6. Appendix

All simulations were implemented in MATLAB (The MathWorks, Inc., Fremingham, Boston, M, USA, http://www.mathworks.com). This appendix consists of the Matlab code used for the simulations carried out in Chapter 3. The code is divided in two files, simulating the first and second sections of the simulations shown in Chapter 3, respectively. Files can be provided on request (info@maxversace.com).

-1-
FILE NAME: first_section_Chapter_3.m

```
warning off MATLAB:divideByZero
clc;clear;close all

%%% Initializing stimuli
% SIMULATION DURATION
n_trials = 10;
n_ext_trials = 6;
total_time = 500;

%%%%%%%%%%%%%%
NAC_AMPL = 10^3;
NAC_IN_VTA = 10;
%%%%%%%%%%%%%%

% Input
CS1_input_starts = 1;
CS1_input_ends = 50;
CS2_input_starts = 0;
CS2_input_ends = 0;
US_input_starts = 150;
US_input_ends = 200;

% CS1 = Input(1),  CS2 = Input(2), US = Input(3)
Input.CS1 = zeros(1,total_time); Input.CS2 = zeros(1,total_time); Input.US = zeros(1,total_time);
Input.CS1(CS1_input_starts:CS1_input_ends) = Input.CS1(CS1_input_starts:CS1_input_ends) + 1;
% Input.CS2(CS2_input_starts:CS2_input_ends) = Input.CS2(CS2_input_starts:CS2_input_ends) + 1;
Input.US(US_input_starts:US_input_ends) = Input.US(US_input_starts:US_input_ends) + 1;

% REPEAT FOR NUMBER OF TRIALS
if n_trials ~= 1
    temp1 = Input.CS1;temp2 = Input.CS2; temp3 = Input.US;
    for i = 1:n_trials
        temp1 = [temp1 Input.CS1];temp2 = [temp2 Input.CS2];temp3 = [temp3 Input.US];
    end
end

% Add test trial only CS
for i = 1:n_ext_trials
    temp1 = [temp1 Input.CS1]; temp2 = [temp2 Input.CS2]; temp3 = [temp3 zeros(size(Input.CS2))];
end

%Put all together
Input.CS1 = temp1; Input.CS2 = temp2; Input.US = temp3;
clear temp1, clear temp2, clear temp3
```

```
% Create reset signal for EC
reset_signal = zeros(size(Input.CS1));
size_reset = 17;
fff = 5;
reset_signal(US_input_ends) = size_reset;
for kjk = 1:n_trials + n_ext_trials
   reset_signal((kjk*total_time + US_input_ends  -fff):(kjk*total_time + US_input_ends +  fff)) = size_reset; % reset at the end of US
   reset_signal((kjk*total_time - fff):(kjk*total_time + fff)) = size_reset; % reset at the end of TRIAL
end
k_reset = 100;

clear temp1, clear temp2, clear temp3, clear temp4, clear temp5, clear temp6,

total_time = size(Input.CS1,2);

%%% Iniziatilize variables
%% C E L L S
% Variables,  Euler integration step ,learning rate
A = 30;
A_pfc = 20;
A_VTA = 5;
B = 2;
B_post = 100;
B_pfc = 5;
B_H = 1; % B spectra
C = .5;
del_t = 0.01;
del_t_NAC = 0.002;
n_ms_sampl_VTA = 50;
eta = 10;
eta_passive_phase = .5;
eta_pfc_VTA = 10;
beta = .2;
VTA_exc_Pfc = .2;
%  and Spectral Timing parameter
A_sp = 0;
alpha_parameter = 0.1:0.01 :.8;
n_channels = size(alpha_parameter,2);
n_H = 8; % power for  f(x)
C_H = 0.0001;
D_H = 1;
E_H = 1;
HIPPO_DA = 10^3; % effect of DA on hippo
ampl_H_x = 10; %amplification factor last x received in hippo - starts AT
VTA_spike_H = 0;
VTA_INH = 0;
epsy = 0.0001;
alpha_E = .5;
% Posterior cortex (x)
x(1) = 0;   % CS1
x(2) = 0;   % CS2
x(3) = 0;   % US
dx_dt = zeros(1,3);
% Anterior cortex (y)
y(1) = 0;   % CS1
y(2) = 0;   % CS2
y(3) = 0;   % US
dy_dt = zeros(1,3);
% Motor cortex (z)
```

```
z(1) = 0;   % CS1
z(2) = 0;   % CS2
z(3) = 0;   % US
dz_dt = zeros(1,3);
% VTA and NAC
tonic_VTA = .8;
VTA = 0*rand;
NAC = 0*rand;
NAC_INH = 0;
F1 = 12;    % DA+ in anterior cortex
F2 = 100;   % DA- in anterior cortex
epsilon = 0.00000001;

% Hippocampus dentate
d_spectra_CA3_dt = zeros(3,n_channels);
spectra_CA3 = zeros(3,n_channels);
d_y_CA3_dt = zeros(3,n_channels);
y_CA3 = zeros(3,n_channels);
d_W_DENT_CA3_dt = zeros(3,n_channels);
W_DENT_CA3 = zeros(3,n_channels);
w_y_VTA = 0;

% Hippocampus  CA3
CA3(1) = 0;   % CS1
CA3(2) = 0;   % CS2
CA3(3) = 0;   % US
%% W E I G H T S
% Posterior cortex (x) => Anterior cortex (y)
w_x1_y1 = .33; w_x1_y2 = 0;w_x1_y3 = 0;
w_x2_y1 = 0; w_x2_y2 = .33;w_x2_y3 = 0;
w_x3_y1 = 0; w_x3_y2 = 0;w_x3_y3 = .33;
dww_x1_y1 = 0; dww_x1_y2 = 0; dww_x1_y3 = 0;
dww_x2_y1 = 0; dww_x2_y2 = 0; dww_x2_y3 = 0;
dww_x3_y1 = 0; dww_x3_y2 = 0; dww_x3_y3 = 0;
% Hippocampus => Hippocampus symmetric weights
w_CA3_1_CA3_2 = 0;
w_CA3_1_CA3_3 = 0;
w_CA3_2_CA3_3 = 0;
% EC => EC symmetric weights
w_EC_1_EC_2 = 0;
w_EC_1_EC_3 = 0;
w_EC_2_EC_3 = 0;

% Record activation
network.x1 = 0;
network.x2 = 0;
network.x3 = 0;
network.y1 = 0;
network.y2 = 0;
network.y3 = 0;
network.w_x1_y1 = w_x1_y1;
network.w_x1_y2 = w_x1_y2;
network.w_x1_y3 = w_x1_y3;
network.w_x2_y1 = w_x2_y1;
network.w_x2_y2 = w_x2_y2;
network.w_x2_y3 = w_x2_y3;
network.w_x3_y1 = w_x3_y1;
network.w_x3_y2 = w_x3_y2;
network.w_x3_y3 = w_x3_y3;
network.VTA = VTA;
```

```
network.NAC = NAC;
network.w_y_VTA = 0;
network.CA3_1 = 0;
network.CA3_2 = 0;
network.CA3_3 = 0;
network.w_CA3_1_CA3_2 = 0;
network.w_CA3_1_CA3_3 = 0;
network.w_CA3_2_CA3_3 = 0;
network.w_EC_1_EC_2 = 0;
network.w_EC_1_EC_3 = 0;
network.w_EC_2_EC_3 = 0;
network.w_y_VTA  = 0;
network.VTA_spike_H = 0;
network.W_DENT_CA3  = [];
EC_1 = 0; EC_1_record = [];
EC_2 = 0; EC_2_record = [];
EC_3 = 0; EC_3_record = [];
network.W_DENT_CA3 = [];

w_y_1_AMY = 0;
w_y_2_AMY = 0;


%%% MAIN LOOP
for j = 1:total_time


    %%%%%%%%!!!!!!!!!!!!! Now Print Provvisorio
    if j < total_time
        if and(Input.US(j+1) == 1, Input.US(j) == 0)
            now_print_provvisorio = 1;
        else
            now_print_provvisorio = 0;
        end
    end

    % CONTROL if NAN or INF
    if any([isnan(x) isnan(y) isnan(VTA) isnan(NAC)]);'Cazzo!!!!',x,y,VTA,NAC,w_y_VTA, j,
        break,
        'Something wrong, testa di cazzo!'
    end
    if any([isinf(x) isinf(y) isinf(VTA) isinf(NAC)]);'Cazzo!!!!', j,
        break, end
    % If no input, allow thr hippo to retrieve info
    if Input.CS1(j) + Input.CS2(j) + Input.US(j) == 0
        % ----- %  % ----- %  % ----- %  % ----- %  % ----- %
        %      H i p p o c a m p a l   r e t r i e v a l
        % ----- %  % ----- %  % ----- %  % ----- %  % ----- %
        %         x(1) = epsilon; x(2)= epsilon; x(3)= epsilon; y(1) = epsilon; y(2)= epsilon; y(3)= epsilon;
        inp_hippo = ones(1,3);
        CA3_temp(1) = inp_hippo(1); CA3_temp(2) = inp_hippo(2); CA3_temp(3) = inp_hippo(3);
        w_CA3_1_CA3_2_temp = w_CA3_1_CA3_2; w_CA3_1_CA3_3_temp = w_CA3_1_CA3_3; w_CA3_2_CA3_3_temp
= w_CA3_2_CA3_3;

        sum_W = w_CA3_1_CA3_2_temp + w_CA3_1_CA3_3_temp + w_CA3_2_CA3_3_temp;
        if sum_W > 0
            % normalize weights
            w_CA3_1_CA3_2_temp = ((w_CA3_1_CA3_2_temp)/(sum_W));
            w_CA3_1_CA3_3_temp = ((w_CA3_1_CA3_3_temp)/(sum_W));
            w_CA3_2_CA3_3_temp = ((w_CA3_2_CA3_3_temp)/(sum_W));
```

```
% network settles
for kk = 1:100
  dCA3_temp_dt(1) = CA3_temp(1)*CA3_temp(2)*w_CA3_1_CA3_2_temp + CA3_temp(1)*CA3_temp(3)*w_CA3_1_CA3_3_temp- sum(CA3_temp)*sum_W;
  dCA3_temp_dt(2) = CA3_temp(1)*CA3_temp(2)*w_CA3_1_CA3_2_temp + CA3_temp(2)*CA3_temp(3)*w_CA3_2_CA3_3_temp- sum(CA3_temp)*sum_W;
  dCA3_temp_dt(3) = CA3_temp(1)*CA3_temp(3)*w_CA3_1_CA3_3_temp + CA3_temp(2)*CA3_temp(3)*w_CA3_2_CA3_3_temp- sum(CA3_temp)*sum_W;
  CA3_temp(1) = CA3_temp(1) + del_t*dCA3_temp_dt(1);
  CA3_temp(2) = CA3_temp(2) + del_t*dCA3_temp_dt(2);
  CA3_temp(3) = CA3_temp(3) + del_t*dCA3_temp_dt(3);
end

% CA3 gives back activation to cortex
x_temp(1) = CA3_temp(1)-1;  y_temp(1) = max(CA3_temp(1)-1,0); x_temp(2) = CA3_temp(2)-1;  ...
  y_temp(2) = max(CA3_temp(2)-1,0); x_temp(3) = CA3_temp(3)-1;  y_temp(3) = max(CA3_temp(3)-1,0);

x_temp(1) = max(x_temp(1)/sum(x_temp),0); x_temp(2) = max(x_temp(2)/sum(x_temp),0); x_temp(3) = max(x_temp(3)/sum(x_temp),0);
% calculate d_dt W from x to y
dw_w_x1_y1 = eta_passive_phase*(1-w_x1_y1)*[x_temp(1)*y_temp(1)] - (w_x1_y1)*[x_temp(1)*VTA_dip];
dw_w_x1_y2 = eta_passive_phase*(1-w_x1_y2)*[x_temp(1)*y_temp(2)] - (w_x1_y2)*[x_temp(1)*VTA_dip];
dw_w_x1_y3 = eta_passive_phase*(1-w_x1_y3)*[x_temp(1)*y_temp(3)] - (w_x1_y3)*[x_temp(1)*VTA_dip];
dw_w_x2_y1 = eta_passive_phase*(1-w_x2_y1)*[x_temp(2)*y_temp(1)] - (w_x2_y1)*[x_temp(2)*VTA_dip];
dw_w_x2_y2 = eta_passive_phase*(1-w_x2_y2)*[x_temp(2)*y_temp(2)] - (w_x2_y2)*[x_temp(2)*VTA_dip];
dw_w_x2_y3 = eta_passive_phase*(1-w_x2_y3)*[x_temp(2)*y_temp(3)] - (w_x2_y3)*[x_temp(2)*VTA_dip];
dw_w_x3_y1 = eta_passive_phase*(1-w_x3_y1)*[x_temp(3)*y_temp(1)] - (w_x3_y1)*[x_temp(3)*VTA_dip];
dw_w_x3_y2 = eta_passive_phase*(1-w_x3_y2)*[x_temp(3)*y_temp(2)] - (w_x3_y2)*[x_temp(3)*VTA_dip];
dw_w_x3_y3 = eta_passive_phase*(1-w_x3_y3)*[x_temp(3)*y_temp(3)] - (w_x3_y3)*[x_temp(3)*VTA_dip];

sum_W =  w_x1_y1 + w_x1_y2 + w_x1_y3 +w_x2_y1 + w_x2_y2 + w_x2_y3 +w_x3_y1 + w_x3_y2 + w_x3_y3;

% Update cortico-cortical weights
w_x1_y1 = (w_x1_y1 + del_t * dw_w_x1_y1)/sum_W; w_x1_y2 = (w_x1_y2 + del_t * dw_w_x1_y2)/sum_W; w_x1_y3 = (w_x1_y3 + del_t * dw_w_x1_y3)/sum_W;
w_x2_y1 = (w_x2_y1 + del_t * dw_w_x2_y1)/sum_W; w_x2_y2 = (w_x2_y2 + del_t * dw_w_x2_y2)/sum_W; w_x2_y3 = (w_x2_y3 + del_t * dw_w_x2_y3)/sum_W;
w_x3_y1 = (w_x3_y1 + del_t * dw_w_x3_y1)/sum_W; w_x3_y2 = (w_x3_y2 + del_t * dw_w_x3_y2)/sum_W; w_x3_y3 = (w_x3_y3 + del_t * dw_w_x3_y3)/sum_W;

network.w_x1_y1 = [network.w_x1_y1 w_x1_y1];   % x1 to y1
network.w_x1_y2 = [network.w_x1_y2 w_x1_y2];   % x1 to y2
network.w_x1_y3 = [network.w_x1_y3 w_x1_y3];   % x1 to y3
network.w_x2_y1 = [network.w_x2_y1 w_x2_y1];   % x2 to y1
network.w_x2_y2 = [network.w_x2_y2 w_x2_y2];   % x2 to y2
network.w_x2_y3 = [network.w_x2_y3 w_x2_y3];   % x2 to y3
network.w_x3_y1 = [network.w_x3_y1 w_x3_y1];   % x3 to y1
network.w_x3_y2 = [network.w_x3_y2 w_x3_y2];   % x3 to y2
network.w_x3_y3 = [network.w_x3_y3 w_x3_y3];   % x3 to y3
network.w_CA3_1_CA3_2  = [network.w_CA3_1_CA3_2 w_CA3_1_CA3_2];
network.w_CA3_1_CA3_3  = [network.w_CA3_1_CA3_3 w_CA3_1_CA3_3];
network.w_CA3_2_CA3_3  = [network.w_CA3_2_CA3_3 w_CA3_2_CA3_3];
end
% ----- %  % ----- %  % ----- %  % ----- %  % ----- %
% E n d   H i p p o c a m p a l  r e t r i e v a l
% ----- %  % ----- %  % ----- %  % ----- %  % ----- %
end
```

```
    % saample VTA n msecs ago
    if j>n_ms_sampl_VTA
       VTA_spike = network.VTA(j-n_ms_sampl_VTA);
       VTA_dip = abs(min(VTA,0));
    else
       VTA_spike = 0;
       VTA_dip = 0;
    end

    %     VTA_exc_Pfc = VTA_spike +  20*w_y_VTA*sum(y)+ tonic_VTA; VTA_inh_Pfc = VTA_dip;
    if VTA_spike>0
       VTA_exc_Pfc = tonic_VTA;
       VTA_inh_Pfc = 0;
    else
       VTA_inh_Pfc = VTA_dip;
       VTA_exc_Pfc =  0;
       VTA_spike = 0;
    end

    % AMYGDALA
    %     AMY = max(Input.US(j) + w_y_1_AMY*y(1) + w_y_2_AMY*y(2),0);
    %     d_w_y_1_AMY_dt = (100-w_y_1_AMY)*y(1)*VTA_spike*AMY - [beta*w_y_1_AMY*AMY];
    %     d_w_y_2_AMY_dt = (100-w_y_2_AMY)*y(2)*VTA_spike*AMY - [beta*w_y_2_AMY*AMY];
    %
    %     w_y_1_AMY = w_y_1_AMY + del_t*d_w_y_1_AMY_dt;
    %     w_y_2_AMY = w_y_2_AMY + del_t*d_w_y_2_AMY_dt;


    %%% CALCULATE d_dt
    % calculate d_dt activation of posterior cortex (x)
    dx_dt(1) = -x(1)*A + (B_post-x(1))*[Input.CS1(j)] - (x(1))*[max(sum(x),0)] - reset_signal(j);
    dx_dt(2) = -x(2)*A + (B_post-x(2))*[Input.CS2(j)] - (x(2))*[max(sum(x),0)] - reset_signal(j);
    dx_dt(3) = -x(3)*A + (B_post-x(3))*[Input.US(j) ] - (x(3))*[max(sum(x),0)] - reset_signal(j);

    % calculate d_dt activation of anterior cortex (y)
    excit_1 = max((x(1)*w_x1_y1 + x(2)*w_x2_y1 + x(3)*w_x3_y1),0);    % to y1
    excit_2 = max((x(1)*w_x1_y2 + x(2)*w_x2_y2 + x(3)*w_x3_y2),0);    % to y2
    excit_3 = max((x(1)*w_x1_y3 + x(2)*w_x2_y3 + x(3)*w_x3_y3),0);    % to y3
    dy_dt(1)  =  -y(1)*A_pfc  +  (B_pfc-y(1))*[excit_1*max((1-VTA_exc_Pfc),0)  +  max(F1*VTA_exc_Pfc*(y(1)^1.1/(1.2  +
y(1)^1.1)),0)] - (y(1))*[F2*VTA_inh_Pfc+ 1*max(y(2)+y(3),0)] - reset_signal(j);
    dy_dt(2)  =  -y(2)*A_pfc  +  (B_pfc-y(2))*[excit_2*max((1-VTA_exc_Pfc),0)  +  max(F1*VTA_exc_Pfc*(y(2)^1.2/(1.5  +
y(2)^1.2)),0)] - (y(2))*[F2*VTA_inh_Pfc+ 1*max(y(1)+y(3),0)] - reset_signal(j);
    dy_dt(3)  =  -y(3)*A_pfc  +  (B_pfc-y(3))*[excit_3*max((1-VTA_exc_Pfc),0)  +  max(F1*VTA_exc_Pfc*(y(3)^1.2/(1.5  +
y(3)^1.2)),0)] - (y(3))*[F2*VTA_inh_Pfc+ 1*max(y(1)+y(2),0)] - reset_signal(j);

    % calculate d_dt for VTA
    dVTA_dt = -VTA*A_VTA + (10-VTA)*[Input.US(j) + w_y_VTA*sum(y)] - NAC_IN_VTA*NAC;
    %      dVTA_dt   =   -VTA*A_VTA   +   (10-VTA)*[Input.US(j)   +   max(w_y_VTA*sum(y)-   NAC_IN_VTA*NAC,0)]   -
NAC_IN_VTA*NAC;

    % weight from pfc (ALL) to VTA
    dw_y_VTA_dt = (1-w_y_VTA)*eta_pfc_VTA*sum(y)*VTA_spike - [beta*w_y_VTA*VTA_dip + w_y_VTA*sum(y)] ;

    % calculate d_dt W from x to y
    dw_w_x1_y1 = eta*(1-w_x1_y1)*[x(1)*y(1)*VTA_spike] - (w_x1_y1)*[10*x(1)*VTA_dip];
    dw_w_x1_y2 = eta*(1-w_x1_y2)*[x(1)*y(2)*VTA_spike] - (w_x1_y2)*[10*x(1)*VTA_dip];
    dw_w_x1_y3 = eta*(1-w_x1_y3)*[x(1)*y(3)*VTA_spike] - (w_x1_y3)*[10*x(1)*VTA_dip];
    dw_w_x2_y1 = eta*(1-w_x2_y1)*[x(2)*y(1)*VTA_spike] - (w_x2_y1)*[10*x(2)*VTA_dip];
    dw_w_x2_y2 = eta*(1-w_x2_y2)*[x(2)*y(2)*VTA_spike] - (w_x2_y2)*[10*x(2)*VTA_dip];
    dw_w_x2_y3 = eta*(1-w_x2_y3)*[x(2)*y(3)*VTA_spike] - (w_x2_y3)*[10*x(2)*VTA_dip];
```

```
    dw_w_x3_y1 = eta*(1-w_x3_y1)*[x(3)*y(1)*VTA_spike] - (w_x3_y1)*[10*x(3)*VTA_dip];
    dw_w_x3_y2 = eta*(1-w_x3_y2)*[x(3)*y(2)*VTA_spike] - (w_x3_y2)*[10*x(3)*VTA_dip];
    dw_w_x3_y3 = eta*(1-w_x3_y3)*[x(3)*y(3)*VTA_spike] - (w_x3_y3)*[10*x(3)*VTA_dip];

    %%%%% SPECTRAL TIMING
    d_VTA_INH_dt = max(alpha_E*(-VTA_INH +(VTA^2/(1+VTA^2))),0);
    VTA_INH = VTA_INH + del_t*d_VTA_INH_dt;
    VTA_spike_H = max(((VTA^n_H/(1+VTA^n_H)) - VTA_INH - epsy), 0);


    VTA_dip = 0;
    decay_W_DENT = 0;
    %     VTA_spike_H = now_print_provvisorio;

    if j > 30   % if we are in training for at least 30 msec, so I can add CSs....
        % For x1
        exc_EC_1 =  Input.CS1(j)+ EC_1 + EC_2*w_EC_1_EC_2 + EC_3*w_EC_1_EC_3;
        d_EC_1_dt = (1-EC_1)*exc_EC_1 - k_reset*reset_signal(j);
        EC_1 = max(EC_1 + .1*d_EC_1_dt,0);
        EC_1_record = [EC_1_record EC_1];
        for hh = 1:n_channels
            d_spectra_CA3_dt(1,hh) = alpha_parameter(hh)*[-A_sp*spectra_CA3(1,hh) + (B_H - spectra_CA3(1,hh))*ampl_H_x*EC_1]; % calculate d_dt 10 spectra for each cell
            d_y_CA3_dt(1,hh) = C_H*(1 - y_CA3(1,hh))- D_H*(spectra_CA3(1,hh).^n_H./(1+spectra_CA3(1,hh).^n_H).*y_CA3(1,hh)); % calculate d_dt transmitter deplition
            d_W_DENT_CA3_dt(1,hh) = E_H*(1-W_DENT_CA3(1,hh))*(spectra_CA3(1,hh).^2./(1+spectra_CA3(1,hh).^2).*y_CA3(1,hh).*...
                [- decay_W_DENT*W_DENT_CA3(1,hh)+ HIPPO_DA * VTA_spike_H - HIPPO_DA * VTA_dip]); % calculate d_dt W from DENTATE to CA3
            %     [- W_DENT_CA3(1,hh)+ HIPPO_DA * Input.US(j) - HIPPO_DA * VTA_dip]); % calculate d_dt W from DENTATE to CA3


        end
        % For x2
        exc_EC_2 =  Input.CS2(j)+ EC_2 + EC_1*w_EC_1_EC_2 + EC_3*w_EC_2_EC_3;
        d_EC_2_dt = (1-EC_2)*[Input.CS2(j) + EC_2] - k_reset*reset_signal(j);
        EC_2 = max(EC_2 + .1*d_EC_2_dt,0);
        EC_2_record = [EC_2_record EC_2];
        for hh = 1:n_channels
            d_spectra_CA3_dt(2,hh) = alpha_parameter(hh)*[-A_sp*spectra_CA3(2,hh) + (B_H - spectra_CA3(2,hh))*ampl_H_x*EC_2]; % calculate d_dt 10 spectra for each cell
            d_y_CA3_dt(2,hh) = C_H*(1 - y_CA3(2,hh))- D_H*(spectra_CA3(2,hh).^n_H./(1+spectra_CA3(2,hh).^n_H).*y_CA3(2,hh)); % calculate d_dt transmitter deplition
            d_W_DENT_CA3_dt(2,hh) = E_H*(1-W_DENT_CA3(2,hh))*(spectra_CA3(2,hh).^2./(1+spectra_CA3(2,hh).^2).*y_CA3(2,hh).*...
                [- decay_W_DENT*W_DENT_CA3(2,hh)+ HIPPO_DA * VTA_spike_H - HIPPO_DA * VTA_dip]); % calculate d_dt W from DENTATE to CA3
            %   [- W_DENT_CA3(2,hh)+ HIPPO_DA * Input.US(j) - HIPPO_DA * VTA_dip]); % calculate d_dt W from DENTATE to CA3

        end
        % For x3
        exc_EC_3 =  Input.US(j)+ EC_3 + EC_1*w_EC_1_EC_3 +EC_2*w_EC_2_EC_3;
        d_EC_3_dt = (1-EC_3)*[Input.US(j)+ EC_3] - k_reset*reset_signal(j);
        EC_3 = max(EC_3 + .1*d_EC_3_dt,0);
        EC_3_record = [EC_3_record EC_3];
        for hh = 1:n_channels
            d_spectra_CA3_dt(3,hh) = alpha_parameter(hh)*[-A_sp*spectra_CA3(3,hh) + (B_H - spectra_CA3(3,hh))*ampl_H_x*EC_3]; % calculate d_dt 10 spectra for each cell
```

```
        d_y_CA3_dt(3,hh) = C_H*(1 - y_CA3(3,hh))-
D_H*(spectra_CA3(3,hh).^n_H./(1+spectra_CA3(3,hh).^n_H).*y_CA3(3,hh)); % calculate d_dt transmitter deplition
        d_W_DENT_CA3_dt(3,hh) = E_H*(1-
W_DENT_CA3(3,hh))*(spectra_CA3(3,hh).^2./(1+spectra_CA3(3,hh)^2).*y_CA3(3,hh).*...
          [- decay_W_DENT*W_DENT_CA3(3,hh)+ HIPPO_DA * VTA_spike_H - HIPPO_DA * VTA_dip]); % calculate
d_dt W from DENTATE to CA3
        %    [- W_DENT_CA3(3,hh)+ HIPPO_DA * Input.US(j) - HIPPO_DA * VTA_dip]); % calculate d_dt W from
DENTATE to CA3
    end

    % update activations and W
    spectra_CA3 = spectra_CA3 + del_t * d_spectra_CA3_dt; spectra_CA3 = max(spectra_CA3,0);
    y_CA3 = y_CA3 + del_t * d_y_CA3_dt; y_CA3 = max(y_CA3,0);
    W_DENT_CA3 = W_DENT_CA3 + del_t * d_W_DENT_CA3_dt; W_DENT_CA3 = max(W_DENT_CA3,0);

    % calculate CA3 activation - rectified version
    CA3 = max(sum(spectra_CA3.^2./(1+spectra_CA3.^2).*y_CA3.*W_DENT_CA3,2),0);

    % CA3 to CA3 connectivity
    dw_CA3_1_CA3_2_dt = (1-w_CA3_1_CA3_2)*[100*eta*CA3(1)*CA3(2)*VTA_spike_H];% -
(w_CA3_1_CA3_2)*[CA3(1)*VTA_dip + CA3(2)*VTA_dip];
    dw_CA3_1_CA3_3_dt = (1-w_CA3_1_CA3_3)*[100*eta*CA3(1)*CA3(3)*VTA_spike_H];% -
(w_CA3_1_CA3_3)*[CA3(1)*VTA_dip + CA3(3)*VTA_dip];
    dw_CA3_2_CA3_3_dt = (1-w_CA3_2_CA3_3)*[100*eta*CA3(2)*CA3(3)*VTA_spike_H];% -
(w_CA3_2_CA3_3)*[CA3(2)*VTA_dip + CA3(3)*VTA_dip];

    % Ec to EC connectivity
    dw_EC_1_EC_2_dt = (10-w_EC_1_EC_2)*[EC_1*EC_2*VTA_spike_H];% - (w_EC_1_EC_2)*[EC_1*VTA_dip +
EC_2*VTA_dip];
    dw_EC_1_EC_3_dt = (10-w_EC_1_EC_3)*[EC_1*EC_3*VTA_spike_H];% - (w_EC_1_EC_3)*[EC_1*VTA_dip +
EC_3*VTA_dip];
    dw_EC_2_EC_3_dt = (10-w_EC_2_EC_3)*[EC_2*EC_3*VTA_spike_H];% - (w_EC_2_EC_3)*[EC_2*VTA_dip +
EC_3*VTA_dip];

    % NAC parameters
    NAC_AMPL = 10;
    alpha_NAC = 1;
    n_N = 3;

    if j > 10
        INP_NAC = (1-NAC)*NAC_AMPL*sum(CA3);
        d_NAC_INH_dt = alpha_NAC*(-NAC_INH +(INP_NAC^n_N/(1+INP_NAC^n_N)));
        NAC_INH = max(NAC_INH + del_t*d_NAC_INH_dt,0);
        N = max(((INP_NAC^n_N/(1+INP_NAC^n_N)) - NAC_INH - 0.0000001), 0);
        NAC = N * 1000;
    end

    %%% CALCULATE ACTIVATION
    % update d_dt activation of posterior cortex (x)
    x(1) = max(x(1) + del_t * dx_dt(1),0);
    x(2) = max(x(2) + del_t * dx_dt(2),0);
    x(3) = max(x(3) + del_t * dx_dt(3),0);
    y(1) = max(y(1) + del_t * dy_dt(1),0);
    y(2) = max(y(2) + del_t * dy_dt(2),0);
    y(3) = max(y(3) + del_t * dy_dt(3),0);
    % VTA
    VTA = VTA + del_t * dVTA_dt;
    VTA = VTA;

    %%% UPDATE WEIGHTS
```

```
        % W x => y
        w_x1_y1 = w_x1_y1 + del_t * dw_w_x1_y1;  w_x1_y2 = w_x1_y2 + del_t * dw_w_x1_y2; w_x1_y3 = w_x1_y3 +
del_t * dw_w_x1_y3;
        w_x2_y1 = w_x2_y1 + del_t * dw_w_x2_y1; dw_w_x2_y2 = dw_w_x2_y2 + del_t * dw_w_x2_y2; w_x2_y3 =
w_x2_y3 + del_t * dw_w_x2_y3;
        w_x3_y1 = w_x3_y1 + del_t * dw_w_x3_y1; w_x3_y2 = w_x3_y2 + del_t * dw_w_x3_y2; w_x3_y3 = w_x3_y3 + del_t
* dw_w_x3_y3;

        % Update Pfc => VTA W
        w_y_VTA = w_y_VTA + del_t * dw_y_VTA_dt;
        % CA3 tp CA3
        w_CA3_1_CA3_2 = w_CA3_1_CA3_2 + del_t * dw_CA3_1_CA3_2_dt;
        w_CA3_1_CA3_3 = w_CA3_1_CA3_3 + del_t * dw_CA3_1_CA3_3_dt;
        w_CA3_2_CA3_3 = w_CA3_2_CA3_3 + del_t * dw_CA3_2_CA3_3_dt;

        % EC to EC
        w_EC_1_EC_2 = w_EC_1_EC_2 + del_t * dw_EC_1_EC_2_dt;
        w_EC_1_EC_3 = w_EC_1_EC_3 + del_t * dw_EC_1_EC_3_dt;
        w_EC_2_EC_3 = w_EC_2_EC_3 + del_t * dw_EC_2_EC_3_dt;

    end

    % save record activation
    network.x1 = [network.x1 x(1)];
    network.x2 = [network.x2 x(2)];
    network.x3 = [network.x3 x(3)];
    network.y1 = [network.y1 y(1)];
    network.y2 = [network.y2 y(2)];
    network.y3 = [network.y3 y(3)];
    network.VTA = [network.VTA VTA];
    network.NAC = [network.NAC NAC];
    network.CA3_1 = [network.CA3_1 CA3(1)];
    network.CA3_2 = [network.CA3_2 CA3(2)];
    network.CA3_3 = [network.CA3_3 CA3(3)];
    network.w_x1_y1 = [network.w_x1_y1 w_x1_y1];   % x1 to y1
    network.w_x1_y2 = [network.w_x1_y2 w_x1_y2];   % x1 to y2
    network.w_x1_y3 = [network.w_x1_y3 w_x1_y3];   % x1 to y3
    network.w_x2_y1 = [network.w_x2_y1 w_x2_y1];   % x2 to y1
    network.w_x2_y2 = [network.w_x2_y2 w_x2_y2];   % x2 to y2
    network.w_x2_y3 = [network.w_x2_y3 w_x2_y3];   % x2 to y3
    network.w_x3_y1 = [network.w_x3_y1 w_x3_y1];   % x3 to y1
    network.w_x3_y2 = [network.w_x3_y2 w_x3_y2];   % x3 to y2
    network.w_x3_y3 = [network.w_x3_y3 w_x3_y3];   % x3 to y3
    network.w_CA3_1_CA3_2  = [network.w_CA3_1_CA3_2 w_CA3_1_CA3_2];
    network.w_CA3_1_CA3_3  = [network.w_CA3_1_CA3_3 w_CA3_1_CA3_3];
    network.w_CA3_2_CA3_3  = [network.w_CA3_2_CA3_3 w_CA3_2_CA3_3];
    network.w_EC_1_EC_2  = [network.w_EC_1_EC_2 w_EC_1_EC_2];
    network.w_EC_1_EC_3  = [network.w_EC_1_EC_3 w_EC_1_EC_3];
    network.w_EC_2_EC_3  = [network.w_EC_2_EC_3 w_EC_2_EC_3];
    network.w_y_VTA  = [network.w_y_VTA w_y_VTA];
    network.VTA_spike_H  = [network.VTA_spike_H VTA_spike_H];
%     network.W_DENT_CA3 = [network.W_DENT_CA3 W_DENT_CA3];
end

% Plot activation% Plot stimuli
set(figure,'DoubleBuffer', 'on')
subplot(5,2,1), plot(Input.CS1), hold on, plot(Input.CS2, 'm'), title('CS 1 (blu), CS2 (magenta)'); axis([0 size(Input.CS2,2) -1
2])
subplot(5,2,3), plot(Input.US, 'r'), title('US'); axis([0 size(Input.CS2,2) -1 2])
subplot(5,2,2),plot(network.x1), hold on, plot(network.x2, 'm'), hold on, ...
```

```
    plot(network.x3, 'r'), title('Posterior cortex'); %axis([0 size(Input.CS2,2) -2 2.1]); drawnow
subplot(5,2,4),plot(network.y1), hold on, plot(network.y2, 'm'), hold on, ...
    plot(network.y3, 'r'), title('Anterior cortex'); %axis([0 total_time -2 2.1]); drawnow
subplot(5,2,7),plot(network.VTA), hold on, plot(network.NAC, 'r'), title('VTA (BLUE) - NAC (RED)'); %axis([0 size(Input.CS2,2) -2 2.1]); drawnow
subplot(5,2,8),plot(network.CA3_1), hold on, plot(network.CA3_2, 'm'), hold on, ...
    plot(network.CA3_3, 'r'), title('CA3 posterior');drawnow
subplot(5,2,9),plot(network.NAC, 'r'), title('NAC');drawnow
subplot(5,2,10),plot(network.w_CA3_1_CA3_2), hold on, plot(network.w_CA3_1_CA3_3, 'm'), hold on, ...
    plot(network.w_CA3_2_CA3_3, 'r'), title('CA3 1=2 (blu), 1=3 (magenta), 2=3 (red)');drawnow
subplot(5,2,6),plot(network.w_x1_y1), hold on, plot(network.w_x1_y3, 'r'), hold on, ...
    plot(network.w_x3_y1, 'm'), hold on, plot(network.w_x3_y3, 'g'), title('W 1=1(blu), 1=3(red), 3=1(mag.), 3=3(green)');drawnow
subplot(5,2,5),  plot(network.w_y_VTA, 'g'), title('Pfc => VTA weight');

figure
subplot(4,1,1),plot(Input.CS1), hold on, plot(Input.CS2, 'm'), hold on, plot(Input.US, 'r'),axis([0 size(Input.CS2,2) -1 2]),title('Input'), grid on
subplot(4,1,2),plot(network.VTA, 'b'),  axis([0 size(Input.CS2,2) -1 2]),title('VTA activation'), grid on
subplot(4,1,3),plot(network.NAC, 'r'), title('NAC'), axis([0 size(Input.CS2,2) min(network.NAC) max(network.NAC)]),grid on
subplot(4,1,4),plot(max(network.VTA,0)), axis([0 size(Input.CS2,2) -1 2]),title('Net VTA output'), grid on

figure
subplot(4,1,1),plot(Input.CS1), hold on, plot(Input.CS2, 'm'), hold on, plot(Input.US, 'r'),axis([0 size(Input.CS2,2) -1 2]),title('Input'), grid on
subplot(4,1,2),plot(network.y1), axis([0 size(Input.CS2,2) -1 2]),title('Anterior cortex, y1')
subplot(4,1,3),plot(network.y2, 'm'), axis([0 size(Input.CS2,2) -1 2]),title('Anterior cortex, y2')
subplot(4,1,4), plot(network.y3, 'r'), axis([0 size(Input.CS2,2) -1 2]),title('Anterior cortex, y3')


start_time1 = 900;      end_time1 = 1400;
start_time2 = 11900; end_time2 = 12500;
start_time3 = 15900; end_time3 = 16300;
start_time4 = 9900; end_time4 = 11500;

figure
subplot(5,1,1),plot(Input.CS1(start_time1:end_time1)), hold on, plot(Input.CS2(start_time1:end_time1), 'm'), hold on, plot(Input.US(start_time1:end_time1), 'r'), title('Early training'),grid on
subplot(5,1,2),plot(network.x1(start_time1:end_time1)), hold on, plot(network.x2(start_time1:end_time1), 'm'), hold on, ...
    plot(network.x3(start_time1:end_time1), 'r'), title('Posterior cortex');
subplot(5,1,3),plot(network.y1(start_time1:end_time1)), hold on, plot(network.y2(start_time1:end_time1), 'm'), hold on, ...
    plot(network.y3(start_time1:end_time1), 'r'), title('Anterior cortex');
subplot(5,1,4),plot(max(network.VTA(start_time1:end_time1),0), 'b'),  grid on
subplot(5,1,5),plot(network.NAC(start_time1:end_time1), 'r'), grid on

% figure
%  subplot(5,1,1),plot(Input.CS1(start_time2:end_time2)), hold on, plot(Input.CS2(start_time2:end_time2), 'm'), hold on, plot(Input.US(start_time2:end_time2), 'r'), title('Late training'),grid on
% subplot(5,1,2),plot(network.x1(start_time2:end_time2)), hold on, plot(network.x2(start_time2:end_time2), 'm'), hold on, ...
%     plot(network.x3(start_time2:end_time2), 'r'), title('Posterior cortex');
% subplot(5,1,3),plot(network.y1(start_time2:end_time2)), hold on, plot(network.y2(start_time2:end_time2), 'm'), hold on, ...
%     plot(network.y3(start_time2:end_time2), 'r'), title('Anterior cortex');
% subplot(5,1,4),plot(max(network.VTA(start_time2:end_time2),0), 'b'), title('VTA'),  grid on
% subplot(5,1,5),plot(network.NAC(start_time2:end_time2), 'r'), title('NAC'), grid on

% figure
%  subplot(5,1,1),plot(Input.CS1(start_time3:end_time3)), hold on, plot(Input.CS2(start_time3:end_time3), 'm'), hold on, plot(Input.US(start_time3:end_time3), 'r'), title('Extinction'),grid on
% subplot(5,1,2),plot(network.x1(start_time3:end_time3)), hold on, plot(network.x2(start_time3:end_time3), 'm'), hold on, ...
%     plot(network.x3(start_time3:end_time3), 'r'), title('Posterior cortex');
```

```
% subplot(5,1,3),plot(network.y1(start_time3:end_time3)), hold on, plot(network.y2(start_time3:end_time3), 'm'), hold on, ...
%     plot(network.y3(start_time3:end_time3), 'r'), title('Anterior cortex');
% subplot(5,1,4),plot(max(network.VTA(start_time3:end_time3),0), 'b'), title('VTA'),  grid on
% subplot(5,1,5),plot(network.NAC(start_time3:end_time3), 'r'), title('NAC'), grid on
%
% figure
% subplot(5,1,1),plot(Input.CS1(start_time4:end_time4)), hold on, plot(Input.CS2(start_time4:end_time4), 'm'), hold on, plot(Input.US(start_time4:end_time4), 'r'), title('wrong trial'),grid on
% subplot(5,1,2),plot(network.x1(start_time4:end_time4)), hold on, plot(network.x2(start_time4:end_time4), 'm'), hold on, ...
%     plot(network.x3(start_time4:end_time4), 'r'), title('Posterior cortex');
% subplot(5,1,3),plot(network.y1(start_time4:end_time4)), hold on, plot(network.y2(start_time4:end_time4), 'm'), hold on, ...
%     plot(network.y3(start_time4:end_time4), 'r'), title('Anterior cortex');
% subplot(5,1,4),plot(max(network.VTA(start_time4:end_time4),0), 'b'), title('VTA'),  grid on
% subplot(5,1,5),plot(network.NAC(start_time4:end_time4), 'r'), title('NAC'), grid on


figure;
subplot(4,1,1), plot(EC_1_record), axis([0 length(EC_3_record) -1 2]), title('EC1'), grid on
subplot(4,1,2),plot(EC_3_record,'r'), axis([0 length(EC_3_record) -1 2]), title('EC3'), grid on
subplot(4,1,3), plot(network.VTA_spike_H), axis([0 length(EC_3_record) -1 2]), title('Now Print'), grid on
subplot(4,1,4), plot(reset_signal, 'g'), title('Reset signal'), hold on, plot(Input.CS1),  hold on, plot(Input.US, 'r'), grid on
```

-2-

FILE NAME: FILE NAME: second_section_Chapter_3.m

```
% -----------------------------------------------------------------------------
% FIRST EXAMPLE
% UNLEARN bad trial
% In this example, the Premotor cortex (intergates PFc at different rates)
% sends activation to the motor cortex from time 1 to 1000 msec.
% Mortor cortes sums the activation of Pfc, releases the action, and the reward is negative (punishment).
% The AT mechanism learns to adaptively close the thalamus at time of the negative reward.
% This allows the motor cortex to refrain from releasing the action.
% -----------------------------------------------------------------------------

% Spectral timing
clc; clear;close all

% Variables and parameters
n_trials = 10;
total_time = 1000;
% Cellular stages
prem_activ = zeros(1,total_time);
prem_activ(1:1000) = 1;
reward = zeros(1,total_time);
reward(900:1000) = -1;
motor_cortex = 0;
thalamus = 0;
striatum = 0;
GP_e = 0;
GP_i = 0;

% AT Parameters
A = 0;
B = 1;
C = 0.0001;
D = 0.125;
beta = .8;
```

```
E = 1;
n = 8;
time_step = 0.01;
alpha_parameter = 0.00001:0.01 :0.2;
n_chan = size(alpha_parameter,2);

% Record
record_motor_cortex = [];
record_thalamus = [];
record_striatum = [];
record_GP_i = [];
record_GP_e = [];
record_act = zeros(n_chan,total_time);
record_trans = zeros(n_chan,total_time);
record_gated = zeros(n_chan,total_time);
record_doubly_gated = zeros(n_chan,total_time);
record_ltm = zeros(n_chan,total_time);

z = 0;
for k = 1:n_trials
   striatum = 0.9;
   % ---------  AT   ---------
   for i = 1:n_chan
      x = 0; y = 1;
      for j = 1:total_time        % Time steps
         dx = alpha_parameter(i)*(-A*x + (1 - B*x)* prem_activ(j));
         x = x + time_step*dx;
         fx = x^n/(beta^n+x^n);
         record_actx(i,j) = fx;
         % Neurotransmitter
         dy = C*(1-y) - D*fx*y;
         y = y + time_step*dy;
         record_trans(i,j) = y;
         % LTM traces
         dz = E*fx*y*(-z + reward(j));
         z = z + time_step*dz;
         record_ltm(i,j) = z;
         % Gated signals
         gated_signal = fx*y;
         record_gated(i,j) = gated_signal;
         % Doubly Gated signals
         doubly_gated_signal = fx*y*z;
         record_doubly_gated(i,j) = doubly_gated_signal;
      end
   end
   % --------- end AT ---------
   GP_i = max(sum(doubly_gated_signal),0);
   GP_e = max(1 - striatum - GP_i,0);
   thalamus = .1 - GP_e;
   motor_cortex = 1*thalamus;

   % Record
   record_motor_cortex = [record_motor_cortex motor_cortex];
   record_thalamus = [record_thalamus thalamus];
   record_striatum = [record_striatum striatum];
   record_GP_i = [record_GP_i GP_i];
   record_GP_e = [record_GP_e GP_e];
end

% Total Gated signals
```

```
total_gated_signal = sum(record_doubly_gated);

% Plots AT
a = figure;
set(a,'Color',[1 1 1])
subplot(3,3,1), plot(prem_activ,'b'), title('Premotor Cortex'), axis([1 total_time 0 2])
subplot(3,3,2), plot(reward,'r'), title('Reward'), axis([1 total_time -2 2])
subplot(3,3,3), plot(record_actx'), title('Activation x')
subplot(3,3,4), plot(record_trans'), title('Transmitter y')
subplot(3,3,5), plot(record_gated'), title('fx*y')
subplot(3,3,6), plot(record_doubly_gated'), title('fx*y*z')
subplot(3,3,7), plot(record_ltm'), title('LTM traces')
subplot(3,3,7), plot(total_gated_signal'), title('Output')

% Plots
a = figure;
set(a,'Color',[1 1 1])
subplot(3,2,1), plot(record_motor_cortex,'b'), title('Motor cortex')
subplot(3,2,2), plot(record_thalamus,'r'), title('Thalamus')
subplot(3,2,3), plot(record_striatum'), title('Striatum')
subplot(3,2,4), plot(record_GP_i'), title('GP_i')
subplot(3,2,5), plot(record_GP_i'), title('GP_e')
subplot(3,2,6), plot(prem_activ'), title('Premotor cortex'), axis([1 total_time 0 2])


% ------------------------------------------------------------------------------
% SECOND EXAMPLE
% LEARN good trial
% In this example, Premotor cortex is activated at time 500
% ------------------------------------------------------------------------------

% Variables and parameters
n_trials = 10;
total_time = 1000;
% Cellular stages
prem_activ = zeros(1,total_time);
prem_activ(500:1000) = 1;
reward = zeros(1,total_time);
reward(900:1000) = 1;
motor_cortex = 0;
thalamus = 0;
striatum = 0;
GP_e = 0;
GP_i = 0;

% Record
record_motor_cortex = [];
record_thalamus = [];
record_striatum = [];
record_GP_i = [];
record_GP_e = [];
record_act = zeros(n_chan,total_time);
record_trans = zeros(n_chan,total_time);
record_gated = zeros(n_chan,total_time);
record_doubly_gated = zeros(n_chan,total_time);
record_ltm = zeros(n_chan,total_time);

z = 0;
for k = 1:n_trials
   striatum = 0.9;
```

```
    % ---------  AT   ---------
    for i = 1:n_chan
        x = 0; y = 1;
        for j = 1:total_time        % Time steps
            dx = alpha_parameter(i)*(-A*x + (1 - B*x)* prem_activ(j));
            x = x + time_step*dx;
            fx = x^n/(beta^n+x^n);
            record_actx(i,j) = fx;
            % Neurotransmitter
            dy = C*(1-y) - D*fx*y;
            y = y + time_step*dy;
            record_trans(i,j) = y;
            % LTM traces
            dz = E*fx*y*(-z + reward(j));
            z = z + time_step*dz;
            record_ltm(i,j) = z;
            % Gated signals
            gated_signal = fx*y;
            record_gated(i,j) = gated_signal;
            % Doubly Gated signals
            doubly_gated_signal = fx*y*z;
            record_doubly_gated(i,j) = doubly_gated_signal;
        end
    end
    % --------- end AT ---------
    GP_i = max(sum(doubly_gated_signal),0);
    GP_e = max(1 - striatum - GP_i,0);
    thalamus = .1 - GP_e;
    motor_cortex = 1*thalamus;

    % Record
    record_motor_cortex = [record_motor_cortex motor_cortex];
    record_thalamus = [record_thalamus thalamus];
    record_striatum = [record_striatum striatum];
    record_GP_i = [record_GP_i GP_i];
    record_GP_e = [record_GP_e GP_e];
end

% Total Gated signals
total_gated_signal = sum(record_doubly_gated);

% Plots AT
a = figure;
set(a,'Color',[1 1 1])
subplot(3,3,1), plot(prem_activ,'b'), title('Premotor Cortex'), axis([1 total_time 0 2])
subplot(3,3,2), plot(reward,'r'), title('Reward'), axis([1 total_time -2 2])
subplot(3,3,3), plot(record_actx'), title('Activation x')
subplot(3,3,4), plot(record_trans'), title('Transmitter y')
subplot(3,3,5), plot(record_gated'), title('fx*y')
subplot(3,3,6), plot(record_doubly_gated'), title('fx*y*z')
subplot(3,3,7), plot(record_ltm'), title('LTM traces')
subplot(3,3,7), plot(total_gated_signal'), title('Output')

% Plots
a = figure;
set(a,'Color',[1 1 1])
subplot(3,2,1), plot(record_motor_cortex,'b'), title('Motor Cortex')
subplot(3,2,2), plot(record_thalamus,'r'), title('Thalamus')
subplot(3,2,3), plot(record_striatum'), title('Striatum')
subplot(3,2,4), plot(record_GP_i'), title('GP_i')
```

```
subplot(3,2,5), plot(record_GP_i'), title('GP_e')
subplot(3,2,6), plot(prem_activ'), title('Premotor Cortex'),axis([1 total_time 0 2])
```