Grayson Gerlich **10801535**

Systems Biology, Homework # 2, Part 2: Chemical Reaction Networks

1. (a) Write a set of six differential equations describing the behaviour of the concentrations of the species, starting from an arbitrary initial concentration profile.

$$\frac{da}{dt} = -k_1 * (a * b)$$

$$\frac{db}{dt} = k_2 * d - k_1 * (a * b)$$

$$\frac{dc}{dt} = k_1 * (a * b) - k_3 * c$$

$$\frac{dd}{dt} = k_1 * (a * b) - k_2 * d$$

$$\frac{de}{dt} = k_3 * c$$

$$\frac{df}{dt} = k_3 * c$$

   (b) Use conservations to reduce the system description to three differential equations and three conservation statements.

      i. Differential equations

$$\frac{db}{dt} = k_2 * d - k_1 * (a * b)$$

$$\frac{dc}{dt} = k_1 * (a * b) - k_3 * c$$

$$\frac{de}{dt} = k_3 * c$$

      ii. Conservation statements

$$\frac{db}{dt} = -\frac{dd}{dt}$$

$$\frac{dc}{dt} = -\frac{da}{dt} - \frac{de}{dt}$$

$$\frac{de}{dt} = \frac{df}{dt}$$

   (c) Open system

i. System of differential equations

$$\frac{da}{dt} = k_0 - k_1 * (a * b)$$

$$\frac{db}{dt} = k_2 * d - k_1 * (a * b)$$

$$\frac{dc}{dt} = k_1 * (a * b) - k_3 * c$$

$$\frac{dd}{dt} = k_1 * (a * b) - k_2 * d$$

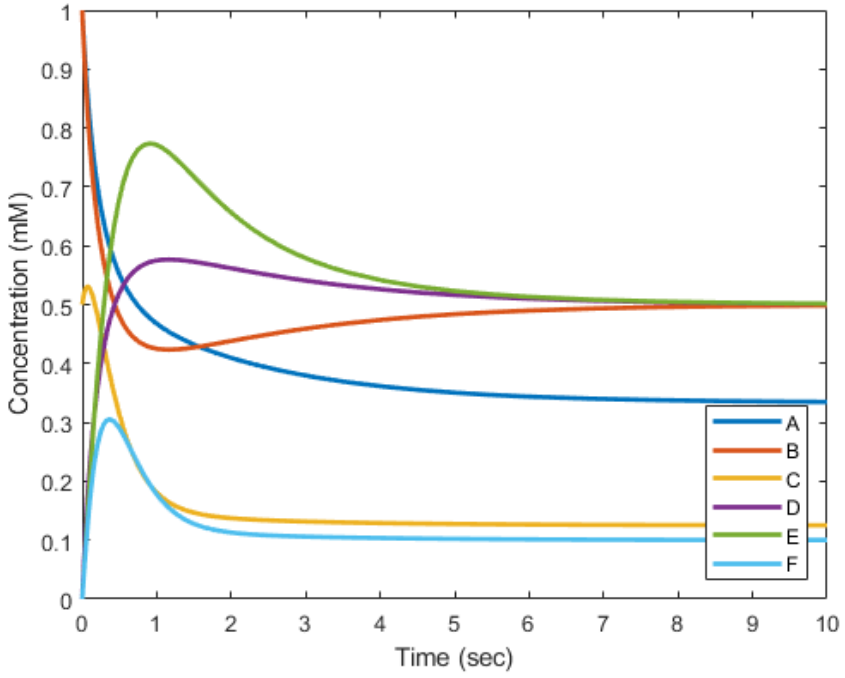
$$\frac{de}{dt} = k_3 * c - k_4 * e$$

$$\frac{df}{dt} = k_3 * c - k_5 * f$$

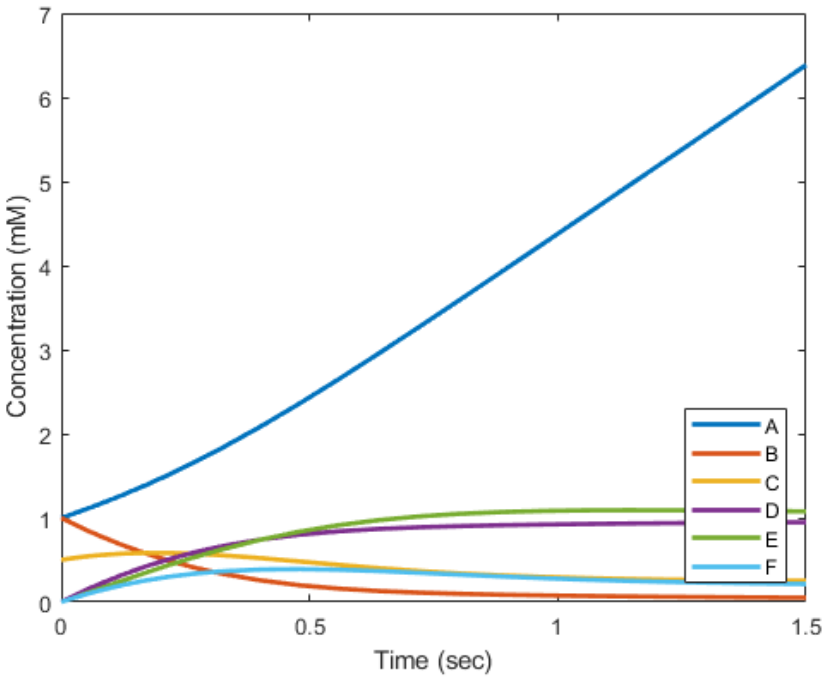
ii. Conservation statements

$$\frac{db}{dt} = -\frac{dd}{dt}$$

(d) Simulation



(e) Explain why is there no steady state if we choose $k_0 = 5$ mM sec^{-1}? (Hint: it depends on the conservation).



Increasing k_0 to this degree drives the reaction out of steady state by wildly overproducing A. The law of mass action drives quick conversion of all available B to C and D. Because of the conservation statement driving the population of B, strong overproduction of D drives the B population into the dirt, limiting the conversion of A to its products, and allowing the population to spiral.

2. Consider the closed system

   (a) Construct a differential equation model of the system. Simulate your model.

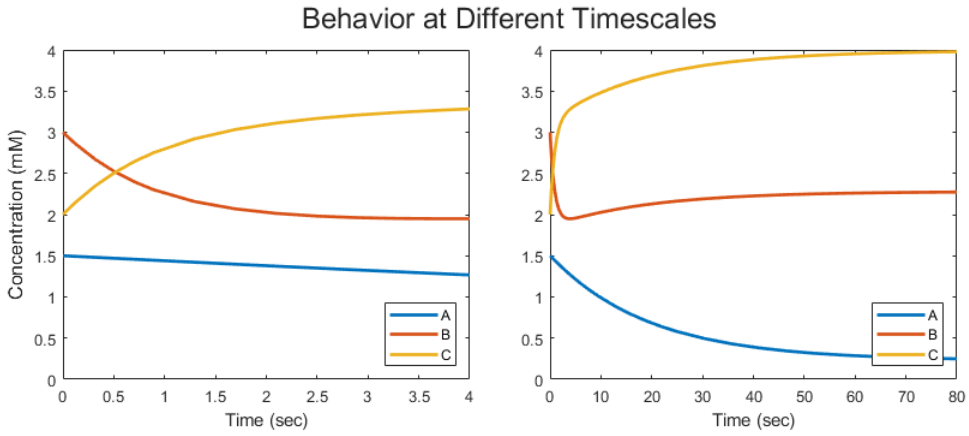
      i. Differential equations

$$\frac{da}{dt} = k_{-1} * b - k_1 * a$$

$$\frac{db}{dt} = -k_{-1} * b + k_1 * a - k_2 * b + k_{-2} * c$$

$$\frac{dc}{dt} = k_2 * b - k_{-2} * c$$

      ii. Simulation



(b) Reduced model

i. Differential equations

$$\frac{da}{dt} = k_{-1} * d - k_1 * a$$

$$\frac{dd}{dt} = k_1 * a - k_{-1}(\frac{k_{-2}}{k_{-2} + k_2} * d)$$

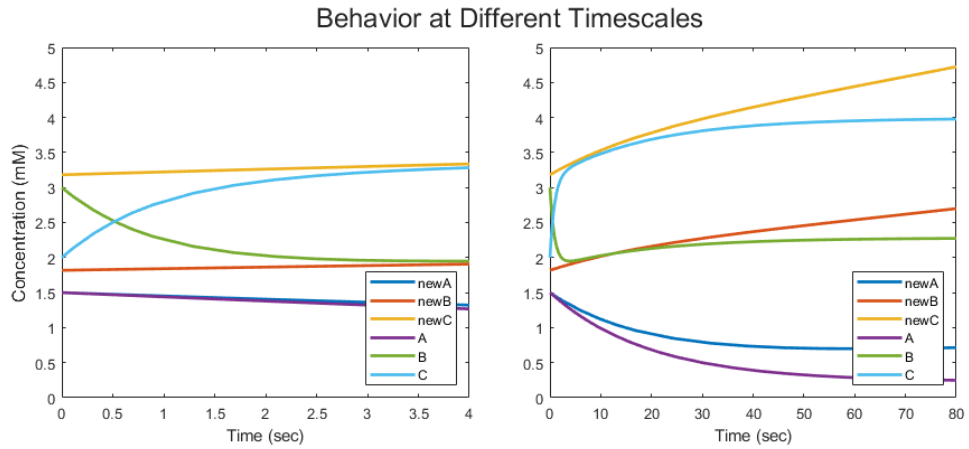
ii. Algebraic expressions

$$b(t) = \frac{k_{-2}}{k_{-2} + k_2} * d(t)$$

$$c(t) = \frac{k_2}{k_{-2} + k_2} * d(t)$$

$$d(0) = b(0) + c(0)$$

(c) Simulation



There is a period of the initial transient where the simulation does not match, but afterwards there is a period of very good agreement. In the long timescale, the simulation begins to diverge again.

3. Model Code for Question 1

```
function dYdt = q1e_model(t,Y)  % TODO - Write the function declaration.
    % Name of the function is lab1_model
    % TODO - Extract a, b, c, and d from input vector Y
    a = Y(1); %a
    b = Y(2); %b
    c = Y(3); %c
    d = Y(4); %d
    e = Y(5); %e
    f = Y(6); %f

    % TODO - Define the constants k1 through k5
    k0 = 0.5; %mM/s
    k1 = 3; %1/(mM*s)
    k2 = 1; %1/s
    k3 = 4; %1/s
    k4 = 1; %1/s
    k5 = 5; %1/s

    % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
    dadt = -(k1*(a*b))+ k0;
    dbdt = (k2*d) - (k1*(a*b));
    dcdt = (k1*(a*b)) - (k3*c);
    dddt = (k1*(a*b)) - (k2*d);
    dedt = (k3*c) - (k4*e);
    dfdt = (k3*c) - (k5*f);


    % Create output column vector dYdt
    dYdt = [dadt; dbdt; dcdt; dddt; dedt; dfdt];
end
```

4. Driver Code for Question 1

```
clear all

% TODO define the timespan to simulation
tRange = [0 1.5];

% TODO define the initial conditions
Y0 = [1,1,0.5,0,0,0];

% call the solver of choice (ode45 is fine)
[tSol,YSol] = ode15s(@q1e_model,tRange,Y0);

% plot solutions to look like figure in lab

plot(tSol,YSol(:,1),'LineWidth',2)
hold on
plot(tSol,YSol(:,2),'LineWidth',2)
plot(tSol,YSol(:,3),'LineWidth',2)
plot(tSol,YSol(:,4),'LineWidth',2)
plot(tSol,YSol(:,5),'LineWidth',2)
plot(tSol,YSol(:,6),'LineWidth',2)
legend('A','B','C','D','E','F','Location','southeast')
xlabel('Time (sec)')
ylabel('Concentration (mM)')
```

5. Model Code for Question 2 Before Rapid Equilibrium

```
function dYdt = q2a_model(t,Y)  % TODO - Write the function declaration.
    % Name of the function is lab1_model
    % TODO - Extract a, b, c, and d from input vector Y
    a = Y(1); %a
    b = Y(2); %b
    c = Y(3); %c

    % TODO - Define the constants k1 through k5
    k1 = 0.05; %mM/s
    k_1 = 0.005; %1/(mM*s)
    k2 = 0.7; %1/s
    k_2 = 0.4; %1/s

    % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
    dadt = (k_1*b) - (k1*a);
    dbdt = -(k_1*b) + (k1*a) - (k2*b) + (k_2*c);
    dcdt = (k2*b) - (k_2*c);


    % Create output column vector dYdt
    dYdt = [dadt; dbdt; dcdt];
end
```

6. Driver Code for Question 2 Before Rapid Equilibrium

```
clear all

% TODO define the timespan to simulation
tRange = [0 4];
tRange1 = [0 80];

% TODO define the initial conditions
Y0 = [1.5,3,2];

% call the solver of choice (ode45 is fine)
[tSol,YSol] = ode15s(@q2a_model,tRange,Y0);
[tSol1,YSol1] = ode15s(@q2a_model,tRange1,Y0);

% plot solutions to look like figure in lab

t = tiledlayout(1,2);

t1 = nexttile;
plot(tSol,YSol(:,1),'LineWidth',2)
hold on
plot(tSol,YSol(:,2),'LineWidth',2)
plot(tSol,YSol(:,3),'LineWidth',2)
legend('A','B','C','Location','southeast')
xlabel('Time (sec)')
hold off

t2 = nexttile;
plot(tSol1,YSol1(:,1),'LineWidth',2)
hold on
plot(tSol1,YSol1(:,2),'LineWidth',2)
plot(tSol1,YSol1(:,3),'LineWidth',2)
legend('A','B','C','Location','southeast')
xlabel('Time (sec)')
hold off

linkaxes([t1,t2],'y');
```

```
title(t, 'Behavior at Different Timescales','FontSize',18);
ylabel(t,'Concentration (mM)', 'FontSize',12);
t.TileSpacing = 'compact';
```

7. Model Code for Question 2 After Rapid Equilibrium

```
function dYdt = q2a_reducedmodel(t,Y)  % TODO - Write the function declaration.
    % Name of the function is lab1_model
    % TODO - Extract a, b, c, and d from input vector Y
    a = Y(1); %a
    d = Y(2); %d

    % TODO - Define the constants k1 through k5
    k1 = 0.05; %mM/s
    k_1 = 0.005; %1/(mM*s)
    k2 = 0.7; %1/s
    k_2 = 0.4; %1/s

    % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
    dadt = (k_1*d) - (k1*a);
    dddt = (k1*a) - k_1*((k_2*d)/(k_2+k2));

    % Create output column vector dYdt
    dYdt = [dadt; dddt];
end
```

8. Driver Code for Question 2 After Rapid Equilibrium

```
clear all

% TODO define the timespan to simulation
tRange = [0 4];
tRange1 = [0 80];

% TODO define the initial conditions
Y0 = [1.5,5];
Y1 = [1.5,3,2];

% TODO - Define the constants k1 through k5
k1 = 0.05; %mM/s
k_1 = 0.005; %1/(mM*s)
k2 = 0.7; %1/s
k_2 = 0.4; %1/s

% call the solver of choice (ode45 is fine)
[tSol,YSol] = ode15s(@q2a_reducedmodel,tRange,Y0);
[tSol1,YSol1] = ode15s(@q2a_reducedmodel,tRange1,Y0);

[tSol2,YSol2] = ode15s(@q2a_model,tRange,Y1);
[tSol3,YSol3] = ode15s(@q2a_model,tRange1,Y1);

% back solve for approximated b and c concentrations
newB = (k_2/(k_2+k2))*YSol(:,2);
newC = (k2/(k_2+k2))*YSol(:,2);
newnewB = (k_2/(k_2+k2))*YSol1(:,2);
newnewC = (k2/(k_2+k2))*YSol1(:,2);

% plot solutions to look like figure in lab

```

```
t = tiledlayout(1,2);

t1 = nexttile;
plot(tSol,YSol(:,1),'LineWidth',2)
hold on
plot(tSol,newB,'LineWidth',2)
plot(tSol,newC,'LineWidth',2)
plot(tSol2,YSol2(:,1),'LineWidth',2)
plot(tSol2,YSol2(:,2),'LineWidth',2)
plot(tSol2,YSol2(:,3),'LineWidth',2)
legend('newA','newB','newC','A','B','C','Location','southeast')
xlabel('Time (sec)')
hold off

t2 = nexttile;
plot(tSol1,YSol1(:,1),'LineWidth',2)
hold on
plot(tSol1,newnewB,'LineWidth',2)
plot(tSol1,newnewC,'LineWidth',2)
plot(tSol3,YSol3(:,1),'LineWidth',2)
plot(tSol3,YSol3(:,2),'LineWidth',2)
plot(tSol3,YSol3(:,3),'LineWidth',2)
legend('newA','newB','newC','A','B','C','Location','southeast')
xlabel('Time (sec)')
hold off

linkaxes([t1,t2],'y');
title(t, 'Behavior at Different Timescales','FontSize',18);
ylabel(t,'Concentration (mM)', 'FontSize',12);
t.TileSpacing = 'compact';
```