis([-1.5 3])
colorbar
xlabel('$x$','Interpreter','latex')
ylabel('$y$','Interpreter','latex')
% yticks([0 0.25 0.50 0.75 1.0 1.25 1.50 1.75 2.0]);
% xticks([0 0.50 1.0 1.50 2.0 2.50 3.0 3.50 4.0]);
set(get(gca,'ylabel'),'rotation',0)
set(gca,'fontsize',axisSize)
pbaspect([2 1 1])
grid on
txt=['Latex/FIGURES/u_' num2str(10+n)];
saveas(gcf,txt,'epsc')

% Plot v
figure(20+n)
contourf(xv,yv,v')
hold on
plot(1,1.5,'r.','markersize',markersize,'linewidth',linewidth)
axis([0 4 0 2])
caxis([-1.5 0])
colorbar
xlabel('$x$','Interpreter','latex')
ylabel('$y$','Interpreter','latex')
% yticks([0 0.25 0.50 0.75 1.0 1.25 1.50 1.75 2.0]);
% xticks([0 0.50 1.0 1.50 2.0 2.50 3.0 3.50 4.0]);
set(get(gca,'ylabel'),'rotation',0)
set(gca,'fontsize',axisSize)
pbaspect([2 1 1])
grid on
txt=['Latex/FIGURES/v_' num2str(20+n)];
saveas(gcf,txt,'epsc')

% Plot Y
figure(30+n)
contourf(xu,yu,u')
hold on
plot(2,0.5,'r.','markersize',markersize,'linewidth',linewidth)
axis([0 4 0 2])
caxis([-0.1 1.1])
colorbar
xlabel('$x$','Interpreter','latex')
ylabel('$y$','Interpreter','latex')
% yticks([0 0.25 0.50 0.75 1.0 1.25 1.50 1.75 2.0]);
% xticks([0 0.50 1.0 1.50 2.0 2.50 3.0 3.50 4.0]);
set(get(gca,'ylabel'),'rotation',0)
set(gca,'fontsize',axisSize)
pbaspect([2 1 1])
grid on
txt=['Latex/FIGURES/Y_' num2str(30+n)];
saveas(gcf,txt,'epsc')
end


function [pu,pv,pY]=probeValues(u,v,Y,M,N,hx,hy)
% Define the points of the different meshes
xu=linspace(0,4,M+1);
yu=linspace(-hy/2,2+hy/2,N+2);
xv=linspace(-hx/2,4+hx/2,M+2);
yv=linspace(0,2,N+1);
xY=linspace(-5*hx/2,4+5*hx/2,M+6);
yY=linspace(-5*hx/2,2+5*hx/2,N+6);
% Probe for u
```

```
pu = ( u(xu==1,find(yu<=0.5,1,'last')) + u(xu==1,find(yu>=0.5,1)) )/2;
% Probe for v
pv = ( v(find(xv<=1,1,'last'),yv==1.5) + v(find(xv>=1,1),yv==1.5) )/2;
% Probe for Y
pY = ( Y(find(xY<=2,1,'last'),find(yY<=0.5,1,'last'))...
    +Y(find(xY<=2,1,'last'),find(yY>0.5,1))...
    +Y(find(xY>2,1),find(yY<=0.5,1,'last'))...
    +Y(find(xY>2,1),find(yY>0.5,1)) )/4;
end


function CurvePlots(pu,pv,pY,R,t)
axisSize=14;
markersize=16;
linewidth=3.5;

% Cut out what hasn't been filled
index=find(t==0,2);
index=index(2);
t(index:end)=[];
pu(index:end)=[];
pv(index:end)=[];
pY(index:end)=[];
R(index:end)=[];

% Probe for u
figure(1)
plot(t,pu,'linewidth',2)
xlim([0 t(end)])
xlabel('$t$','Interpreter','latex')
ylabel('$u(1,0.5,t)$','Interpreter','latex')
set(gca,'fontsize',axisSize)
grid on
txt='Latex/FIGURES/probeu';
saveas(gcf,txt,'epsc')

% Probe for v
figure(2)
plot(t,pv,'linewidth',2)
xlim([0 t(end)])
xlabel('$t$','Interpreter','latex')
ylabel('$v(1,1.5,t)$','Interpreter','latex')
set(gca,'fontsize',axisSize)
grid on
txt='Latex/FIGURES/probev';
saveas(gcf,txt,'epsc')

% Probe for Y
figure(3)
plot(t,pY,'linewidth',2)
xlim([0 t(end)])
xlabel('$t$','Interpreter','latex')
ylabel('$Y(2,0.5,t)$','Interpreter','latex')
set(gca,'fontsize',axisSize)
grid on
txt='Latex/FIGURES/probeY';
saveas(gcf,txt,'epsc')

% Performance parameter R
figure(4)
plot(t,R,'linewidth',2)
xlim([0 t(end)])
xlabel('$t$','Interpreter','latex')
ylabel('$R(t)$','Interpreter','latex')
set(gca,'fontsize',axisSize)
grid on
txt='Latex/FIGURES/Rplot';
saveas(gcf,txt,'epsc')
end
```

```
clear all; close all; format long; clc

Lx=4;
Ly=2;
M=64;
N=32;
CFL = 0.8;
Re = 20;
Sc=0.5;

put5=zeros(6,1);
pvt5=zeros(6,1);
pYt5=zeros(6,1);
```

```
Rt5=zeros(6,1);

put10=zeros(6,1);
pvt10=zeros(6,1);
pYt10=zeros(6,1);
Rt10=zeros(6,1);

Tu5=nan(5,7);
Tv5=nan(5,7);
TY5=nan(5,7);

Tu10=nan(5,7);
Tv10=nan(5,7);
TY10=nan(5,7);


i=0;
checku5=10;
checkv5=10;
checkY5=10;

outputTime=[1 2 3 5 7 10];
endtime=outputTime(end);

while ( checku5>0.02 || checkv5>0.02 || checkY5>0.4 )
```

```
% while ( checku>1 && checkv>1.2 && checkY>6 )
    i=i+1;
    M=2*M
    N=2*N;
    hx = Lx/M;
    hy = Ly/N;
    if hx~=hy
        error('Cells not square')
    end
    time=0
    n=1;
    tstep=1;

    % Define initial values
    [u,v,Y]=initialization(M,N,hx,hy);
    dt=calcTimeStep(u,v,hx,hy,CFL);

    Hu=zeros(size(u));
    Hv=zeros(size(v));

    pu=zeros(5e2,1);
    pv=zeros(5e2,1);
    pY=zeros(5e2,1);
    R=zeros(5e2,1);
    t=zeros(5e2,1);

        [pu(tstep,1),pv(tstep,1),pY(tstep,1)]=probeValues(u,v,Y,M,N,hx,hy);
    R(tstep,1)=performance(Y,M,N,hx,hy,Lx,Ly);

    while time < endtime
```

Calculate Time Step

```
        if (time < outputTime(n) && time+dt >= outputTime(n))
            dt=outputTime(n)-time;
            n=n+1;
        else
            dt=calcTimeStep(u,v,hx,hy,CFL);
        end
```

Solve Hyperbolic part of Y equation

```
        HY=HyperbolicY(Y,M,N,hx,hy,dt,u,v);
```

Solve Y equation

```
        Y=ADI_Y(Y,M,N,dt,hx,hy,Re,Sc,HY);
```

Solve Burger's viscous equation

```
        [u,v,Hu,Hv]=solveBurgers2D(u,v,Hu,Hv,M,N,hx,hy,dt,Re,time);
```

Next time step

```
        time=time+dt;
        tstep=tstep+1;
```

Store probe values and performance parameter

```
        if mod(tstep,5e2)==0
            pu=[pu;zeros(5e2,1)];
            pv=[pv;zeros(5e2,1)];
            pY=[pY;zeros(5e2,1)];
            R=[R;zeros(5e2,1)];
            t=[t;zeros(5e2,1)];
        end
        [pu(tstep,1),pv(tstep,1),pY(tstep,1)]=probeValues(u,v,Y,M,N,hx,hy);
        R(tstep,1)=performance(Y,M,N,hx,hy,Lx,Ly);
        t(tstep)=time; % time vector used to plot
        if time==5
            put5(i)=pu(tstep);
            pvt5(i)=pv(tstep);
            pYt5(i)=pY(tstep);
            Rt5(i)=R(tstep);
        end
        if time==10
            put10(i)=pu(tstep);
            pvt10(i)=pv(tstep);
            pYt10(i)=pY(tstep);
            Rt10(i)=R(tstep);
        end
```

Outputs

```
        if ( M==256 && N==128 )
            if ismember(time,outputTime)
                ContourPlots(u,v,Y,M,N,hx,hy,n-1)
            end
            if time==endtime
                CurvePlots(pu,pv,pY,R,t)
            end
        end
```

Save progress

```
        if ismember(time,outputTime)
            time
            txt=['DataSaved/Save_M',num2str(M),'_time',num2str(time)];
            save(txt)
        end
```

```
    end
```

GCI analysis

Values at t=5

```
    Tu5(i,1)=M;  Tu5(i,2)=N;
    Tv5(i,1)=M;  Tv5(i,2)=N;
    TY5(i,1)=M;  TY5(i,2)=N;
    if i>=3
        [Tu5,Tv5,TY5]=GCIanalysis(put5,pvt5,pYt5,i,Tu5,Tv5,TY5);
        checku5=abs(Tu5(i,5));
        checkv5=abs(Tv5(i,5));
        checkY5=abs(TY5(i,5));
    end
    % Values at t=10
    Tu10(i,1)=M;  Tu10(i,2)=N;
    Tv10(i,1)=M;  Tv10(i,2)=N;
    TY10(i,1)=M;  TY10(i,2)=N;
    if i>=3
        [Tu10,Tv10,TY10]=GCIanalysis(put10,pvt10,pYt10,i,Tu10,Tv10,TY10);
        checku10=abs(Tu10(i,5));
        checkv10=abs(Tv10(i,5));
        checkY10=abs(TY10(i,5));
    end
```

```
end
```

```
Tu5=array2table(Tu5,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})
Tv5=array2table(Tv5,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})
TY5=array2table(TY5,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})

Tu10=array2table(Tu10,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})
Tv10=array2table(Tv10,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})
TY10=array2table(TY10,'VariableNames',{'M','N','p','phi0','GCI12','GCI23','beta'})
```

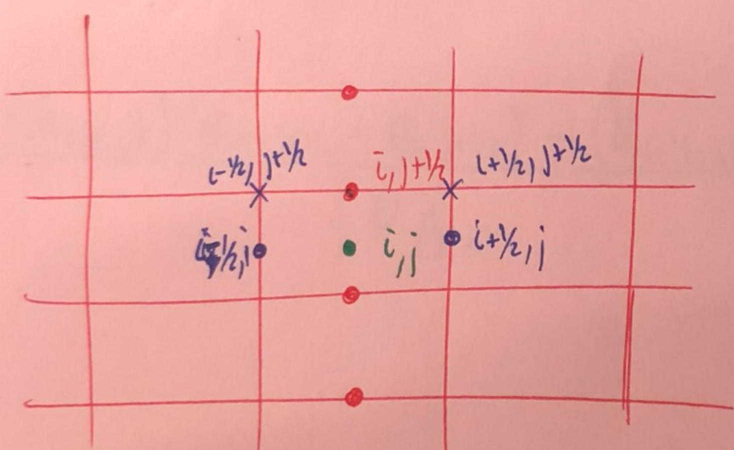
Published with MATLAB® R2017a

Hyperbolic part for the v equation:

$$\frac{v^{n+1}_{i,j+1/2} - v^{n}_{i,j+1/2}}{\Delta t} = \frac{3}{2}\left[-\left.\frac{\partial uv}{\partial x}\right|^{n}_{i,j+1/2} - \left.\frac{\partial vv}{\partial y}\right|^{n}_{i,j+1/2}\right]$$

$$-\frac{1}{2}\left[-\left.\frac{\partial uv}{\partial x}\right|^{n-1}_{i,j+1/2} - \left.\frac{\partial vv}{\partial y}\right|^{n-1}_{i,j+1/2}\right]$$

$$+\frac{\nu}{2}\left[\left.\frac{\partial^2 v}{\partial x^2}\right|^{n}_{i,j+1/2} + \left.\frac{\partial^2 v}{\partial y^2}\right|^{n}_{i,j+1/2} + \left.\frac{\partial^2 v}{\partial x^2}\right|^{n+1}_{i,j+1/2} + \left.\frac{\partial^2 v}{\partial y^2}\right|^{n+1}_{i,j+1/2}\right]$$



$$\left.\begin{aligned}
\left.\frac{\partial vv}{\partial y}\right|_{i,j+1/2} &= \frac{(v_{i,j+1})^2 - (v_{i,j})^2}{\Delta y} \\
\left.\frac{\partial uv}{\partial x}\right|_{i,j+1/2} &= \frac{(uv)_{i+1/2,j+1/2} - (uv)_{i-1/2,j+1/2}}{\Delta x}
\end{aligned}\right\} \text{Convective terms}$$

Note that:

$$v_{i,j} = \frac{1}{2}\left(v_{i,j+1/2} + v_{i,j-1/2}\right)$$

$$v_{i+1/2,j+1/2} = \frac{1}{2}\left(v_{i+1,j+1/2} + v_{i,j+1/2}\right)$$

$$u_{i+1/2,j+1/2} = \frac{1}{2}\left(u_{i+1/2,j+1} + u_{i+1/2,j}\right)$$

Then:

$$\left.\frac{\partial vv}{\partial y}\right|_{i,j+1/2} = \frac{(v_{i,j+1})^2 - (v_{i,j})^2}{\Delta y} =$$

$$= \frac{v_{i,j+3/2}^2 + \cancel{v_{i,j+1/2}^2} + 2v_{i,j+3/2}v_{i,j+1/2} - \cancel{v_{i,j+1/2}^2} - v_{i,j-1/2}^2 - 2v_{i,j+1/2}v_{i,j-1/2}}{4\Delta y}$$

$$= \frac{v_{i,j+3/2}^2 + 2v_{i,j+1/2}(v_{i,j+3/2} - v_{i,j-1/2}) - v_{i,j-1/2}^2}{4\Delta y}$$

$$\left.\frac{\partial uv}{\partial x}\right|_{i,j+1/2} = \frac{(uv)_{i+1/2,j+1/2} - (uv)_{i-1/2,j+1/2}}{\Delta x} =$$

$$= \frac{(u_{i+1/2,j+1} + u_{i+1/2,j})(v_{i+1,j+1/2} + v_{i,j+1/2}) - (u_{i-1/2,j+1} + u_{i-1/2,j})(v_{i,j+1/2} + v_{i-1,j+1/2})}{4\Delta x}$$

$$\Delta t_u = \frac{\min(\Delta x, \Delta y)}{\max(|2u|) + \max(|v|)} \; ; \; \Delta t_v = \frac{\min(\Delta x, \Delta y)}{\max(|u|) + \max(|2v|)}$$

$$\Delta t = CFL \cdot \min(\Delta t_u, \Delta t_v)$$

This ~~satisifie~~ is stable for the Y equation as well