Robert Heeter
BIOE 446
8 September 2023

Lab 2: PK/PD Modeling

CHALLENGE PROBLEMS

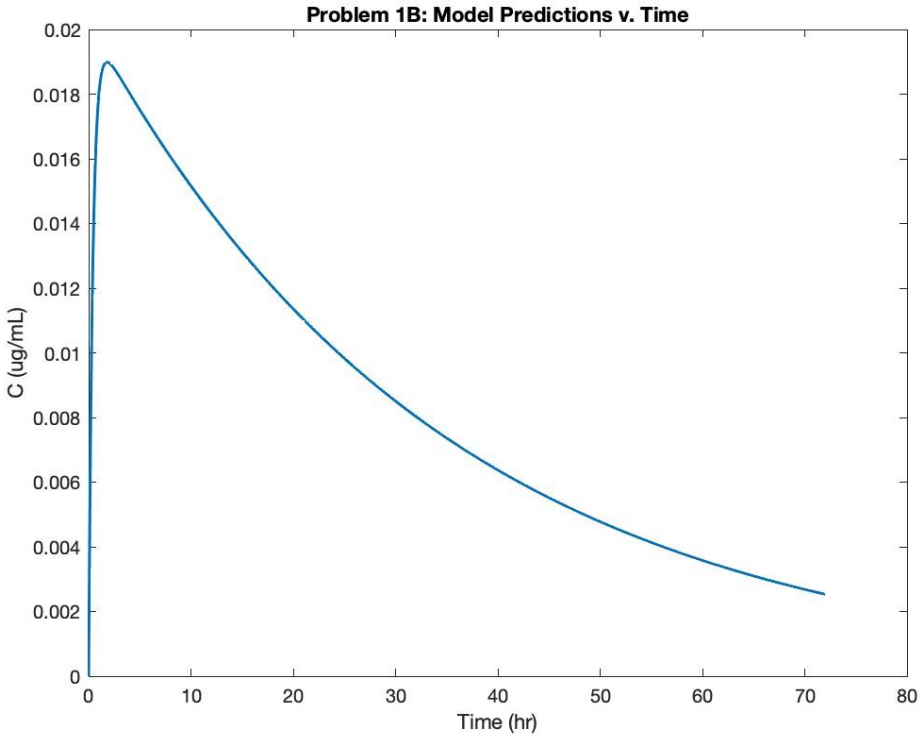
1. The estimated half life is 48.00 hrs.

ANSWERS

1. Modeling s.c. bolus therapeutic delivery with a two-compartment model (DEMO)

a. dC1/dt = -250.00 ug/(mL hr)
   dC2/dt = 0.05 ug/(mL hr)

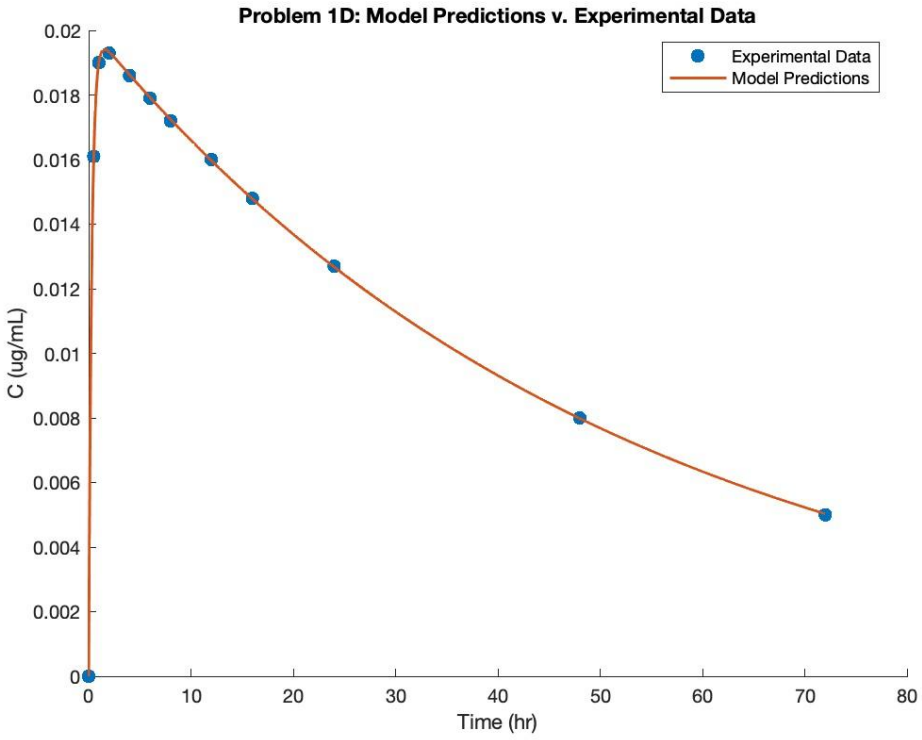
b. Figure



c. The error between model predictions and experimental data is 0.000035820.

d. k12_opt = 3.3175
   kcl_opt = 0.0192
   t_half = 36.0088 hrs

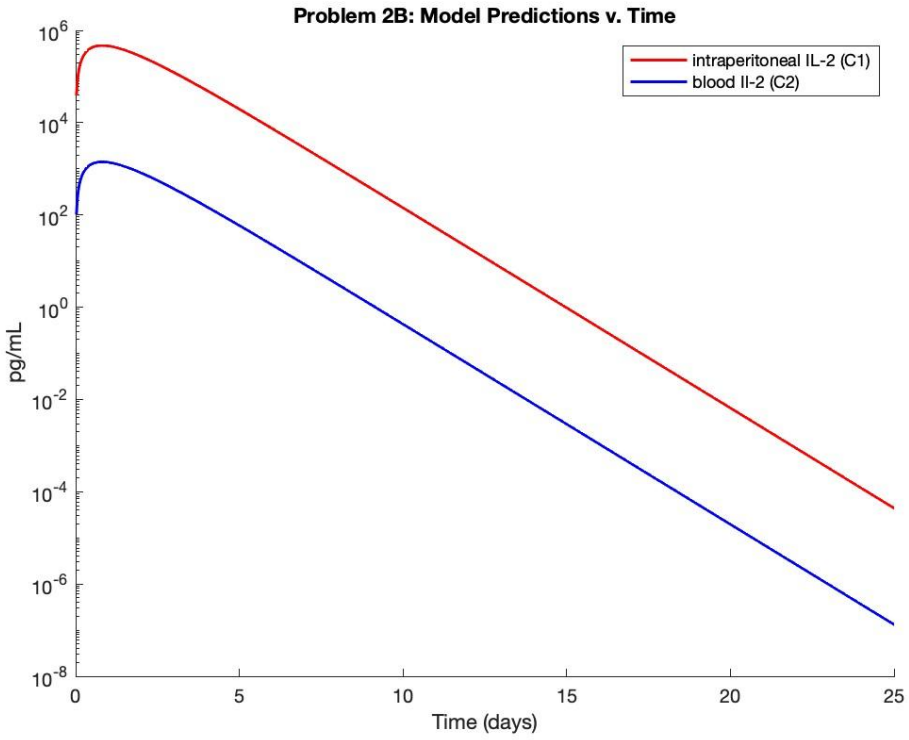
   Figure



2. Modeling IL-2 intraperitoneal cell therapy with a two-compartment model

a. dC1/dt = 1586160.00 pg/(mL day)
   dC2/dt = -416.67 pg/(mL day)

b. Figure

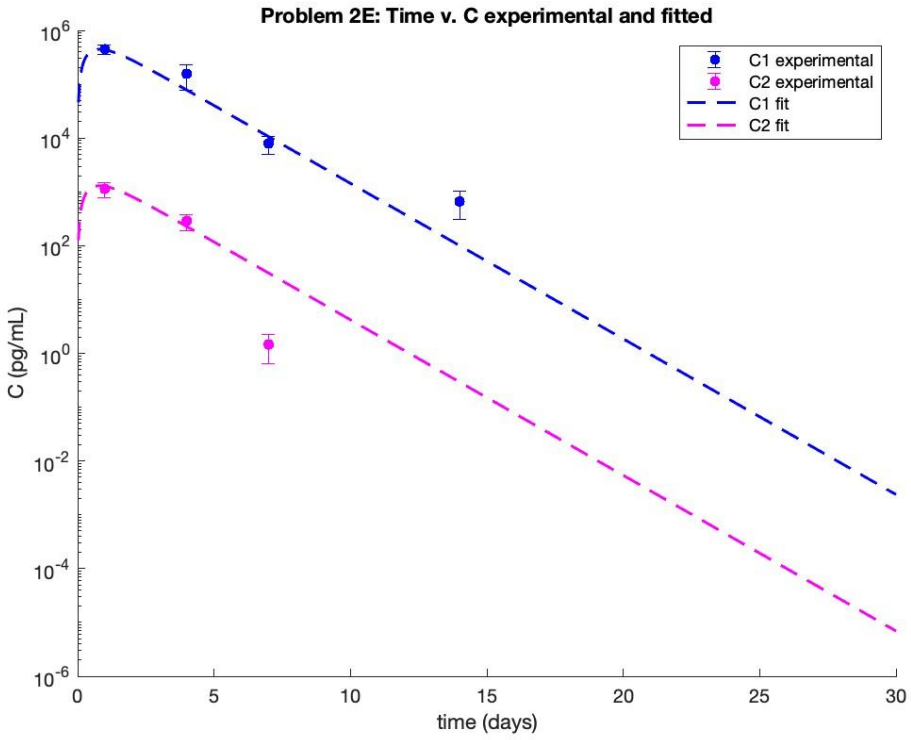


c. C1_error = 1.8706
   C2_error = 0.2719
   total_error = 2.1425

d. total_err = 2.1425

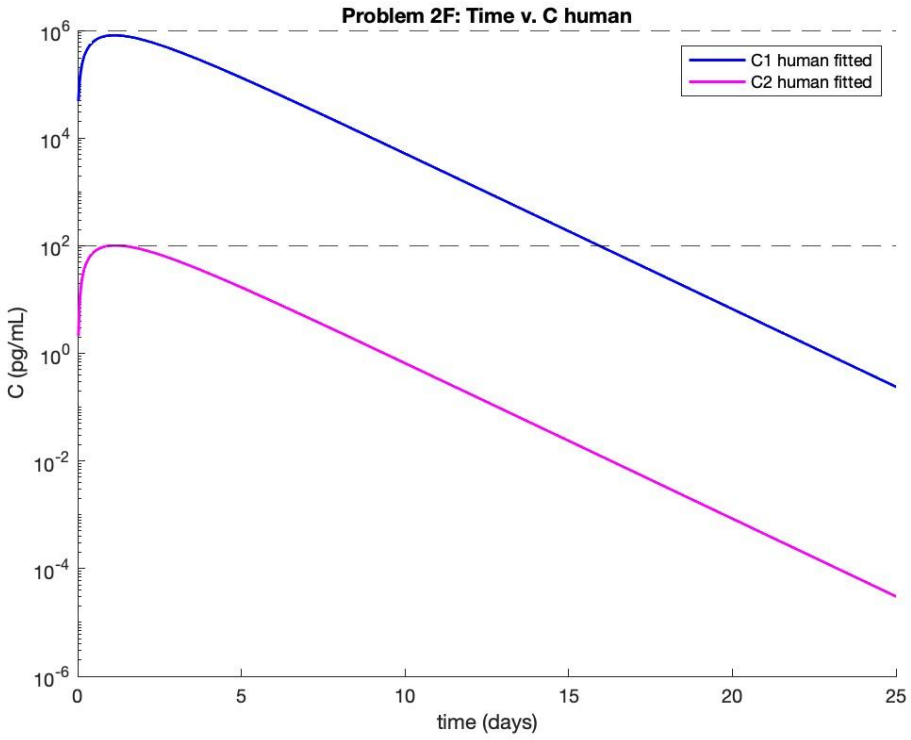
e. lambda_opt = 0.6657
   ktrans_opt = 2.0816
   kclear_opt = 714.5154
   t_half = 0.00097009

   Figure



f. ktrans_opt_h = 1.1569
   kclear_opt_h = 184936.3104

   Figure



The thresholds on the figure show that the IP IL-2 concentration approximately reaches the toxic threshold of ~1000 ng/mL (1e6 pg/mL), indicating that this treatment may not be safe for some patients since there is no safety buffer between the maximum concentration and the toxic threshold. The blood IL-2 concentration approaches the 0.1 ng/mL (100 pg/mL) threshold, but does not significantly exceed it, indicating that it may not cause T-cell activation in the blood. Furthermore, since this is just an approximate model built from mouse data, further testing in a human model is necessary to draw any definitive conclusions.

g. C1_opt_h_final = 132312.9448
C2_opt_h_final = 16.8364
ratio = 7858.7424

C1_opt_h_final_ktransmod = 0.07117906
cC2_opt_h_final_ktransmod = 0.00001814
ratio = 3924.3855

C1_opt_h_final_7500 = 0.35778267
C2_opt_h_final_7500 = 0.00004558
ratio = 7848.8235

When the transport rate (ktrans_opt_h) is doubled, the ratio of concentrations is halved. However, the ratio of concentrations is independent of the dosage (number of capsules N0); the number of capsules does affect the actual final concentrations proportionally.

This indicates that the model's predictions are dependent on these parameters.

CODE

```
close all
clc

%% Challenge Problem 1
disp('CHALLENGE PROBLEM 1')
clear

% initializing data and system parameters
t = [.5; 1; 1.5; 2; 2.5; 3; 6; 8; 12; 16; 24; 36; 54; 72];
Cvals = [.197; .194; .192; .189; .186; .183; .168; .159; .141; .126; .1; .07; .05;
.025]; % ug/mL
D = 1000; % ug
V = 5000; % mL

% defining one compartment equation solution
func = @(t, D, V, kcl) (D./V)*exp(-kcl*t);

% defining equation to minimize
fit_func = @(kcl) sum((func(D, V, kcl, t) - Cvals).^2);

% initial guess is two day half life (educated guess)
k0 = log(2)/48;

% solving with fsolve
k_opt = fminsearch(fit_func, k0);

% converting to half life
t_half = log(2)/k_opt;

% printing
fprintf('The estimated half life is %.2f hrs.\n\n', t_half);

%% Problem 1 (Demo): Modeling s.c. bolus therapeutic delivery with a two-compartment
model
disp('PROBLEM 1')
clear

t = [.5; 1; 1.5; 2; 2.5; 3; 6; 8; 12; 16; 24; 36; 54; 72];
Cvals = [0; .0161; .019; .0193; .0186; .0179; .0172; .016; .0148; .0127; .008; .005];
% ug/mL

%% Problem 1, Part A
D = 100; % ug
k12 = 2.5;
kcl = log(2)/24;
V1 = 1;
V2 = 5000;
p = [k12, kcl, V1, V2];
c = [D/V1, 0];
t = 0;
```

```
dCdt = two_comp_rate_laws(t, c, p);
fprintf('Problem 1A:\ndC1/dt = %.2f ug/(mL hr)\ndC2/dt = %.2f ug/(mL hr)\n\n',
dCdt(1), dCdt(2));

%% Problem 1, Part B
tspan = linspace(0, 72, 1000);
C = simulate_two_comp_model(tspan, p, D);
figure(1)
clf
plot(tspan,C(:,2), 'LineWidth', 1.5)
title('Problem 1B: Model Predictions v. Time')
xlabel('Time (hr)')
ylabel('C (ug/mL)')

%% Problem 1, Part C
D = 100; % ug
k12 = 2.5;
kcl = log(2)/24;
V1 = 1;
V2 = 5000;
p_kinetic = [k12, kcl];
p_system = [V1, V2, D];
t_exp = [0; .5; 1; 2; 4; 6; 8; 12; 16; 24; 48; 72];
c_exp = [0; .0161; .019; .0193; .0186; .0179; .0172; .016; .0148; .0127; .008; .005];
% ug/mL

% initializing Data
err = calculate_sse(t_exp, c_exp, p_kinetic, p_system);
fprintf('Problem 1C:\nThe error between model predictions and experimental data is
%.9f\n\n',err);

%% Problem 1, Part D
% set known parameters
p_system = [V1, V2, D];

% define our optimization function
fit_func = @(p) calculate_sse(t_exp, c_exp, p, p_system);
p0 = [0.9, log(2)/24];
[p_opt, err] = fminsearch(fit_func, p0);
p = [p_opt, V1, V2];
tspan = linspace(0, 72, 1000);
c_predictions = simulate_two_comp_model(tspan, p, D);
c2_predictions = c_predictions(:,2);

figure(2)
clf
hold on
plot(t_exp,c_exp, '.', 'MarkerSize', 20) % experimental data
plot(tspan,c2_predictions, 'LineWidth', 1.5) % predictions
title('Problem 1D: Model Predictions v. Experimental Data')
ylabel('C (ug/mL)')
```

```
xlabel('Time (hr)')
legend('Experimental Data', 'Model Predictions', 'Location', 'Best')

k12_opt = p_opt(1);
kcl_opt = p_opt(2);
t_half = log(2)/kcl_opt;
fprintf('Problem 1D:\nk12_opt = %.4f\nkcl_opt = %.4f\nt_half = %.4f hrs\n\n',
k12_opt, kcl_opt, t_half);

%% Problem 2: Modeling IL-2 intraperitoneal cell therapy with a two-compartment model
clear

%% Problem 2, Part A
kprod = 7930.8; % pg capsule^-1 day^-1
N0 = 200; % capsules
lambda = 1;
ktrans = 1.5;
kclear = 500;
V1 = 1; % mL
V2 = 1.2; % mL

C = [0; 1];
p = [kprod, N0, lambda, ktrans, kclear, V1, V2];
t = 0;

dCdt = comp_model_2(t, C, p);

fprintf('Problem 2A:\ndC1/dt = %.2f pg/(mL day)\ndC2/dt = %.2f pg/(mL day)\n\n',
dCdt(1), dCdt(2));

%% Problem 2, Part B
tspan = linspace(0, 25, 1000);
C0 = [0; 0];
C = simulate_comp_model_2(tspan, p, C0);
figure(3)
clf
hold on
plot(tspan, C(:,1), '-r', 'LineWidth', 1.5)
plot(tspan, C(:,2), '-b', 'LineWidth', 1.5)
title('Problem 2B: Model Predictions v. Time')
legend('intraperitoneal IL-2 (C1)', 'blood Il-2 (C2)')
xlabel('Time (days)')
ylabel('pg/mL')
set(gca, 'YScale', 'log')

%% Problem 2, Part C
tspan = [0 1 4 7 14 21 30]; % days
C = simulate_comp_model_2(tspan, p, C0);
C1 = C(:,1);
C2 = C(:,2);
C1_exp = [0 448568.7 156181.1 7948.23 661.44 0 0]; % pg/mL
```

```
C2_exp = [0 1135.15 288.04 1.46 0 0 0]; % pg/mL
D = 30; % pg/mL threshold

C1_err = calculate_sse_2(C1, C1_exp, D);
C2_err = calculate_sse_2(C2, C2_exp, D);

fprintf('Problem 2C:\nC1_error = %.4f\nC2_error = %.4f\ntotal_error = %.4f\n\n',
C1_err, C2_err, C1_err+C2_err);

%% Problem 2, Part D
p_kinetic = [lambda, ktrans, kclear];
p_system = [kprod, N0, V1, V2];
total_err = simulate_sse(tspan, C1_exp, C2_exp, p_kinetic, p_system);
fprintf('Problem 2D:\ntotal_err = %.4f\n\n', total_err);

%% Problem 2, Part E
tspan = tspan';
C1_exp = C1_exp';
C2_exp = C2_exp';

p0 = [1 2 500];

kprod = 7930.8; % pg capsule^-1 day^-1
N0 = 200; % capsules
V1 = 1; % mL
V2 = 1.2; % mL

func = @(p) simulate_sse(tspan, C1_exp, C2_exp, p, p_system);
[p_opt, err_opt] = fminsearch(func, p0);

lambda_opt = p_opt(1);
ktrans_opt = p_opt(2);
kclear_opt = p_opt(3);

t_half = log(2)/kclear_opt;

fprintf('Problem 2C:\nlambda_opt = %.4f\nktrans_opt = %.4f\nkclear_opt = %.4f\nt_half
= %.8f\n\n', lambda_opt, ktrans_opt, kclear_opt, t_half);

p = [kprod, N0, lambda_opt, ktrans_opt, kclear_opt, V1, V2];
C0 = [0, 0];
t = linspace(0, 30, 1000);
C_opt = simulate_comp_model_2(t, p, C0);
C1_opt = C_opt(:, 1);
C2_opt = C_opt(:, 2);

C1_err = [0.01 87937.6 77579.66 2885.65 355.29 0.01 0.01];
C2_err = [0.01 350.05 95.27 0.82 0.01 0.01 0.01];

figure(4)
hold on
```

```
errorbar(tspan, C1_exp, C1_err, '.b', 'MarkerSize', 15)
errorbar(tspan, C2_exp, C2_err, '.m', 'MarkerSize', 15)
plot(t, C1_opt, '--b', 'LineWidth', 1.5)
plot(t, C2_opt, '--m', 'LineWidth', 1.5)
legend('C1 experimental', 'C2 experimental', 'C1 fit', 'C2 fit')
title('Problem 2E: Time v. C experimental and fitted')
xlabel('time (days)')
ylabel('C (pg/mL)')
set(gca, 'YScale', 'log')
hold off

%% Problem 2, Part F
ktrans_opt_h = ktrans_opt*(70000/25)^(-0.074);
kclear_opt_h = kclear_opt*(70000/25)^(0.70);

fprintf('Problem 2F:\nktrans_opt_h = %.4f\nkclear_opt_h = %.4f\n\n', ktrans_opt_h,
kclear_opt_h);

N0 = 5000; % capsules
kprod = 7930.8; % pg capsule^-1 day^-1
V1 = 20; % mL
V2 = 5000; % mL

p = [kprod, N0, lambda_opt, ktrans_opt_h, kclear_opt_h, V1, V2];
C0 = [0, 0];
t = linspace(0, 25, 1000);
C_opt_h = simulate_comp_model_2(t, p, C0);
C1_opt_h = C_opt_h(:, 1);
C2_opt_h = C_opt_h(:, 2);

figure(5)
hold on
plot(t, C1_opt_h, '-b', 'LineWidth', 1.5)
plot(t, C2_opt_h, '-m', 'LineWidth', 1.5)
yline(1e6, '--k')
yline(100, '--k')
legend('C1 human fitted', 'C2 human fitted')
title('Problem 2F: Time v. C human')
xlabel('time (days)')
ylabel('C (pg/mL)')
set(gca, 'YScale', 'log')
hold off

%{
The thresholds on the figure show that the IP IL-2 concentration
approximately reaches the toxic threshold of ~1000 ng/mL (1e6 pg/mL),
indicating that this treatment may not be safe for some patients since
there is no safety buffer between the maximum concentration and the toxic
threshold. The blood IL-2 concentration approaches the 0.1 ng/mL (100
pg/mL) threshold, but does not significantly exceed it, indicating that it
may not cause T-cell activation in the blood. Furthermore, since this is
```

```
just an approximate model built from mouse data, further testing in a human
model is necessary to draw any definitive conclusions.
%}

%% Problem 2, Part G
C0 = [0, 0];
t = linspace(0, 5, 1000);
C_opt_h_final = simulate_comp_model_2(t, p, C0);
C1_opt_h_final = C_opt_h_final(end, 1);
C2_opt_h_final = C_opt_h_final(end, 2);
ratio = C1_opt_h_final/C2_opt_h_final;
fprintf('Problem 2G:\nC1_opt_h_final = %.4f\nC2_opt_h_final = %.4f\nratio =
%.4f\n\n', C1_opt_h_final, C2_opt_h_final, ratio);

ktrans_opt_h_mod = ktrans_opt_h*2;
p = [kprod, N0, lambda_opt, ktrans_opt_h_mod, kclear_opt_h, V1, V2];
C0 = [0, 0];
t = linspace(0, 25, 1000);
C_opt_h_final_ktransmod = simulate_comp_model_2(t, p, C0);
C1_opt_h_final_ktransmod = C_opt_h_final_ktransmod(end, 1);
C2_opt_h_final_ktransmod = C_opt_h_final_ktransmod(end, 2);
ratio = C1_opt_h_final_ktransmod/C2_opt_h_final_ktransmod;
fprintf('Problem 2G:\nC1_opt_h_final_ktransmod = %.8f\ncC2_opt_h_final_ktransmod =
%.8f\nratio = %.4f\n\n', C1_opt_h_final_ktransmod, C2_opt_h_final_ktransmod, ratio);

N0_mod = 7500;
p = [kprod, N0_mod, lambda_opt, ktrans_opt_h, kclear_opt_h, V1, V2];
C0 = [0, 0];
t = linspace(0, 25, 1000);
C_opt_h_final_7500 = simulate_comp_model_2(t, p, C0);
C1_opt_h_final_7500 = C_opt_h_final_7500(end, 1);
C2_opt_h_final_7500 = C_opt_h_final_7500(end, 2);
ratio = C1_opt_h_final_7500/C2_opt_h_final_7500;
fprintf('Problem 2G:\nC1_opt_h_final_7500 = %.8f\nC2_opt_h_final_7500 = %.8f\nratio =
%.4f\n\n', C1_opt_h_final_7500, C2_opt_h_final_7500, ratio);

%{
When the transport rate (ktrans_opt_h) is doubled, the ratio of
concentrations is halved. However, the ratio of concentrations is
independent of the the dosage (number of capsules N0); the number of
capsules does affect the actual final concentrations proportionally. This
indicates that the model's predictions are dependent on these parameters.
%}

%% Functions
% Problem 1, Part A
function dCdt = two_comp_rate_laws(t, c, p)
% Function that calculates rates of change in two compartment model given
% time t, vector of concentrations C, and vector of parameters p.
      % initialize species
      C1 = c(1);
```

```
        C2 = c(2);

        % initialize parameters
        k12 = p(1);
        kcl = p(2);
        V1 = p(3);
        V2 = p(4);

        % calculate rates of change
        dCdt = zeros(size(c)); % dC1/dt
        dCdt(1) = -k12*C1;
        dCdt(2) = ((k12*V1)/V2)*C1 - kcl*C2;
end

% Problem 1, Part B
function C = simulate_two_comp_model(tspan, p, D)
% Function that simulates the two compartment model at timepoints in tspan
% with parameters p and dose amount D.
        % setting up rate laws
        dCdt = @(t, c) two_comp_rate_laws(t, c, p);

        % setting up initial conditions
        V1 = p(3);
        C0 = [D/V1, 0];

        % simulate the model
        [~, C] = ode15s(dCdt, tspan, C0);
end

% Problem 1, Part C
function err = calculate_sse(t_exp, c_exp, p_kinetic, p_system)
% Function for calculating standard squared error between model predictions
% and experimental measurements.
        % combining kinetic parameters and system parameters
        p = [p_kinetic, p_system(1:2)];

        % setting dose amount
        D = p_system(3);

        % simulating model
        c_predicted = simulate_two_comp_model(t_exp, p, D);
        c_predicted = c_predicted(:, 2);

        % calculating SSE between model predictions and experimental data
        err = (c_predicted - c_exp).^2;
        err = sum(err);
end

% Problem 2, Part A
function dCdt = comp_model_2(t, C, p)
        % initialize species
```

```
        C1 = C(1);
        C2 = C(2);

        % initialize parameters
        kprod = p(1);
        N0 = p(2);
        lambda = p(3);
        ktrans = p(4);
        kclear = p(5);
        V1 = p(6);
        V2 = p(7);

        % calculate rates of change
        dCdt = zeros(size(C)); % dC/dt
        dCdt(1) = ((kprod/V1)*N0*exp(-1.*lambda.*t)) - ktrans.*C1;
        dCdt(2) = (ktrans.*(V1./V2).*C1) - ((kclear./V2).*C2);
end

% Problem 2, Part B
function C = simulate_comp_model_2(tspan, p, C0)
        % setting up rate laws
        dCdt = @(t, c) comp_model_2(t, c, p);

        % simulate the model
        [~, C] = ode15s(dCdt, tspan, C0);
end

% Problem 2, Part C
function err = calculate_sse_2(C, C_exp, D)

% calculate error according to equation in 2C
err = zeros(1,length(C));
        for i = 1:length(C)
                if C_exp(i) >= D
                        err(i) = ((C(i)-C_exp(i))^2)/(1+C_exp(i)^2);
                elseif (C_exp(i) < D && C(i) >= D)
                        err(i) = ((C(i)-D)^2)/(1+D^2);
                elseif (C_exp(i) < D && C(i) < D)
                        err(i) = 0;
                end
        end
        err = sum(err);
end

% Problem 2, Part D
function total_err = simulate_sse(tspan, C1_exp, C2_exp, p_kinetic, p_system)
        kprod = p_system(1);
        N0 = p_system(2);
        lambda = p_kinetic(1);
        ktrans = p_kinetic(2);
        kclear = p_kinetic(3);
```

```
        V1 = p_system(3);
        V2 = p_system(4);

        p = [kprod, N0, lambda, ktrans, kclear, V1, V2];
        C0 = [0, 0];
        C = simulate_comp_model_2(tspan, p, C0);
        C1 = C(:,1);
        C2 = C(:,2);

        D = 30; % pg/mL
        C1_err = calculate_sse_2(C1, C1_exp, D);
        C2_err = calculate_sse_2(C2, C2_exp, D);

        total_err = C1_err + C2_err;
end
```