Matlab Code

1. Neuron Approximation Generation

‘key_cylindrical’

```
%  this acts as a key to keep track of neuronal and glial information by
%  index value

% SCALE:  25 microm = 1 unit

%_____________________NEURONS_____________________________________

% 1 = intial time of firing
% 2 = phenotype
   % 1 = L2py
   % 2 = L3py
   % 3 = L4sp
   % 4 = L4ss
   % 5 = L4py
   % 6 = L5st
   % 7 = L5tt
   % 8 = L6cc
   % 9 = L6ct
   % 10 = L6invpy

% # variables +

% PHENOTYPE DIMENSIONS

% 1 = neuron catalog number
% 2 = start (1 = start)
% 3 = x initial
% 4 = y initial
% 5 = z initial
% 6 = soma radius
% 7 = 0
% 8 = soma height
% 9 = dendrite radius
% 10 = 0;
% 11 = dendrite height (+)
% 12 = dendrite height (-)
% 13 = axon radius
% 14 = 0
% 15 = axon height (+)
% 16 = axon height (-)

% DISTANCES

% 17 - (num_glia + 16) = neurons -> glia
% (num_glia + 17) - (num_glia + num_neurons + 16) = neuron -> neuron
% (num_glia + num_neurons + 17)
%  - (num_glia + num_neurons + num_exglia + 16) = neuron -> exglia

%_____________________GLIA_____________________________________
```

```
% 1 = x initial
% 2 = y initial
% 3 = z initial

% # glia variables +

% 4 - (size_GliaVariables + num_neurons + 3) = glia -> neuron

%_____________________________EX-GLIA_____________________________________

% 1 = x initial
% 2 = y initial
% 3 = z initial

% # exglia variables +

% 4 - (size_GliaVariables + num_neurons + 3) = exglia -> neuron
```

‘set_up_cylindrical’

```
global num_neurons num_glia num_exglia neuron glial exglial size_NeuronVariables ...
      size_GliaVariables size_ExGliaVariables neuronPrP

%This script establised the semi-randomly choosen number of neurons and
%glia. The ratio of neurons to glia is approx 2:1

A = 2;
B = 2;
C = 0.38;
D = 0.80;

X = 100;
Y = 100;
Z = 80;
space = zeros(X,Y,Z);

%axis are (x,y,z)
%space is a visual guide that places the initial coordinates of cells
%will eventually be expanded to a 3D volume proportional to the number of
%cells generated

num_neurons = randi([A,B],1,1);
num_glia = round(((C-D).*rand(1,1)+D)*num_neurons);
num_exglia = round(((C-D).*rand(1,1)+D)*num_neurons);

%this matrix contains the coordinates of specific concentrations of PrP
%that are subject to change as time evolves
neuronPrP = zeros(12,5,num_neurons);

%neuron and glia and invading glia 3D arrays composed of time, individual
%concentrations, positions relevant to other cells, and for each neurons
neuron = zeros(1,(16 + num_neurons + num_glia + num_exglia...
   + size_NeuronVariables + (3*(num_neurons*6))),num_neurons);
glial = zeros(1,(3 + num_neurons + size_GliaVariables),num_glia);
exglial = zeros(1,(3 + num_neurons + size_ExGliaVariables),num_exglia);
```

```
%_____________________________________________________________________%

%cells are placed randomly and assigned coordinates so that they dont
%overlap

for i = 1:num_neurons;
   neuron(1,(size_NeuronVariables+1),i) = i;
   x = randi([(900/25),X-(900/25)],1,1);
   y = randi([(900/25),Y-(900/25)],1,1);
   d = randi([(150/25),Z-(150/25)],1,1);
   if space(x,y,d) == 0 && space(x+1,y,d) == 0 && space(x-1,y,d) == 0 ...
         && space(x,y+1,d) == 0 && space(x,y-1,d) == 0 ...
         && space(x,y,d+1) == 0 && space(x,y,d-1) == 0;
      neuron(1,(size_NeuronVariables+3),i) = x;
      neuron(1,(size_NeuronVariables+4),i) = y;
      neuron(1,(size_NeuronVariables+5),i) = d;
      space(x,y,d) = i;
      space(x+1,y,d) = i;
      space(x-1,y,d) = i;
      space(x,y+1,d) = i;
      space(x,y-1,d) = i;
      space(x,y,d+1) = i;
      space(x,y,d-1) = i;
   elseif space(x,y,d) ~= 0;
         while space(x,y,d) ~= 0 && space(x+1,y,d) ~= 0 && space(x-1,y,d) ~= 0 ...
          && space(x,y+1,d) ~= 0 && space(x,y-1,d) ~= 0 ...
          && space(x,y,d+1) ~= 0 && space(x,y,d-1) ~= 0;
              x = randi([(900/25),X-(900/25)],1,1);
              y = randi([(900/25),Y-(900/25)],1,1);
              d = randi([(150/25),Z-(150/25)],1,1);
              neuron(1,(size_NeuronVariables+3),i) = x;
              neuron(1,(size_NeuronVariables+4),i) = y;
              neuron(1,(size_NeuronVariables+5),i) = d;
         end
      space(x,y,d) = i;
      space(x+1,y,d) = i;
      space(x-1,y,d) = i;
      space(x,y+1,d) = i;
      space(x,y-1,d) = i;
      space(x,y,d+1) = i;
      space(x,y,d-1) = i;
   end
end

for j = 1:num_glia;
   w = randi([(100/25),X-(100/25)],1,1);
   z = randi([(100/25),Y-(100/25)],1,1);
   b = randi([(100/25),Z-(100/25)],1,1);
   if space(w,z,b) == 0;
      glial(1,(size_GliaVariables+1),j) = w;
      glial(1,(size_GliaVariables+2),j) = z;
      glial(1,(size_GliaVariables+3),j) = b;
      space(w,z,b) = j;
   elseif space(w,z,d) ~= 0;
         while space(w,z,d) ~= 0;
              w = randi([(100/25),X-(100/25)],1,1);
```

```
            z = randi([(100/25),Y-(100/25)],1,1);
            b = randi([(100/25),Z-(100/25)],1,1);
            glial(1,(size_GliaVariables+1),j) = w;
            glial(1,(size_GliaVariables+2),j) = z;
            glial(1,(size_GliaVariables+3),j) = b;
        end
        space(w,z,d) = j;
    end
end

for k = 1:num_exglia;
    e = randi([(100/25),(200/25)],1,1);
    f = randi([(100/25),(200/25)],1,1);
    g = randi([(100/25),(200/25)],1,1);
    if space(e,f,g) == 0;
      exglial(1,(size_ExGliaVariables+1),k) = e;
      exglial(1,(size_ExGliaVariables+2),k) = f;
      exglial(1,(size_ExGliaVariables+3),k) = g;
      space(e,f,g) = k;
    elseif space(e,f,g) ~= 0;
        while space(e,f,g) ~= 0;
            e = randi([(100/25),(200/25)],1,1);
            f = randi([(100/25),(200/25)],1,1);
            g = randi([(100/25),(200/25)],1,1);
            exglial(1,(size_ExGliaVariables+1),k) = e;
            exglial(1,(size_ExGliaVariables+2),k) = f;
            exglial(1,(size_ExGliaVariables+3),k) = g;
        end
        space(e,f,g) = k;
    end
end

%_____________________________start_____________________________________%

%one neuron is randomly selected to be the intially infected
start = randi([1,num_neurons],1,1);
neuron(1,(size_NeuronVariables+2),start) = 1;

%_____________________________________________________________________%
%Assignment of radii representing somas(spheres), dendrites and axon (cylinders)
%Establishes approximate radii for individual neurons depending on its
%previously determined depth

for l = 1:num_neurons;
  if neuron(:,size_NeuronVariables + 5,l) >= 60;
    %neuron is an approximate L2 pyramidal (units in microm)
    neuron(:,2,l) = 1;
    neuron(:,size_NeuronVariables + 6,l) = ((12.9-16.5).*rand(1,1)+16.5)/25;        %soma r
    neuron(:,size_NeuronVariables + 7,l) = 0;
    neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;        %soma z
    neuron(:,size_NeuronVariables + 9,l) = ((468-642).*rand(1,1)+642)/25;           %dendrite r
    neuron(:,size_NeuronVariables + 10,l) = 0;
    max_d = ((185-325).*rand(1,1)+325)/25;
     if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
       neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+
```

```
  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1500-1600).*rand(1,1)+1600)/50;       %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1500-1600).*rand(1,1)+1600)/25;
  if neuron(:,size_NeuronVariables + 5) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
  end


elseif (60 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 52);
  %neuron is an approximate L3 pyramidal (units in microm)
  neuron(:,2,l) = 2;
  neuron(:,size_NeuronVariables + 6,l) = ((12.9-16.5).*rand(1,1)+16.5)/25;       %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;       %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((316-392).*rand(1,1)+392)/25;          %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((315-579).*rand(1,1)+579)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1600-1700).*rand(1,1)+1700)/50;       %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1700-1800).*rand(1,1)+1800)/25;
  if neuron(:,size_NeuronVariables + 5) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
```

```
  end
end

cell4= 0;
%1 = L4sp, 2 = L4ss, 3 = L4py

if ((52 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 48)) || ...
      (((44 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 36)));
  cell4 = randi([1,2],1,1);

elseif (48 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l)) >= 44;
  cell4 = randi([1,3],1,1);

elseif (36 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l)) >= 32;
   cell4 = 1;

end

if cell4 == 1;
  %neuron is an approximate L4sp pyramidal (units in microm)
  neuron(:,2,l) = 3;
  neuron(:,size_NeuronVariables + 6,l) = ((12.9-16.5).*rand(1,1)+16.5)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;        %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((237-371).*rand(1,1)+371)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((170-468).*rand(1,1)+468)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1200-1300).*rand(1,1)+1300)/50;        %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1500-1600).*rand(1,1)+1600)/25;
  if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
  end
```

```
elseif cell4 == 2;
  %neuron is an approximate L4ss pyramidal (units in microm)
  neuron(:,2,l) = 4;
  neuron(:,size_NeuronVariables + 6,l) = ((12.9-16.5).*rand(1,1)+16.5)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;        %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((250-275).*rand(1,1)+275)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((280-310).*rand(1,1)+310)/25;
   if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((700-800).*rand(1,1)+800)/50;          %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1400-1500).*rand(1,1)+1500)/25;
  if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                  %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                  %axon z-
  end

 elseif cell4 == 3;
  %neuron is an approximate L4py pyramidal (units in microm)
  neuron(:,2,l) = 5;
  neuron(:,size_NeuronVariables + 6,l) = ((12.9-16.5).*rand(1,1)+16.5)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;        %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((287-359).*rand(1,1)+359)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((453-531).*rand(1,1)+531)/25;
   if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end
```

```
  neuron(:,size_NeuronVariables + 13,l) = ((1200-1300).*rand(1,1)+1300)/50;       %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1600-1700).*rand(1,1)+1700)/25;
  if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                  %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                  %axon z-
  end

end

cell5= 0;
%1 = L5st, 2 = L5tt

if (32 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 28);
  cell5 = 1;

elseif (28 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 20);
  cell5 = randi([1,2],1,1);

elseif (20 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 16);
   cell5 = 2;

end

if cell5 == 1;
  %neuron is an approximate L5st pyramidal (units in microm)
  neuron(:,2,l) = 6;
  neuron(:,size_NeuronVariables + 6,l) = ((15.4-23.2).*rand(1,1)+23.2)/25;       %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((16.8-20.6).*rand(1,1)+16.8)/25;       %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((379-499).*rand(1,1)+499)/25;          %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((950-1154).*rand(1,1)+1154)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
    neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
    neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
    neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);      %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
    neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1400-1500).*rand(1,1)+1500)/50;       %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1700-1800).*rand(1,1)+1800)/25;
```

```
  if (neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z)) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif (neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z)) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                  %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                  %axon z-
  end

elseif cell5 == 2;
  %neuron is an approximate L5tt pyramidal (units in microm)
  neuron(:,2,l) = 7;
  neuron(:,size_NeuronVariables + 6,l) = ((14.8-21.4).*rand(1,1)+21.4)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((19.6-29.4).*rand(1,1)+29.4)/25;        %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((449-707).*rand(1,1)+707)/25;          %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((1050-1210).*rand(1,1)+1210)/25;
   if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((900-1000).*rand(1,1)+1000)/50;        %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1700-1800).*rand(1,1)+1800)/25;
  if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                  %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                  %axon z-
  end

end

cell6= 0;
%1 = L6cc, 2 = L6ct, 3 = L6inv py

if (16 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 12);
  L6 = [1 3];
  pick_L6 = randi([1,2],1,1);
  cell6 = L6(1,(pick_L6));

elseif (12 > neuron(:,size_NeuronVariables + 5,l)) && ...
      (neuron(:,size_NeuronVariables + 5,l) >= 8);
```

```
  cell6 = randi([1,3],1,1);

elseif (8 > neuron(:,size_NeuronVariables + 5,l)) && ...
       (neuron(:,size_NeuronVariables + 5,l) >= 1);
   cell6 = 2;

end

if cell6 == 1;
  %neuron is an approximate L6cc pyramidal (units in microm)
  neuron(:,2,l) = 8;
  neuron(:,size_NeuronVariables + 6,l) = ((14.8-21.4).*rand(1,1)+21.4)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((19.6-29.4).*rand(1,1)+29.4)/25;        %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((493-597).*rand(1,1)+597)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((443-815).*rand(1,1)+815)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1700-1800).*rand(1,1)+1800)/50;        %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1700-1800).*rand(1,1)+1800)/25;
  if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
    neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
    neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
       neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
       neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
  end

elseif cell6 == 2;
  %neuron is an approximate L6ct (units in microm)
  neuron(:,2,l) = 9;
  neuron(:,size_NeuronVariables + 6,l) = ((14.8-21.4).*rand(1,1)+21.4)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((15-25).*rand(1,1)+25)/25;              %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((260-372).*rand(1,1)+372)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((616-904).*rand(1,1)+904)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
```

```
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1200-1300).*rand(1,1)+1300)/50;        %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1600-1700).*rand(1,1)+1700)/25;
   if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
     neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
     neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
        neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
        neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
  end

elseif cell6 == 3;
  %neuron is an approximate L6inv pyramidal (units in microm)
  neuron(:,2,l) = 10;
  neuron(:,size_NeuronVariables + 6,l) = ((14.8-21.4).*rand(1,1)+21.4)/25;        %soma r
  neuron(:,size_NeuronVariables + 7,l) = 0;
  neuron(:,size_NeuronVariables + 8,l) = ((15-25).*rand(1,1)+25)/25;              %soma z
  neuron(:,size_NeuronVariables + 9,l) = ((306-611.5).*rand(1,1)+611.5)/25;           %dendrite r
  neuron(:,size_NeuronVariables + 10,l) = 0;
  max_d = ((492-642).*rand(1,1)+642)/25;
  if max_d > (72 - neuron(:,size_NeuronVariables + 5,l));
     neuron(:,size_NeuronVariables + 11,l) = 72 - neuron(:,size_NeuronVariables + 5,l);  %dendrite z+

  elseif max_d <= (72 - neuron(:,size_NeuronVariables + 5,l))
     neuron(:,size_NeuronVariables + 11,l) = max_d;
  end

  if max_d > neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = neuron(:,size_NeuronVariables + 5,l);       %dendrite z-
  elseif max_d <= neuron(:,size_NeuronVariables + 5,l);
     neuron(:,size_NeuronVariables + 12,l) = max_d;
  end

  neuron(:,size_NeuronVariables + 13,l) = ((1800-2200).*rand(1,1)+2200)/50;        %axon r
  neuron(:,size_NeuronVariables + 14,l) = 0;
  max_z = ((1800-1800).*rand(1,1)+1800)/25;
   if neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) > 72;
     neuron(:,size_NeuronVariables + 15,l) = 72 - neuron(:,size_NeuronVariables + 5,l);      %axon z+
     neuron(:,size_NeuronVariables + 16,l) = max_z - neuron(:,size_NeuronVariables + 15,l);  %axon z-

  elseif neuron(:,size_NeuronVariables + 5,l) + (0.9*max_z) <= 72;
        neuron(:,size_NeuronVariables + 15,l) = (0.9*max_z);                    %axon z+
        neuron(:,size_NeuronVariables + 16,l) = (0.1*max_z);                    %axon z-
  end

end
```

```
end

%_____seperation_____%

%the distance from a given neuron to glial and other neuron is measured
%and recorded in microm
for l = 1:num_neurons;
   for m = (size_NeuronVariables + 17):(size_NeuronVariables + num_glia + 16);
         neuron(:,m,l) = dis_cy(neuron(:,(size_NeuronVariables+3),l),glial(:,(size_GliaVariables+1),(m-
(size_NeuronVariables+16))),...
            neuron(:,(size_NeuronVariables+4),l),glial(:,(size_GliaVariables+2),(m-
(size_NeuronVariables+16))), ...
            neuron(:,(size_NeuronVariables+5),l),glial(:,(size_GliaVariables+3),(m-
(size_NeuronVariables+16))));
   end
    for p = (size_NeuronVariables + num_glia + 17):(size_NeuronVariables + num_neurons + num_glia +
16);
      if neuron(1,(size_NeuronVariables+1),(p-(16 + size_NeuronVariables + num_glia))) ~= l;
        neuron(:,p,l) = dis_cy(neuron(:,(size_NeuronVariables+3),l),neuron(:,(size_NeuronVariables+3),(p-
(16 + size_NeuronVariables + num_glia))),...
           neuron(:,(size_NeuronVariables+4),l),neuron(:,(size_NeuronVariables+4),(p-(16 +
size_NeuronVariables +num_glia))),...
           neuron(:,(size_NeuronVariables+5),l),neuron(:,(size_NeuronVariables+5),(p-(16 +
size_NeuronVariables +num_glia))));
      end
    end

    for q = (size_NeuronVariables + num_glia + num_neurons + 17): ...
         (size_NeuronVariables + num_glia + num_neurons + num_exglia + 16);
        neuron(:,q,l) = dis_cy(neuron(:,(size_NeuronVariables+3),l),exglial(:,(size_GliaVariables+1),(q-
(size_NeuronVariables+16 + num_glia + num_neurons))),...
           neuron(:,(size_NeuronVariables+4),l),exglial(:,(size_GliaVariables+2),(q-
(size_NeuronVariables+16 + num_glia + num_neurons))),...
           neuron(:,(size_NeuronVariables+5),l),exglial(:,(size_GliaVariables+3),(q-
(size_NeuronVariables+16 + num_glia + num_neurons))));
    end
end

%the distance from a given glial to neuron is measured and recorded
for q = 1:num_glia;
   for r = (size_GliaVariables + 4):(size_GliaVariables + num_neurons + 3);
      glial(1,r,q) = dis_cy(glial(:,(size_GliaVariables+1),q),neuron(:,(size_NeuronVariables+3),(r-
(size_GliaVariables+3))),...
         glial(:,(size_GliaVariables+2),q),neuron(:,(size_NeuronVariables+4),(r-(size_GliaVariables+3))),...
         glial(:,(size_GliaVariables+3),q),neuron(:,(size_NeuronVariables+5),(r-(size_GliaVariables+3))));
   end
end

for s = 1:num_exglia;
   for t = (size_ExGliaVariables + 4):(size_ExGliaVariables + num_neurons + 3);
      exglial(1,t,s) = dis_cy(exglial(:,(size_ExGliaVariables+1),s),neuron(:,(size_NeuronVariables+3),(t-
(size_ExGliaVariables+3))),...
         exglial(:,(size_ExGliaVariables+2),s),neuron(:,(size_NeuronVariables+4),(t-
(size_ExGliaVariables+3))),...
         exglial(:,(size_ExGliaVariables+3),s),neuron(:,(size_NeuronVariables+5),(t-
(size_ExGliaVariables+3))));
```

```
   end
end


%_____________________________TIME REFERENCE_____________________________________%

for fire = 1:num_neurons;
   times = [0.2 0.4 0.6 0.8 1];
   if neuron(1,(size_NeuronVariables+2),fire) == 1;
      randomtime = randi([1,length(times)],1,1);
      neuron(1,1,fire) = round(5*(times(randomtime)));
      prime = neuron(1,:,fire);
   end
end

for fire2 = 1:num_neurons;
   if neuron(1,(size_NeuronVariables+2),fire2) ~= 1;
      neuron(1,1,fire2) = round(5*(prime(1,1) + ((25*prime(1,(fire2-1)...
         + size_NeuronVariables + num_glia + 17))/(3*10^6))));
   end
end

%_____________________________________VISUAL_____________________________________%

for i = 1:num_neurons;
%plot the soma (sphere)
   hold on

   rad = neuron(:,size_NeuronVariables + 6,i);
   z_adj = neuron(:,size_NeuronVariables+5,i);
   x_adj = neuron(:,size_NeuronVariables+4,i);
   y_adj = neuron(:,size_NeuronVariables+3,i);

   [X,Y,Z] = sphere();
   Z = (rad*Z+ z_adj);
   X = (rad*X+ x_adj);
   Y = (rad*Y+ y_adj);
   testsubject = surf(X,Y,Z);

   grid on
   axis ([0 200 0 200 0 200])
   if neuron(:,size_NeuronVariables+2,i) == 1;
      set(testsubject,'FaceColor','red','FaceAlpha',0.5,'EdgeColor','black','EdgeAlpha',0.1,...
      'DiffuseStrength',1,'AmbientStrength',1)
   elseif neuron(:,size_NeuronVariables+2,i) == 0;
      set(testsubject,'FaceColor','cyan','FaceAlpha',0.5,'EdgeColor','black','EdgeAlpha',0.1,...
      'DiffuseStrength',1,'AmbientStrength',1)
   end

%plot the dendrites (cylinder)
   hold on
   rad = neuron(:,size_NeuronVariables + 9,i);
   z_minus = neuron(:,size_NeuronVariables +5,i) - (neuron(:,size_NeuronVariables + 12,i));
   z_plus = neuron(:,size_NeuronVariables +5,i) + (neuron(:,size_NeuronVariables + 11,i));
   x_adj = neuron(:,size_NeuronVariables+3,i);
   y_adj = neuron(:,size_NeuronVariables+4,i);
```

```
R=[rad rad];
N=10000;

[X,Y,Z] = cylinder(R,N);
Z(1, :) = z_minus;
Z(2, :) = z_plus;
X(1,:) = X(1,:)+x_adj;
X(2,:) = X(2,:)+x_adj;
Y(1,:) = Y(1,:)+y_adj;
Y(2,:) = Y(2,:)+y_adj;
testsubject = surf(Y,X,Z);


   set(testsubject,'FaceColor','yellow','FaceAlpha',0.5,'EdgeColor','black','EdgeAlpha',0,...
   'DiffuseStrength',1,'AmbientStrength',1)


radius = linspace(0,rad,10);
theta = (pi/180)*[0:1:360];
[R,T] = meshgrid(radius,theta);
Y = R.*cos(T);
X = R.*sin(T);

A = ones(361,10)*z_plus;

alpha = ones(361,10)*z_minus;

X = X+x_adj;
Y = Y+y_adj;

hold on
test = surf(Y,X,A);

    set(test,'FaceColor','yellow','FaceAlpha',0.5,'EdgeColor','black','EdgeAlpha',0.1,...
   'DiffuseStrength',1,'AmbientStrength',1)



hold on
test2 = surf(Y,X,alpha);

    set(test2,'FaceColor','yellow','FaceAlpha',0.5,'EdgeColor','black','EdgeAlpha',0.1,...
   'DiffuseStrength',1,'AmbientStrength',1)


%plot the axons (cylinder)
  hold on
  rad = neuron(:,size_NeuronVariables+13,i);
  z_minus = neuron(:,size_NeuronVariables +5,i) - (neuron(:,size_NeuronVariables + 16,i));
  z_plus = neuron(:,size_NeuronVariables +5,i) + (neuron(:,size_NeuronVariables + 15,i));
  x_adj = neuron(:,size_NeuronVariables+3,i);
  y_adj = neuron(:,size_NeuronVariables+4,i);
  R=[rad rad];
  N=10000;

  [X,Y,Z] = cylinder(R,N);
```

```
    Z(1, :) = z_minus;
    Z(2, :) = z_plus;
    X(1,:) = X(1,:)+x_adj;
    X(2,:) = X(2,:)+x_adj;
    Y(1,:) = Y(1,:)+y_adj;
    Y(2,:) = Y(2,:)+y_adj;
    testsubject = surf(Y,X,Z);

       set(testsubject,'FaceColor','red','FaceAlpha',0.3,'EdgeColor','red','EdgeAlpha',0,...
       'DiffuseStrength',1,'AmbientStrength',1)


     radius = linspace(0,rad,10);
     theta = (pi/180)*[0:1:360];
     [R,T] = meshgrid(radius,theta);
     Y = R.*cos(T);
     X = R.*sin(T);

     A = ones(361,10)*z_plus;

     alpha = ones(361,10)*z_minus;

     X = X+x_adj;
     Y = Y+y_adj;

     hold on

     test = surf(Y,X,A);

     set(test,'FaceColor','red','FaceAlpha',0.3,'EdgeColor','black','EdgeAlpha',0.1,...
    'DiffuseStrength',1,'AmbientStrength',1)



     hold on
     test2 = surf(Y,X,alpha);

     set(test2,'FaceColor','red','FaceAlpha',0.3,'EdgeColor','black','EdgeAlpha',0.1,...
    'DiffuseStrength',1,'AmbientStrength',1)


end

for m = 1:num_glia;
   two(m,1) = glial(:,size_GliaVariables+1,m);
   one(m,1) = glial(:,size_GliaVariables+2,m);
   depth(m,1) = glial(:,size_GliaVariables+3,m);
end

for ex = 1:num_exglia;
   vert(ex,1) = exglial(1,size_ExGliaVariables+1,ex);
   horiz(ex,1) = exglial(1,size_ExGliaVariables+2,ex);
   in(ex,1) = exglial(1,size_ExGliaVariables+3,ex);
end
```

```
hold on
scatter3(two,one,depth,500,'r','*')

hold on
scatter3(vert,horiz,in,500,'c','*')
```

‘run_cylindrical’

```
%The initial concentrations for dependent and independent variables

global k r main_neuron main_glia main_exglia time num_neurons num_glia num_exglia ...
      size_NeuronVariables size_GliaVariables size_ExGliaVariables ...
      neuron glial exglial PrPSc

PrPSc = 0.04e-17;                              %initial PrPSc

%rate constants for neurons
k = zeros(10,1);

k(1) = 0.1;                                    %PrPC -> PrPSc

%rate constants for glia
r = zeros(10,1);


%_____________________________________________________________________________


size_NeuronVariables = 25;
size_GliaVariables = 3;
size_ExGliaVariables = 3;

set_up_cylindrical

%_____________________________values for neurons_____________________________

for i = 1:num_neurons;
   %independent variables for neurons
   %neuron(:,1,i) = 0;                                %time of intial firing
   %neuron(:,2,i) = 0;                                %neuronal phenotype (1-10)
   %dependent variables for neurons (10^-17)
   neuron(:,3,i) = (3.04 - 6.08).*rand(1,1)+(6.08);          %surface PrPC
   neuron(:,4,i) = 0;                                %surface PrPSc total
   neuron(:,5,i) = 0;                                %surface PrPSc monomers
   neuron(:,6,i) = 0;                                %surface PrPSc amorphous agrregate
   neuron(:,7,i) = 0;                                %surface PrPSc fibriles
   neuron(:,8,i) = (10/100);                            %EE PrPC
   neuron(:,9,i) = 0;                                %EE PrPSc monomers
   neuron(:,10,i) = 0;                               %EE PrPSc amorphous agrregate
   neuron(:,11,i) = (10/100);                           %RE PrPC
   neuron(:,12,i) = 0;                               %RE PrPSc monomers
   neuron(:,13,i) = 0;                               %RE PrPSc amorphous agrregate
   neuron(:,14,i) = (1/100);                            %LE PrPC
   neuron(:,15,i) = 0;                               %LE PrPSc monomers
```

```
   neuron(:,16,i) = 0;                          %LE PrPSc amorphous agrregate
   neuron(:,17,i) = (1/100);                      %EL PrPC
   neuron(:,18,i) = 0;                          %EL PrPSc monomers
   neuron(:,19,i) = 0;                          %EL PrPSc amorphous agrregate
   neuron(:,20,i) = (1/100);                      %degraded PrPC
   neuron(:,21,i) = 0;                          %partially degraded PrPSc
   neuron(:,22,i) = 0;                          %infected?
   neuron(:,23,i) =  0;                         %released PrPSc

   %[] dependent rate equations for neurons
   neuron(:,24,i) = 0.1*(neuron(:,1,i));              %surface PrPC -> surface PrPSc (orig PrPSc)
   neuron(:,25,i) = 0.1*(neuron(:,3,i));              %surface PrPC -> surface PrPSc (new PrPSc)


end
%the neuron array has been populated with intitial values

%_____________________values for glia_____________________________________

for j = 1:num_glia;
   %independent variables for glia
   glial(:,1,j) = 0.04e-17;

   %dependent variables for glia
   glial(:,2,j) = (3.04e-17 - 6.08e-17).*rand(1,1)+(6.08e-17);

   %[] dependent rate equations for glia
   glial(:,3,j) = 10;
end
%the glia have been populated by intial values

for l = 1:num_exglia;
   %independent variables for exglia
   exglial(:,1,l) = 0.04e-17;

   %dependent variables for exglia
   exglial(:,2,l) = (3.04e-17 - 6.08e-17).*rand(1,1)+(6.08e-17);

   %[] dependent rate equations for exglia
   exglial(:,3,l) = 10;
end
%the exglia have been populated by intial values

%PrP_plot_cylindrical

main_neuron = zeros(1440,(4+num_neurons+num_glia+size_NeuronVariables),num_neurons);
main_glia = zeros(1440,(2+num_neurons+size_GliaVariables),num_glia);
main_exglia = zeros(1440,(2+num_neurons+size_ExGliaVariables),num_exglia);

time = 1;

% endocytosis

% for t = 1:0.2:3600;
%     endocytosis;
% end
```

```
%disp(neuron)
```

‘Plot_cylindrical’

```
global   neuron  size_NeuronVariables  num_neurons

%Generation and Distribution of trackable species in individual neurons

%Moves = zeros((num_neurons)+1,1,3,num_neurons);

% dim 3(1) = PrPC
   % dim 1(1->(num_neurons-1)) = synapses
      % [local [PrP] x y z]
   % dim 1(num_neurons + 1) = free terminals
      % [[PrP] x y z]
   % dim 1(num_neurons + 2) = EE
   % dim 1(num_neurons + 3) = MVB
   % dim 1(num_neurons + 4) = RE
   % dim 1(num_neurons + 5) = LE
   % dim 1(num_neurons + 6) = EndoLys
   % dim 1(num_neurons + 7) = exosome
   % dim 1(num_neurons + 8) = Golgi cis
   % dim 1(num_neurons + 9) = Golgi mid
   % dim 1(num_neurons + 10) = Golgi trans
   % dim 1(num_neurons + 11) = network
   % dim 1(num_neurons + 12) = phogolysosome

% dim 3(2) = PrPSc
   % dim 1 = same for PrPC

% dim 3(3) = cytokines/ chemoattractants

% dim 4 = neuron index

% each neuron has its own assigned 3 dimensional move matrix

 for a = 1: num_neurons;
     [probD, probA, T] = pick_equation(neuron(:,2,a),0,0);
    termsize = size(T);
    num_term = termsize(1);
    terminals = [];
    for b = 1:num_term;
      theta = ((0-(2*pi)).*rand(1,1)+(2*pi));
      x = ((10^3)*T(b,1)/25)*cos(theta) + neuron(:,size_NeuronVariables + 3,a);
      y = ((10^3)*T(b,1)/25)*sin(theta) + neuron(:,size_NeuronVariables + 4,a);
      z = ((10^3)*T(b,2)/25);
```

```
      % [x y z r h]
      terminals = [terminals ; x y z (10^3)*T(b,2) (10^3)*T(b,1)];
    end

   synapses = [];
   f_terminals = [];
   dim12 = [];

   for c = 1:num_neurons;
       if neuron(:,size_NeuronVariables + 1,c) ~= a;
 %_____________________________________________________________________%

          % find the region of overlap for axon densities
          % soma
          if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a))
< ...
             (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables +
6,c)))...
             && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables +
15,a)) > ...
             (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables +
6,c))));
             % overlap axon on soma!

             ds = dis_cy(neuron(:,size_NeuronVariables +
3,a),neuron(:,size_NeuronVariables + 3,c)...
                ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables +
4,c),0,0);
             if (ds-neuron(:,size_NeuronVariables + 6,c)) < neuron(:,size_NeuronVariables
+ 13,a);
                % radial overlap!
                for r = 0:1:25*neuron(:,size_NeuronVariables + 6,c);
                   for h = ((neuron(:,size_NeuronVariables + 3,c) -
neuron(:,size_NeuronVariables + 6,c)))*25:...
                         ((neuron(:,size_NeuronVariables + 3,c) +
neuron(:,size_NeuronVariables + 6,c)))*25;
                      for theta = 0:(2*pi);
                         z = h/25;
                         x = (r*cos(theta))/25 + neuron(:,size_NeuronVariables + 3,c);
                         y = (r*sin(theta)/25) + neuron(:,size_NeuronVariables + 4,c);
                         d = dis_cy(neuron(:,size_NeuronVariables+3,a)...

,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);

                         if d < neuron(:,size_NeuronVariables+13,a);
                            %overlap point!
```

```
                        %disp('wow')
                        pc = neuron(:,2,c);
                        [probD, probA, Axterminals] = pick_equation(pc,r,h);
                        %disp(probA)
                        if probA > 0.8;

                           % synapse! [local [PrP] x y z]
                           synapses = [synapses 1 0 x y z];
                        end
                     end
                  end
               end
            end
         end


      if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a))
< ...
         (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables +
11,c)))...
         && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables +
15,a)) > ...
         (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables +
12,c))));

          % overlap axon on dendrite!
          %disp('wow')
         ds = dis_cy(neuron(:,size_NeuronVariables +
3,a),neuron(:,size_NeuronVariables + 3,c)...
            ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables +
4,c),0,0);
         if (ds-neuron(:,size_NeuronVariables + 9,c)) < neuron(:,size_NeuronVariables
+ 13,a);
            %disp('doublewow')
            % radial overlap!
            if (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables +
15,a))...
                  <= (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 11,c));
               top = (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 11,c))*25;
            elseif (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables
+ 15,a))...
                  > (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 11,c));
```

```
                top = (neuron(:,size_NeuronVariables +
5,a)+neuron(:,size_NeuronVariables + 15,a))*25;
            end
            if (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables +
16,a))...
                <= (neuron(:,size_NeuronVariables + 5,c)-
neuron(:,size_NeuronVariables + 12,c));
                bottom = (neuron(:,size_NeuronVariables + 5,c)-
neuron(:,size_NeuronVariables + 12,c))*25;
            elseif (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables
+ 16,a))...
                > (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables
+ 12,c));
                bottom = (neuron(:,size_NeuronVariables + 5,a) -
neuron(:,size_NeuronVariables + 16,a))*25;
            end
            for r = 0:10:25*neuron(:,size_NeuronVariables + 9,c);
                for h = bottom:10:top;
                    for theta = 0:(2*pi);
                        z = h/25;
                        x = (r*cos(theta)/25)+ neuron(:,size_NeuronVariables + 3,c);
                        y = (r*sin(theta)/25)+ neuron(:,size_NeuronVariables + 4,c);
                        d = dis_cy(neuron(:,size_NeuronVariables+3,a)...

,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);
                        if d < neuron(:,size_NeuronVariables+13,a);
                            %overlap point!
                            %disp('oh yeah')
                            [probD, A, t1] = pick_equation(neuron(:,2,a),r,h);
                            [D, probA, t2] = pick_equation(neuron(:,2,c),r,h);
                            disp(A)
                            disp(D)
                            if (A > 0.8) && (D > 0.8) && (D >= probA);
                                %disp('bop')
                                % synapse! [local [PrP] x y z]
                                synapses = [synapses 2 0 x y z];

                            elseif (A > 0.8) && (D > 0.8) && (D < probA);
                                %disp('wham')
                                synapses = [synapses 3 0 x y z];
                            end
                        end
                    end
                end
            end
```

```
            end

        end

        if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a))
< ...
            (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables +
15,c)))...
            && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables +
15,a)) > ...
            (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables +
16,c))));

             % overlap axon on dendrite!
            ds = dis_cy(neuron(:,size_NeuronVariables +
3,a),neuron(:,size_NeuronVariables + 3,c)...
                ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables +
4,c),0,0);
            if (ds-neuron(:,size_NeuronVariables + 13,c)) < neuron(:,size_NeuronVariables
+ 13,a);
                % radial overlap!
                if (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables +
15,a))...
                        <= (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 15,c));
                    top = (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 15,c))*25;
                elseif (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables
+ 15,a))...
                        > (neuron(:,size_NeuronVariables +
5,c)+neuron(:,size_NeuronVariables + 15,c));
                    top = (neuron(:,size_NeuronVariables +
5,a)+neuron(:,size_NeuronVariables + 15,a))*25;
                end
                if (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables +
16,a))...
                        <= (neuron(:,size_NeuronVariables + 5,c)-
neuron(:,size_NeuronVariables + 16,c));
                    bottom = (neuron(:,size_NeuronVariables + 5,c)-
neuron(:,size_NeuronVariables + 16,c))*25;
                elseif (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables
+ 16,a))...
                        > (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables
+ 16,c));
                    bottom = (neuron(:,size_NeuronVariables + 5,a) -
neuron(:,size_NeuronVariables + 16,a))*25;
```

```
            end

            for r = 0:10:25*neuron(:,size_NeuronVariables + 13,c);
               for h = bottom:10:top;
                  for theta = 0:(2*pi);
                     z = h/25;
                     x = (r*cos(theta)/25)+ neuron(:,size_NeuronVariables + 3,c);
                     y = (r*sin(theta)/25)+ neuron(:,size_NeuronVariables + 4,c);
                     d = dis_cy(neuron(:,size_NeuronVariables+3,a)...

,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);
                     if d < neuron(:,size_NeuronVariables+13,a) && (d >
neuron(:,size_NeuronVariables+9,a));
                        %overlap point!
                        [probD, A, t1] = pick_equation(neuron(:,2,a),r,h);
                        [D, probA, t2] = pick_equation(neuron(:,2,c),r,h);
                        if (A > 0.8) && (probA > 0.8);
                           % synapse! [local [PrP] x y z]
                           synapses = [synapses 3 0 x y z];
                        end
                     end
                  end
               end
            end
         end

%
%_____________________________________________________________________%

         % find the region of overlap for terminals
         % soma
         for d = 1:num_term;
               term_d = dis_cy(terminals(d,1),neuron(:,size_NeuronVariables + 3,c),...
                  terminals(d,2),neuron(:,size_NeuronVariables +
4,c),terminals(d,3),neuron(:,size_NeuronVariables + 5,c));

               if term_d < neuron(:,size_NeuronVariables + 6,l);
                  % soma overlap! = 1
                  synapses = [synapses 1 0 terminals(d,1) terminals(d,2) terminals(d,3)];


               elseif (term_d < neuron(:,size_NeuronVariables + 9,l)) && ...
                     (term_d > neuron(:,size_NeuronVariables + 6,l));
                  % dendrite or axon overlap!
```

```
                [probD, probA, Axterminals] = pick_equation(neuron(:,2,c),terminals(d,4),
terminals(d,3));
                if (probD > 0.8) && (probD > probA);
                  % synapse! = 2
                  synapse = [synapses 2 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;

                elseif (probA > 0.8) && (probA > probD);
                   % synapse! = 3
                   synapses = [synapses 3 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
                end

             elseif (term_d < neuron(:,size_NeuronVariables + 13,l)) && ...
                   (term_d > neuron(:,size_NeuronVariables + 9,l));
                % just axon overlap!

                [probD, probA, Axterminals] =
pick_equation(neuron(:,2,c),terminals(d,4),terminals(d,3));
                if (probA > 0.8);
                   % synapse = 3
                   synapses = [synapses 3 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
                elseif (probA < 0.9);
                   % free terminal!
                   f_terminals = [f_terminals 0 terminals(d,1) terminals(d,2)
terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
                end

             end
       end

      end
```

```
%       if neuron(:,size_NeuronVariables + 1,c) ~= a;
%         % Moves = zeros((num_neurons*2)+10,1,3,num_neurons);
%         dimsize = size(dim12);
% %           if dimsize(1) == 0;
% %              dim12 = [dim12 ; synapses];
% %           elseif dimsize(1) > 0;
%             c = size(synapses);
%             if dimsize(2) >= c(2);
%                 synapses = [synapses zeros(1,dimsize(2)-c(2))];
%             elseif c(2) > dimsize(2);
%                dim12 = [dim12 zeros(dimsize(1),c(2)-dimsize(2))];
%             end
%
%             dim12 = [dim12; synapses];
%             disp(dim12)
% %           end
%       elseif neuron(:,size_NeuronVariables + 1,a) == c;
%          dimsize = size(dim12);
%          dim12 = [dim12; zeros(1,dimsize(2))];
%       end
%    end
%
%    for e = 1:num_term;
%        if terminals(e,1) ~= 0;
%            f_terminals = [f_terminals 0 terminals(e,1) terminals(e,2) terminals(e,3)];
%        end
%    end
%    dimsize = size(dim12);
%    ft = size(f_terminals);
%    if dimsize(2) >= ft(2);
%        f_terminals = [f_terminals zeros(1,dimsize(2)-ft(2))];
%    elseif ft(2) > dimsize(2);
%       dim12 = [dim12 zeros(dimsize(1),ft(2)-dimsize(2))];
%    end
%
%    dim12 = [dim12; f_terminals];
%
%    if a == 1;
%       Moves = dim12;
%    elseif a ~= 1;
%       dimsize = size(dim12);
%       movesize = size(Moves);
%       if dimsize(2) >= movesize(2);
%           Moves = [Moves zeros(movesize(1),dimsize(2)-movesize(2))];
%       elseif movesize(2) > dimsize(2);
```

```
%            dim12 = [dim12 zeros(dimsize(1), movesize(2)-dimsize(2))];
%        end
%
%        Moves(:,:,a) =  dim12;
%    end
%
   end
   disp(synapses)
 end

% for q = 1:num_neurons;
%
%    PrPC = neuron(1,3,q);                    %moles PrPC/cell
%    PrPC_same = neuron(1,3,q);                   %moles PrPC/cell copy
%
%    amount = 1;
%    coord_x = 2;
%    coord_y = 3;
%    coord_z = 4;
%    k = 1;
%
%    while PrPC > 0;
%
%        neuronPrP(:,amount,q) = 0;
%        neuronPrP(:,coord_x,q) = 0;
%        neuronPrP(:,coord_y,q) = 0;
%        neuronPrP(:,coord_z,q) = 0;
%
%    end
%end

%
%_____________________________________visual_____________________________________

% M = size(Moves);
%
%
% X_1 = [];
% Y_1 = [];
% Z_1 = [];
% X_2 = [];
% Y_2 = [];
% Z_2 = [];
% X_3 = [];
% Y_3 = [];
% Z_3 = [];
```

```
% X_4 = [];
% Y_4 = [];
% Z_4 = [];
% % X_5 = [];
% % Y_5 = [];
% % Z_5 = [];
% % X_6 = [];
% % Y_6 = [];
% % Z_6 = [];
% % X_7 = [];
% % Y_7 = [];
% % Z_7 = [];
% % X_8 = [];
% % Y_8 = [];
% % Z_8 = [];
% % X_9 = [];
% % Y_9 = [];
% % Z_9 = [];
% % X_10 = [];
% % Y_10 = [];
% % Z_10 = [];
% % X_11 = [];
% % Y_11 = [];
% % Z_11 = [];
% % X_12 = [];
% % Y_12 = [];
% % Z_12 = [];
% %
%
% % synapses! [local [PrP] x y z]
% % free terminals! [[PrP] x y z]
% for m = 3:5:M(2)-2;
%     %disp(Moves(2,m,1))
%     %disp(Moves(3,m-1,1))
%     if Moves(2,m,1) ~= 0;
%         X_1 = [X_1 Moves(2,m,1)];
%     end
% %     if Moves(3,m-1,1) ~= 0;
% %         X_2 = [X_2 Moves(3,m-1,1)];
% %     end
% %     if Moves(1,m,2) ~= 0;
% %         X_3 = [X_3 Moves(1,m,2)];
% %     end
% %     if Moves(3,m-1,2) ~= 0;
% %         X_4 = [X_4 Moves(3,m-1,2)];
% %     end
```

```
% %     if neuronPrP(5,m,1) ~= 0;
% %         X_5 = [X_5 neuronPrP(5,m,1)];
% %     end
% %     if neuronPrP(6,m,1) ~= 0;
% %         X_6 = [X_6 neuronPrP(6,m,1)];
% %     end
% %     if neuronPrP(7,m,1) ~= 0;
% %         X_7 = [X_7 neuronPrP(7,m,1)];
% %     end
% %     if neuronPrP(8,m,1) ~= 0;
% %         X_8 = [X_8 neuronPrP(8,m,1)];
% %     end
% %     if neuronPrP(9,m,1) ~= 0;
% %         X_9 = [X_9 neuronPrP(9,m,1)];
% %     end
% %     if neuronPrP(10,m,1) ~= 0;
% %         X_10 = [X_10 neuronPrP(10,m,1)];
% %     end
% %     if neuronPrP(11,m,1) ~= 0;
% %         X_11 = [X_11 neuronPrP(11,m,1)];
% %     end
% %     if neuronPrP(12,m,1) ~= 0;
% %         X_12 = [X_12 neuronPrP(12,m,1)];
% %     end
% end
% %
% for n = 4:5:M(2)-1;
%     if Moves(2,n,1) ~= 0;
%         Y_1 = [Y_1 Moves(2,n,1)];
%     end
% %     if Moves(3,n-1,1) ~= 0;
% %         Y_2 = [Y_2 Moves(3,n-1,1)];
% %     end
% %     if Moves(1,n,2) ~= 0;
% %         Y_3 = [Y_3 Moves(1,n,2)];
% %     end
% %     if Moves(3,n-1,2) ~= 0;
% %         Y_4 = [Y_4 Moves(3,n-1,2)];
% %     end
% %     if neuronPrP(5,n,1) ~= 0;
% %         Y_5 = [Y_5 neuronPrP(5,n,1)];
% %     end
% %     if neuronPrP(6,n,1) ~= 0;
% %         Y_6 = [Y_6 neuronPrP(6,n,1)];
% %     end
% %     if neuronPrP(7,n,1) ~= 0;
```

```
% %         Y_7 = [Y_7 neuronPrP(7,n,1)];
% %     end
% %     if neuronPrP(8,n,1) ~= 0;
% %         Y_8 = [Y_8 neuronPrP(8,n,1)];
% %     end
% %     if neuronPrP(9,n,1) ~= 0;
% %         Y_9 = [Y_9 neuronPrP(9,n,1)];
% %     end
% %     if neuronPrP(10,n,1) ~= 0;
% %         Y_10 = [Y_10 neuronPrP(10,n,1)];
% %     end
% %     if neuronPrP(11,n,1) ~= 0;
% %         Y_11 = [Y_11 neuronPrP(11,n,1)];
% %     end
% %     if neuronPrP(12,n,1) ~= 0;
% %         Y_12 = [Y_12 neuronPrP(12,n,1)];
% %     end
% end
% %
% for o = 5:5:M(2);
%     if Moves(2,o,1) ~= 0;
%         Z_1 = [Z_1 Moves(2,o,1)];
%     end
% %     if Moves(3,o-1,1) ~= 0;
% %         Z_2 = [Z_2 Moves(3,o-1,1)];
% %     end
% %     if Moves(1,o,2) ~= 0;
% %         Z_3 = [Z_3 Moves(1,o,2)];
% %     end
% %     if Moves(3,o-1,2) ~= 0;
% %         Z_4 = [Z_4 Moves(3,o-1,2)];
% %     end
% %     if neuronPrP(5,o,1) ~= 0;
% %         Z_5 = [Z_5 neuronPrP(5,o,1)];
% %     end
% %     if neuronPrP(6,o,1) ~= 0;
% %         Z_6 = [Z_6 neuronPrP(6,o,1)];
% %     end
% %     if neuronPrP(7,o,1) ~= 0;
% %         Z_7 = [Z_7 neuronPrP(7,o,1)];
% %     end
% %     if neuronPrP(8,o,1) ~= 0;
% %         Z_8 = [Z_8 neuronPrP(8,o,1)];
% %     end
% %     if neuronPrP(9,o,1) ~= 0;
% %         Z_9 = [Z_9 neuronPrP(9,o,1)];
```

```
% %     end
% %     if neuronPrP(10,o,1) ~= 0;
% %         Z_10 = [Z_10 neuronPrP(10,o,1)];
% %     end
% %     if neuronPrP(11,o,1) ~= 0;
% %         Z_11 = [Z_11 neuronPrP(11,o,1)];
% %     end
% %     if neuronPrP(12,o,1) ~= 0;
% %         Z_12 = [Z_12 neuronPrP(12,o,1)];
% %     end
%  end
% %
%
% disp(size(X_1))
% disp(size(Y_1))
% scatter3(X_1,Y_1,Z_1,100,'c','h','filled')
% hold on
%
% % scatter3(X_2,Y_2,Z_2,500,'m','h','filled')
% % hold on
% %
% % scatter3(X_3,Y_3,Z_3,500,'c','h','filled')
% % hold on
% %
% % scatter3(X_4,Y_4,Z_4,500,'r','h','filled')
% % hold on
% %
% % scatter3(X_5,Y_5,Z_5,500,'g','h','filled')
% % hold on
% %
% % scatter3(X_6,Y_6,Z_6,500,'b','h','filled')
% % hold on
% %
% % scatter3(X_7,Y_7,Z_7,500,'k','h','filled')
% % hold on
% %
% % scatter3(X_8,Y_8,Z_8,500,'y','o','filled')
% % hold on
% %
% % scatter3(X_9,Y_9,Z_9,500,'m','o','filled')
% % hold on
% %
% % scatter3(X_10,Y_10,Z_10,500,'c','o','filled')
% % hold on
% %
% % scatter3(X_11,Y_11,Z_11,500,'r','o','filled')
```

```
% % hold on
% %
% % scatter3(X_12,Y_12,Z_12,500,'g','o','filled')
```

‘neuron_generator’

```
%These generate dendrite and axonal densities, which will be used to derive
%probability distrubutionsto be used generally for the model

%L2 pyramidal
type = 2;
scale = 5.0125;
axon_radius = ((1500-1600).*rand(1,1)+1600)/2;
ap = 93;
b = 95;
A = 20;
cut = 2.5;
depth = randi([1500,1800],1,1);
length_a = (185-325).*rand(1,1)+325;
prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
prime_t = randi([27,31],1,1) - prime_o;
prime_d = randi([18,22],1,1);
apical_path = ((234-321).*rand(1,1)+321);
basal_path = ((119-169).*rand(1,1)+169);
length_ax = ((1500-1600).*rand(1,1)+1600);
para = [(apical_path/5) (basal_path/10)  90];
sigma_1 = 10000;
sigma_2 = 100000;
mean_1 = 250;
mean_2 = 1100;
weight_1 = 5;
weight_2 = 1;
%disp('L2py ____________________________________________________________')
[Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
    axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
    length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


% %L3 pyramidal
% type = 2;
% scale = 11.6861;
% axon_radius = ((1600-1700).*rand(1,1)+1700)/2;
% ap = 90;
% b = 90;
% A = 86;
% cut = 1.1;
% depth = randi([1300,1500],1,1);
% length_a = (315-579).*rand(1,1)+579;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([17,21],1,1) - prime_o;
% prime_d = randi([24,28],1,1);
% apical_path = (130.5-197.5).*rand(1,1)+197.5;
% basal_path = ((119-169).*rand(1,1)+169);
% length_ax = ((1700-1800).*rand(1,1)+1800);
% para = [(apical_path/3) (basal_path/4)  90];
```

```
% sigma_1 = 20000;
% sigma_2 = 100000;
% mean_1 = 300;
% mean_2 = 1100;
% weight_1 = 1.5;
% weight_2 = 1;
% %disp('L3py ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L4sp
% type = 2;
% scale = 13.75;
% axon_radius = ((1200-1300).*rand(1,1)+1300)/2;
% ap = 95;
% b = 90;
% A = 97;
% cut = .7;
% depth = randi([800,1300],1,1);
% length_a = (170-468).*rand(1,1)+468;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([18,28],1,1) - prime_o;
% prime_d = randi([10,18],1,1);
% apical_path = (72-129).*rand(1,1)+129;
% basal_path = (216.5-330.5).*rand(1,1)+330.5;
% length_ax = ((1500-1600).*rand(1,1)+1600);
% para = [(apical_path/4.7) (basal_path/3)  100];
% sigma_1 = 30000;
% sigma_2 = 100000;
% mean_1 = 350;
% mean_2 = 850;
% weight_1 = 2;
% weight_2 = 1;
% %disp('L4sp ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);

%L4ss
% type = 2;
% scale = 9.25;
% axon_radius = ((700-800).*rand(1,1)+800)/2;
% ap = 0;
% b = 70;
% A = 95;
% cut = .45;
% depth = randi([1100,1300],1,1);
% length_a = 0;
% prime_o = 0;
% prime_t = 0;
% prime_d = round((3.6-6.2).*rand(1,1)+6.2);
% apical_path = 0;
% basal_path = (280-310).*rand(1,1)+310;
% length_ax = (1400-1500).*rand(1,1)+1500;
```

```
% para = [0 (basal_path/3)  90];
% sigma_1 = 30000;
% sigma_2 = 100000;
% mean_1 = 450;
% mean_2 = 750;
% weight_1 = 1;
% weight_2 = 1;
% %disp('L4ss ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L4py
% type = 2;
% scale = 10;
% axon_radius = ((1200-1300).*rand(1,1)+1300)/2;
% ap = 93;
% b = 80;
% A = 96;
% cut = .3;
% depth = randi([1100,1200],1,1);
% length_a = (453-531).*rand(1,1)+531;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([6,7],1,1);
% prime_d = round((6-7.2).*rand(1,1)+7.2);
% apical_path = (260-332).*rand(1,1)+332;
% basal_path = (224.5-335.5).*rand(1,1)+335.5;
% length_ax = (1600-1700).*rand(1,1)+1700;
% para = [(apical_path/7) (basal_path/3)  95];
% sigma_1 = 30000;
% sigma_2 = 50000;
% mean_1 = 350;
% mean_2 = 1250;
% weight_1 = 1.6;
% weight_2 = 1;
% %disp('L4py ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L5st
% type = 2;
% scale = 20;
% axon_radius = ((1400-1500).*rand(1,1)+1500)/2;
% ap = 90;
% b = 92;
% A = 93;
% cut = .5;
% depth = randi([500,800],1,1);
% length_a = (950-1154).*rand(1,1)+1154;
% prime_o  = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([16,19],1,1) - prime_o;
% prime_d = randi([30,32],1,1);
% apical_path = (269-417).*rand(1,1)+417;
```

```
% basal_path = (260-372).*rand(1,1)+372;
% length_ax = ((1700-1800).*rand(1,1)+1800);
% para = [(apical_path/13) (basal_path/13)  300];
% sigma_1 = 10000;
% sigma_2 = 60000;
% mean_1 = 250;
% mean_2 = 1150;
% weight_1 = 3.75;
% weight_2 = 1;
% %disp('L5st ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);



%L5tt
% type = 2;
% scale = 6.25;
% axon_radius = ((900-1000).*rand(1,1)+1000)/2;
% ap = 40;
% b = 85;
% A = 95;
% cut = 1;
% depth = randi([400,700],1,1);
% length_a = (1050-1210).*rand(1,1)+1210;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([3,11],1,1) - prime_o;
% prime_d = randi([20,20],1,1);
% apical_path = ((112-142).*rand(1,1)+142);
% basal_path = ((138-176).*rand(1,1)+176);
% length_ax = (1700-1800).*rand(1,1)+1800;
% para = [(apical_path/3) (basal_path/3)  100];
% sigma_1 = 5000;
% sigma_2 = 60000;
% mean_1 = 150;
% mean_2 = 1250;
% weight_1 = 1;
% weight_2 = 2;
% %disp('L5tt ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L6cc
% type = 2;
% scale = 15;
% axon_radius = ((1700-1800).*rand(1,1)+1800)/2;
% ap = 90;
% b = 90;
% A = 95;
% cut = .25;
% depth = randi([200,400],1,1);
% length_a = (443-815).*rand(1,1)+815;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
```

```
% prime_t = randi([12,16],1,1) - prime_o;
% prime_d = randi([15,21],1,1);
% apical_path = (276-444).*rand(1,1)+444;
% basal_path = (402-506).*rand(1,1)+506;
% length_ax = (1700-1800).*rand(1,1)+1800;
% para = [(apical_path/12) (basal_path/10)  400];
% sigma_1 = 50000;
% sigma_2 = 100000;
% mean_1 = 450;
% mean_2 = 1050;
% weight_1 = 1;
% weight_2 = 2;
% %disp('L6cc _____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L6ct
% type = 1;
% scale = 2.5;
% axon_radius = ((1200-1300).*rand(1,1)+1300)/2;
% ap = 92;
% b = 92;
% A = 80;
% cut = .8;
% depth = randi([50,300],1,1);
% length_a = (616-904).*rand(1,1)+904;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([22,28],1,1)-prime_o;
% prime_d = randi([17,23],1,1);
% apical_path = (220-322).*rand(1,1)+322;
% basal_path = (201.5-286.5).*rand(1,1)+286.5;
% length_ax = (1600-1700).*rand(1,1)+1700;
% para = [(apical_path/10) (basal_path/10)  90];
% sigma_1 = 90000;
% sigma_2 = 1;
% mean_1 = 1600;
% mean_2 = 0;
% weight_1 = 1;
% weight_2 = 0;
% %disp('L6ct _____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);


%L6inv py
% type = 1;
% scale = 6.25;
% axon_radius = ((1800-2200).*rand(1,1)+2200)/2;
% ap = 80;
% b = 80;
% A = 80;
% cut = .4;
% depth = randi([200,400],1,1);
```

```
% length_a = (492-642).*rand(1,1)+642;
% prime_o = round((length_a/100)*((1.8-3.6).*rand(1,1)+3.6));
% prime_t = randi([8,14],1,1) - prime_o;
% prime_d = randi([8,14],1,1);
% apical_path = (306-528).*rand(1,1)+528;
% basal_path = (384.5-611.5).*rand(1,1)+611.5;
% length_ax = (2000-2150).*rand(1,1)+2150;
% para = [(apical_path/11) (basal_path/6)  200];
% sigma_1 = 140000;
% sigma_2 = 1;
% mean_1 = 1450;
% mean_2 = 0;
% weight_1 = 1;
% weight_2 = 0;
% % disp('L6invpy ____________________________________________________________')
% [Apical, Basal, Axonal, terminals] = generator(type, para, scale,...
%     axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path, basal_path,...
%     length_ax, sigma_1, sigma_2, mean_1, mean_2, weight_1, weight_2, cut);
%
% Termsize = length(terminals);
% Terminal = [];
% for L2 = 1:2:Termsize;
%     Terminal = [Terminal ; terminals(1,L2)/10^3 terminals(1,L2+1)/10^3];
% end
%
% disp(Terminal)
```

‘generator’

```
function [Apical, Basal, Axonal,terminals] = generator(type, para, scale,...
   axon_radius,ap,b,A, depth, length_a,prime_o, prime_t, prime_d, apical_path,...
   basal_path, length_ax, sigma_1, sigma_2, Mean_1, Mean_2, weight_1, weight_2,cut)

apical_endings = [];
basal_endings = [];
terminals = [];
Apical = [];

if length_a ~= 0;

   Apical = [];

   branch_a = 0;
   total_length_a = 0;
   l = 0;
   for i = (1:prime_o);
      %add = [numel rad r z length];

      add = [1 0 ...
         0 (depth+((1-length_a).*rand(1,1)+length_a)) 5*para(1,1)];
      Apical = [Apical add];
      l = l+ add(1,5);
      %branch_a = branch_a +.5;
```

```
end

for j = (1:prime_t);
   add = [1 0 ...
      0 (depth+length_a) 5*para(1,1)];
   Apical = [Apical add];

   %branch_a = branch_a +.5;
   l = l+ add(1,5);

end

index = 5*(prime_o + prime_t);
order = 1;
branch_index = index;

while branch_index > 0;
   add = [];
   new = 0;
   branch_i = 0;

   for k = 5:5:index;
      branch = randi([1,100],1,1);
      if (Apical(order,k) >= para(1,1)) && (branch >= ap);
        % disp('poop')
         %add = [numel rad r z length];
         slice = (para(1,1)-Apical(order,k)).*rand(1,1)+Apical(order,k);
         Apical(order,k-3) = (((((3*(.45^2)*(Apical(order,k)-slice))/4)^(1/3))-slice).*rand(1,1)+slice);
         l_1 = (para(1,1)-((4*para(1,1))/(order+1))).*rand(1,1)+((4*para(1,1))/(order+1));
         rad1 = (((((3*(.45^2)*(Apical(order,k)))/4)^(1/3))...
               -slice).*rand(1,1)+slice);
         rad2 = (((((3*(.45^2)*(Apical(order,k)))/4)^(1/3))...
               -l_1).*rand(1,1)+l-1);

         theta = ((-(pi/2)) - (pi/2)).*rand(1,1)+(pi/2);
         z = Apical(order,k-1) + (Apical(order,k-3)*sin(theta));
         phi = (0 - (2*pi)).*rand(1,1)+(pi*2);
         x = cos(theta) * (Apical(order,k-3)*sin(theta));
         y = sin(abs(pi - phi))*x;
         zeta = abs(atan(y/Apical(order,k-3)));
         r = abs(y/sin(zeta));


         branch_1 = [Apical(order,k-4)+1 rad1 r z l_1];
         branch_2 = [Apical(order,k-4)+1 rad2 r z (Apical(order,k)-slice)];
         add = [add branch_1 branch_2];
         Apical(order,k) = slice;
         new = new+(2*5);
         branch_i = branch_i + (2*5);

         branch_a = branch_a +1;

      elseif Apical(order,k) >= para(1,1) && branch < ap;
         Apical(order,k-3) = (((((3*(.45^2)*(Apical(order,k)))/4)^(1/3))...
               -Apical(order,k)).*rand(1,1)+Apical(order,k));
```

```
            add = [add Apical(order,k-4) Apical(order,k-3)...
               Apical(order,k-2) Apical(order,k-1) Apical(order,k)];
            Apical(order,k-4) = 0;
            Apical(order,k-3) = 0;
            Apical(order,k-2) = 0;
            Apical(order,k-1) = 0;
            Apical(order,k) = 0;
            new = new + 5;

         elseif Apical(order,k) < para(1,1);
               Apical(order,k-3) = (((((3*(.45^2)*(Apical(order,k)))/4)^(1/3))...
                  -Apical(order,k)).*rand(1,1)+Apical(order,k));
               apical_endings = apical_endings+1;
         end

      end
      if branch_i == 0;
         for r = 2:5:length(add);
            if add(1,r) == 0;
               add(1,r) = (((((3*(.45^2)*(add(1,(r+3))))/4)^(1/3))...
                  -add(1,(r+3))).*rand(1,1)+add(1,(r+3)));
               total_length_a = total_length_a + add(1,(r+3));
               apical_endings = apical_endings+1;
            end
         end
      end
      if new ~= 0;
         a = size(Apical);
         c = size(add);
         if a(2) >= c(2);
             add = [add zeros(1,a(2)-c(2))];
         elseif c(2) > a(2);
            Apical = [Apical zeros(a(1),c(2)-a(2))];
         end

         Apical = [Apical; add];

      end

      order = order+1;
      index = new;
      branch_index = branch_i;
   end
end

Basal = [];
branch_b = 0;
total_length_b = 0;

for l = (1:prime_d);
   %add = [numel rad r z length];
   add = [1 0 0 depth 5*para(1,2)];
   Basal = [Basal add];

   %branch_b = branch_b+1;
end
```

```
index = 5*(prime_d);
order = 1;
branch_index = index;

while branch_index > 0;
   add = [];
   new = 0;
   branch_i = 0;
   %disp(branch_index)
   for m = 5:5:index;
      branch = randi([1,100],1,1);
      if Basal(order,m) >= para(1,2) && branch >= b;
            %disp('poop')
             %add = [numel rad r z length];
            slice = (para(1,2)-Basal(order,m)).*rand(1,1)+Basal(order,m);
            Basal(order,m-3) = (((((3*(.45^2)*(Basal(order,m)-slice))/4)^(1/3))-slice).*rand(1,1)+slice);
            l_1 = (para(1,2)-((4*para(1,2))/(order+1))).*rand(1,1)+((4*para(1,2))/(order+1));

            rad1 = (((((3*(.45^2)*(Basal(order,m)))/4)^(1/3))...
                  -slice).*rand(1,1)+slice);
            rad2 = (((((3*(.45^2)*(Basal(order,m)))/4)^(1/3))...
                  -l_1).*rand(1,1)+l_1);
            theta = ((-(pi/2)) - (pi/2)).*rand(1,1)+(pi/2);
            z = Basal(order,m-1) + (Basal(order,m-3)*sin(theta));
            phi = (0 - (2*pi)).*rand(1,1)+(pi*2);
            x = cos(theta) * (Basal(order,m-3)*sin(theta));
            y = sin(abs(pi - phi))*x;
            zeta = abs(atan(y/Basal(order,m-3)));
            r = abs(y/sin(zeta));

            branch_1 = [Basal(order,m-4)+1 rad1 r z l_1];
            branch_2 = [Basal(order,m-4)+1 rad2 r z (Basal(order,m)-slice)];
            add = [add branch_1 branch_2];
            Basal(order,m) = slice;


            new = new+(2*5);
            branch_i = branch_i + (2*5);

            branch_b = branch_b +1;

      elseif Basal(order,m) >= para(1,2) && branch < b;
            Basal(order,m-3) = (((((3*(.45^2)*(Basal(order,m)))/4)^(1/3))...
                  -Basal(order,m)).*rand(1,1)+Basal(order,m));
            add = [add Basal(order,m-4) Basal(order,m-3)...
               Basal(order,m-2) Basal(order,m-1) Basal(order,m)];
            Basal(order,m-4) = 0;
            Basal(order,m-3) = 0;
            Basal(order,m-2) = 0;
            Basal(order,m-1) = 0;
            Basal(order,m) = 0;
            new = new + 5;


      elseif Basal(order,m) < para(1,2);
```

```
                Basal(order,m-3) = (((((3*(.45^2)*(Basal(order,m)))/4)^(1/3))...
                    -Basal(order,m)).*rand(1,1)+Basal(order,m));

                 basal_endings = basal_endings+1;
        end
    end
    if branch_i == 0;
          for r = 2:5:length(add);
             if add(1,r) == 0;
                add(1,r) = (((((3*(.45^2)*(add(1,(r+3))))/4)^(1/3))...
                    -add(1,(r+3))).*rand(1,1)+add(1,(r+3)));
                total_length_b = total_length_b + add(1,(r+3));
                basal_endings = basal_endings+1;
             end
          end
     end
    if new ~= 0;
        a = size(Basal);
        c = size(add);
        if a(2) >= c(2);
            add = [add zeros(1,a(2)-c(2))];
        elseif c(2) > a(2);
            Basal = [Basal zeros(a(1),c(2)-a(2))];
        end

        Basal = [Basal; add];
    end

    order = order+1;
    index = new;
    branch_index = branch_i;
end

Axonal = [];

if (length_a+depth) >= 1800;
    top = (length_a+depth);
    bottom =  (length_a+depth) - length_ax;
elseif (length_a+depth) <= 1800;
    top = 1800;
    bottom =  1800 - length_ax;
end

total_length = 0;
branch_A = 0;
count = 0;
c = 1;
mean_1 = (top-Mean_1);
mean_2 = (top-Mean_2);
if weight_1 >= weight_2;
    density_max = (mean_1);
elseif weight_2 > weight_1;
    density_max = (mean_2);
end

for h = (density_max):-50:bottom;
```

```
   mean = density_eq(type, scale, sigma_1,sigma_2,mean_1,mean_2,weight_1,weight_2,h);
   num_branch = (mean*1000)/axon_radius;
      for q = 1:round(num_branch);

         %add = [numel rad r z length];
         r = (0 - axon_radius ).*rand(1,1)+(axon_radius);
         add = [c 0 r h axon_radius];
         Axonal = [Axonal add];
         count = count+1;
         c = c+1;
         branch_A = branch_A+.5;

      end
end
for h = (density_max):50:top;
   mean = density_eq(type, scale, sigma_1,sigma_2,mean_1,mean_2,weight_1,weight_2,h);
   num_branch = (mean*1000)/axon_radius;
      for q = 1:round(num_branch);
         r = (0 - axon_radius ).*rand(1,1)+(axon_radius);
         add = [c 0 r h axon_radius];
         Axonal = [Axonal add];
         count = count+1;
         c = c+1;
         branch_A = branch_A+.5;

      end
end

index = 5*(count);
order = 1;
branch_index = index;

while branch_index > 0;
   new = 0;
   add = [];
   branch_i = 0;
   for n = 5:5:index;
      branch = randi([1,100],1,1);
      if Axonal(order,n) >= para(1,3) && branch >= A;
         %disp('poop')
         slice = (para(1,3)-Axonal(order,n)).*rand(1,1)+Axonal(order,n);
         Axonal(order,n-3) = (((((3*(.15^2)*(Axonal(order,n)-slice))/4)^(1/3))-slice).*rand(1,1)+slice);
         l_1 = (1/(cut*(order+1)))*axon_radius;

         theta = ((-(pi/2)) - (pi/2)).*rand(1,1)+(pi/2);
         z = Axonal(order,n-1) + (Axonal(order,n-3)*sin(theta));
         phi = (0 - (2*pi)).*rand(1,1)+(pi*2);
         x = cos(theta) * (Axonal(order,n-3)*sin(theta));
         y = sin(abs(pi - phi))*x;
         zeta = abs(atan(y/Axonal(order,n-3)));
         r = abs(y/sin(zeta));

         branch_1 = [Axonal(order,n-4) 0 r z l_1];
         branch_2 = [Axonal(order,n-4) 0 r z (Axonal(order,n)-slice)];
         add = [add branch_1 branch_2];
         Axonal(order,n) = slice;
```

```
        new = new + (2*5);
        branch_i = branch_i + (2*5);

        branch_A = branch_A+1;
        total_length = total_length +  branch_1(1,5) + branch_2(1,5);

    elseif Axonal(order,n) >= para(1,3) && branch < A;

          add = [add Axonal(order,n-4) Axonal(order,n-3)...
             Axonal(order,n-2) Axonal(order,n-1) Axonal(order,n)];
          Axonal(order,n-4) = 0;
          Axonal(order,n-3) = 0;
          Axonal(order,n-2) = 0;
          Axonal(order,n-1) = 0;
          Axonal(order,n) = 0;
          new = new + 5;



    elseif Axonal(order,n) < para(1,3);
             %add = [numel rad r z length];
             Axonal(order,n-3) = (((((3*(.15^2)*(Axonal(order,n)...
                -Axonal(order,n)))/4)^(1/3))-Axonal(order,n)).*rand(1,1)+Axonal(order,n));
             terminals = [terminals Axonal(order,n-3) Axonal(order,n-2) ...
                Axonal(order,n-1) Axonal(order,n)];
    end
  end
  if branch_i == 0;
       for r = 2:5:length(add);
          if add(1,r) == 0;
             add(1,r) = (((((3*(.15^2)*(add(1,(r+3))))/4)^(1/3))...
                -add(1,(r+3))).*rand(1,1)+ add(1,(r+3)));
             total_length = total_length + add(1,(r+3));
             terminals = [terminals add(1,r) add(1,r+1) ...
                add(1,r+2) add(1,r+3)];
          end
       end
   end
  if new ~= 0;
    a = size(Axonal);
    c = size(add);
    if a(2) >= c(2);
        add = [add zeros(1,a(2)-c(2))];
    elseif c(2) > a(2);
       Axonal = [Axonal zeros(a(1),c(2)-a(2))];
    end

    Axonal = [Axonal; add];
  end

  order = order+1;
  index = new;
  branch_index = branch_i;
end
```

```
% disp('apical_____')
% %disp(A_order)
% disp((total_length_a)/1000)
% disp(round(branch_a))

% disp('basal_____')
% % disp(B_order)
% disp((total_length_b)/1000)
% disp(round(branch_b))

% disp('axonal_____')
% disp(size(Axonal))
% disp((total_length)/1000)
% disp(round(branch_A))

end
```

2. Analysis

‘generator_anlysis’

```
One = [];

Two = [];

Three = [];

Four = [];

Five = [];

Six = [];

Seven = [];

Eight = [];

Nine = [];

Ten  = [];

Eleven = [];

Twelve = [];

Thirt = [];

Fourt = [];

Fift = [];

Sixt = [];

Sevent = [];

Eightt = [];

Ninet = [];

Twenty = [];


for gen = 1:10;
```

```
neruon_generator

a = size(Apical);
c = size(Basal);

if a(2) >= c(2);
    Basal = [Basal zeros(c(1),a(2)-c(2))];
elseif c(2) > a(2);
   Apical = [Apical zeros(a(1),c(2)-a(2))];
end



dendrites = [Apical; Basal];
DendriteSize = size(dendrites);
Dendrite = [];
for l2 = 1:1:DendriteSize(1);
   for L2 = 1:5:DendriteSize(2);
      if dendrites(l2,L2) ~= 0 && dendrites(l2,L2+1) ~= 0;
         Dendrite = [Dendrite ; dendrites(l2,L2)...
            dendrites(l2,L2+1) dendrites(l2,L2+2) dendrites(l2,L2+3)...
            dendrites(l2,L2+4)];
      end
   end
end

%    Axsize = size(Axonal);
%    Axon = [];
%    for l2 = 1:1:Axsize(1);
%       for L2 = 1:5:Axsize(2);
%          if Axonal(l2,L2) ~= 0;
%             Axon = [Axon ; Axonal(l2,L2)...
%                Axonal(l2,L2+1) Axonal(l2,L2+2) Axonal(l2,L2+3)...
%                Axonal(l2,L2+4)];
%          end
%       end
%    end
```

```
new = [];
size1 = size(Dendrite);
%   size1 = size(Axon);
for i = 2:size1;
    density = (pi*(.45^2)*Dendrite(i,5))/((4/3)*pi*(Dendrite(i,2)));
    add = [Dendrite(i,:) density];
%       density = (pi*(.15^2)*Axon(i,5))/((4/3)*pi*(Axon(i,2)));
%       add = [Axon(i,:) density];
    new = [new ; add];

end

size2 = size(new);

X = [];
Y = [];
Z = [];
for j = 1:size2;

    X = [X ; new(j,3)];
    Y = [Y ; new(j,4)];
    Z = [Z ; new(j,6)];
end

B = sort(Z);
L = size(B);
l = L(1);
count = 0;
if l >= 20;
count = count+1;
    for i = 1:L;
        if B(l) == Z(i);
            One = [One ; X(i) Y(i) Z(i)];
        end
        if B(l-1) == Z(i);
            Two = [Two; X(i) Y(i) Z(i)];
        end
```

```
if B(l-2) == Z(i);
    Three = [Three; X(i) Y(i) Z(i)];
end
if B(l-3) == Z(i);
    Four = [Four; X(i) Y(i) Z(i)];
end
if B(l-4) == Z(i);
    Five = [Five; X(i) Y(i) Z(i)];
end
if B(l-5) == Z(i);
    Six = [Six; X(i) Y(i) Z(i)];
end
if B(l-6) == Z(i);
    Seven = [Seven; X(i) Y(i) Z(i)];
end
if B(l-7) == Z(i);
    Eight = [Eight; X(i) Y(i) Z(i)];
end
if B(l-8) == Z(i);
    Nine = [Nine; X(i) Y(i) Z(i)];
end
if B(l-9) == Z(i);
    Ten = [Ten ; X(i) Y(i) Z(i)];
end
if B(l-10) == Z(i);
    Eleven = [Eleven; X(i) Y(i) Z(i)];
end
if B(l-11) == Z(i);
    Twelve = [Twelve; X(i) Y(i) Z(i)];
end
if B(l-12) == Z(i);
    Thirt = [Thirt; X(i) Y(i) Z(i)];
end
if B(l-13) == Z(i);
    Fourt = [Fourt; X(i) Y(i) Z(i)];
end
if B(l-14) == Z(i);
```

```
            Fift = [Fift; X(i) Y(i) Z(i)];
        end
        if B(l-15) == Z(i);
            Sixt = [Sixt; X(i) Y(i) Z(i)];
        end
        if B(l-16) == Z(i);
            Sevent = [Sevent; X(i) Y(i) Z(i)];
        end
        if B(l-17) == Z(i);
            Eightt = [Eightt; X(i) Y(i) Z(i)];
        end
        if B(l-18) == Z(i);
            Ninet = [Ninet; X(i) Y(i) Z(i)];
        end
        if B(l-19) == Z(i);
            Twenty = [Twenty; X(i) Y(i) Z(i)];
        end

    end

  end
end

  [one,two,three,four,five,six,seven,eight,nine,ten,...
  eleven,twelve,thirt,fourt,fift,sixt,sevent,eightt,ninet,twenty,...
  diff1, diff2, diff3, diff4, diff5, diff6, diff7, diff8, diff9, diff10,...
  diff11,diff12,diff13,diff14,diff15,diff16,diff17,diff18,diff19,diff20]...
  = averager(One,Two,Three,Four, Five,Six,Seven,Eight,Nine,Ten,...
  Eleven,Twelve,Thirt,Fourt,Fift,Sixt,Sevent,Eightt,Ninet,Twenty,count);

  sig = zeros(20,1);

  sig(1) = 100; % sqrt((1/20)*diff1);
  sig(2) = 100; % sqrt((1/20)*diff2);
  sig(3) = 100; % sqrt((1/20)*diff3);
  sig(4) = 100; % sqrt((1/20)*diff4);
  sig(5) = 100; % sqrt((1/20)*diff5);
```

```
sig(6) = 100; % sqrt((1/20)*diff6);
sig(7) = 100; % sqrt((1/20)*diff7);
sig(8) = 100; % sqrt((1/20)*diff8);
sig(9) = 100; % sqrt((1/20)*diff9);
sig(10) = 100; %  sqrt((1/20)*diff10);
sig(11) = 100; % sqrt((1/20)*diff11);
sig(12) = 100; % sqrt((1/20)*diff12);
sig(13) = 100; % sqrt((1/20)*diff13);
sig(14) = 100; % sqrt((1/20)*diff14);
sig(15) = 100; % sqrt((1/20)*diff15);
sig(16) = 100; % sqrt((1/20)*diff16);
sig(17) = 100; % sqrt((1/20)*diff17);
sig(18) = 100; % sqrt((1/20)*diff18);
sig(19) = 100; % sqrt((1/20)*diff19);
sig(20)= 100; % sqrt((1/20)*diff20);

w = zeros(20,1);

w(1) = one(3)*10^5;
w(2) = two(3)*10^5;
w(3) = three(3)*10^5;
w(4) = four(3)*10^5;
w(5) = five(3)*10^5;
w(6) = six(3)*10^5;
w(7) = seven(3)*10^5;
w(8) = eight(3)*10^5;
w(9) = nine(3)*10^5;
w(10) = ten(3)*10^5;
w(11) = eleven(3)*10^5;
w(12) = twelve(3)*10^5;
w(13) = thirt(3)*10^5;
w(14) = fourt(3)*10^5;
w(15) = fift(3)*10^5;
w(16) = sixt(3)*10^5;
w(17) = sevent(3)*10^5;
w(18) = eightt(3)*10^5;
w(19) = ninet(3)*10^5;
```

```
   w(20) = twenty(3)*10^5;

%   scatter3(X,Y,Z,'r')
%
%    hold on
%
   [x,y] = meshgrid([0:10:3000]);
   z = ((w(1)*(1/(2*pi*sig(1)^2))*exp(-((x-one(1)).^2+(y-one(2)).^2)/(2*sig(1)^2))...
   +  (w(2)*(1/(2*pi*sig(2)^2))*exp(-((x-two(1)).^2+(y-two(2)).^2)/(2*sig(2)^2)))...
   + (w(3)*(1/(2*pi*sig(3)^2))*exp(-((x-three(1)).^2+(y-three(2)).^2)/(2*sig(3)^2)))...
   + (w(4)*(1/(2*pi*sig(4)^2))*exp(-((x-four(1)).^2+(y-four(2)).^2)/(2*sig(4)^2)))...
   + (w(5)*(1/(2*pi*sig(5)^2))*exp(-((x-five(1)).^2+(y-five(2)).^2)/(2*sig(5)^2)))...
   + (w(6)*(1/(2*pi*sig(6)^2))*exp(-((x-six(1)).^2+(y-six(2)).^2)/(2*sig(6)^2)))...
   + (w(7)*(1/(2*pi*sig(7)^2))*exp(-((x-seven(1)).^2+(y-seven(2)).^2)/(2*sig(7)^2)))...
   + (w(8)*(1/(2*pi*sig(8)^2))*exp(-((x-eight(1)).^2+(y-eight(2)).^2)/(2*sig(8)^2)))...
   + (w(9)*(1/(2*pi*sig(9)^2))*exp(-((x-nine(1)).^2+(y-nine(2)).^2)/(2*sig(9)^2)))...
   + (w(10)*(1/(2*pi*sig(10)^2))*exp(-((x-ten(1)).^2+(y-ten(2)).^2)/(2*sig(10)^2)))...
   + (w(11)*(1/(2*pi*sig(11)^2))*exp(-((x-eleven(1)).^2+(y-eleven(2)).^2)/(2*sig(11)^2)))...
   + (w(12)*(1/(2*pi*sig(12)^2))*exp(-((x-twelve(1)).^2+(y-twelve(2)).^2)/(2*sig(12)^2)))...
   + (w(13)*(1/(2*pi*sig(13)^2))*exp(-((x-thirt(1)).^2+(y-thirt(2)).^2)/(2*sig(13)^2)))...
   + (w(14)*(1/(2*pi*sig(14)^2))*exp(-((x-fourt(1)).^2+(y-fourt(2)).^2)/(2*sig(14)^2)))...
   + (w(15)*(1/(2*pi*sig(15)^2))*exp(-((x-fift(1)).^2+(y-fift(2)).^2)/(2*sig(15)^2)))...
   + (w(16)*(1/(2*pi*sig(16)^2))*exp(-((x-sixt(1)).^2+(y-sixt(2)).^2)/(2*sig(16)^2)))...
   + (w(17)*(1/(2*pi*sig(17)^2))*exp(-((x-sevent(1)).^2+(y-sevent(2)).^2)/(2*sig(17)^2)))...
   + (w(18)*(1/(2*pi*sig(18)^2))*exp(-((x-eightt(1)).^2+(y-eightt(2)).^2)/(2*sig(18)^2)))...
   + (w(19)*(1/(2*pi*sig(19)^2))*exp(-((x-ninet(1)).^2+(y-ninet(2)).^2)/(2*sig(19)^2)))...
   + (w(20)*(1/(2*pi*sig(20)^2))*exp(-((x-twenty(1)).^2+(y-twenty(2)).^2)/(2*sig(20)^2)))...
   )/20);



   h = surf(x,y,z);
   set(h, 'edgecolor','none')

   xlabel('r')
   ylabel('h')
   zlabel('density')
```

```
%    %Weights_____________________________________________________
% disp('weights')
%    disp(w(1))
%    disp(w(2))
%    disp(w(3))
%    disp(w(4))
%    disp(w(5))
%    disp(w(6))
%    disp(w(7))
%    disp(w(8))
%    disp(w(9))
%    disp(w(10))
%    disp(w(11))
%    disp(w(12))
%    disp(w(13))
%    disp(w(14))
%    disp(w(15))
%    disp(w(16))
%    disp(w(17))
%    disp((w(18)))
%    disp(w(19))
%    disp(w(20))
%
% %___________________mur and muh_____________________________
% disp('mur and muh')
% disp(one)
% disp(two)
% disp(three)
% disp(four)
% disp(five)
% disp(six)
% disp(seven)
% disp(eight)
% disp(nine)
```

```
% disp(ten)
% disp(eleven)
% disp(twelve)
% disp(thirt)
% disp(fourt)
% disp(fift)
% disp(sixt)
% disp(sevent)
% disp(eightt)
% disp(ninet)
% disp(twenty)
%
% tot = Prob_write(w,sig, one, two, three, four, five,...
%     six,seven,eight,nine,ten,eleven,twelve,thirt,fourt,fift,sixt,sevent,...
%     eightt,ninet,twenty);
%
% disp(tot)
```

‘Prob_write’

```
%this script solves the integration of the density equations
function [tot] = Prob_write(w,sig, one, two, three, four, five,...
    six,seven,eight,nine,ten,eleven,twelve,thirt,fourt,fift,sixt,sevent,...
    eightt,ninet,twenty)

    sum_w = 0;
    for  i = 1:20;
        sum_w = sum_w + w(i);
    end

    eps = (1/(20*100^2));

    %one_____________________
    A1 = (-(100^2)*exp(-((1800-one(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-one(1))^2)/(2*100^2)));

    B1 = (one(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-one(1))/(2^(1/2))*100)))...
        -(((1/2)*(pi^(1/2))*erf((-one(1))/(2^(1/2))*100)));

    C1 = (((pi^(1/2))*exp((-(one(2)^2)/(2*100^2))-((((2*one(2))/(2*100^2))^2)...
        /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*one(2))/...
        (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
        -(((pi^(1/2))*exp((-(one(2)^2)/(2*100^2))-((((2*one(2))/(2*100^2))^2)...
        /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*one(2))/...
        (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

    ONE = (A1+B1)*C1*w(1);
```

```
%two_____________________
A2 = (-(100^2)*exp(-((1800-two(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-two(1))^2)/(2*100^2)));

B2 = (two(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-two(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-two(1))/(2^(1/2))*100)));

C2 = (((pi^(1/2))*exp((-(two(2)^2)/(2*100^2))-((((2*two(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*two(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(two(2)^2)/(2*100^2))-((((2*two(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*two(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

TWO = (A2+B2)*C2*w(2);

%three___________________
A3 = (-(100^2)*exp(-((1800-three(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-three(1))^2)/(2*100^2)));

B3 = (three(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-three(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-three(1))/(2^(1/2))*100)));

C3 = (((pi^(1/2))*exp((-(three(2)^2)/(2*100^2))-((((2*three(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*three(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(three(2)^2)/(2*100^2))-((((2*three(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*three(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

THREE = (A3+B3)*C3*w(3);

%four____________________
A4 = (-(100^2)*exp(-((1800-four(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-four(1))^2)/(2*100^2)));

B4 = (four(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-four(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-four(1))/(2^(1/2))*100)));

C4 = (((pi^(1/2))*exp((-(four(2)^2)/(2*100^2))-((((2*four(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*four(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(four(2)^2)/(2*100^2))-((((2*four(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*four(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

FOUR = (A4+B4)*C4*w(4);

%five____________________
A5 = (-(100^2)*exp(-((1800-five(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-five(1))^2)/(2*100^2)));

B5 = (five(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-five(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-five(1))/(2^(1/2))*100)));

C5 = (((pi^(1/2))*exp((-(five(2)^2)/(2*100^2))-((((2*five(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*five(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(five(2)^2)/(2*100^2))-((((2*five(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*five(2))/...
```

```
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

FIVE = (A5+B5)*C5*w(5);

%six_____________________________________________________________
A6 = (-(100^2)*exp(-((1800-six(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-six(1))^2)/(2*100^2)));

B6 = (six(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-six(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-six(1))/(2^(1/2))*100)));

C6 = (((pi^(1/2))*exp((-(six(2)^2)/(2*100^2))-((((2*six(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*six(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(six(2)^2)/(2*100^2))-((((2*six(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*six(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

SIX = (A6+B6)*C6*w(6);

%seven___________________________________________________________

A7 = (-(100^2)*exp(-((1800-seven(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-seven(1))^2)/(2*100^2)));

B7 = (seven(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-seven(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-seven(1))/(2^(1/2))*100)));

C7 = (((pi^(1/2))*exp((-(seven(2)^2)/(2*100^2))-((((2*seven(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*seven(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(seven(2)^2)/(2*100^2))-((((2*seven(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*seven(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

SEVEN = (A7+B7)*C7*w(7);

%eight___________________________________________________________

A8 = (-(100^2)*exp(-((1800-eight(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-eight(1))^2)/(2*100^2)));

B8 = (eight(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-eight(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-eight(1))/(2^(1/2))*100)));

C8 = (((pi^(1/2))*exp((-(eight(2)^2)/(2*100^2))-((((2*eight(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*eight(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(eight(2)^2)/(2*100^2))-((((2*eight(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*eight(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

EIGHT = (A8+B8)*C8*w(8);

%nine____________________________________________________________

A9 = (-(100^2)*exp(-((1800-nine(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-nine(1))^2)/(2*100^2)));

B9 = (nine(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-nine(1))/(2^(1/2))*100)))...
```

```
   -(((1/2)*(pi^(1/2))*erf((-nine(1))/(2^(1/2))*100)));

C9 = (((pi^(1/2))*exp((-(nine(2)^2)/(2*100^2))-((((2*nine(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*nine(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(nine(2)^2)/(2*100^2))-((((2*nine(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*nine(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

NINE = (A9+B9)*C9*w(9);

%ten_____________________________________________________________

A10 = (-(100^2)*exp(-((1800-ten(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-ten(1))^2)/(2*100^2)));

B10 = (ten(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-ten(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-ten(1))/(2^(1/2))*100)));

C10 = (((pi^(1/2))*exp((-(ten(2)^2)/(2*100^2))-((((2*ten(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*ten(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(ten(2)^2)/(2*100^2))-((((2*ten(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*ten(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

TEN = (A10+B10)*C10*w(10);

%eleven__________________________________________________________
A11 = (-(100^2)*exp(-((1800-eleven(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-eleven(1))^2)/(2*100^2)));

B11 = (eleven(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-eleven(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-eleven(1))/(2^(1/2))*100)));

C11 = (((pi^(1/2))*exp((-(eleven(2)^2)/(2*100^2))-((((2*eleven(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*eleven(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(eleven(2)^2)/(2*100^2))-((((2*eleven(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*eleven(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

ELEVEN = (A11+B11)*C11*w(11);

%twelve__________________________________________________________
A12 = (-(100^2)*exp(-((1800-twelve(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-twelve(1))^2)/(2*100^2)));

B12 = (one(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-twelve(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-twelve(1))/(2^(1/2))*100)));

C12 = (((pi^(1/2))*exp((-(twelve(2)^2)/(2*100^2))-((((2*twelve(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*twelve(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(twelve(2)^2)/(2*100^2))-((((2*twelve(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*twelve(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

TWELVE = (A12+B12)*C12*w(12);
```

```
%thirt_____________________________________________________________
A13 = (-(100^2)*exp(-((1800-thirt(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-thirt(1))^2)/(2*100^2)));

B13 = (thirt(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-thirt(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-thirt(1))/(2^(1/2))*100)));

C13 = (((pi^(1/2))*exp((-(thirt(2)^2)/(2*100^2))-((((2*thirt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*thirt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(thirt(2)^2)/(2*100^2))-((((2*thirt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*thirt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

THIRT = (A13+B13)*C13*w(13);

%fourt_____________________________________________________________
A14 = (-(100^2)*exp(-((1800-fourt(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-fourt(1))^2)/(2*100^2)));

B14 = (fourt(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-fourt(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-fourt(1))/(2^(1/2))*100)));

C14 = (((pi^(1/2))*exp((-(fourt(2)^2)/(2*100^2))-((((2*fourt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*fourt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(fourt(2)^2)/(2*100^2))-((((2*fourt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*fourt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

FOURT = (A14+B14)*C14*w(14);

%fift_____________________________________________________________
A15 = (-(100^2)*exp(-((1800-fift(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-fift(1))^2)/(2*100^2)));

B15 = (fift(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-fift(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-fift(1))/(2^(1/2))*100)));

C15 = (((pi^(1/2))*exp((-(fift(2)^2)/(2*100^2))-((((2*fift(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*fift(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(fift(2)^2)/(2*100^2))-((((2*fift(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*fift(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

FIFT = (A15+B15)*C15*w(15);

%sixt_____________________________________________________________
A16 = (-(100^2)*exp(-((1800-sixt(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-sixt(1))^2)/(2*100^2)));

B16 = (sixt(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-sixt(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-sixt(1))/(2^(1/2))*100)));

C16 = (((pi^(1/2))*exp((-(sixt(2)^2)/(2*100^2))-((((2*sixt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*sixt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(sixt(2)^2)/(2*100^2))-((((2*sixt(2))/(2*100^2))^2)...
```

```
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*sixt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

SIXT = (A16+B16)*C16*w(16);

%sevent_____________________________________________________________
A17 = (-(100^2)*exp(-((1800-sevent(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-sevent(1))^2)/(2*100^2)));

B17 = (sevent(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-sevent(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-sevent(1))/(2^(1/2))*100)));

C17 = (((pi^(1/2))*exp((-(sevent(2)^2)/(2*100^2))-((((2*sevent(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*sevent(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(sevent(2)^2)/(2*100^2))-((((2*sevent(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*sevent(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

SEVENT = (A17+B17)*C17*w(17);

%eightt_____________________________________________________________
A18 = (-(100^2)*exp(-((1800-eightt(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-eightt(1))^2)/(2*100^2)));

B18 = (eightt(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-eightt(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-eightt(1))/(2^(1/2))*100)));

C18 = (((pi^(1/2))*exp((-(eightt(2)^2)/(2*100^2))-((((2*eightt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*eightt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(eightt(2)^2)/(2*100^2))-((((2*eightt(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*eightt(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

EIGHTT = (A18+B18)*C18*w(18);

%ninet_____________________________________________________________
A19 = (-(100^2)*exp(-((1800-ninet(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-ninet(1))^2)/(2*100^2)));

B19 = (ninet(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-ninet(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-ninet(1))/(2^(1/2))*100)));

C19 = (((pi^(1/2))*exp((-(ninet(2)^2)/(2*100^2))-((((2*ninet(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*ninet(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
   -(((pi^(1/2))*exp((-(ninet(2)^2)/(2*100^2))-((((2*ninet(2))/(2*100^2))^2)...
   /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*ninet(2))/...
   (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

NINET = (A19+B19)*C19*w(19);

%twenty_____________________________________________________________

A20 = (-(100^2)*exp(-((1800-twenty(1))^2)/(2*100^2)))-(-(100^2)*exp(-((-twenty(1))^2)/(2*100^2)));

B20 = (twenty(1)*(2^(1/2)))*100*(((1/2)*(pi^(1/2))*erf((1800-twenty(1))/(2^(1/2))*100)))...
   -(((1/2)*(pi^(1/2))*erf((-twenty(1))/(2^(1/2))*100)));
```

```
   C20 = (((pi^(1/2))*exp((-(twenty(2)^2)/(2*100^2))-((((2*twenty(2))/(2*100^2))^2)...
      /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*1800)+((2*twenty(2))/...
      (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))))...
      -(((pi^(1/2))*exp((-(twenty(2)^2)/(2*100^2))-((((2*twenty(2))/(2*100^2))^2)...
      /(4*(-1/(2*100^2)))))*erfi(((2*(-1/(2*100^2))*0)+((2*twenty(2))/...
      (2*100^2)))/(2*((-1/(2*100^2))^(1/2)))))/(2*((-1/(2*100^2))^(1/2))));

   TWENTY = (A20+B20)*C20*w(20);


tot = (ONE + TWO + THREE + FOUR + FIVE + SIX + SEVEN + EIGHT + NINE + TEN +...
   ELEVEN + TWELVE + THIRT + FOURT + FIFT + SIXT + SEVENT + EIGHTT + NINET +
TWENTY)/20;

end
```

Here I provide one example of a script containing a probability density function.

‘L2py’

```
function [probD, probA, Axterminals] = L2py(x,y)

probD = -10/log10((1/(8.6279e+10))*((9.2264e+05*(1/(2*pi*100^2))*exp(-((x-0).^2+(y-
1.9033*10^3).^2)/(2*100^2))...
  +  (2.8289e+05*(1/(2*pi*100^2))*exp(-((x-0.1069*10^3).^2+(y-1.9584*10^3).^2)/(2*100^2)))...
  + (2.2350e+05*(1/(2*pi*100^2))*exp(-((x-0).^2+(y-1.7710*10^3).^2)/(2*100^2)))...
  + (1.7744e+05*(1/(2*pi*100^2))*exp(-((x-0.0584*10^3).^2+(y-1.9647*10^3).^2)/(2*100^2)))...
  + (1.7207e+05*(1/(2*pi*100^2))*exp(-((x-0.0478*10^3).^2+(y-1.9286*10^3).^2)/(2*100^2)))...
  + (1.4022e+05*(1/(2*pi*100^2))*exp(-((x-0.0150*10^3).^2+(y-1.7815*10^3).^2)/(2*100^2)))...
  + (1.3144e+05*(1/(2*pi*100^2))*exp(-((x-0.0106*10^3).^2+(y-1.9409*10^3).^2)/(2*100^2)))...
  + (1.2717e+05*(1/(2*pi*100^2))*exp(-((x-0.0223*10^3).^2+(y-1.9755*10^3).^2)/(2*100^2)))...
  + (1.2146e+05*(1/(2*pi*100^2))*exp(-((x-0.0552*10^3).^2+(y-1.8952*10^3).^2)/(2*100^2)))...
  + (1.1901e+05*(1/(2*pi*100^2))*exp(-((x-0*10^3).^2+(y-1.9648*10^3).^2)/(2*100^2)))...
  + (1.1714e+05*(1/(2*pi*100^2))*exp(-((x-0.0312*10^3).^2+(y-1.9408*10^3).^2)/(2*100^2)))...
  + (1.0851e+05*(1/(2*pi*100^2))*exp(-((x-0.0434*10^3).^2+(y-2.0059*10^3).^2)/(2*100^2)))...
  + (9.7069e+04*(1/(2*pi*100^2))*exp(-((x-0.1391*10^3).^2+(y-1.9310*10^3).^2)/(2*100^2)))...
  + (9.6446e+04*(1/(2*pi*100^2))*exp(-((x-0).^2+(y-1.9648*10^3).^2)/(2*100^2)))...
  + (9.6324e+04*(1/(2*pi*100^2))*exp(-((x-0.0592*10^3).^2+(y-1.9384*10^3).^2)/(2*100^2)))...
  + (9.4100e+04*(1/(2*pi*100^2))*exp(-((x-0).^2+(y-1.9648*10^3).^2)/(2*100^2)))...
  + (8.2197e+04*(1/(2*pi*100^2))*exp(-((x-0).^2+(y-1.9648*10^3).^2)/(2*100^2)))...
  + (8.1885e+04*(1/(2*pi*100^2))*exp(-((x-0*10^3).^2+(y-1.7710*10^3).^2)/(2*100^2)))...
  + (7.9498e+04*(1/(2*pi*100^2))*exp(-((x-0*10^3).^2+(y-1.7710*10^3).^2)/(2*100^2)))...
  + (1.2146e+05*(1/(2*pi*100^2))*exp(-((x-0.0552*10^3).^2+(y-1.8952*10^3).^2)/(2*100^2)))...
   )/20));


probA = -10/log10((1/(3.1628e+11))*((5.2239e+05*(1/(2*pi*100^2))*exp(-((x-0.0522*10^3).^2+(y-
1.7033*10^3).^2)/(2*100^2))...
   +  (1.1500e+05*(1/(2*pi*100^2))*exp(-((x-0.0845*10^3).^2+(y-1.7400*10^3).^2)/(2*100^2)))...
   + (1.0816e+05*(1/(2*pi*100^2))*exp(-((x-0.0305*10^3).^2+(y-1.1594*10^3).^2)/(2*100^2)))...
   + (8.1223e+04*(1/(2*pi*100^2))*exp(-((x-0.0989*10^3).^2+(y-1.5363*10^3).^2)/(2*100^2)))...
   + (7.7952e+04*(1/(2*pi*100^2))*exp(-((x-0.0939*10^3).^2+(y-1.8570*10^3).^2)/(2*100^2)))...
```

```
+ (7.5921e+04*(1/(2*pi*100^2))*exp(-((x-234.5335).^2+(y-730.7494).^2)/(2*100^2)))...
+ (7.4613e+04*(1/(2*pi*100^2))*exp(-((x-59.0885).^2+(y-889.9172).^2)/(2*100^2)))...
+ (7.2115e+04*(1/(2*pi*100^2))*exp(-((x-0.1107*10^3).^2+(y-1.8109*10^3).^2)/(2*100^2)))...
+ (7.0458e+04*(1/(2*pi*100^2))*exp(-((x-0.0488*10^3).^2+(y-1.7622*10^3).^2)/(2*100^2)))...
+ (5.7876e+04*(1/(2*pi*100^2))*exp(-((x-0.0726*10^3).^2+(y-2.3051*10^3).^2)/(2*100^2)))...
+ (5.5776e+04*(1/(2*pi*100^2))*exp(-((x-0.4125*10^3).^2+(y-1.8668*10^3).^2)/(2*100^2)))...
+ (5.5468e+04*(1/(2*pi*100^2))*exp(-((x-0.6140*10^3).^2+(y-2.2923*10^3).^2)/(2*100^2)))...
+ (5.3986e+04*(1/(2*pi*100^2))*exp(-((x-0.5510*10^3).^2+(y-1.7891*10^3).^2)/(2*100^2)))...
+ (5.1490e+04*(1/(2*pi*100^2))*exp(-((x-186.7996).^2+(y-992.4296).^2)/(2*100^2)))...
+ (4.2583e+04*(1/(2*pi*100^2))*exp(-((x-0.0218*10^3).^2+(y-1.8891*10^3).^2)/(2*100^2)))...
+ (4.1880e+04*(1/(2*pi*100^2))*exp(-((x-0.1356*10^3).^2+(y-1.0125*10^3).^2)/(2*100^2)))...
+ (3.9746e+04*(1/(2*pi*100^2))*exp(-((x-0.0635*10^3).^2+(y-1.4763*10^3).^2)/(2*100^2)))...
+ (3.6422e+04*(1/(2*pi*100^2))*exp(-((x-0.0099*10^3).^2+(y-1.6842*10^3).^2)/(2*100^2)))...
+ (3.5137e+04*(1/(2*pi*100^2))*exp(-((x-0.5188*10^3).^2+(y-1.1391*10^3).^2)/(2*100^2)))...
+ (7.0458e+04*(1/(2*pi*100^2))*exp(-((x-0.0488*10^3).^2+(y-1.7622*10^3).^2)/(2*100^2)))...
)/20));

%1.0e+03 *
Axterminals = [  0.0307    0.5996;
2.0799    0.0315;
0.0340    0.1788;
0.9117    0.0473;
0.0266    0.4101;
1.2544    0.0573;
0.0508    0.6773;
1.1258    0.0576;
0.0000    0.3699;
1.4963    0.0899;
0.0102    0.0692;
1.4235    0.0197;
0.0103    0.0370;
1.4583    0.0403;
0.0137    0.0664;
1.5619    0.0450;
0.0018    0.0771;
1.6323    0.0095;
0.0061    0.0881;
1.2381    0.0069;
0.0217    0.0545;
1.4841    0.0338;
0.0115    0.1294;
2.2080    0.0168;
0.0161    0.1333;
1.3307    0.0183;
0.0350    0.1183;
1.5010    0.0677;
0.0065    0.0818;
1.6546    0.0601;
0.0237    0.2995;
1.8446    0.0728;
0.0077    0.0550;
1.5045    0.0150;
0.0158    0.0500;
1.5189    0.0615;
0.0209    0.0911;
0.9767    0.0593;
```

0.0060 0.0464;
0.9138 0.0375;
0.0203 0.0698;
0.7655 0.0401;
0.0271 0.0738;
0.8012 0.0461;
0.0136 0.0634;
1.2321 0.0508;
0.0179 0.0226;
1.1465 0.0271;
0.0234 0.0983;
0.8702 0.0387;
0.0261 0.0825;
0.6631 0.0554;
0.0591 0.0358;
0.3894 0.0658;
0.0175 0.0164;
0.5755 0.0221;
0.0113 0.1166;
1.7445 0.0257;
0.0770 0.1792;
1.7996 0.0781;
0.0223 0.0060;
1.6002 0.0349;
0.0207 0.0088;
1.6263 0.0482;
0.0172 0.0316;
1.5800 0.0460;
0.0416 0.0923;
1.4942 0.0558;
0.0585 0.0830;
1.7466 0.0612;
0.0139 0.0182;
1.5629 0.0388;
0.0113 0.3187;
1.7970 0.0375;
0.0238 0.0033;
1.6044 0.0440;
0.0101 0.0720;
1.4418 0.0476;
0.0053 0.0629;
1.6678 0.0218;
0.0212 0.0868;
1.6019 0.0611;
0.0023 0.0827;
1.7517 0.0029;
0.0177 0.0255;
1.4648 0.0780;
0.0024 0.0255;
1.4648 0.0035;
0.0487 0.0535;
1.4782 0.0780;
0.0033 0.0535;
1.4782 0.0146;
0.0095 0.0247;
1.5494 0.0780;

0.0056 0.0247;
1.5494 0.0123;
0.0557 0.0156;
1.5384 0.0780;
0.0011 0.0156;
1.5384 0.0026;
0.0039 0.1443;
1.6454 0.0780;
0.0628 0.2309;
1.5696 0.0780;
0.0177 0.0661;
1.6458 0.0780;
0.0074 0.0661;
1.6458 0.0093;
0.0100 0.0145;
1.2462 0.0780;
0.0010 0.0145;
1.2462 0.0027;
0.0734 0.0221;
1.3218 0.0780;
0.0551 0.0822;
1.4268 0.0780;
0.0023 0.0822;
1.4268 0.0073;
0.0185 0.1757;
1.5517 0.0780;
0.0275 0.0802;
2.2119 0.0780;
0.0068 0.0802;
2.2119 0.0071;
0.0510 0.0015;
1.3322 0.0780;
0.0005 0.0015;
1.3322 0.0007;
0.0315 0.0889;
1.5751 0.0780;
0.0000 0.0889;
1.5751 0.0000;
0.0495 0.0765;
1.7682 0.0780;
0.0000 0.0765;
1.7682 0.0081;
0.0425 0.0311;
1.5701 0.0780;
0.0105 0.0311;
1.5701 0.0131;
0.0413 0.0263;
1.4477 0.0780;
0.0044 0.0263;
1.4477 0.0083;
0.0145 0.0551;
1.5406 0.0780;
0.0036 0.0551;
1.5406 0.0044;
0.0578 0.1944;
1.3122 0.0780;

0.0098 0.0539;
1.6368 0.0780;
0.0002 0.0539;
1.6368 0.0017;
0.0665 0.0475;
1.4963 0.0780;
0.0058 0.0475;
1.4963 0.0126;
0.0706 0.0730;
1.5648 0.0780;
0.0045 0.0730;
1.5648 0.0074;
0.0053 0.0213;
1.5724 0.0780;
0.0174 0.0213;
1.5724 0.0524;
0.0170 0.0380;
1.0004 0.0780;
0.0043 0.0380;
1.0004 0.0048;
0.0546 0.0767;
0.9074 0.0780;
0.0077 0.0767;
0.9074 0.0124;
0.0391 0.0078;
0.9333 0.0780;
0.0237 0.0507;
0.9401 0.0780;
0.0002 0.0507;
0.9401 0.0004;
0.0302 0.0454;
0.7483 0.0780;
0.0007 0.0454;
0.7483 0.0056;
0.0340 0.0295;
0.7805 0.0780;
0.0681 0.0249;
0.7806 0.0780;
0.0002 0.0249;
0.7806 0.0011;
0.0250 0.0061;
0.6506 0.0780;
0.0028 0.0061;
0.6506 0.0099;
0.0412 0.0926;
0.7274 0.0780;
0.0055 0.0926;
0.7274 0.0060;
0.0015 0.0581;
1.1745 0.0780;
0.0088 0.0581;
1.1745 0.0112;
0.0187 0.0860;
0.8848 0.0780;
0.0072 0.0860;
0.8848 0.0120;

0.0440 0.0835;
0.7501 0.0780;
0.0385 0.0835;
0.7501 0.0790;
0.0398 0.0240;
0.6781 0.0780;
0.0058 0.0240;
0.6781 0.0082;
0.0473 0.0807;
0.5178 0.0780;
0.0044 0.0807;
0.5178 0.0046;
0.0686 0.0220;
0.5705 0.0780;
0.0001 0.0220;
0.5705 0.0004;
0.0709 0.0881;
0.7784 0.0780;
0.0381 0.0881;
0.7784 0.0410;
0.0339 0.0988;
0.6019 0.0780;
0.0023 0.0988;
0.6019 0.0032;
0.0094 0.0452;
0.6579 0.0780;
0.0003 0.0452;
0.6579 0.0009;
0.0457 0.0143;
0.6245 0.0780;
0.0466 0.0143;
0.6245 0.0627;
0.0738 0.0433;
0.3569 0.0780;
0.0012 0.0433;
0.3569 0.0049;
0.0236 0.0828;
0.5863 0.0780;
0.0028 0.0828;
0.5863 0.0061;
0.0254 0.0586;
0.7737 0.0780;
0.0078 0.0586;
0.7737 0.0108;
0.0620 0.1004;
0.7600 0.0780;
0.0320 0.0385;
0.5452 0.0780;
0.0003 0.0385;
0.5452 0.0003;
0.0720 0.0225;
1.4714 0.0780;
0.0019 0.0225;
1.4714 0.0036;
0.0386 0.0865;
1.4485 0.0780;

0.0010 0.0865;
1.4485 0.0032;
0.0199 0.0630;
1.7857 0.0780;
0.0020 0.0630;
1.7857 0.0068;
0.0127 0.0101;
1.5945 0.0780;
0.0042 0.0101;
1.5945 0.0107;
0.0696 0.0251;
1.5910 0.0780;
0.0022 0.0251;
1.5910 0.0053;
0.0441 0.0723;
1.6525 0.0780;
0.0600 0.0220;
1.6044 0.0780;
0.0066 0.0220;
1.6044 0.0114;
0.0377 0.0148;
1.1739 0.0780;
0.0007 0.0148;
1.1739 0.0054;
0.0329 0.0693;
1.5543 0.0780;
0.0014 0.0693;
1.5543 0.0042;
0.0067 0.0989;
1.5835 0.0780;
0.0004 0.0989;
1.5835 0.0012;
0.0388 0.1006;
1.5822 0.0780;
0.0061 0.1006;
1.5822 0.0082;
0.0772 0.0351;
1.7394 0.0780;
0.0020 0.0351;
1.7394 0.0040;
0.0367 0.0381;
1.7160 0.0780;
0.0156 0.0087;
1.5715 0.0780;
0.0046 0.0087;
1.5715 0.0108;
0.0348 0.0846;
1.8590 0.0780;
0.0012 0.0846;
1.8590 0.0025;
0.0241 0.0969;
1.5445 0.0780;
0.0134 0.0807;
1.4084 0.0780;
0.0050 0.0807;
1.4084 0.0137;

0.0582 0.0773;
1.6829 0.0780;
0.0060 0.0773;
1.6829 0.0338;
0.0372 0.0008;
1.6019 0.0780;
0.0071 0.0008;
1.6019 0.0081;
0.0741 0.0271;
1.4174 0.0780;
0.0067 0.0271;
1.4174 0.0115;
0.0407 0.0552;
1.4288 0.0780;
0.0127 0.0552;
1.4288 0.0377;
0.0071 0.0403;
1.7242 0.0780;
0.0242 0.0403;
1.7242 0.0332;
0.0707 0.0940;
1.6577 0.0780;
0.0037 0.0940;
1.6577 0.0078;
0.0149 0.0835;
1.8377 0.0780;
0.0068 0.0835;
1.8377 0.0102;
0.0758 0.0604;
1.8136 0.0780;
0.0341 0.1976;
1.5308 0.0624;
0.0236 0.0274;
1.4686 0.0624;
0.0000 0.0274;
1.4686 0.0000;
0.0174 0.0564;
1.6492 0.0624;
0.0078 0.0564;
1.6492 0.0187;
0.0500 0.2930;
1.8587 0.0624;
0.0306 0.2930;
1.8587 0.0647;
0.0296 0.0140;
1.6957 0.0624;
0.0247 0.0140;
1.6957 0.0705;
0.0579 0.1273;
1.2680 0.0624;
0.0213 0.1769;
1.3918 0.0624;
0.0460 0.0423;
1.6820 0.0624;
0.0031 0.0423;
1.6820 0.0042;

0.0357 0.1967;
1.3727 0.0624;
0.0351 0.0188;
1.5228 0.0624;
0.0032 0.0188;
1.5228 0.0055;
0.0071 0.0460;
1.6039 0.0624;
0.0022 0.0460;
1.6039 0.0065;
0.0035 0.1925;
0.7430 0.0624;
0.0066 0.1925;
0.7430 0.0067;
0.0233 0.0430;
0.7575 0.0624;
0.0114 0.0430;
0.7575 0.0186;
0.0400 0.0154;
0.5469 0.0624;
0.0044 0.0601;
0.7315 0.0624;
0.0001 0.0601;
0.7315 0.0001;
0.0042 0.1607;
0.7464 0.0624;
0.0034 0.0168;
1.7566 0.0624;
0.0071 0.0168;
1.7566 0.0096;
0.0094 0.0908;
1.7309 0.0624;
0.0541 0.0908;
1.7309 0.0558;
0.0154 0.0925;
1.2131 0.0624;
0.0095 0.0925;
1.2131 0.0491;
0.0459 0.0224;
1.6273 0.0624;
0.0017 0.0224;
1.6273 0.0050;
0.0316 0.0788;
1.6799 0.0624;
0.0074 0.3350;
1.7212 0.0624;
0.0634 0.3350;
1.7212 0.0826;
0.0002 0.0083;
1.7205 0.0624;
0.0434 0.0083;
1.7205 0.0442;
0.0554 0.1081;
1.5564 0.0624;
0.0006 0.1081;
1.5564 0.0029;

0.0074 0.0100;
1.5459 0.0624;
0.0474 0.0673;
1.7581 0.0624;
0.0055 0.0673;
1.7581 0.0275;
0.0508 0.0472;
1.4654 0.0520;
0.0019 0.0472;
1.4654 0.0026;
0.0437 0.0672;
1.5297 0.0520;
0.0090 0.0672;
1.5297 0.0175;
0.0333 0.0880;
1.2332 0.0520;
0.0137 0.0880;
1.2332 0.0380;
0.0024 0.0959;
1.4002 0.0520;
0.0367 0.0959;
1.4002 0.0472;
0.0223 0.0617;
1.3064 0.0520;
0.0034 0.0617;
1.3064 0.0312;
0.0176 0.1078;
0.5456 0.0520;
0.0263 0.0138;
0.5331 0.0520;
0.0016 0.0138;
0.5331 0.0018;
0.0169 0.1438;
0.7185 0.0520;
0.0028 0.1438;
0.7185 0.0100;
0.0134 0.0444;
1.7050 0.0520;
0.0064 0.0444;
1.7050 0.0070;
0.0291 0.0181;
1.5342 0.0520;
0.0014 0.0181;
1.5342 0.0059;
0.0395 0.0115;
1.6792 0.0520;
0.0007 0.0115;
1.6792 0.0074;
0.0223 0.0242;
1.3693 0.0446;
0.0286 0.0242;
1.3693 0.0435;
0.0430 0.0796;
1.2368 0.0446;
0.0019 0.0796;
1.2368 0.0032;

```
   0.0138    0.0857;
   0.5271    0.0446;
   0.0237    0.0857;
   0.5271    0.0378];
end
```

“run_classical”

```
%this is the script to run the ode solver, considering only the kinetic
%equations.

global K XI

num_neurons = 3;
num_glia = 0;

neurons = [];
microglia = [];

N = [];
M = [];

for i = 1:num_neurons;
   [XO,Xi,k] = pick_initial(1,i);
   if i == 1;
      neurons = [XO Xi k];
   elseif i > 1;
      neurons(:,:,i) = [XO Xi k];
   end
end

for j = 1:num_glia;
   [XO,Xi,k] = pick_initial(2,j);
   if j == 1;
      microglia = [XO Xi k];
   elseif j > 1;
      microglia(:,:,j) = [XO Xi k];
   end
end

c = 10;   %proprotionality constant

for k = 1:10;

   %in order to run multiple matrices throught the ODE solver and update
   %variables shared by diffeent matrices the following iterates through
   %the previous time step's solved values to update concentrations for
   %the next iteration

   for l = 1:num_neurons;
      exosomePrPSc = 0;
      CCL2 = 0;
      CCL2m = 0;
```

```
    Na = 0;
    Ca = 0;
    glu = 0;
    BDNF = 0;
    NGF = 0;
    TNF = 0;
    IL1beta = 0;
    FasL = 0;
    CCL5  = 0;
    fractalkine = 0;
    UDP = 0;
    ATP = 0;
    IFNgamma = 0;
    CCL21 = 0;
    IL4 = 0;

    for m = 1:num_neurons;
      if m ~= l;
        if (neurons(14,1,m) - ((1/(c*(abs(l-m))))*neurons(14,1,m))) >= 0;
          exosomePrPSc = exosomePrPSc + (1/(c*(abs(l-m))))*neurons(14,1,m);
          neurons(14,1,m) = neurons(14,1,m) - ((1/(c*(abs(l-m))))*neurons(14,1,m));
        end
        if (neurons(335,1,m) - ((1/(c*(abs(l-m))))*neurons(335,1,m))) >= 0;
          CCL2 = CCL2 + (1/(c*(abs(l-m))))*neurons(335,1,m);
          neurons(335,1,m) = neurons(335,1,m) - ((1/(c*(abs(l-m))))*neurons(335,1,m));
        end
        if (neurons(331,1,m) - ((1/(c*(abs(l-m))))*neurons(331,1,m))) >= 0;
          Na = Na + (1/(c*(abs(l-m))))*neurons(335,1,m);
          neurons(331,1,m) = neurons(331,1,m) - ((1/(c*(abs(l-m))))*neurons(331,1,m));
        end
%           disp(size(neurons))
%           disp(m)
%           disp(neurons(735,1,m))
%           if (neurons(735,1,m) - ((1/c(c*(abs(l-m))))*neurons(735,1,m))) >= 0;
%             Ca = Ca + (1/(c*(abs(l-m))))*neurons(735,1,m);
%             neurons(735,1,m) = neurons(735,1,m) - ((1/(c*(abs(l-m))))*neurons(735,1,m));
%           end
        if (neurons(737,1,m) - ((1/(c*(abs(l-m))))*neurons(737,1,m))) >= 0;
          glu = glu + (1/(c*(abs(l-m))))*neurons(737,1,m);
          neurons(737,1,m) = neurons(737,1,m) - ((1/(c*(abs(l-m))))*neurons(737,1,m));
        end
        if (neurons(350,1,m) - ((1/(c*(abs(l-m))))*neurons(350,1,m))) >= 0;
          BDNF = BDNF + (1/(c*(abs(l-m))))*neurons(350,1,m);
          neurons(350,1,m) = neurons(350,1,m) - ((1/(c*(abs(l-m))))*neurons(350,1,m));
        end
        if (neurons(339,1,m) - ((1/(c*(abs(l-m))))*neurons(339,1,m))) >= 0;
          TNF = TNF + (1/(c*(abs(l-m))))*neurons(339,1,m);
          neurons(339,1,m) = neurons(339,1,m) - ((1/(c*(abs(l-m))))*neurons(339,1,m));
        end
        if (neurons(284,1,m) - ((1/(c*(abs(l-m))))*neurons(284,1,m))) >= 0;
          IL1beta = IL1beta + (1/(c*(abs(l-m))))*neurons(284,1,m);
          neurons(284,1,m) = neurons(284,1,m) - ((1/(c*(abs(l-m))))*neurons(284,1,m));
        end
        if (neurons(358,1,m) - ((1/(c*(abs(l-m))))*neurons(358,1,m))) >= 0;
          FasL = FasL + (1/(c*abs(l-m)))*neurons(358,1,m);
```

```
            neurons(358,1,m) = neurons(358,1,m) - ((1/(c*(abs(l-m))))*neurons(358,1,m));
          end

        end
      end
      for n = 1:num_glia
        if (microglia(744,1,n) - ((1/(c*(num_neurons+num_glia)))*microglia(744,1,n))) >= 0;
          CCL2 = CCL2 + (1/(c*(num_neurons+num_glia)))*microglia(744,1,n);
          microglia(744,1,n) = microglia(744,1,n) -
((1/(c*(num_neurons+num_glia)))*microglia(744,1,n));
        end
        if (microglia(742,1,n) - ((1/(c*(num_neurons)))*microglia(742,1,n))) >= 0;
          BDNF = BDNF + (1/(c*(num_neurons)))*microglia(742,1,n);
          microglia(742,1,n) = microglia(742,1,n) - ((1/(c*(num_neurons)))*microglia(742,1,n));
        end
        if (microglia(536,1,n) - ((1/(c*(num_neurons)))*microglia(536,1,n))) >= 0;
          NGF = NGF + (1/(c*(num_neurons)))*microglia(536,1,n);
          microglia(536,1,n) = microglia(536,1,n) - ((1/(c*(num_neurons)))*microglia(536,1,n));
        end
        if (microglia(745,1,n) - ((1/(c*(num_neurons+num_glia)))*microglia(745,1,n))) >= 0;
          TNF = TNF + (1/(c*(num_neurons+num_glia)))*microglia(745,1,n);
          microglia(745,1,n) = microglia(745,1,n) -
((1/(c*(num_neurons+num_glia)))*microglia(745,1,n));
        end
        if (microglia(746,1,n) - ((1/(c*(num_neurons)))*microglia(746,1,n))) >= 0;
          IL1beta = IL1beta + (1/(c*(num_neurons)))*microglia(746,1,n);
          microglia(746,1,n) = microglia(746,1,n) - ((1/(c*(num_neurons)))*microglia(746,1,n));
        end
        if (microglia(743,1,n) - ((1/(c*(num_neurons)))*microglia(743,1,n))) >= 0;
          FasL = FasL + (1/(c*(num_neurons)))*microglia(743,1,n);
          microglia(743,1,n) = microglia(743,1,n) - ((1/(c*(num_neurons)))*microglia(743,1,n));
        end
      end
      neurons(15,1,l) = neurons(15,1,l) + 0.5*exosomePrPSc;
      neurons(19,1,l) = neurons(15,1,l) + 0.5*exosomePrPSc;
      neurons(335,1,l) = neurons(335,1,l) + CCL2;
      neurons(331,1,l) = neurons(331,1,l) + Na;
      neurons(737,1,l) = neurons(737,1,l) + glu;
      neurons(735,1,l) = neurons(735,1,l) + Ca;
      neurons(350,1,l) = neurons(350,1,l) + BDNF;
      neurons(747,1,l) = neurons(747,1,l) + NGF;
      neurons(339,1,l) = neurons(339,1,l) + TNF;
      neurons(284,1,l) = neurons(284,1,l) + IL1beta;
      neurons(358,1,l) = neurons(358,1,l) + FasL;


    end
    for o = 1:num_glia;
      TNF = 0;
      CCL2 = 0;
      CCL5  = 0;
      fractalkine = 0;
      UDP = 0;
      ATP = 0;
      IFNgamma = 0;
```

```
    CCL21 = 0;
    IL4 = 0;

    for p = 1:num_neurons;
      if (neurons(339,1,p) - ((1/(c*num_glia))*neurons(339,1,p))) >= 0;
         TNF = TNF + (1/(c*num_glia))*neurons(339,1,p);
         neurons(339,1,p) = neurons(339,1,m) - ((1/(c*num_glia))*neurons(339,1,p));
      end
      if (neurons(335,1,p) - ((1/(c*num_glia))*neurons(335,1,p))) >= 0;
        CCL2 = CCL2 + (1/(c*num_glia))*neurons(335,1,p);
        neurons(335,1,p) = neurons(335,1,p) - ((1/(c*num_glia))*neurons(335,1,p));
      end
      if (neurons(348,1,p) - ((1/(c*num_glia))*neurons(348,1,p))) >= 0;
        CCL5 = CCL5 + (1/(c*num_glia))*neurons(348,1,p);
        neurons(348,1,p) = neurons(348,1,p) - ((1/(c*num_glia))*neurons(348,1,p));
      end
      if (neurons(748,1,p) - ((1/(c*num_glia))*neurons(748,1,p))) >= 0;
        fractalkine = fractalkine + (1/(c*num_glia))*neurons(748,1,p);
        neurons(748,1,p) = neurons(748,1,p) - ((1/(c*num_glia))*neurons(748,1,p));
      end
      if (neurons(207,1,p) - ((1/(c*num_glia))*neurons(207,1,p))) >= 0;
        UDP = UDP + (1/(c*num_glia))*neurons(207,1,p);
        neurons(207,1,p) = neurons(207,1,p) - ((1/(c*num_glia))*neurons(207,1,p));
      end
      if (neurons(208,1,p) - ((1/(c*num_glia))*neurons(208,1,p))) >= 0;
        ATP = ATP + (1/(c*num_glia))*neurons(208,1,p);
        neurons(208,1,p) = neurons(208,1,p) - ((1/(c*num_glia))*neurons(208,1,p));
      end
      if (neurons(346,1,p) - ((1/(c*num_glia))*neurons(346,1,p))) >= 0;
        CCL21 = CCL21 + (1/(c*num_glia))*neurons(346,1,p);
        neurons(346,1,p) = neurons(346,1,p) - ((1/(c*num_glia))*neurons(346,1,p));
      end
      if (neurons(354,1,p) - ((1/(c*num_glia))*neurons(354,1,p))) >= 0;
        IL4 = IL4 + (1/(c*num_glia))*neurons(354,1,p);
        neurons(354,1,p) = neurons(354,1,p) - ((1/(c*num_glia))*neurons(354,1,p));
      end
    end
    for q = 1: num_glia;
      if (microglia(745,1,q) - ((1/(c*(num_neurons+num_glia)))*microglia(745,1,q))) >= 0;
        TNF = TNF + (1/(c*(num_neurons+num_glia)))*microglia(745,1,q);
        microglia(745,1,q) = microglia(745,1,q) -
((1/(c*(num_neurons+num_glia)))*microglia(745,1,q));
      end
      if (microglia(744,1,q) - ((1/(c*(num_neurons+num_glia)))*microglia(744,1,q))) >= 0;
        CCL2 = CCL2 + (1/(c*(num_neurons+num_glia)))*microglia(744,1,q);
        microglia(744,1,q) = microglia(744,1,q) -
((1/(c*(num_neurons+num_glia)))*microglia(744,1,q));
      end
      if (microglia(738,1,q) - ((1/(c*(num_neurons+num_glia)))*microglia(738,1,q))) >= 0;
        IFNgamma = IFNgamma + (1/(c*(num_neurons+num_glia)))*microglia(738,1,q);
        microglia(738,1,q) = microglia(738,1,q) -
((1/(c*(num_neurons+num_glia)))*microglia(738,1,q));
      end
      if (microglia(739,1,q) - ((1/(c*(num_neurons+num_glia)))*microglia(739,1,q))) >= 0;
        IL4 = IL4 + (1/(c*(num_neurons+num_glia)))*microglia(739,1,q);
```

```
            microglia(739,1,q) = microglia(738,1,q) -
((1/(c*(num_neurons+num_glia)))*microglia(739,1,q));
          end
        end
        microglia(745,1,o) = microglia(745,1,o) + TNF;
        microglia(744,1,o) = microglia(744,1,o) + CCL2;
        microglia(751,1,o) = microglia(751,1,o) + CCL5;
        microglia(430,1,o) = microglia(430,1,o) + fractalkine;
        microglia(749,1,o) = microglia(749,1,o) + UDP;
        microglia(750,1,o) = microglia(750,1,o) + ATP;
        microglia(738,1,o) = microglia(738,1,o) + IFNgamma;
        microglia(752,1,o) = microglia(752,1,o) + CCL21;
        microglia(739,1,o) = microglia(739,1,o) + IL4;
    end
    n = [];
    m = [];
    for r = 1:num_neurons;
        XI = neurons(:,2,r);
        K = neurons(:,3,r);
        [t,X] = ode15s(@cy_equations,0:.1:1,neurons(:,1,r));
        for x = 1:760;
            neurons(x,1,r) = X(length(t),x);
        end
        if r == 1;
            n = X;
        elseif r > 1;
            n(:,:,r) = X;
        end
    end
    for s = 1:num_glia;
        XI = microglia(:,2,s);
        K = microglia(:,3,s);
        [t,Y] = ode15s(@cy_equations,0:1:1,microglia(:,1,s));
        m(:,:,s) = Y;
    end

    N = [N ; n];
    M = [M ; m];
End
```

“cy_equations”

```
% cylindrical equations

function [dX] = cy_equations(~,X)

    global K XI

    dX = zeros(760,1);

    %NEED TO INTRODUCE PROPORTIONALITY CONSTANT FOR SPATIAL MODEL
```

```
%_____CONCENTRATION EQUATIONS_____

   %PrP and PrPSx localization
   if XI(124) == 1;
      if X(15)/X(1) >= 1; %PrPSc aggregates are endocytosed spontaneously
        dX(15) = X(756)*X(1) - K(1)*X(15) - K(3)*X(15) + X(560)*X(18);   %total surface PrPSc
        dX(53) = K(1)*X(15) - X(556)*X(53);                    %PrPSc/AP-2/clatherin/dynamin
        dX(760) =  K(1)*X(65) + K(1)*X(66) + K(147)*X(66)...
           + K(147)*X(65) + K(1)*X(68) + K(147)*X(68)...
           - (2/3)*X(556)*X(760);                              %AP-2/clatherin/dynamin surface
        dX(65) = - (K(1)*X(65)) + .5*K(2)*X(67) - K(147)*X(65);         %free clatherin
        dX(66) = - (K(1)*X(66)) + .5*K(2)*X(67) - K(147)*X(66);         %free AP-2
        dX(68) = - (K(1)*X(68)) + (1/3)*K(2)*X(65) - K(556)*X(68);      %free dynamin
        dX(313) =  (K(1)*X(555)) + X(560)*X(76) + K(13)*X(78) - K(148)*X(15);          %total available
sphingolipid/cholesterol surface
        dX(555) = - (K(1)*X(555)) + K(148)*X(15);                       %total sphingolipid/cholesterol
LR

      elseif X(15)/X(1) < 1;
        dX(15) = X(756)*X(1) - K(3)*X(15) + X(560)*X(18);            %total surface PrPSc

        dX(53) = - X(556)*X(53);                               %PrPSc/AP-2/clatherin/dynamin
        dX(760) =  K(147)*X(66) + K(147)*X(65) + K(147)*X(68)...
           - (2/3)*X(556)*X(760);                              %AP-2/clatherin/dynamin surface
        dX(65) = - K(1)*X(65) + .5*K(2)*X(67) - K(147)*X(65);          %free clatherin
        dX(66) = - K(1)*X(66) + .5*K(2)*X(67) - K(147)*X(65);          %free AP-2
        dX(68) = - (K(1)*X(68)) + (1/3)*K(2)*X(65) - K(556)*X(68);      %free dynamin
        dX(313) =  K(1)*X(555) + X(560)*X(76) + K(13)*X(78) - K(148)*X(15);          %total available
sphingolipid/cholesterol surface
        dX(555) = - K(1)*X(555) + K(148)*X(15);                         %total sphingolipid/cholesterol
LR
      end

      dX(1) = -X(756)*X(1) - K(1)*X(1) - K(3)*X(1) + X(560)*X(6)...
         + K(13)*X(8);                                         %total surface PrPC
      dX(52) = K(147)*X(1) - X(556)*X(52);                     %PrPC/AP-2/clatherin
      dX(2) = K(3)*X(1) + K(3)*X(54) - X(756)*X(2);                    %total surface PrPC/Cu2+
      dX(3) = X(556)*X(52) - K(2)*X(3) + K(4)*X(50) - X(757)*X(3);          %total EE PrPC
      dX(4) = K(2)*X(3) - K(9)*X(4) - K(14)*X(4) - K(758)*X(4);             %total LE/MVB PrPC
      dX(5) =  K(14)*X(4) - X(563)*X(5);                       %total Endolys PrPC
      dX(6) = K(9)*X(4) - X(560)*X(6) + K(12)*X(8);            %total RE PrPC
      dX(7) = 0;                                               %total retrograde transport PrPC
      dX(8) = - K(12)*X(8) - K(13)*X(8) + K(47)*X(9);          %total trans Golgi network PrPC
      dX(9) = K(46)*X(10) - K(47)*X(9);                        %total trans Golgi PrPC
      dX(10) = K(45)*X(11) - K(46)*X(10);                      %total media Golgi PrPC
      dX(11) = K(44)*X(12) - K(45)*X(11);                      %total cis golgi PrPC
      dX(12) = K(43)*X(75) - K(44)*X(12);                      %total cis golgi nascent PrPC
      dX(13) = K(29)*X(47) - X(572)*X(13) - X(573)*X(13) - K(42)*X(13);   %total RER nacent PrPC
      if X(19) + X(754) >= 900;
        dX(14) = K(146)*X(754) + K(146)*X(19);                 %total exosome PrPSc
        dX(754) = K(2)*X(753) - K(9)*X(754) - K(10)*X(754)...
           - K(14)*X(754) - K(146)*X(754) + K(758)*X(4);             %total MVB PrPSc
        dX(19) = X(556)*X(17) - K(14)*X(19) - K(146)*X(19);      %total LE/MVB PrPSc/Cu2+
      elseif X(19) + X(754) < 900;
```

```
    dX(15) = 0;
    dX(754) = K(2)*X(753) - K(9)*X(754) - K(10)*X(754)...
      - K(14)*X(754) + K(758)*X(4);                    %total MVB PrPSc
    dX(19) = X(556)*X(17) - K(14)*X(19);               %total LE/MVB PrPSc/Cu2+
  end
  dX(753) = X(556)*X(53) - K(2)*X(753) + X(757)*X(3);          %total EE PrPSc
  dX(755) = K(14)*X(754) - X(561)*X(755) - X(564)*X(755);      %total endolys PrPSc
  dX(16) = K(3)*X(15) + K(3)*X(54) + X(756)*X(2) - K(1)*X(16);  %total surface PrPSc/Cu2+
  dX(50) = K(1)*X(2) + K(1)*X(65) + K(1)*X(66) - K(4)*X(50);    %PrPC/Cu2+/AP-2/clatherin
  dX(51) = K(1)*X(16) + K(1)*X(65) + K(1)*X(66) - K(2)*X(51);   %PrPSc/Cu2+/AP-2/clatehrin
  dX(17) = K(2)*X(51) - X(556)*X(17);                %total EE PrPSc/Cu2+
  dX(18) = K(9)*X(754) - X(560)*X(18) + K(12)*X(32);           %total RE PrPSc
  dX(20) = K(14)*X(19) - X(561)*X(20) - X(564)*X(20);          %total endolys PrPSc/Cu2+
  dX(21) = K(10)*X(754) - K(11)*X(21);               %total retrograde transport PrPSc
  dX(22) = X(564)*X(20) + X(564)*X(755) - K(16)*X(22) - K(17)*X(22)...
    - K(22)*X(22) + X(571)*X(43);                    %total cytoplasmic PrPSc/Cu2+
  dX(23) = X(561)*X(20) + X(561)*X(755) - X(564)*X(23);        %total partially degraded
PrPSc/Cu2+ endolys
  dX(24) = X(564)*X(23) - K(16)*X(24) - K(17)*X(24) - K(18)*X(24)...
    - K(22)*X(24);                                   %total cytoplasmic partially degraded PrPSc
  dX(25) = K(16)*X(24) + K(16)*X(178) + K(16)*X(22) - K(18)*X(22);  %PrPSc/14-3-3/partially
digested PrpSc/Cu2+ (cyt)
  dX(26) = K(17)*X(24) + K(17)*X(364) + K(17)*X(22) - X(568)*X(26);  %PrPC/p62/partially
digested PrPSc/Cu2+ (cyt agrresome)
  dX(27) = K(18)*X(24) + K(18)*X(82) + K(18)*X(22) - X(565)*X(27);  %PrPSc/Hcs70/partially
digested PrpSc/Cu2+ (cyt)
  dX(28) = K(22)*X(24) + K(22)*X(22) - K(23)*X(28);            %ubiquinated PrPSc (cyt)
  dX(29) = K(23)*X(28);                              %proteosome/PrPSc/Ub
  dX(30) = X(568)*XI(122) + X(568)*X(26) - K(21)*X(30);        %PrPC/p62/partially digested
PrPSc/LC3-II/Cu2+ (phagophore)
  dX(31) = K(21)*X(30) + K(21)*XI(5);                %PrPC/p62/partially digested PrPSc/LC3-
II/Cu2+ (autophage)
  dX(32) = K(11)*X(21) - K(12)*X(32) - K(37)*X(32);            %PrPSc trans Golgi network
  dX(33) = K(37)*X(32) - K(38)*X(33);                %PrPSc trans golgi
  dX(34) = K(38)*X(33) - K(39)*X(34);                %PrPSc media golgi
  dX(35) = K(39)*X(34) - K(40)*X(35);                %PrPSc cis golgi
  dX(74) = K(40)*X(35) + K(40)*X(96) + K(40)*X(73) - K(41)*X(74);  %cisGogi
PrPSc/COPI/Rab6?
  dX(36) = X(573)*X(13) - X(35)*X(36) + K(41)*X(74);           %RER PrPSc
  dX(37) = (1/3)*K(35)*X(38) + (1/3)*K(35)*X(574)...
    + (1/3)*K(35)*X(575) + X(35)*X(36);              %RER PrPSc/Bip
  dX(39) = X(572)*X(13);                             %RER misfolded PrPC
  dX(40) = K(30)*X(38) + K(30)*X(574) + K(30)*X(575) - K(31)*X(41);  %RER misfolded
PrPC/Bip
  dX(41) = K(31)*X(41) + K(31)*X(760) - K(32)*X(41);           %misfolded PrPC/Sec61
  dX(42) = K(32)*X(41) - K(33)*X(42);                %misfolded PrPC/Sec61/Ub
  dX(43) = - X(571)*X(43) - K(26)*X(43) + K(33)*X(42);         %misfolded PrPC/Ub in
cytoplasm
  dX(44) = K(26)*X(43) + K(26)*X(84) - K(27)*X(44);            %misfolded PrPC/Ub/proteosome
  dX(45) = K(27)*X(44);                              %degraded misfolded PrPC/Ub
  dX(46) = 0;                                        %PrPC mRNA
  dX(47) = K(28)*X(48) + K(28)*X(89) - K(29)*X(47);            %PrPC mRNA/ribisome
  dX(48) = X(563)*X(5);                              %degraded PrPC endolysosome
  dX(49) = X(562)*X(23);                             %degraded PrPSc endolysosome
  dX(54) = - 2*K(3)*X(54) - X(557)*X(54) - X(558)*X(54);       %Cu2+ free (from synapses)
```

```
    dX(55) = K(4)*X(50) + X(557)*X(54) + X(558)*X(54) + X(562)*X(23);   %Cu2+ free (cyt)
    dX(56) = K(6)*X(55) - K(8)*X(56);                                   %Cu+/CCS
    dX(57) = K(7)*X(55) - X(559)*X(57);                                 %Cu2+/Hah1
    dX(58) = K(5)*X(55);                                                %Cu+/COX17
    dX(59) = K(8)*X(56);                                                %free CCS
    dX(60) = X(559)*X(57);                                              %free Hah1
    dX(61) = 0;                                                         %SOD1 (cyt)
    dX(62) = K(8)*X(56) + K(8)*XI(121) + K(49)*XI(123) + K(49)*X(63);   %SOD1/Cu2+
    dX(63) = X(559)*X(57) - K(49)*X(63);                                %trans Golgi Cu2+
    dX(64) = 0;                                                         %SOD1 trans golgi
    dX(67) = X(556)*X(760) - K(2)*X(67);                                %clatherin/AP-2 EE
    dX(69) = - K(9)*X(69) + X(560)*X(7);                                %free Rab5
    dX(70) = K(9)*X(69) - X(560)*X(7);                                  %RE Rab5
    dX(71) = -K(10)*X(71) + K(11)*X(72);                                %free Rab9
    dX(72) = K(10)*X(71) - K(11)*X(72);                                 %retrograde Rab9
    dX(73) = - K(40)*X(73) + K(41)*X(74);                               %free Rab6?
    dX(75) = K(42)*X(95) + K(42)*X(13) - K(43)*X(75);                   %COPII/PrPC
    dX(76) = - X(560)*X(76) + K(12)*X(78);                              %RE cholesterol/sphingolipid
    dX(78) = - K(12)*X(78) - K(13)*X(78);                               %trans golgi
network/cholesterol/sphingolipid
    dX(79) = 0;                                                         %trans golgi cholesterol/sphingolipid
    dX(80) = K(15)*XI(5);                                               %lysosozymes endolys
    dX(84) = - K(23)*X(28) - K(26)*X(84) + K(27)*X(44);                 %proteosome
    dX(85) = X(568)*X(86) - K(21)*X(85);                                %ATGs phagophore
    dX(86) = - X(568)*X(86) + K(21)*X(85);                              %free ATGs
    dX(87) = K(21)*XI(5);                                               %autophage lysosozymes
    dX(88) = - K(31)*X(760) + K(33)*X(42);                              %chaperone/Sec61 complex
    dX(89) = - K(28)*X(89) + K(29)*X(47);                               %robosomes
    dX(90) = + K(32)*X(41) - 3*K(36)*X(90);                             %free ER BiP
    dX(91) = (1/3)*K(30)*X(574) + (1/3)*K(35)*X(574) - K(36)*X(91);     %free ER Ire1
    dX(92) = (1/3)*K(30)*X(575) + (1/3)*K(35)*X(575)- K(36)*X(92);      %free ER Perk
    dX(94) = (1/3)*K(30)*X(38) + (1/3)*K(35)*X(38) - K(36)*X(94)...
       - K(48)*X(94);                                                   %free ER ATF6
    dX(95) = - K(42)*X(95) + K(43)*X(75);                               %free COPII
    dX(96) = - K(40)*X(96) + K(41)*X(74);                               %free COPI
    dX(97) = K(48)*X(94);                                               %active ATF6
    dX(98) = K(34)*X(36);                                               %ER stress
    dX(99) = X(570)*X(569) - K(24)*X(99);                               %Nrf2 active
    dX(102) = - X(640)*X(102) + K(25)*X(103);                           %free MTOR
    dX(103) = X(640)*X(102) + X(640)*X(155) - K(25)*X(103);             %free p-MTOR
    dX(104) = X(639)*X(566) - X(567)*X(104) + K(19)*X(105)...
       - K(20)*X(104);                                                  %free BECN1
    dX(105) = X(567)*X(104) + X(567)*X(2) - K(19)*X(105);               %BECN1/PIK3C3
    dX(759) = K(19)*X(105);                                             %PIK3C3
    dX(110) = X(639)*X(566) + X(639)*X(155);                            %free p-Bcl-2
    dX(566) = - X(639)*X(566) + K(20)*X(104) + K(20)*X(386);            %Bcl-2/BECN1
    dX(38) = - (1/3)*K(30)*X(38) - (1/3)*K(35)*X(38) + K(36)*X(94);     %Bip/ATF6
    dX(574) = - (1/3)*K(30)*X(574) - (1/3)*K(35)*X(574) + K(36)*X(91);  %BiP/Ire1
    dX(575) = - (1/3)*K(30)*X(575) - (1/3)*K(35)*X(575) + K(36)*X(92);  %BiP/PERK
```

“Moves_generator”

```
global   neuron  size_NeuronVariables  num_neurons num_glia
```

```
%Generation and Distribution of trackable species in individual neurons

Moves = [];

% dim 3(1) = PrPC
   % dim 1(1->(num_neurons-1)) = synapses
      % [local [PrP] x y z]
   % dim 1(num_neurons + 1) = free terminals
      % [[PrP] x y z]
   % dim 1(num_neurons + 2) = EE
   % dim 1(num_neurons + 3) = MVB
   % dim 1(num_neurons + 4) = RE
   % dim 1(num_neurons + 7) = exosome


% dim 3(2) = PrPSc
   % dim 1 = same for PrPC

% dim 3(3) = cytokines/ chemoattractants

% dim 4 = neuron index

% each neuron has its own assigned 3 dimensional move matrix

%_____FOR  N E U R O N S_____

%_____SYNAPTIC & TERMINAL PLACEMENT_____

 for a = 1: num_neurons;

    [probD, probA, T] = pick_equation(neuron(:,2,a),0,0);
    termsize = size(T);
    num_term = termsize(1);
    terminals = [];
    for b = 1:num_term;
      theta = ((0-(2*pi)).*rand(1,1)+(2*pi));
      x = ((10^3)*T(b,1)/25)*cos(theta) + neuron(:,size_NeuronVariables + 3,a);
      y = ((10^3)*T(b,1)/25)*sin(theta) + neuron(:,size_NeuronVariables + 4,a);
      z = ((10^3)*T(b,2)/25);
      % [x y z r h]
      terminals = [terminals ; x y z (10^3)*T(b,2) (10^3)*T(b,1)];
    end

   Synapses = [];
   f_terminals = [];

   for c = 1:num_neurons;
       synapses = [];
       if neuron(:,size_NeuronVariables + 1,c) ~= a;

          % find the region of overlap for axon densities
          % soma
          if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a)) < ...
            (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables + 6,c)))...
```

```
        && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables + 15,a)) > ...
        (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables + 6,c))));
        % overlap axon on soma!

        ds = dis_cy(neuron(:,size_NeuronVariables + 3,a),neuron(:,size_NeuronVariables + 3,c)...
          ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables + 4,c),0,0);
        if (ds-neuron(:,size_NeuronVariables + 6,c)) < neuron(:,size_NeuronVariables + 13,a);
          % radial overlap!
          for r = 0:1:25*neuron(:,size_NeuronVariables + 6,c);
            for h = ((neuron(:,size_NeuronVariables + 3,c) - neuron(:,size_NeuronVariables +
6,c)))*25:...
                ((neuron(:,size_NeuronVariables + 3,c) + neuron(:,size_NeuronVariables + 6,c)))*25;
              for theta = 0:(2*pi);
                z = h/25;
                x = (r*cos(theta))/25 + neuron(:,size_NeuronVariables + 3,c);
                y = (r*sin(theta)/25) + neuron(:,size_NeuronVariables + 4,c);
                d = dis_cy(neuron(:,size_NeuronVariables+3,a)...
                  ,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);

                if d < neuron(:,size_NeuronVariables+13,a);
                  %overlap point!
                  %disp('wow')
                  pc = neuron(:,2,c);
                  [probD, probA, Axterminals] = pick_equation(pc,r,h);
                  %disp(probA)
                  if probA > 0.8;

                    % synapse! [local [PrP] x y z]
                    synapses = [synapses 1 0 x y z];
                  end
                end
              end
            end
          end
        end
      end


    if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a)) < ...
        (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables + 11,c)))...
        && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables + 15,a)) > ...
        (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables + 12,c))));

        % overlap axon on dendrite!
        %disp('wow')
        ds = dis_cy(neuron(:,size_NeuronVariables + 3,a),neuron(:,size_NeuronVariables + 3,c)...
          ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables + 4,c),0,0);
        if (ds-neuron(:,size_NeuronVariables + 9,c)) < neuron(:,size_NeuronVariables + 13,a);
          %disp('doublewow')
          % radial overlap!
          if (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))...
              <= (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 11,c));
            top = (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 11,c))*25;
          elseif (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))...
              > (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 11,c));
```

```
            top = (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))*25;
        end
        if (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables + 16,a))...
            <= (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 12,c));
          bottom = (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables +
12,c))*25;
        elseif (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables + 16,a))...
            > (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 12,c));
          bottom = (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables +
16,a))*25;
        end
        for r = 0:10:25*neuron(:,size_NeuronVariables + 9,c);
          for h = bottom:10:top;
            for theta = 0:(2*pi);
              z = h/25;
              x = (r*cos(theta)/25)+ neuron(:,size_NeuronVariables + 3,c);
              y = (r*sin(theta)/25)+ neuron(:,size_NeuronVariables + 4,c);
              d = dis_cy(neuron(:,size_NeuronVariables+3,a)...
                ,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);
              if d < neuron(:,size_NeuronVariables+13,a);
                %overlap point!
                %disp('oh yeah')
                [probD, A, t1] = pick_equation(neuron(:,2,a),r,h);
                [D, probA, t2] = pick_equation(neuron(:,2,c),r,h);
                if (A > 0.8) && (D > 0.8) && (D >= probA);
                  %disp('bop')
                  % synapse! [local [PrP] x y z]
                  synapses = [synapses 2 0 x y z];

                elseif (A > 0.8) && (D > 0.8) && (D < probA);
                  %disp('wham')
                  synapses = [synapses 3 0 x y z];
                end
              end
            end
          end
        end

      end

    end

    if ((neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,a)) < ...
      (neuron(:,size_NeuronVariables + 5,c) + neuron(:,size_NeuronVariables + 15,c)))...
      && (((neuron(:,size_NeuronVariables + 5,c)+ neuron(:,size_NeuronVariables + 15,a)) > ...
      (neuron(:,size_NeuronVariables + 5,c) - neuron(:,size_NeuronVariables + 16,c))));

       % overlap axon on dendrite!
      ds = dis_cy(neuron(:,size_NeuronVariables + 3,a),neuron(:,size_NeuronVariables + 3,c)...
        ,neuron(:,size_NeuronVariables +4,a),neuron(:,size_NeuronVariables + 4,c),0,0);
      if (ds-neuron(:,size_NeuronVariables + 13,c)) < neuron(:,size_NeuronVariables + 13,a);
         % radial overlap!
         if (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))...
            <= (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 15,c));
          top = (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 15,c))*25;
```

```
            elseif (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))...
                > (neuron(:,size_NeuronVariables + 5,c)+neuron(:,size_NeuronVariables + 15,c));
              top = (neuron(:,size_NeuronVariables + 5,a)+neuron(:,size_NeuronVariables + 15,a))*25;
            end
            if (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables + 16,a))...
                <= (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,c));
              bottom = (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables +
16,c))*25;
            elseif (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables + 16,a))...
                > (neuron(:,size_NeuronVariables + 5,c)-neuron(:,size_NeuronVariables + 16,c));
              bottom = (neuron(:,size_NeuronVariables + 5,a) - neuron(:,size_NeuronVariables +
16,a))*25;
            end

            for r = 0:10:25*neuron(:,size_NeuronVariables + 13,c);
              for h = bottom:10:top;
                for theta = 0:(2*pi);
                  z = h/25;
                  x = (r*cos(theta)/25)+ neuron(:,size_NeuronVariables + 3,c);
                  y = (r*sin(theta)/25)+ neuron(:,size_NeuronVariables + 4,c);
                  d = dis_cy(neuron(:,size_NeuronVariables+3,a)...
                    ,x,neuron(:,size_NeuronVariables+4,a),y,neuron(:,size_NeuronVariables+5,a),z);
                  if d < neuron(:,size_NeuronVariables+13,a) && (d >
neuron(:,size_NeuronVariables+9,a));
                    %overlap point!
                    [probD, A, t1] = pick_equation(neuron(:,2,a),r,h);
                    [D, probA, t2] = pick_equation(neuron(:,2,c),r,h);
                    if (A > 0.8) && (probA > 0.8);
                      % synapse! [local [PrP] x y z]
                      synapses = [synapses 3 0 x y z];
                    end
                  end
                end
              end
            end
          end
        end

%           find the region of overlap for terminals
%           soma
          for d = 1:num_term;
              term_d = dis_cy(terminals(d,1),neuron(:,size_NeuronVariables + 3,c),...
                terminals(d,2),neuron(:,size_NeuronVariables +
4,c),terminals(d,3),neuron(:,size_NeuronVariables + 5,c));

              if term_d < neuron(:,size_NeuronVariables + 6,l);
                % soma overlap! = 1
                synapses = [synapses 1 0 terminals(d,1) terminals(d,2) terminals(d,3)];
%                   disp('here')

              elseif (term_d < neuron(:,size_NeuronVariables + 9,l)) && ...
                    (term_d > neuron(:,size_NeuronVariables + 6,l));
                % dendrite or axon overlap!
                [probD, probA, Axterminals] = pick_equation(neuron(:,2,c),terminals(d,4),
terminals(d,3));
```

```
                if (probD > 0.8) && (probD > probA);
                  % synapse! = 2
                  synapse = [synapses 2 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
%                      disp('here')
                elseif (probA > 0.8) && (probA > probD);
                   % synapse! = 3
                   synapses = [synapses 3 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
%                      disp('here')
                end

             elseif (term_d < neuron(:,size_NeuronVariables + 13,l)) && ...
                   (term_d > neuron(:,size_NeuronVariables + 9,l));
                % just axon overlap!

                  [probD, probA, Axterminals] =
pick_equation(neuron(:,2,c),terminals(d,4),terminals(d,3));

                if (probA > 0.8);
                   % synapse = 3
                   synapses = [synapses 3 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
                   %disp('here')
                elseif (probA < 0.75);
                   % free terminal!
                   f_terminals = [f_terminals 0 terminals(d,1) terminals(d,2) terminals(d,3)];
                   terminals(d,1) = 0;
                   terminals(d,2) = 0;
                   terminals(d,3) = 0;
                   terminals(d,4) = 0;
                   %disp('there')
                end
             end

          end


       elseif neuron(:,size_NeuronVariables + 1,c) == a;
          synapses = [0];

       end
        if isempty(synapses) == 1;
           synapses = [0];
        end
```

```
      S = size(Synapses);
      s = length(synapses);
      if S(2) > s;
         synapses = [synapses zeros(1,abs(S(2)-s))];
      elseif s > S(2);
         Synapses = [Synapses zeros(S(1),(s-S(2)))];
      end
      Synapses = [Synapses ; synapses];
  end

   S = size(Synapses);
   S_l = S(2);
   T = length(f_terminals);

   if S_l > T;

      f_terminals = [f_terminals zeros(1,(S_l-T))];
   elseif T > S_l;
      Synapses = [Synapses zeros(S(1),(T-S(2)))];
   end
   add = [Synapses ; f_terminals];

   if a == 1;
     Moves = add;
   elseif a ~= 1;
     M = size(Moves);
     M_l = M(2);
     A = size(add);
     A_l = A(2);
     if M_l > A_l;
        add = [add zeros(A(1),(M_l - A_l))];
     elseif A_l > M_l && a == 2;
        Moves = [Moves zeros(M(1),(A_l - M_l))];

     elseif A_l > M_l && a > 2;
        Moves = [Moves zeros(M(1),(A_l - M_l),M(3))];
     end

     Moves(:,:,a) = add;

   end
end
 %  disp(Moves(:,:,2))

for b = 1:num_neurons;
   M = size(Moves);
   num_synapse = 0;
   num_terms = 0;
   disp(M(2))
   for n = 1:num_neurons;
      if n ~= b;
         for i = 1:5:M(2)-4;
            %synapse! [local [PrP] x y z]
            if Moves(n,i,b) ~= 0;
              num_synapse = num_synapse + 1;
```

```
        end
      end
    end
end
for j = 1:4:M(2)-4
    %[PrP] x y z]
    if Moves((num_neurons+1),j+1,b) == 0;
      num_terms = num_terms+1;
    end
end
```