3-dimensional stochastic system of CAR T Cell model

$$dN = r_N N \log\left(\frac{K_N}{N+C}\right)dt$$

$$dC = \left[\rho_C + \frac{b(N+C-K_N)^2}{a(N+C)^2+(N+C-K_N)^2}\right]C\log\left(\frac{K_C}{N+C}\right)dt + \tau_1 C\log\left(\frac{K_C}{N+C}\right)dW_1$$

$$dB = \left(r_B B - \gamma_B B\frac{C}{k_B+C}\right)dt + \tau_2 B dW_2 - \tau_3 B\frac{C}{k_B+C}dW_3$$

where

$N(t)$ count of normal T cells at time t

$C(t)$ count of CAR-T cells at time t

$B(t)$ count of tumor cells at time t

r_N net growth rate of normal T cell

K_N carrying capacity of normal T cell

K_C carrying capacity of CAR T cell

ρ_C baseline net growth rate of CAR T cell

b immune reconstitution impact

a signalling inefficiency factor provided by all T cells

r_B net growth rate of tumor cell

γ_B tumor killing rate caused by CAR T cell

k_B killing rate saturation parameter

τ_1, τ_2, τ_3 constants representing noise intensity

W_1, W_2, W_3 standard Wiener processes

Execution options

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot deterministic path of CAR T cell model (all noises are zero)
exe_deterministic_path_CART = 0;
% plot solution paths when r_B is much smaller than 1/2*tau2^2 -- Fig1a
exe_stochastic_path_CART_1a = 0;
% plot solution paths when r_B is much larger than 1/2*tau2^2 -- Fig1b
exe_stochastic_path_CART_1b = 0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot solution paths and histograms of time to cure and time to progress
% in 3 different treatment Protocols
```

```
% Protocol (i): Lymphodepletion before day 0 and one CAR dose at day 0;

% Protocol (ii): Lymphodepletion before day 0, first CAR dose at day 0,
% and second CAR dose at day 15;

% Protocol (iii): Lymphodepletion before day 0, first CAR dose at day 0,
% second lymphodepletion at day 10, and second CAR dose at day 15;

% plot tumor-eradicated and tumor-progression paths for Protocol (i) -- Fig2b
exe_extract_tumor_path_CART = 0;
% plot histogram of time to cure and time to progress for Protocol (i)
exe_cure_and_progress_CART = 0; % Fig2a
% plot a typical solution path for Protocol (ii) -- Fig3b
exe_solution_path_second_dose_no_LD = 0;
% plot histogram of time to cure and time to progress for Protocol (ii)
exe_second_dose_no_LD = 0; % Fig3a
% plot a typical solution path for Protocol (iii) -- Fig4b
exe_solution_path_second_dose_LD = 0;
% plot histogram of time to cure and time to progress for Protocol (iii)
exe_second_dose_LD = 0; % Fig4a
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot probability of cure as a function of hyper-parameter sigma
% where sigma = [0.001 0.01 0.05 0.1 0.15 0.2 0.25]
% plot histogram of time to cure wrt sigma using violin plot
exe_violin_plot = 1; % Fig 5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot probability of cure as a function of the time of 2nd CAR dose
exe_probability_of_cure = 0; % Fig6
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Violin plot - Figure 5

```
if exe_violin_plot
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    sigma = [0.001,0.01,0.05,0.1,0.15];
```

```
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    Dt = 5*10^(-3);
    n = 1000;
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%
    m = length(sigma);
    time_to_cure = zeros(n,m);
    time_to_progress = zeros(n,m);
    time_cure = zeros(n,m);
    time_progress = zeros(n,m);
    Kc = zeros(n,m);
    Kp = zeros(n,m);
    PoC = zeros(1,m); % Probability of cure
    PoP = zeros(1,m); % Probability of progress
    simulated_data = cell(1,m); % create simulated data of time-to-cure for violin
plot
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for j = 1:m
        for i = 1:n
            [Kc(i,j),time_cure(i,j),Kp(i,j),time_progress(i,j)] = ...
            cure_or_progress(base_para,sigma(j),noise,Dt,Xzero);
            if Kc(i,j) >= 1
                time_to_cure(i,j) = time_cure(i,j);
                PoC(j) = PoC(j) + 1;
            end
            if Kp(i,j) >= 1
                time_to_progress(i,j) = time_progress(i,j);
                PoP(j) = PoP(j) + 1;
            end
        end
        simulated_data{j} = nonzeros(time_to_cure(:,j));
        PoC(j) = PoC(j)/n;
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Plot probability of cure as a function of sigma
    %%%%%%%%%%
    h1 = subplot(121);
    set(gca,'FontName','Times','FontSize', 16);
    plot(sigma,PoC,'.','MarkerSize',25),
    xlabel('patient std $\sigma$','Interpreter','latex',...
           'FontName', 'Times','FontSize',16);
    ylabel('probability of cure', 'FontName', 'Times','FontSize',16);
    ylim([0 1])
    title('(a) Effect of parameter variability on probability of cure',...
          'FontName','Times','FontSize',16)
    %%%%%%%%%%%
    % Plot histogram of time-to-cure for each value of sigma using violin plot
    h2 = subplot(122);
    colormat = [
    255,255,204;
```

```
    217,240,163;
    173,221,142;
    120,198,121;
    65,171,93;
    35,132,67;
    0,90,50]/255;

    set(gca, 'FontName', 'Times', 'FontSize', 16);
    label = {'0.001','0.01','0.05','0.1','0.15'};
    violin(simulated_data,'facecolor',colormat,'edgecolor','k','bw',0.3);
    xlabel('patient std $\sigma$','Interpreter','latex',...
           'FontName', 'Times','FontSize',16);
    ylabel('time to cure', 'FontName', 'Times','FontSize',16);
    set(gca,'xTickLabel',label)
    % ylim([0 140])
    title('(b) Effect of parameter variability on time to cure',...
          'FontName','Times','FontSize',16)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    set(0,'Units','normalized')
    set(h2,'position',[.57 .12 .4 .8])
    set(h1,'position',[.07 .12 .4 .8])
    set(gcf, 'PaperSize', [12 5], 'PaperPosition', [0 0 12 5])
    print('Fig5','-dpdf')
end
```

Plot solution paths of deterministic CAR T cell model

```
if exe_deterministic_path_CART
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    tspan = [0,1000];
    func = @f_drift;
    [Time,Xrk] = ode45(@(t,X) func(X,base_para),tspan,Xzero);
    % Plot solution path
    clf
    subplot(3,1,1)
    plot(Time,Xrk(:,1),'g*','LineWidth',0.5,'MarkerSize',2)
    xlabel('time (days)'); ylabel('normal T cells')

    subplot(3,1,2)
    plot(Time,Xrk(:,2),'b-','LineWidth',1)
```

```
    xlabel('time (days)'); ylabel('CAR T cells')

    subplot(3,1,3)
    plot(Time,Xrk(:,3),'r-','LineWidth',1)
    xlabel('time (days)'); ylabel('tumor cells')
    axis([0 100 0 10^(2)])
end
```

Plot solution paths of stochastic CAR T Cell model

```
if exe_stochastic_path_CART_1a
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-3); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    T1 = 0;
    T2 = 1000;
    Dt = 5*10^(-3);
    rng(250);
    [Time,Xrk] = RK_stochastic_CART(base_para,noise,T1,T2,Dt,Xzero);
    Xrk1 = [Xzero(1),Xrk(1,:)];
    Xrk2 = [Xzero(2),Xrk(2,:)];
    Xrk3 = [Xzero(3),Xrk(3,:)];
    % Plot solution path
    clf
    h1 = subplot(2,1,1);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
    plot(Time,Xrk2,'b-','LineWidth',1.5), hold on
    plot(Time,Xrk3,'r-','LineWidth',1.5);
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
    ylabel('T cell count','FontName', 'Times','FontSize',18);
    legend('normal T cells','CAR T cells','tumor cells', 'FontName',...
        'Times','FontSize',18,'Location','southwest');
    set(gca, 'YScale', 'log');
    ylim([0.01 20*K_N]);
    xlim([0 50]);
    title(['\textbf{(a) transient dynamics of solution paths as} ' ...
        '$r_B<\frac{\tau_2^2}{2}$'],'Interpreter','latex','FontSize',18)
```

```
    h2 = subplot(2,1,2);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
    plot(Time,Xrk2,'b-','LineWidth',1.5), hold on
    plot(Time,Xrk3,'r-','LineWidth',1.5);
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
    ylabel('T cell count','FontName','Times','FontSize',18);
    legend('normal T cells','CAR T cells','tumor cells', 'FontName',...
           'Times','FontSize',18,'Location','best');
    set(gca, 'YScale', 'log');
    ylim([10^(-10) 20*K_N]);
    xlim([0 1000]);
    title(['\textbf{(b) long-term dynamics of solution paths as} ' ...
           '$r_B<\frac{\tau_2^2}{2}$'],'Interpreter','latex','FontSize',18)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig1a','-dpdf')
end

if exe_stochastic_path_CART_1b
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 15*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    T1 = 0;
    T2 = 1000;
    Dt = 5*10^(-3);
    rng(350);
    [Time,Xrk] = RK_stochastic_CART(base_para,noise,T1,T2,Dt,Xzero);
    Xrk1 = [Xzero(1),Xrk(1,:)];
    Xrk2 = [Xzero(2),Xrk(2,:)];
    Xrk3 = [Xzero(3),Xrk(3,:)];
    % Plot solution path
    clf
    h1 = subplot(2,1,1);
```

```
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
    plot(Time,Xrk2,'b-','LineWidth',1.5), hold on
    plot(Time,Xrk3,'r-','LineWidth',1.5);
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
    ylabel('T cell count','FontName', 'Times','FontSize',18);
    legend('normal T cells','CAR T cells','tumor cells', 'FontName',...
           'Times','FontSize',18,'Location','best');
    set(gca, 'YScale', 'log');
    ylim([0.01 20*K_N]);
    xlim([0 50]);
    title(['\textbf{(c) transient dynamics of solution paths as} ' ...
           '$r_B>\frac{\tau_2^2}{2}$'],'Interpreter','latex','FontSize',18)

    h2 = subplot(2,1,2);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
    plot(Time,Xrk2,'b-','LineWidth',1.5), hold on
    plot(Time,Xrk3,'r-','LineWidth',1.5);
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
    ylabel('T cell count','FontName', 'Times','FontSize',18);
    legend('normal T cells','CAR T cells','tumor cells', 'FontName',...
           'Times','FontSize',18,'Location','best');
    set(gca, 'YScale', 'log');
    ylim([10^(-10) 20*K_N]);
    xlim([0 1000]);
    title(['\textbf{(d) long-term dynamics of solution paths as} ' ...
           '$r_B>\frac{\tau_2^2}{2}$'],'Interpreter','latex','FontSize',18)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig1b','-dpdf')
end
```

Extract tumor paths - cure and progress

```
if exe_extract_tumor_path_CART
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
```

```
    tau2 = 0.2;
    tau3 = 0.1;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    T1 = 0;
    T2 = 1000;
    Dt = 5*10^(-3);
    [Time,Xrk_cure,Xrk_progress] =
extract_tumor_path(base_para,noise,T1,T2,Dt,Xzero);
    % Plot solution path
    clf

    h1 = subplot(2,1,1);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk_cure,'r-','LineWidth',1.5)
    xlabel('time (days)','FontName','Times','FontSize',18);
    ylabel('tumor cells','FontName','Times','FontSize',18);
    set(gca, 'YScale', 'log');
    title('(c) tumor sample path - complete response or cure','FontName','Times',...
          'FontSize',18)
    xlim([0 100]);
    ylim([0 1.5*Xzero(3)])

    h2 = subplot(2,1,2);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk_progress,'r-','LineWidth',1.5), hold on
    xlabel('time (days)','FontName','Times','FontSize',18);
    ylabel('tumor cells','FontName','Times','FontSize',18);
    set(gca, 'YScale', 'log');
    title('(d) tumor sample path - progression','FontName','Times','FontSize',18)
    axis([0 500 0 1.5*Xzero(3)])

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig2b','-dpdf')
end
```

Time to cure & Time to progression

```
if exe_cure_and_progress_CART
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
```

```
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    sigma = 0.05;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    Dt = 5*10^(-3);
    n = 1000;
    Time_cure = zeros(1,n);
    Time_progress = zeros(1,n);
    Kc = zeros(1,n);
    Kp = zeros(1,n);
    time_to_cure = zeros(1,n);
    time_to_progress = zeros(1,n);
    PoC = 0;
    PoP = 0;
    % rng(500)
    for i = 1:n
        [Kc(i),Time_cure(i),Kp(i),Time_progress(i)] = cure_or_progress(base_para,...

sigma,noise,Dt,Xzero);
        if Kc(i) >= 1
            time_to_cure(i) = Time_cure(i);
            PoC = PoC + 1;
        end
        if Kp(i) >= 1
            time_to_progress(i) = Time_progress(i);
            PoP = PoP + 1;
        end
    end
    time_to_cure = nonzeros(time_to_cure);
    probability_of_cure = PoC/n;
    time_to_progress = nonzeros(time_to_progress);
    probability_of_progress = PoP/n;

    % plot histogram

    h1 = subplot(211);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    histogram(time_to_cure,50,'Normalization','probability',...
              'FaceColor','green','EdgeColor','white','LineWidth',1)
    xlabel('days post CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(a) time to cure distribution','FontName','Times','FontSize',18)
    xlim([0 140])
    ylim([0 0.15])
    txt1 = ['Probability of cure = ' num2str(probability_of_cure)];
    text(90,0.1,txt1,'FontName','Times','FontSize',14)
```

```
    h2 = subplot(212);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    histogram(time_to_progress,50,'Normalization','probability',...
              'FaceColor','red','EdgeColor','white','LineWidth',1)
    xlabel('days post CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(b) time to progress distribution (defined as 1.2 of initial tumor)',...
          'FontName','Times','FontSize',18)
    xlim([0 600])
    ylim([0 0.15])
    txt2 = ['Probability of progress = ' num2str(probability_of_progress)];
    text(360,0.1,txt2,'FontName','Times','FontSize',14)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig2a','-dpdf')
end
```

Extract solution paths - 2nd dose w/o LD

```
if exe_solution_path_second_dose_no_LD
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    T1 = 0;
    second_CART_time = 15;
    T2 = 1000;
    Dt = 5*10^(-3);
    [Time,Xrk1,Xrk2,Xrk3] = extract_solution_paths_2nd_dose_no_LD(base_para,...
                                  noise,T1,T2,Dt,Xzero,second_CART_time);
    % Plot solution path
    clf
    h1 = subplot(2,1,1);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
```

```
    plot(Time,Xrk2,'b-','LineWidth',1.5)
    xlabel('time (days)','FontName','Times','FontSize',18);
    ylabel('T cell count','FontName','Times','FontSize',18);
    legend('normal T cells','CAR T cells','FontName','Times','FontSize',18);
    set(gca, 'YScale', 'log');
    ylim([10^8 K_N]);
    xticks(0:5:50);
    xtickangle(0);
    xlim([0 50]);
    txt = '\downarrow 2nd CAR w/o LD: day 15';
    text(second_CART_time,1.5*Xrk2(second_CART_time/Dt),txt,'FontName','Times',...
        'FontSize',18);
    title('(c) Second CAR without lymphodepletion
(LD)','FontName','Times','FontSize',18)

    h2 = subplot(2,1,2);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk3,'r-','LineWidth',1.5)
    xlabel('time (days)','FontName','Times','FontSize',18);
    ylabel('tumor cells','FontName','Times','FontSize',18);
    set(gca, 'YScale', 'log');
    xlim([0 200])
    ylim([0 1.2*Xzero(3)])
    title('(d) Second CAR without lymphodepletion
(LD)','FontName','Times','FontSize',18)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig3b','-dpdf')
end
```

Second dose with no LD

```
if exe_second_dose_no_LD
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    sigma = 0.05;
```

```
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    Dt = 5*10^(-3);
    second_CART_time = 15;
    n = 1000;
    Time_cure = zeros(1,n);
    Time_progress = zeros(1,n);
    Kc = zeros(1,n);
    Kp = zeros(1,n);
    time_to_cure = zeros(1,n);
    time_to_progress = zeros(1,n);
    PoC = 0;
    PoP = 0;
    % rng(300)
    for i = 1:n
        [Kc(i),Time_cure(i),Kp(i),Time_progress(i)] =
second_dose_no_LD(base_para,...

sigma,noise,Dt,Xzero,second_CART_time);
        if Kc(i) >= 1
            time_to_cure(i) = Time_cure(i);
            PoC = PoC + 1;
        end
        if Kp(i) >= 1
            time_to_progress(i) = Time_progress(i);
            PoP = PoP + 1;
        end
    end
    time_to_cure = nonzeros(time_to_cure);
    probability_of_cure = PoC/n;
    time_to_progress = nonzeros(time_to_progress);
    probability_of_progress = PoP/n;

    % Plot histograms

    h1 = subplot(211);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    histogram(time_to_cure,50,'Normalization','probability',...
              'FaceColor','green','EdgeColor','white','LineWidth',1)
    xlabel('days post first CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(a) time to cure distribution - second CAR without LD',...
          'FontName','Times','FontSize',18)
    xlim([0 140])
    ylim([0 0.15])
    txt1 = ['Probability of cure = ' num2str(probability_of_cure)];
    text(90,0.1,txt1,'FontName','Times','FontSize',14)

    h2 = subplot(212);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
```

```
    histogram(time_to_progress,50,'Normalization','probability',...
              'FaceColor','red','EdgeColor','white','LineWidth',1)
    xlabel('days post first CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(b) time to progress distribution - second CAR without LD',...
          'FontName','Times','FontSize',18)
    xlim([0 600])
    ylim([0 0.15])
    txt2 = ['Probability of progress = ' num2str(probability_of_progress)];
    text(360,0.1,txt2,'FontName','Times','FontSize',14)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig3a','-dpdf')
end
```

Extract solution paths - 2nd dose w/ LD

```
if exe_solution_path_second_dose_LD
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    T1 = 0;
    LD_time = 10;
    second_CART_time = LD_time + 5;
    T2 = 1000;
    Dt = 5*10^(-3);
    [Time,Xrk1,Xrk2,Xrk3] = extract_solution_paths_2nd_dose_LD(base_para,...
                            noise,T1,T2,Dt,Xzero,LD_time,second_CART_time);
    % Plot solution path
    clf
    h1 = subplot(2,1,1);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk1,'g-','LineWidth',1.5), hold on
    plot(Time,Xrk2,'b-','LineWidth',1.5)
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
```

```
    ylabel('T cell count', 'FontName', 'Times','FontSize',18);
    legend('normal T cells','CAR T cells', 'FontName', 'Times','FontSize',18,...
           'Location','SouthEast');
    set(gca, 'YScale', 'log');
    ylim([10^6 2*K_N]);
    xticks(0:5:50);
    xtickangle(0);
    xlim([0 50]);
    txt1 = '\leftarrow LD day 10';
    text(10,Xrk1(10/Dt),txt1, 'FontName', 'Times','FontSize',18);
    txt2 = '\leftarrow 2nd CAR day 15';
    text(15,Xzero(2),txt2, 'FontName', 'Times','FontSize',18);
    title('(c) Second CAR with lymphodepletion (LD)', 'FontName',
'Times','FontSize',18)

    h2 = subplot(2,1,2);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    plot(Time,Xrk3,'r-','LineWidth',1.5)
    xlabel('time (days)', 'FontName', 'Times','FontSize',18);
    ylabel('tumor cells', 'FontName', 'Times','FontSize',18);
    set(gca, 'YScale', 'log');
    xlim([0 200])
    ylim([0 1.2*Xzero(3)])
    title('(d) Second CAR with lymphodepletion (LD)', 'FontName',
'Times','FontSize',18)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig4b','-dpdf')

end
```

Second dose with LD

```
if exe_second_dose_LD
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 1.15; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
```

```
    sigma = 0.05;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    Dt = 5*10^(-3);
    LD_time = 10;
    second_CART_time = LD_time + 5;
    n = 1000;
    Time_cure = zeros(1,n);
    Time_progress = zeros(1,n);
    Kc = zeros(1,n);
    Kp = zeros(1,n);
    time_to_cure = zeros(1,n);
    time_to_progress = zeros(1,n);
    PoC = 0;
    PoP = 0;
    % rng(100)
    for i = 1:n
        [Kc(i),Time_cure(i),Kp(i),Time_progress(i)] = second_dose_LD(base_para,...

sigma,noise,Dt,Xzero,LD_time,second_CART_time);
        if Kc(i) >= 1
            time_to_cure(i) = Time_cure(i);
            PoC = PoC + 1;
        end
        if Kp(i) >= 1
            time_to_progress(i) = Time_progress(i);
            PoP = PoP + 1;
        end
    end
    time_to_cure = nonzeros(time_to_cure);
    probability_of_cure = PoC/n;
    time_to_progress = nonzeros(time_to_progress);
    probability_of_progress = PoP/n;

    h1 = subplot(211);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    histogram(time_to_cure,50,'Normalization','probability',...
              'FaceColor','green','EdgeColor','white','LineWidth',1)
    xlabel('days post first CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(a) time to cure distribution - second CAR with LD','FontName',...
          'Times','FontSize',18)
    xlim([0 140])
    ylim([0 0.15])
    txt1 = ['Probability of cure = ' num2str(probability_of_cure)];
    text(90,0.1,txt1,'FontName','Times','FontSize',14)

    h2 = subplot(212);
    set(gca, 'FontName', 'Times', 'FontSize', 18);
    histogram(time_to_progress,50,'Normalization','probability',...
```

```
            'FaceColor','red','EdgeColor','white','LineWidth',1)
    xlabel('days post first CAR infusion','FontName','Times','FontSize',18);
    ylabel('probability','FontName','Times','FontSize',18);
    title('(b) time to progress distribution - second CAR with LD','FontName',...
        'Times','FontSize',18)
    xlim([0 600])
    ylim([0 0.15])
    txt2 = ['Probability of progress = ' num2str(probability_of_progress)];
    text(360,0.1,txt2,'FontName','Times','FontSize',14)

    set(0,'Units','normalized')
    set(h2,'position',[.12 .1 .8 .35])
    set(h1,'position',[.12 .58 .8 .35])
    set(gcf, 'PaperSize', [8 8], 'PaperPosition', [0 0 8 8])
    print('Fig4a','-dpdf')
end
```

Probability of cure - 2nd dose with and without LD

```
if exe_probability_of_cure
    K_N = 2.5*10^(11); % cells - normal T cell carrying capacity
    K_C = 6.96*10^(10); % cells - CAR T cell carrying capacity
    r_N = 1.7*10^(-1); % day^(-1) - normal T cell growth rate
    rho_C = 2.51*10^(-2); % day^(-1) - baseline CAR T cell net growth rate
    a = 4.23*10^(-1); % dimensionless - signaling ineffieciency factor in r_C
    b = 5.25*10^(-1); % day^(-1) - immune reconstitution impact in r_C
    r_B = 10*10^(-2); % day^(-1) - tumor net growth rate
    k_B = 2.024*10^9; % cells - killing rate saturation parameter
    gamma_B = 0.9; % day^(-1) - tumor killing rate (by effector CAR)
    base_para = [K_N,K_C,r_N,rho_C,a,b,r_B,k_B,gamma_B];
    tau1 = 0.1;
    tau2 = 0.2;
    tau3 = 0.1;
    sigma = 0.05;
    noise = [tau1,tau2,tau3];
    Xzero = [3*10^9; 1.8*10^8; 9.486*10^(10)];
    Dt = 5*10^(-3);
    T1 = 0;
    T2 = 1000;
    Time_2nd_dose_LD = [7 11 15 19 23 27 31 35 39 43];
    Time_2nd_dose_nLD = [7 11 15 19 23 27 31 35 39 43];
    LD_time = Time_2nd_dose_LD - 5;
    m_LD = length(Time_2nd_dose_LD);
    m_nLD = length(Time_2nd_dose_nLD);
    n = 100;
    Theta_LD = zeros(1,n);
    Theta_nLD = zeros(1,n);
    cure_LD = zeros(m_LD,n);
    cure_nLD = zeros(m_nLD,n);
    PoC_LD = zeros(1,m_LD);
```

```
    PoC_nLD = zeros(1,m_nLD);
    % Calculate probability of cure for 2nd dose w/ LD
    for i = 1:m_LD
        for j = 1:n
            [Time,Xrk1,Xrk2,Xrk3] = extract_solution_paths_2nd_dose_LD(base_para,...

sigma,noise,T1,T2,Dt,Xzero,LD_time(i),Time_2nd_dose_LD(i));
            if min(Xrk3) <= 1
                cure_LD(i,j) = 1;
            end
        end
        PoC_LD(i) = sum(cure_LD(i,:))/n;
    end
    % Calculate probability of cure for 2nd dose w/o LD
    for i = 1:m_nLD
        for j = 1:n
            [Time,Xrk1,Xrk2,Xrk3] =
extract_solution_paths_2nd_dose_no_LD(base_para,...

sigma,noise,T1,T2,Dt,Xzero,Time_2nd_dose_nLD(i));
            if min(Xrk3) <= 1
                cure_nLD(i,j) = 1;
            end
        end
        PoC_nLD(i) = sum(cure_nLD(i,:))/n;
    end
    % Plot PoC_LD and PoC_nLD
    clf

    h1 = subplot(1,2,1);
    set(gca, 'FontName', 'Times', 'FontSize', 16);
    plot(Time_2nd_dose_LD,PoC_LD,'.','MarkerSize',25),
    xlabel('Day of second CAR infusion', 'FontName', 'Times','FontSize',16);
    ylabel('Probability of cure', 'FontName', 'Times','FontSize',16);
    xticks([7 11 15 19 23 27 31 35 39 43])
    ylim([0 1])
    title('(a) Second CAR with LD 5 days before','FontName','Times','FontSize',16)

    h2 = subplot(1,2,2);
    set(gca, 'FontName', 'Times', 'FontSize', 16);
    plot(Time_2nd_dose_nLD,PoC_nLD,'.','MarkerSize',25),
    xlabel('Day of second CAR infusion', 'FontName', 'Times','FontSize',16);
    ylabel('Probability of cure', 'FontName', 'Times','FontSize',16);
    xticks([7 11 15 19 23 27 31 35 39 43])
    ylim([0 1])
    title('(b) Second CAR without LD','FontName','Times','FontSize',16)

    set(0,'Units','normalized')
    set(h2,'position',[.57 .12 .4 .8])
    set(h1,'position',[.07 .12 .4 .8])
```

```
    set(gcf, 'PaperSize', [12 5], 'PaperPosition', [0 0 12 5])
    print('Fig6','-dpdf')
end
```

Local function - tumor paths for cure, and progress

```
function [Time,Xrk_cure,Xrk_progress] =
extract_tumor_path(base_para,sigma,noise,T1,T2,Dt,Xzero)

rho_C = base_para(4);
r_B = base_para(7);
gamma_B = base_para(9);
% Pick values of rho_C, r_B, gamma_B from normal distribution
% with mean rho_C, r_B, gamma_B and std sigma*rho_C,
% sigma*r_B, sigma*gamma_B, respectively
base_para(4) = rho_C + randn*sigma*rho_C;
base_para(7) = r_B + randn*sigma*r_B;
base_para(9) = gamma_B + randn*sigma*gamma_B;
%k1 = 0;
k2 = 0;
k3 = 0;
while 1
    [Time,Xrk] = RK_stochastic_CART(base_para,noise,T1,T2,Dt,Xzero);
    Xrk3 = Xrk(3,:);
    %Time_fail = (Time >= 0 & Time <= 10);
    Time_cure = (Time > 10 & Time <= 150);
    Time_progress = (Time > 150 & Time <= 500);
    % Check the solution path is "fail-path" or not
    %if max(Xrk3(Time_fail)) >= 1.2*Xzero(3)
    %    k1 = k1 + 1;
    %    Xrk_fail = [Xzero(3),Xrk3];
    %end
    % Check the solution path is "cure-path" or not
    if min(Xrk3(Time_cure)) <= 1
        k2 = k2 + 1;
        Xrk_cure = [Xzero(3),Xrk3];
    end
    % Check the solution path is "progress-path" or not
    if min(Xrk3(Time_cure)) > 1 && max(Xrk3(Time_progress)) >= 1.2*Xzero(3)
        k3 = k3 + 1;
        Xrk_progress = [Xzero(3),Xrk3];
    end
    % Check if all kinds of paths are extracted
    if k2 >= 1 && k3 >= 1
        break;
    end
end
end
```

Local function - time to cure or time to progression

```
function [Kc,Time_cure,Kp,Time_progress] =
cure_or_progress(base_para,sigma,noise,Dt,Xzero)
dt = (Dt)^2;
N = Dt/dt;

rho_C = base_para(4);
r_B = base_para(7);
gamma_B = base_para(9);
% Pick values of rho_C, r_B, gamma_B from normal distribution
% with mean rho_C, r_B, gamma_B and std sigma*rho_C,
% sigma*r_B, sigma*gamma_B, respectively
base_para(4) = rho_C + randn*sigma*rho_C;
base_para(7) = r_B + randn*sigma*r_B;
base_para(9) = gamma_B + randn*sigma*gamma_B;

Kc = 0; % Track the number of patients who are cured
Kp = 0; % Track the number of patients who progressed
Time = 0;
Time_cure = 0;
Time_progress = 0;
Xtilde = zeros(3,7);
Xfinal = Xzero;

while 1
    % EM method for dW32,dW31,dW3,dW23,dW21,dW2,dW13,dW12,dW1
    X1 = 0; X2 = 0; X3 = 0;
    Y1 = 0; Y2 = 0; Y3 = 0;
    Z1 = 0; Z2 = 0; Z3 = 0;
    for m = 1:N
        dW1 = sqrt(dt)*randn;
        dW2 = sqrt(dt)*randn;
        dW3 = sqrt(dt)*randn;
        X1 = X1 + X3*dW2;
        X2 = X2 + X3*dW1;
        X3 = X3 + dW3;
        Y1 = Y1 + Y3*dW3;
        Y2 = Y2 + Y3*dW1;
        Y3 = Y3 + dW2;
        Z1 = Z1 + Z3*dW3;
        Z2 = Z2 + Z3*dW2;
        Z3 = Z3 + dW1;
    end

    % Runge-Kutta method for the stochastic CAR T cell model
    X = Xfinal;
    Xtilde(:,1) = X + f_drift(X,base_para)*Dt;
    Xtilde(:,2) = Xtilde(:,1) + g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
```

```
                              + g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,3) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              + g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,4) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xtilde(:,5) = Xtilde(:,1) - g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                              - g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,6) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              - g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,7) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xfinal = X + 0.5*(f_drift(X,base_para) + f_drift(Xtilde(:,1),base_para))*Dt...
               + Z3*g1_diff(X,base_para,noise)...
               + 0.5*(g1_diff(Xtilde(:,2),base_para,noise)...
               - g1_diff(Xtilde(:,5),base_para,noise))*sqrt(Dt)...
               + Y3*g2_diff(X,base_para,noise)...
               + 0.5*(g2_diff(Xtilde(:,3),base_para,noise)...
               - g2_diff(Xtilde(:,6),base_para,noise))*sqrt(Dt)...
               + X3*g3_diff(X,base_para,noise)...
               + 0.5*(g3_diff(Xtilde(:,4),base_para,noise)...
               - g3_diff(Xtilde(:,7),base_para,noise))*sqrt(Dt);
    Time = Time + Dt;
    if Xfinal(3) <= 1
        Kc = Kc + 1;
        Time_cure = Time;
    end
    if (Xfinal(3) > 1) && (Xfinal(3) >= 1.2*Xzero(3)) && (Time>10)
        Kp = Kp + 1;
        Time_progress = Time;
    end
    if (Kc >= 1) || (Kp >= 1)
        break;
    end
end
end
```

Local function - solution paths for 2nd dose w/ LD

```
function [Time,Xrk1,Xrk2,Xrk3] = extract_solution_paths_2nd_dose_LD(base_para,...

sigma,noise,T1,T2,Dt,Xzero,LD_time,second_CART_time)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   rho_C = base_para(4);
    r_B = base_para(7);
```

```
    gamma_B = base_para(9);
    % Pick values of rho_C, r_B, gamma_B from normal distribution
    % with mean rho_C, r_B, gamma_B and std sigma*rho_C,
    % sigma*r_B, sigma*gamma_B, respectively
    base_para(4) = rho_C + randn*sigma*rho_C;
    base_para(7) = r_B + randn*sigma*r_B;
    base_para(9) = gamma_B + randn*sigma*gamma_B;
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    [Time_1,Xrk_1] = RK_stochastic_CART(base_para,noise,T1,LD_time,Dt,Xzero);
    Xzero_2 = zeros(3,1);
    Xzero_2(1) = base_para(1)*(Xzero(1)/base_para(1))^(exp(5*base_para(3)));
    Xzero_2(2) = 0;
    Xzero_2(3) = Xrk_1(3,end);
    [Time_2,Xrk_2] =
RK_stochastic_CART(base_para,noise,LD_time,second_CART_time,Dt,Xzero_2);
    Xzero_3 = zeros(3,1);
    Xzero_3(1) = Xrk_2(1,end);
    Xzero_3(2) = Xzero(2);
    Xzero_3(3) = Xrk_2(3,end);
    [Time_3,Xrk_3] =
RK_stochastic_CART(base_para,noise,second_CART_time,T2,Dt,Xzero_3);
    Time = [Time_1(1:end-1),LD_time + Time_2(1:end-1),second_CART_time + Time_3];
    Xrk1 =
[Xzero(1),Xrk_1(1,1:end-1),Xzero_2(1),Xrk_2(1,1:end-1),Xzero_3(1),Xrk_3(1,:)];
    Xrk2 =
[Xzero(2),Xrk_1(2,1:end-1),Xzero_2(2),Xrk_2(2,1:end-1),Xzero_3(2),Xrk_3(2,:)];
    Xrk3 =
[Xzero(3),Xrk_1(3,1:end-1),Xzero_2(3),Xrk_2(3,1:end-1),Xzero_3(3),Xrk_3(3,:)];
end
```

Local function 2nd dose with Lymphodepletion

```
function [Kc,Time_cure,Kp,Time_progress] = second_dose_LD(base_para,...
                            sigma,noise,Dt,Xzero,LD_time,second_CART_time)
dt = (Dt)^2;
N = Dt/dt;

rho_C = base_para(4);
r_B = base_para(7);
gamma_B = base_para(9);
% Pick values of rho_C, r_B, gamma_B from normal distribution
% with mean rho_C, r_B, gamma_B and std sigma*rho_C,
% sigma*r_B, sigma*gamma_B, respectively
base_para(4) = rho_C + randn*sigma*rho_C;
base_para(7) = r_B + randn*sigma*r_B;
base_para(9) = gamma_B + randn*sigma*gamma_B;

Kc = 0;
Kp = 0;
Time_cure = 0;
```

```
Time_progress = 0;
Time = 0;
Xtilde = zeros(3,7);
Xfinal = Xzero;
while 1
    % EM method for dW32,dW31,dW3,dW23,dW21,dW2,dW13,dW12,dW1
    X1 = 0; X2 = 0; X3 = 0;
    Y1 = 0; Y2 = 0; Y3 = 0;
    Z1 = 0; Z2 = 0; Z3 = 0;
    for m = 1:N
        dW1 = sqrt(dt)*randn;
        dW2 = sqrt(dt)*randn;
        dW3 = sqrt(dt)*randn;
        X1 = X1 + X3*dW2;
        X2 = X2 + X3*dW1;
        X3 = X3 + dW3;
        Y1 = Y1 + Y3*dW3;
        Y2 = Y2 + Y3*dW1;
        Y3 = Y3 + dW2;
        Z1 = Z1 + Z3*dW3;
        Z2 = Z2 + Z3*dW2;
        Z3 = Z3 + dW1;
    end

    % Runge-Kutta method for the stochastic HPV model
    X = Xfinal;
    Xtilde(:,1) = X + f_drift(X,base_para)*Dt;
    Xtilde(:,2) = Xtilde(:,1) + g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                              + g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,3) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              + g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,4) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xtilde(:,5) = Xtilde(:,1) - g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                              - g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,6) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              - g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,7) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xfinal = X + 0.5*(f_drift(X,base_para)+f_drift(Xtilde(:,1),base_para))*Dt...
              + Z3*g1_diff(X,base_para,noise)...
              + 0.5*(g1_diff(Xtilde(:,2),base_para,noise)...
              - g1_diff(Xtilde(:,5),base_para,noise))*sqrt(Dt)...
              + Y3*g2_diff(X,base_para,noise)...
```

```
                + 0.5*(g2_diff(Xtilde(:,3),base_para,noise)...
                - g2_diff(Xtilde(:,6),base_para,noise))*sqrt(Dt)...
                + X3*g3_diff(X,base_para,noise)...
                + 0.5*(g3_diff(Xtilde(:,4),base_para,noise)...
                - g3_diff(Xtilde(:,7),base_para,noise))*sqrt(Dt);
    Time = Time + Dt;
    % Lymphodepletion at LD_time
    if Time == LD_time
        Xfinal(1) = base_para(1)*(Xzero(1)/base_para(1))^(exp(5*base_para(3)));
        Xfinal(2) = 0;
    %    Xfinal(2) = base_para(2)*(Xzero(2)/base_para(2))^(exp(5*base_para(4)));
    end
    % Second CART dose at second_CART_time
    if Time == second_CART_time
        Xfinal(2) = Xzero(2);
    end
    % Check whether tumor cells are eradicated or progressed
    if Xfinal(3) <= 1
        Kc = Kc + 1;
        Time_cure = Time;
    end
    if (Xfinal(3) > 1) && (Xfinal(3) >= 1.2*Xzero(3)) && (Time>10)
        Kp = Kp + 1;
        Time_progress = Time;
    end
    if (Kc >= 1) || (Kp >= 1)
        break;
    end
end
end
```

Local function - solution paths for 2nd dose w/o LD

```
function [Time,Xrk1,Xrk2,Xrk3] = extract_solution_paths_2nd_dose_no_LD(base_para,...

sigma,noise,T1,T2,Dt,Xzero,second_CART_time)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    rho_C = base_para(4);
    r_B = base_para(7);
    gamma_B = base_para(9);
    % Pick values of rho_C, r_B, gamma_B from normal distribution
    % with mean rho_C, r_B, gamma_B and std sigma*rho_C,
    % sigma*r_B, sigma*gamma_B, respectively
    base_para(4) = rho_C + randn*sigma*rho_C;
    base_para(7) = r_B + randn*sigma*r_B;
    base_para(9) = gamma_B + randn*sigma*gamma_B;
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    [Time_1,Xrk_1] =
RK_stochastic_CART(base_para,noise,T1,second_CART_time,Dt,Xzero);
    Xzero_2 = ones(3,1);
```

```
    Xzero_2(1) = Xrk_1(1,end);
    Xzero_2(2) = Xzero(2) + Xrk_1(2,end);
    Xzero_2(3) = Xrk_1(3,end);
    [Time_2,Xrk_2] =
RK_stochastic_CART(base_para,noise,second_CART_time,T2,Dt,Xzero_2);
    Time = [Time_1(1:end-1),second_CART_time + Time_2];
    Xrk1 = [Xzero(1),Xrk_1(1,1:end-1),Xzero_2(1),Xrk_2(1,:)];
    Xrk2 = [Xzero(2),Xrk_1(2,1:end-1),Xzero_2(2),Xrk_2(2,:)];
    Xrk3 = [Xzero(3),Xrk_1(3,1:end-1),Xzero_2(3),Xrk_2(3,:)];
end
```

Local function 2nd dose with No Lymphodepletion

```
function [Kc,Time_cure,Kp,Time_progress] = second_dose_no_LD(base_para,...
                                     sigma,noise,Dt,Xzero,second_CART_time)
dt = (Dt)^2;
N = Dt/dt;

rho_C = base_para(4);
r_B = base_para(7);
gamma_B = base_para(9);
% Pick values of rho_C, r_B, gamma_B from normal distribution
% with mean rho_C, r_B, gamma_B and std sigma*rho_C,
% sigma*r_B, sigma*gamma_B, respectively
base_para(4) = rho_C + randn*sigma*rho_C;
base_para(7) = r_B + randn*sigma*r_B;
base_para(9) = gamma_B + randn*sigma*gamma_B;

Kc = 0;
Kp = 0;
Time = 0;
Time_cure = 0;
Time_progress = 0;
Xtilde = zeros(3,7);
Xfinal = Xzero;

while 1
    % EM method for dW32,dW31,dW3,dW23,dW21,dW2,dW13,dW12,dW1
    X1 = 0; X2 = 0; X3 = 0;
    Y1 = 0; Y2 = 0; Y3 = 0;
    Z1 = 0; Z2 = 0; Z3 = 0;
    for m = 1:N
        dW1 = sqrt(dt)*randn;
        dW2 = sqrt(dt)*randn;
        dW3 = sqrt(dt)*randn;
        X1 = X1 + X3*dW2;
        X2 = X2 + X3*dW1;
        X3 = X3 + dW3;
        Y1 = Y1 + Y3*dW3;
        Y2 = Y2 + Y3*dW1;
```

```
        Y3 = Y3 + dW2;
        Z1 = Z1 + Z3*dW3;
        Z2 = Z2 + Z3*dW2;
        Z3 = Z3 + dW1;
    end

    % Runge-Kutta method for the stochastic HPV model
    X = Xfinal;
    Xtilde(:,1) = X + f_drift(X,base_para)*Dt;
    Xtilde(:,2) = Xtilde(:,1) + g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                              + g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,3) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              + g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,4) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              + g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              + g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xtilde(:,5) = Xtilde(:,1) - g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                              - g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,6) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                              - g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,7) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                              - g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                              - g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xfinal = X + 0.5*(f_drift(X,base_para)+f_drift(Xtilde(:,1),base_para))*Dt...
               + Z3*g1_diff(X,base_para,noise)...
               + 0.5*(g1_diff(Xtilde(:,2),base_para,noise)...
               - g1_diff(Xtilde(:,5),base_para,noise))*sqrt(Dt)...
               + Y3*g2_diff(X,base_para,noise)...
               + 0.5*(g2_diff(Xtilde(:,3),base_para,noise)...
               - g2_diff(Xtilde(:,6),base_para,noise))*sqrt(Dt)...
               + X3*g3_diff(X,base_para,noise)...
               + 0.5*(g3_diff(Xtilde(:,4),base_para,noise)...
               - g3_diff(Xtilde(:,7),base_para,noise))*sqrt(Dt);
    Time = Time + Dt;

    % Second CART dose at second_CART_time
    if Time == second_CART_time
        Xfinal(2) = Xfinal(2) + Xzero(2);
    end
    % Check whether tumor cells are eradicated or progressed
    if Xfinal(3) <= 1
        Kc = Kc + 1;
        Time_cure = Time;
    end
    if (Xfinal(3) > 1) && (Xfinal(3) >= 1.2*Xzero(3)) && (Time>10)
        Kp = Kp + 1;
```

```
        Time_progress = Time;
    end
    if (Kc >= 1) || (Kp >= 1)
        break;
    end
end
end
```

Local function - Stochastic Runge-Kutta Algorithm

```
function [Time,Xrk] = RK_stochastic_CART(base_para,noise,T1,T2,Dt,Xzero)
T = T2 - T1;
n = T/Dt;
dt = (Dt)^2;
N = Dt/dt;

Time = zeros(1,n+1);
Xrk = zeros(3,n);
Xtilde = zeros(3,7);
Xfinal = Xzero;

for k = 1:n
    % EM method for dW32,dW31,dW3,dW23,dW21,dW2,dW13,dW12,dW1
    X1 = 0; X2 = 0; X3 = 0;
    Y1 = 0; Y2 = 0; Y3 = 0;
    Z1 = 0; Z2 = 0; Z3 = 0;
    for m = 1:N
        dW1 = sqrt(dt)*randn;
        dW2 = sqrt(dt)*randn;
        dW3 = sqrt(dt)*randn;
        X1 = X1 + X3*dW2;
        X2 = X2 + X3*dW1;
        X3 = X3 + dW3;
        Y1 = Y1 + Y3*dW3;
        Y2 = Y2 + Y3*dW1;
        Y3 = Y3 + dW2;
        Z1 = Z1 + Z3*dW3;
        Z2 = Z2 + Z3*dW2;
        Z3 = Z3 + dW1;
    end
    % Runge-Kutta method for the stochastic CAR T cell model
    X = Xfinal;
    Xtilde(:,1) = X + f_drift(X,base_para)*Dt;
    Xtilde(:,2) = Xtilde(:,1) + g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                               + g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                               + g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,3) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                               + g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                               + g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,4) = Xtilde(:,1) + g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
```

```
                                + g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                                + g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xtilde(:,5) = Xtilde(:,1) - g1_diff(X,base_para,noise)*(Z3^2-Dt)/(2*sqrt(Dt))...
                                - g2_diff(X,base_para,noise)*Y2/sqrt(Dt)...
                                - g3_diff(X,base_para,noise)*X2/sqrt(Dt);
    Xtilde(:,6) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z2/sqrt(Dt)...
                                - g2_diff(X,base_para,noise)*(Y3^2-Dt)/(2*sqrt(Dt))...
                                - g3_diff(X,base_para,noise)*X1/sqrt(Dt);
    Xtilde(:,7) = Xtilde(:,1) - g1_diff(X,base_para,noise)*Z1/sqrt(Dt)...
                                - g2_diff(X,base_para,noise)*Y1/sqrt(Dt)...
                                - g3_diff(X,base_para,noise)*(X3^2-Dt)/(2*sqrt(Dt));
    Xfinal = X + 0.5*(f_drift(X,base_para) + f_drift(Xtilde(:,1),base_para))*Dt...
               + Z3*g1_diff(X,base_para,noise)...
               + 0.5*(g1_diff(Xtilde(:,2),base_para,noise)...
               - g1_diff(Xtilde(:,5),base_para,noise))*sqrt(Dt)...
               + Y3*g2_diff(X,base_para,noise)...
               + 0.5*(g2_diff(Xtilde(:,3),base_para,noise)...
               - g2_diff(Xtilde(:,6),base_para,noise))*sqrt(Dt)...
               + X3*g3_diff(X,base_para,noise)...
               + 0.5*(g3_diff(Xtilde(:,4),base_para,noise)...
               - g3_diff(Xtilde(:,7),base_para,noise))*sqrt(Dt);
    Xrk(:,k) = Xfinal;
    Time(k+1) = k*Dt;
end
end
```

Local function f_drift

```
function y = f_drift(X,base_para)
K_N = base_para(1);
K_C = base_para(2);
r_N = base_para(3);
rho_C = base_para(4);
a = base_para(5);
b = base_para(6);
r_B = base_para(7);
k_B = base_para(8);
gamma_B = base_para(9);
y = [r_N*X(1)*log(K_N/(X(1)+X(2)));
     (rho_C + b*(X(1)+X(2)-K_N)^2/(a*(X(1)+X(2))^2+(X(1)+X(2)-K_N)^2))...
     *X(2)*log(K_C/(X(1)+X(2)));
     r_B*X(3) - gamma_B*X(3)*X(2)/(k_B+X(2))];
% Note: X(1) represents the variable N - normal T cells
%       X(2) represents the variable C - CAR T cells
%       X(3) represents the variable B - tumor cells
end
```

Local function g1_diff

```
function y = g1_diff(X,base_para,noise)
```

```
K_C = base_para(2);
tau1 = noise(1);
y = [0; tau1*X(2)*log(K_C/(X(1)+X(2))); 0];
end
```

Local function g2_diff

```
function y = g2_diff(X,base_para,noise)
tau2 = noise(2);
y = [0; 0; tau2*X(3)];
end
```

Local function g3_diff

```
function y = g3_diff(X,base_para,noise)
k_B = base_para(8);
tau3 = noise(3);
y = [0; 0; -tau3*X(3)*X(2)/(k_B+X(2))];
end
```

Local function: violin plot function

```
% violin.m - Simple violin plot using matlab default kernel density estimation
% Last update: 10/2015
%__________________________________________________________________________
% This function creates violin plots based on kernel density estimation
% using ksdensity with default settings. Please be careful when comparing pdfs
% estimated with different bandwidth!
%
% Differently to other boxplot functions, you may specify the x-position.
% This is usefule when overlaying with other data / plots.
%__________________________________________________________________________
%
% Please cite this function as:
% Hoffmann H, 2015: violin.m - Simple violin plot using matlab default kernel
% density estimation. INRES (University of Bonn), Katzenburgweg 5, 53115 Germany.
% hhoffmann@uni-bonn.de
%
%__________________________________________________________________________
%
% INPUT
%
% Y:     Data to be plotted, being either
%        a) n x m matrix. A 'violin' is plotted for each column m, OR
%        b) 1 x m Cellarry with elements being numerical colums of nx1 length.
%
% varargin:
% xlabel:    xlabel. Set either [] or in the form {'txt1','txt2','txt3',...}
% facecolor: FaceColor. (default [1 0.5 0]); Specify abbrev. or m x 3 matrix (e.g.
[1 0 0])
```

```
% edgecolor: LineColor. (default 'k'); Specify abbrev. (e.g. 'k' for black); set
either [],'' or 'none' if the mean should not be plotted
% facealpha: Alpha value (transparency). default: 0.5
% mc:        Color of the bars indicating the mean. (default 'k'); set either [],''
or 'none' if the mean should not be plotted
% medc:      Color of the bars indicating the median. (default 'r'); set either
[],'' or 'none' if the mean should not be plotted
% bw:        Kernel bandwidth. (default []); prescribe if wanted as follows:
%            a) if bw is a single number, bw will be applied to all
%            columns or cells
%            b) if bw is an array of 1xm or mx1, bw(i) will be applied to cell or
column (i).
%            c) if bw is empty (default []), the optimal bandwidth for
%            gaussian kernel is used (see Matlab documentation for
%            ksdensity()
%
% OUTPUT
%
% h:     figure handle
% L:     Legend handle
% MX:    Means of groups
% MED:   Medians of groups
% bw:    bandwidth of kernel
%__________________________________________________________________________
%{
% Example1 (default):
disp('this example uses the statistical toolbox')
Y=[rand(1000,1),gamrnd(1,2,1000,1),normrnd(10,2,1000,1),gamrnd(10,0.1,1000,1)];
[h,L,MX,MED]=violin(Y);
ylabel('\Delta [yesno^{-2}]','FontSize',14)
%Example2 (specify facecolor, edgecolor, xlabel):
disp('this example uses the statistical toolbox')
Y=[rand(1000,1),gamrnd(1,2,1000,1),normrnd(10,2,1000,1),gamrnd(10,0.1,1000,1)];
violin(Y,'xlabel',{'a','b','c','d'},'facecolor',[1 1 0;0 1 0;.3 .3 .3;0 0.3
0.1],'edgecolor','b',...
'bw',0.3,...
'mc','k',...
'medc','r--')
ylabel('\Delta [yesno^{-2}]','FontSize',14)
%Example3 (specify x axis location):
disp('this example uses the statistical toolbox')
Y=[rand(1000,1),gamrnd(1,2,1000,1),normrnd(10,2,1000,1),gamrnd(10,0.1,1000,1)];
violin(Y,'x',[-1 .7 3.4 8.8],'facecolor',[1 1 0;0 1 0;.3 .3 .3;0 0.3
0.1],'edgecolor','none',...
'bw',0.3,'mc','k','medc','r-.')
axis([-2 10 -0.5 20])
ylabel('\Delta [yesno^{-2}]','FontSize',14)
%Example4 (Give data as cells with different n):
disp('this example uses the statistical toolbox')
Y{:,1}=rand(10,1);
```

```
Y{:,2}=rand(1000,1);
violin(Y,'facecolor',[1 1 0;0 1 0;.3 .3 .3;0 0.3
0.1],'edgecolor','none','bw',0.1,'mc','k','medc','r-.')
ylabel('\Delta [yesno^{-2}]','FontSize',14)
%}
%%
function[h,L,MX,MED,bw]=violin(Y,varargin)
%defaults:
%_____________________
xL=[];
fc=[1 0.5 0];
lc='k';
alp=0.5;
mc='k';
medc='r';
b=[]; %bandwidth
plotlegend=0;
plotmean=0;
plotmedian=0;
x = [];
%_____________________
%convert single columns to cells:
if iscell(Y)==0
    Y = num2cell(Y,1);
end
%get additional input parameters (varargin)
if isempty(find(strcmp(varargin,'xlabel')))==0
    xL = varargin{find(strcmp(varargin,'xlabel'))+1};
end
if isempty(find(strcmp(varargin,'facecolor')))==0
    fc = varargin{find(strcmp(varargin,'facecolor'))+1};
end
if isempty(find(strcmp(varargin,'edgecolor')))==0
    lc = varargin{find(strcmp(varargin,'edgecolor'))+1};
end
if isempty(find(strcmp(varargin,'facealpha')))==0
    alp = varargin{find(strcmp(varargin,'facealpha'))+1};
end
if isempty(find(strcmp(varargin,'mc')))==0
    if isempty(varargin{find(strcmp(varargin,'mc'))+1})==0
        mc = varargin{find(strcmp(varargin,'mc'))+1};
        plotmean = 1;
    else
        plotmean = 0;
    end
end
if isempty(find(strcmp(varargin,'medc')))==0
    if isempty(varargin{find(strcmp(varargin,'medc'))+1})==0
        medc = varargin{find(strcmp(varargin,'medc'))+1};
        plotmedian = 1;
```

```
    else
        plotmedian = 0;
    end
end
if isempty(find(strcmp(varargin,'bw')))==0
    b = varargin{find(strcmp(varargin,'bw'))+1}
    if length(b)==1
        disp(['same bandwidth bw = ',num2str(b),' used for all cols'])
        b=repmat(b,size(Y,2),1);
    elseif length(b)~=size(Y,2)
        warning('length(b)~=size(Y,2)')
        error('please provide only one bandwidth or an array of b with same length
as columns in the data set')
    end
end
if isempty(find(strcmp(varargin,'plotlegend')))==0
    plotlegend = varargin{find(strcmp(varargin,'plotlegend'))+1};
end
if isempty(find(strcmp(varargin,'x')))==0
    x = varargin{find(strcmp(varargin,'x'))+1};
end
%%
if size(fc,1)==1
    fc=repmat(fc,size(Y,2),1);
end
%% Calculate the kernel density
i=1;
for i=1:size(Y,2)

    if isempty(b)==0
        [f, u, bb]=ksdensity(Y{i},'bandwidth',b(i));
    elseif isempty(b)
        [f, u, bb]=ksdensity(Y{i});
    end

    f=f/max(f)*0.3; %normalize
    F(:,i)=f;
    U(:,i)=u;
    MED(:,i)=nanmedian(Y{i});
    MX(:,i)=nanmean(Y{i});
    bw(:,i)=bb;

end
%%
%-------------------------------------------------------------------------
% Put the figure automatically on a second monitor
% mp = get(0, 'MonitorPositions');
% set(gcf,'Color','w','Position',[mp(end,1)+50 mp(end,2)+50 800 600])
%-------------------------------------------------------------------------
%Check x-value options
```

```
if isempty(x)
    x = zeros(size(Y,2));
    setX = 0;
else
    setX = 1;
    if isempty(xL)==0
        disp('_________________________________________________________________')
        warning('Function is not designed for x-axis specification with string
label')
        warning('when providing x, xlabel can be set later anyway')
        error('please provide either x or xlabel. not both.')
    end
end
%% Plot the violins
i=1;
for i=i:size(Y,2)
    if isempty(lc) == 1
        if setX == 0
            h(i)=fill([F(:,i)+i;flipud(i-F(:,i))],
[U(:,i);flipud(U(:,i))],fc(i,:),'FaceAlpha',alp,'EdgeColor','none');
        else
            h(i)=fill([F(:,i)+x(i);flipud(x(i)-F(:,i))],
[U(:,i);flipud(U(:,i))],fc(i,:),'FaceAlpha',alp,'EdgeColor','none');
        end
    else
        if setX == 0
            h(i)=fill([F(:,i)+i;flipud(i-F(:,i))],
[U(:,i);flipud(U(:,i))],fc(i,:),'FaceAlpha',alp,'EdgeColor',lc);
        else
            h(i)=fill([F(:,i)+x(i);flipud(x(i)-F(:,i))],
[U(:,i);flipud(U(:,i))],fc(i,:),'FaceAlpha',alp,'EdgeColor',lc);
        end
    end
    hold on
    if setX == 0
        if plotmean == 1
            p(1)=plot([interp1(U(:,i),F(:,i)+i,MX(:,i)),
interp1(flipud(U(:,i)),flipud(i-F(:,i)),MX(:,i)) ],[MX(:,i)
MX(:,i)],mc,'LineWidth',2);
        end
        if plotmedian == 1
            p(2)=plot([interp1(U(:,i),F(:,i)+i,MED(:,i)),
interp1(flipud(U(:,i)),flipud(i-F(:,i)),MED(:,i)) ],[MED(:,i)
MED(:,i)],medc,'LineWidth',2);
        end
    elseif setX == 1
        if plotmean == 1
            p(1)=plot([interp1(U(:,i),F(:,i)+i,MX(:,i))+x(i)-i,
interp1(flipud(U(:,i)),flipud(i-F(:,i)),MX(:,i))+x(i)-i],[MX(:,i)
MX(:,i)],mc,'LineWidth',2);
```

```
        end
        if plotmedian == 1
            p(2)=plot([interp1(U(:,i),F(:,i)+i,MED(:,i))+x(i)-i,
interp1(flipud(U(:,i)),flipud(i-F(:,i)),MED(:,i))+x(i)-i],[MED(:,i)
MED(:,i)],medc,'LineWidth',2);
        end
    end
end
% Add legend if requested
if plotlegend==1 & plotmean==1 | plotlegend==1 & plotmedian==1

    if plotmean==1 & plotmedian==1
        L=legend([p(1) p(2)],'Mean','Median');
    elseif plotmean==0 & plotmedian==1
        L=legend([p(2)],'Median');
    elseif plotmean==1 & plotmedian==0
        L=legend([p(1)],'Mean');
    end

    set(L,'box','off','FontSize',14)
else
    L=[];
end
% Set axis
if setX == 0
    axis([0.5 size(Y,2)+0.5, min(U(:)) max(U(:))]);
elseif setX == 1
    axis([min(x)-0.05*range(x) max(x)+0.05*range(x), min(U(:)) max(U(:))]);
end
%% Set x-labels
xL2={''};
i=1;
for i=1:size(xL,2)
    xL2=[xL2,xL{i},{''}];
end
set(gca,'TickLength',[0 0],'FontSize',12)
box on
if isempty(xL)==0
    set(gca,'XtickLabel',xL2)
end
%-------------------------------------------------------------------------
end
```