% updates the position

function [x,y,tForceX,tForceY,rnX,rnY] = TheUpdater(N,L,dt,vx,vy,x,y,T)

% N = 23; % Number of particles
% L = 10; % Size of system
% dt = .02; % time step
% T = 5;

tStepMax = T/dt; % total steps in time

% Initialize Variables required by function
rCut = 4;
fCut = 100; % cap for force

TForce = zeros(1,N);
tForceX = zeros(1,N);
tForceY = zeros(1,N);

x = padarray(x,[tStepMax-2,0], 'post'); % pre-allocate matrices to save computation
y = padarray(y,[tStepMax-2,0], 'post');

Error using TheUpdater (line 10)
Not enough input arguments.

Main Loops

for t = 2:1:tStepMax-1 % loop for all time
    for i = 1:N % loop for particle i
        for j = 1:N % for each particle i, loop through all particles i
            if i ~= j % i can't interact with itself

                % X calculations

                if abs(x(t,i)-x(t,j))> L - abs(x(t,i)-x(t,j)) % check for shortest
                    rnX = L - (x(t,i)-x(t,j)); % let rn be the smaller one
                else
                    rnX = (x(t,i)-x(t,j));
                end

                % Y Calculations

                if abs(y(t,i)-y(t,j))> L - abs(y(t,i)-y(t,j)) % check for shortest
                    rnY = L - (y(t,i)-y(t,j)); % let rn be the smaller one
                else
                    rnY = (y(t,i)-y(t,j));
                end

                rn = sqrt(rnX^2+rnY^2); % define rn as total distance between i and

                % Force calculations

                if sqrt(rn)<rCut % if the particle is close enough to interact

result = 

end

end

end
f = .24*((2/rn^13)-(1/rn^7)); %force between i and j calculated by Lennard-Jones potential

% f = 24*((2/rnY^13)-(1/rnY^7)); %force between i and j calculated by Lennard-Jones potential

end

% Calculate x and y components of force

% tForceY(i) = tForce(i)+f; %Cumulate the total force on i

% tForce(i) = tForce(i)+f; %Cumulate the total force on i

end

for i=1:N %loop again through i to calculate all position

% Calculate the new x and y coordinates

x(t+1,i) = 2*x(t,i) - x(t-1,i) + tForceX(i)*dt^2; %calculate new x coordinate for particle i from force and two past positions
y(t+1,i) = 2*y(t,i) - y(t-1,i) + tForceY(i)*dt^2; %calculate new y coordinate for particle i from force and two past positions

% Apply periodic boundary conditions

if x(t+1,i) > L
    x(t+1,i) = x(t+1,i) - L; %if a particle leaves the right side of the box, it returns to the left while keeping its velocity
elseif x(t+1,i) < 0
    x(t+1,i) = x(t+1,i) + L;

end

if y(t+1,i) > L
    y(t+1,i) = y(t+1,i) - L; %if a particle leaves the top side of the box, it returns to the left while keeping its velocity
elseif y(t+1,i) < 0
    y(t+1,i) = y(t+1,i) + L;

end

end

end

We should make sure to count each interaction just once

Published with MATLAB® R2014a