```
% ### EXcoffee.m ###    11.16.14    {C. Bergevin}
% [modified version of EXrandomWalk2D.m motivated by the problem shown
in
% Giordano (1997) Fig.7.18ff]
% **NOTE**: There is a minor bug in this version such that it is
possible for
% some 'cream' to leave the 'cup' (despite the specifed boundary
conditions)
clear;
% -------------
N= 10;      % one plus sqrt of total # of (independent) walkers (each
starts at unique x,y point about origin)
M= 300;      % Total # of steps for each walker
method= 2;  % see comments above
BND= 10;    % bounding limits for initial grid of walkers at t=0
axisLim= 100;   % size of coffee cup
diffC= 1;   % diffusion const. (i.e., scaling factor for step size)
framerate= 1/30;    % pause length [s] for animation
Sgrid= 8;           % grid spacing for entropy calculation
% -------------
% +++
space= (2*BND)/N;
[X,Y]= meshgrid(-BND:space:BND,-BND:space:BND);
E= size(X,1); % # of elements
SgridX= linspace(-axisLim,axisLim,Sgrid);   % set grid bounds for
entropy calc.
SgridY= linspace(-axisLim,axisLim,Sgrid);

figure(1); clf; grid on; xlabel('x-postion'); ylabel('y-postion');
% visualize before onset?
if (1==1), plot(X,Y,'ko','MarkerSize',5); axis([-axisLim axisLim
-axisLim axisLim]);   end
% +++
% To do
% - apply boundary condition (i.e., ensure no steps past walls)
% - fix entropy calc. (i.e., if prob.=0??)
for r= 1:M

    if method==1
        % random L/R and U/D step with equal probability
        tempX= rand(E,E);   tempY= rand(E,E);  % determine random
vals.
        temp2X= tempX<0.5;  temp2Y= tempY<0.5;  % determine L vs R and
U vs D
        X(temp2X)= X(temp2X)+1; X(~temp2X)= X(~temp2X)-1;
        Y(temp2Y)= Y(temp2Y)+1; Y(~temp2Y)= Y(~temp2Y)-1;
    else
        % sample step from normal distribution
        stepX= randn(E,E); stepY= randn(E,E);
        X= X+ diffC*stepX;  Y= Y+ diffC*stepY;
```

```
        % verify step is not past walls; if so, bounce back in
opposite direction
        [aa,bb]= find(abs(X)>axisLim); [cc,dd]= find(abs(Y)>axisLim);

        % +++ --> correct for points that have moved past the walls
        % not quite right, but kinda works
        X(aa,bb)= X(aa,bb)-2*diffC*stepX(aa,bb); Y(cc,dd)=
Y(cc,dd)-2*diffC*stepY(cc,dd);

        % more right (I think), but doesn't work
        %X(aa,bb)= sign(X(aa,bb))*2*axisLim-X(aa,bb); Y(cc,dd)=
Y(cc,dd)-2*diffC*stepY(cc,dd);

        % uncomment to allow for flagging when 'cream' leaves the cup
        if(max(abs(X(:))>axisLim)), return;  end
    end
    % visualize
    figure(1)
    plot(X,Y,'ko','MarkerSize',5); axis([-axisLim axisLim -axisLim
axisLim]); pause(framerate);
    % do binning to determine 'probability' distribution
    histS= hist2(X(:),Y(:),SgridX,SgridY)/E^2; % use external function
hist2.m; and normalize to a probability
    histS= histS(:);    % convert to a single column vector
    zeroI= ~histS==0; % need to filter out states with zero elements
so to avoid computational error (since 0*log(0)= NaN)
    S(r)= -sum(histS(zeroI).*log(histS(zeroI))); % calculate entropy
(S)
end;

figure(2)
plot(S,'LineWidth',2); hold on; grid on;
xlabel('time step'); ylabel('entropy');
```