流体的数值模拟初步

In [ ]:

```
```

1. shallow water simulation

In [6]:

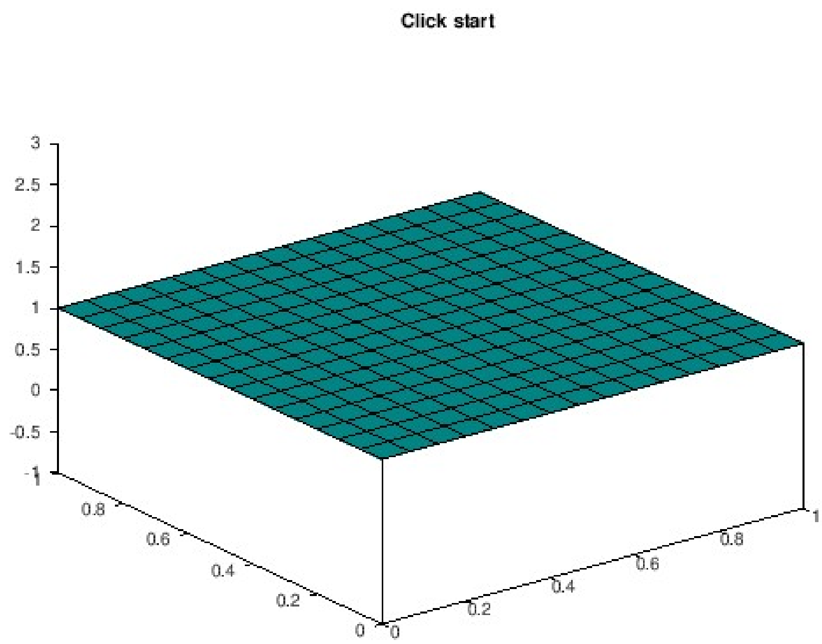
```
function D = droplet(height,width)
% DROPLET  2D Gaussian
% D = droplet(height,width)
   [x,y] = ndgrid(-1:(2/(width-1)):1);
   D = height*exp(-5*(x.^2+y.^2));
endfunction
```

In [8]:

```
function [surfplot,top] = initgraphics(n)
% INITGRAPHICS  Initialize graphics for waterwave.
% [surfplot,top,start,stop] = initgraphics(n)
% returns handles to a surface plot, its title, and two uicontrol toggles.
   clf
   shg
   set(gcf,'numbertitle','off','name','Shallow_water')
   x = (0:n-1)/(n-1);
   surfplot = surf(x,x,ones(n,n),zeros(n,n));
   grid off
   axis([0 1 0 1 -1 3])
   caxis([-1 1])
   shading faceted
   c = (1:64)'/64;
   cyan = [0*c c c];
   colormap(cyan)
   top = title('Click start');
 %  start = uicontrol('position',[20 20 80 20],'style','toggle','string','start');
 %  stop = uicontrol('position',[120 20 80 20],'style','toggle','string','stop');
endfunction
```

In [5]:
```
initgraphics(16)
```

```
ans = -3.4009
```

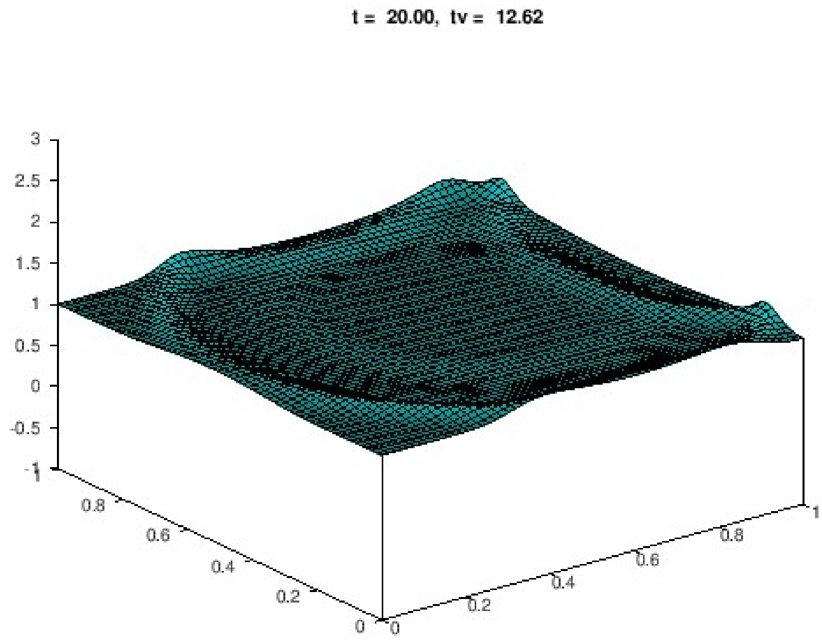


In [ ]:
```
% WATER WAVE
% 2D Shallow Water Model
%
% Lax-Wendroff finite difference method.
% Reflective boundary conditions.
% Random water drops initiate gravity waves.
% Surface plot displays height colored by momentum.
% Plot title shows t = simulated time and tv = a measure of total variation.
% An exact solution to the conservation law would have constant tv.
% Lax-Wendroff produces nonphysical oscillations and increasing tv.
%
% See:
%    http://en.wikipedia.org/wiki/Shallow_water_equations
%    http://www.amath.washington.edu/~rjl/research/tsunamis
%    http://www.amath.washington.edu/~dgeorge/tsunamimodeling.html
%    http://www.amath.washington.edu/~claw/applications/shallow/www
```

In [ ]:
```

```

In [9]:
```
% Parameters
n = 64;                  % grid size
g = 9.8;                 % gravitational constant
dt = 0.02;               % hardwired timestep
dx = 1.0;
dy = 1.0;
nplotstep = 8;           % plot interval
ndrops = 5;              % maximum number of drops
dropstep = 500;          % drop interval
D = droplet(1.5,21);     % simulate a water drop

% Initialize graphics

[surfplot,top] = initgraphics(n);

% Outer loop, restarts.

%while get(stop,'value') == 0
 %  set(start,'value',0)

   H = ones(n+2,n+2);   U = zeros(n+2,n+2);  V = zeros(n+2,n+2);
   Hx = zeros(n+1,n+1); Ux = zeros(n+1,n+1); Vx = zeros(n+1,n+1);
   Hy = zeros(n+1,n+1); Uy = zeros(n+1,n+1); Vy = zeros(n+1,n+1);
   ndrop = ceil(rand*ndrops);

   nstep = 0;
   while nstep < 1000   % Inner loop, time steps.
       nstep = nstep + 1;

       % Random water drops
       if mod(nstep,dropstep) == 0 && nstep <= ndrop*dropstep
           w = size(D,1);
           i = ceil(rand*(n-w))+(1:w);
           j = ceil(rand*(n-w))+(1:w);
           H(i,j) = H(i,j) + rand*D;
       end

       % Reflective boundary conditions
       H(:,1) = H(:,2);      U(:,1) = U(:,2);       V(:,1) = -V(:,2);
       H(:,n+2) = H(:,n+1);  U(:,n+2) = U(:,n+1);   V(:,n+2) = -V(:,n+1);
       H(1,:) = H(2,:);      U(1,:) = -U(2,:);      V(1,:) = V(2,:);
       H(n+2,:) = H(n+1,:);  U(n+2,:) = -U(n+1,:);  V(n+2,:) = V(n+1,:);

       % First half step

       % x direction
       i = 1:n+1;
       j = 1:n;

       % height
       Hx(i,j) = (H(i+1,j+1)+H(i,j+1))/2 - dt/(2*dx)*(U(i+1,j+1)-U(i,j+1));

       % x momentum
       Ux(i,j) = (U(i+1,j+1)+U(i,j+1))/2 -  ...
                 dt/(2*dx)*((U(i+1,j+1).^2./H(i+1,j+1) + g/2*H(i+1,j+1).^2) - ...
                            (U(i,j+1).^2./H(i,j+1) + g/2*H(i,j+1).^2));

       % y momentum
       Vx(i,j) = (V(i+1,j+1)+V(i,j+1))/2 - ...
                 dt/(2*dx)*((U(i+1,j+1).*V(i+1,j+1)./H(i+1,j+1)) - ...
                            (U(i,j+1).*V(i,j+1)./H(i,j+1)));

       % y direction
```

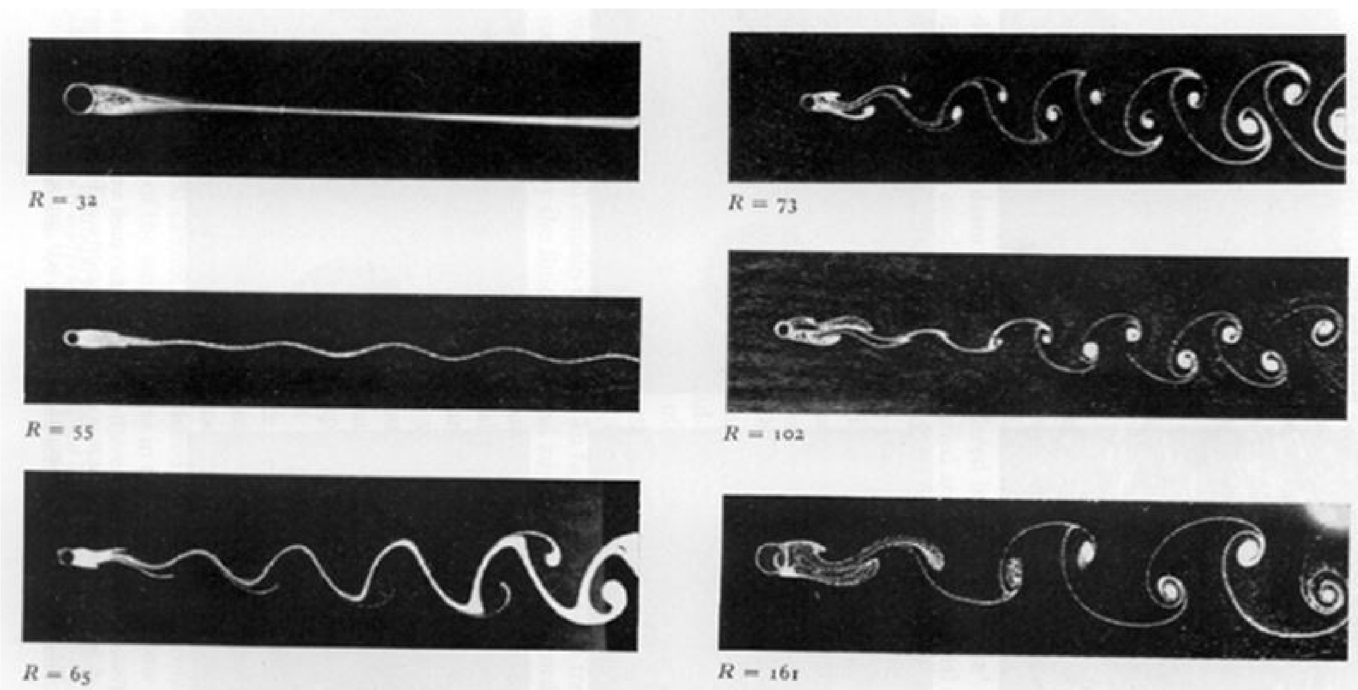
```
        i = 1:n;
        j = 1:n+1;

        % height
        Hy(i,j) = (H(i+1,j+1)+H(i+1,j))/2 - dt/(2*dy)*(V(i+1,j+1)-V(i+1,j));

        % x momentum
        Uy(i,j) = (U(i+1,j+1)+U(i+1,j))/2 - ...
                  dt/(2*dy)*((V(i+1,j+1).*U(i+1,j+1)./H(i+1,j+1)) - ...
                             (V(i+1,j).*U(i+1,j)./H(i+1,j)));
        % y momentum
        Vy(i,j) = (V(i+1,j+1)+V(i+1,j))/2 - ...
                  dt/(2*dy)*((V(i+1,j+1).^2./H(i+1,j+1) + g/2*H(i+1,j+1).^2) - ...
                             (V(i+1,j).^2./H(i+1,j) + g/2*H(i+1,j).^2));

        % Second half step
        i = 2:n+1;
        j = 2:n+1;

        % height
        H(i,j) = H(i,j) - (dt/dx)*(Ux(i,j-1)-Ux(i-1,j-1)) - ...
                          (dt/dy)*(Vy(i-1,j)-Vy(i-1,j-1));
        % x momentum
        U(i,j) = U(i,j) - (dt/dx)*((Ux(i,j-1).^2./Hx(i,j-1) + g/2*Hx(i,j-1).^2) -
...
                          (Ux(i-1,j-1).^2./Hx(i-1,j-1) + g/2*Hx(i-1,j-1).^2)) ...
                        - (dt/dy)*((Vy(i-1,j).*Uy(i-1,j)./Hy(i-1,j)) - ...
                          (Vy(i-1,j-1).*Uy(i-1,j-1)./Hy(i-1,j-1)));
        % y momentum
        V(i,j) = V(i,j) - (dt/dx)*((Ux(i,j-1).*Vx(i,j-1)./Hx(i,j-1)) - ...
                          (Ux(i-1,j-1).*Vx(i-1,j-1)./Hx(i-1,j-1))) ...
                        - (dt/dy)*((Vy(i-1,j).^2./Hy(i-1,j) + g/2*Hy(i-1,j).^2) -
...
                          (Vy(i-1,j-1).^2./Hy(i-1,j-1) + g/2*Hy(i-1,j-1).^2));

        % Update plot
        if mod(nstep,nplotstep) == 0
           C = abs(U(i,j)) + abs(V(i,j));  % Color shows momemtum
           t = nstep*dt;
           tv = norm(C,'fro');
           set(surfplot,'zdata',H(i,j),'cdata',C);
           set(top,'string',sprintf('t = %6.2f,  tv = %6.2f',t,tv))
           drawnow
        end

end
```



关于“流函数”

2. 不可压缩流动

2020年5月5日，广东[虎门大桥发生异常抖动 (https://new.qq.com/omn/20200506/20200506A0TDUS00.html)](https://new.qq.com/omn/20200506/20200506A0TDUS00.html)。事实上，大桥“异常”抖动或晃动的状况时有发生——这是流体力学中重要的现象“[卡门涡街 (https://zhuanlan.zhihu.com/p/129273764)](https://zhuanlan.zhihu.com/p/129273764)”。

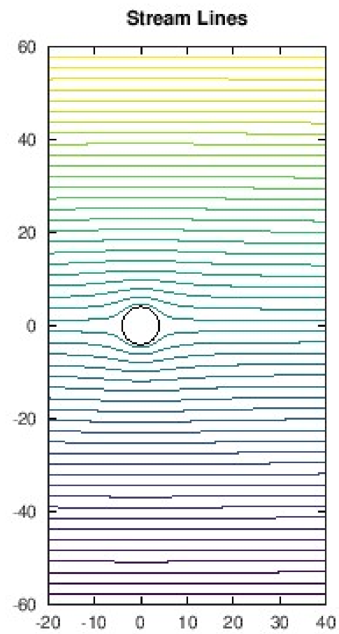


类似的现象在历史上发生过多次：如，2010年，俄国南部伏尔加河的大桥就曾发生波浪状的“离奇”摇晃，当时好几辆正行驶在桥上的车子也跟着不断摇摆;美国的塔科马海峡吊桥（Tacoma Narrow Bridge）事件，这次事件的过程有完整拍摄成影片。塔科马海峡吊桥吊装完成后，只要有 4英里/小时的“小风”吹过来，大桥主体就发生轻微的上下欺负。在建造过程中工人就已经注意到这样的现象。遗憾的是，最终在仅通车4个月后，大桥主题轰然倒塌。

塔科玛海峡大桥的毁坏， 是由周期性旋涡共振引起。设计人想建造一个较便宜结构塔科玛海峡大桥的毁坏， 是由周期性旋涡共振引起。“卡门涡街”引起的桥梁共振。 在必定的风速规模内，穿过大桥气流会周期性地产 生两串平行的反向旋涡，连续性会对被绕 的桥梁产生周期性浸染力，这 种浸染力和大桥震动的频率接近时，就会 产生共振。越强，大桥摆动扭曲 的幅度便会越大。

In [5]:

```
U_i = 20;               % Ambient velocity
a = 4;                  % cylinder radius

c = -a*5;               % starting coordinate (x)
b = a*10;               % ending coordinate (x)
d = -60;                % starting coordinate (y)
e = 60;                 % ending coordinate (y)

n = a*50;               % number of intervals (step size in grid)

[x,y] = meshgrid([c:(b-c)/n:b],[d:(e-d)/n:e]');

for i = 1:length(x)
    for k = 1:length(x);
        f = sqrt(x(i,k).^2 + y(i,k).^2);
        if f < a
            x(i,k) = 0;
            y(i,k) = 0;
        end
    end
end

% Definition of polar variables
r = sqrt(x.^2+y.^2);
theta = atan2(y,x);

%% Creation of Streamline function
z = U_i.*r.*(1-a^2./r.^2).*sin(theta);%- G*log(r)/(2*pi);

%% Creation of Figure

m = 100;
s = ones(1,m+1)*a;
t = [0:2*pi/m:2*pi];

%% Streamline plot

contour(x,y,z,50);
hold on
polar(t,s,'-k');
axis equal;
title('Stream Lines');
grid off
```



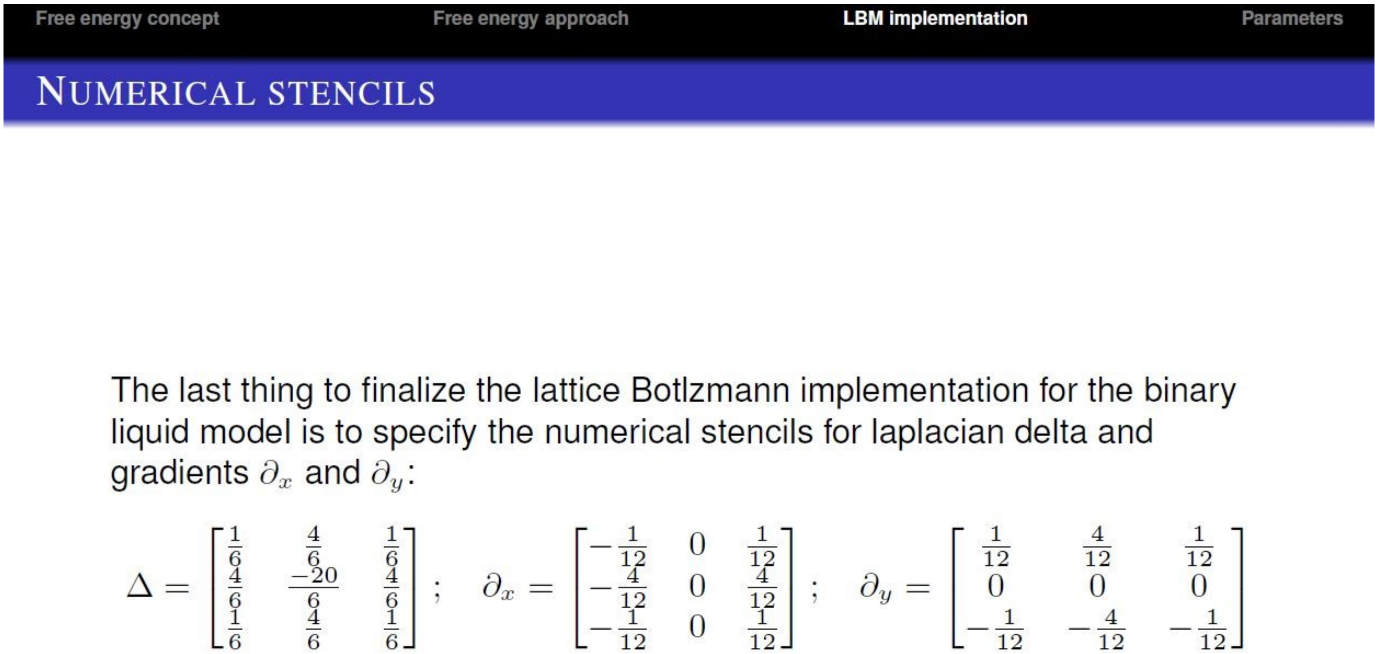
Remark

- 不幸的是，圆柱绕流（flow past a cylinder）的matlab程序并不简短。网上很容易找到另外一个Benchmark流体问题：driven cavity的matlab代码。
- 利用流体力学专门的软件演示，可以参考bilibili：[卡门涡街的数值计算 (https://www.bilibili.com/video/av925385267/)](https://www.bilibili.com/video/av925385267/) 这里我们给出一个基于Navier-Stokes方程计算的模拟
- **读一读** 一个更精致的benchmrk算例[flow past a cylinder (https://www.grc.nasa.gov/WWW/wind/valid/lamcyl/Study1_files/Study1.html)](https://www.grc.nasa.gov/WWW/wind/valid/lamcyl/Study1_files/Study1.html)是科学计算研究的一个入门问题。这里是一个三维算例 (https://pdf.sciencedirectassets.com/272600/1-s2.0-S0889974609X0006X/1-s2.0-S0889974609000218/main.pdf?X-Amz-Security-Token=IQoJb3JpZ2luX2VjEFUaCXVzLWVhc3QtMSJHMEUCIQCAgyGhHz6bGNVKM1eYRmPmJwc%2FK56P91h3JyDp9c%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FARADGgwwNTkwMDM1NDY4NjUiDFgQDnuFQWrErrtJNCqRA74bbQOMOLSkoryqif2RxoYQuvsEGv659IOozc1U%2B%2FHyB39I7aX8WndxjaJcil8oJQA%3D%3D&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20200708T140901Z&X-Amz-SignedHeaders=host&X-Amz-Expires=300&X-Amz-Credential=ASIAQ3PHCVTYUTBECM6C%2F20200708%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Signature=f63ace540084c1ea049080bb03caf57f69ba802748a2f069bec00f65c68ff233&hash=4e09293de122237806f0a7bafc799acf49db80b89ecc16e8dd1e0caa8bf9123b&host=68042c943591013ac2b2430a89b270f6af2c76d8dfd086a07176afe7c76c2c61&pii=S0889974609000218&tid=spdf-ed862dd2-4c0b-4f66-a6af-b6d08411826a&sid=4eac6e403f13844c824b6496694732fbe898gxrqa&type=client)

```
In [ ]:
```

3. Lattice Boltzmann 模拟

LBM是流体力学数值模拟的另一个有效方法，对于介观问题的数值模拟尤为有效。LBM的优点在于直接从物理原理进行建模，并不需要太多关于微分方程的知识。下面这片PPT是一位从事LBM研究学者的报告中截取的一片，它说明了微分算子在离散状态下等同于作用一个3x3矩阵：



这里，我们只展示一个简单的气泡上升的模拟算例，这个实现是2008级李定华同学所做的[毕业论文 (LBM/bylw-li.pdf)](LBM/bylw-li.pdf)中改写的一个两相流格子Boltzmann方法。改写成模拟其他流体现象也是没有本质困难的，感兴趣的读者可以自行寻找相关材料。

In [ ]:

```
%% periodic flow version
clear all;
clc;

% Macroscopic density and velocities
NX=16;
NY=16;
NPOP=9;
NSTEPS=10000;

rho0=1;  umax=0.001;

rho=ones(NX,NY); ux=zeros(NX,NY); uy=zeros(NX,NY);
uxinit=zeros(NX,NY); uyinit=zeros(NX,NY);
[xx,yy] = meshgrid((1:NX)/NX, (1:NY)/NY);

feq=zeros(NPOP); f1=zeros(NPOP,NX,NY); f2=zeros(NPOP,NX,NY);

weights = [4/9  1/9  1/9  1/9  1/9  1/36 1/36 1/36 1/36];
cx      = [0    1    0    -1    0    1    -1   -1    1  ];
cy      = [0    0    1     0   -1    1     1   -1   -1  ];
omega   = 1.0;

%% initialize rho, u and v
for y=1:NY
    for x=1:NX
        rho(x,y)=rho0+3*0.25*umax^2*(cos(4*pi*(x-1)/NX)-cos(4*pi*(y-1)/NY));
        ux(x,y)=umax*sin(2*pi*(x-1)/NX)*sin(2*pi*(y-1)/NY);
        uy(x,y)=umax*cos(2*pi*(x-1)/NX)*cos(2*pi*(y-1)/NY);
        vx=ux(x,y);
        vy=uy(x,y);

        for k=1:NPOP
            feq(k)=weights(k)*rho(x,y)*(1+3*(vx*cx(k)+vy*cy(k)) ...
               + 9/2*((cx(k)*cx(k)-1/3)*vx*vx+2*cx(k)*cy(k)*vx*vy+(cy(k)*cy(k)-1/3)
*vy*vy));
            f1(k,x,y)=feq(k);
            f2(k,x,y)=feq(k);
        end

    end
end

for counter=1:NSTEPS  %% evolving with LBM

    for y=1:NY
        for x=1:NX
            dense=0;  vx=0;  vy=0;
            for k=1:NPOP
                dense=dense+f1(k,x,y);
                vx=vx+cx(k)*f1(k,x,y);
                vy=vy+cy(k)*f1(k,x,y);
            end

            rho(x,y)=dense;
            vx = vx/dense;  vy = vy/dense;
            ux(x,y)=vx; uy(x,y)=vy;

            for k=1:NPOP
                feq(k)=weights(k)*rho(x,y)*(1+3*(vx*cx(k)+vy*cy(k)) ...
                   +9/2*((cx(k)*cx(k)-1/3)*vx*vx+2*cx(k)*cy(k)*vx*vy+(cy(k)*cy(k)-1
/3)*vy*vy));
```

```
                newx=1+mod(x-1+cx(k)+NX,NX);
                newy=1+mod(y-1+cy(k)+NY,NY);

                f1(k,x,y)=f1(k,x,y)*(1-omega)+feq(k)*omega;
                f2(k,newx,newy)=f1(k,x,y);
            end

        end
    end

    track(counter)=ux(NX/4+1,NY/4+1);
    decay(counter)=umax*exp(-1/3*(1/omega-0.5)*counter*2*(2*pi/NX)^2);
    f1=f2;

    if(mod(counter, 10)==0)
        quiver(ux,uy); axis tight; % contour(rho);
        title(sprintf('Round %d ...\n', counter)); drawnow; pause(0.05);
    end
end

%% L2 error:
sum=0
for y=1:NX
    for x=1:NX
        ux_value=umax*sin(2*pi*(x-1)/NX)*sin(2*pi*(y-1)/NY);
        uy_value=umax*cos(2*pi*(x-1)/NX)*cos(2*pi*(y-1)/NY);

        sum=sum+(ux(x,y)-ux_value)^2+(uy(x,y)-uy_value)^2;
    end
end
error=sqrt(sum/(NX*NY))
```

请注意，jupyter对于图像输出并不是太友好，建议在octave的集成开发环境下运行该脚本。

练一练：把上述模拟例子中输出的不同时刻的状态数据，或保存为png图像，并最终制作成gif动画以便于展示

想一想：对于长时间数值模拟，如何有效地保存中间状态？ 如何达到计算时间与存储空间的平衡？ 如何能更高效地调参？

In [ ]: