```
%%Modeling the Spatial Dynamics of Life in the Clouds of Venus
%%

%Debha Amatya
%Venus Life
%Team Stanford-Brown, iGEM 2012

%This is a spatial model of bacteria in the clouds of Venus. The
bacteria
%are initially inserted in the 50 to 60 KM cloud layer range, then
%move with eddy diffusion and wind advection, and are examined for
%persistence in the 50-60km habitable zone. The equations are based on
the
%species transport, advective diffusion partial differential equations.

%%
clc; clear all; close all;

maxcycle = 1100; % Defining the number of cycles for a simulation
%%
%%Defining grid in the horizontal direction
%%
xmin=40000.0; xmax=70000.0; xnum=150;
xstp=(xmax-xmin)/(xnum-1);
ymin=40000.0; ymax=70000.0; ynum=150;
ystp=(ymax-ymin)/(ynum-1);

%Defining Material Properties
kappa = 10000; %Eddy Diffusion constant, m^2/s, scaled to 1e4 larger
than actual for simulation purposes
%%
%%Defining initial bacterial concentration structure
%%
bmin=0;
bmax=1e6;
for xn = 1:xnum
    for yn = 1:ynum
        BC0(xn,yn) = bmin; % Background bacterial concentration,
        if ((xn-1)/(xnum-1) > 0.50 && (xn-1)/(xnum-1) < 0.51 && (yn-1)/
(ynum-1) > 0.40 && (yn-1)/(ynum-1) < 0.42)
            BC0(xn,yn)=bmax; % Rectangular insertion point near the
middle
        end
    end
end

BCinit=BC0; % Store the initial conditions
%%
%%Defining rotation velocity structure
%%
timestep =1e0; %Explicit timestep of 1 unit in simulation time

%We have to create now a velocity field. Letís try with a rotation
%around the center of the computational domain:

p = 1.1; %density in g/cm^3
d = 1e-4; %diameter of bacteria in centimeters
```

```
vGrav = -1e4*3.2*9.8*p*(d^2);


% Defining rotation velocity structure
%-30:0.6:30
[VX,VY] = meshgrid(-125:3.35/2:125, -125:3.35/2:125);
for xn = 1:xnum
    for yn = 1:ynum %add Gaussian noise
        VX(xn,yn) = 10*randn(1) + VX(xn,yn);
        VY(xn,yn) = vGrav + 10*randn(1) + VY(xn,yn);
    end
end
VY = -VY;

time = 0;
%%
%%Solving temperature equation by explicit method
%%
for cycle = 0:maxcycle
    for xn = 1:xnum
        for yn = 1:ynum
            % Boundary nodes:
            if (xn==1 || xn==xnum || yn==1 || yn==ynum)
                BC1(xn,yn)=bmin; % B=const
            else

                %Add the diffusion component
                BC1(xn,yn)= BC0(xn,yn)+ timestep*kappa*( BC0(xn-1,yn)-
2.0*BC0(xn,yn)+BC0(xn+1,yn) )/(xstp*xstp); %x contribution
                BC1(xn,yn)= BC1(xn,yn)+ timestep*kappa*( BC0(xn,yn-1)-
2.0*BC0(xn,yn)+BC0(xn,yn+1) )/(ystp*ystp); %y contribution

                %Add the advection component
                BC1(xn,yn)=BC1(xn,yn)-
0.5*VX(xn,yn).*timestep.*(BC0(xn,yn)-BC0(xn-1,yn))./xstp-
0.5*VX(xn,yn).*timestep.*(BC0(xn+1,yn)-BC0(xn,yn))./xstp;
                BC1(xn,yn)=BC1(xn,yn)-
0.5*VY(xn,yn).*timestep.*(BC0(xn,yn)-BC0(xn,yn-1))./ystp-
0.5*VY(xn,yn).*timestep.*(BC0(xn,yn+1)-BC0(xn,yn))./ystp;

            end
        end

    end
    BC0 = BC1;% Reloading solutions to BC0

    % Creating vectors for x and y axes
    x = xmin:xstp:xmax;
    y = ymin:ystp:ymax;
    x=x/1000.;% renormalizes m in km, only for plotting
    y=y/1000.;% renormalizes m in km, only for plotting

    %Visualizing results
    % Ploting bacterial concentration as a surface
%     if mod(cycle,10) == 0 %capture every 10 frames
%         figure(1);
%         BCrot = rot90(BC0);
```

```
%         surf(x,y,BCrot);
%         %caxis([1 10000])
%         view (0,90);
%         title('Bacterial Persistance in the Habitable
Zone','FontSize', 18,'FontWeight', 'bold');
%         xlabel(sprintf('%f days', time),'FontSize',15,'FontWeight',
'bold');
%         ylabel('Altitude (KM)','FontSize',15,'FontWeight', 'bold');
%         %Capture the title and axes also
%         winsize = get(figure(1),'Position');
%         winsize(1:2) = [0 0];
%         %write the gif file of the simulation
%         frame = getframe(figure(1),winsize);
%         im = frame2im(frame);
%         [imind,cm] = rgb2ind(im,256);
%         if cycle == 0
%             imwrite(imind,cm,'AdDifEx.gif','gif', 'Loopcount',inf);
%         else
%
imwrite(imind,cm,'AdDifEx.gif','gif','WriteMode','append');
%         end
%
%     end
    time = time + (10000*timestep)/(3600*24); %Keep track of days, and
convert to real time
end
```