

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

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

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

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

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

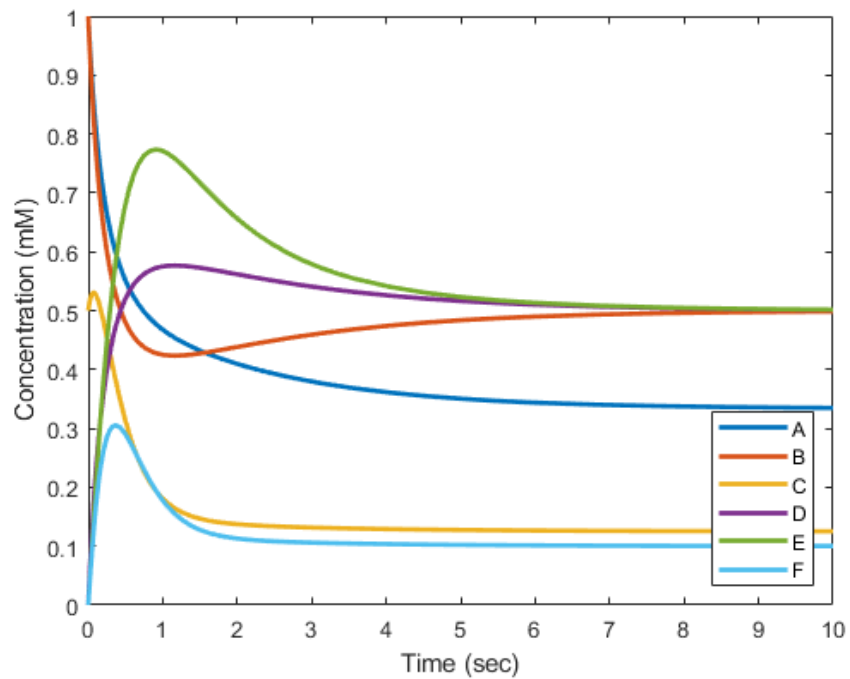
$$\frac{de}{dt} = k_3 * c - k_4 * e$$

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

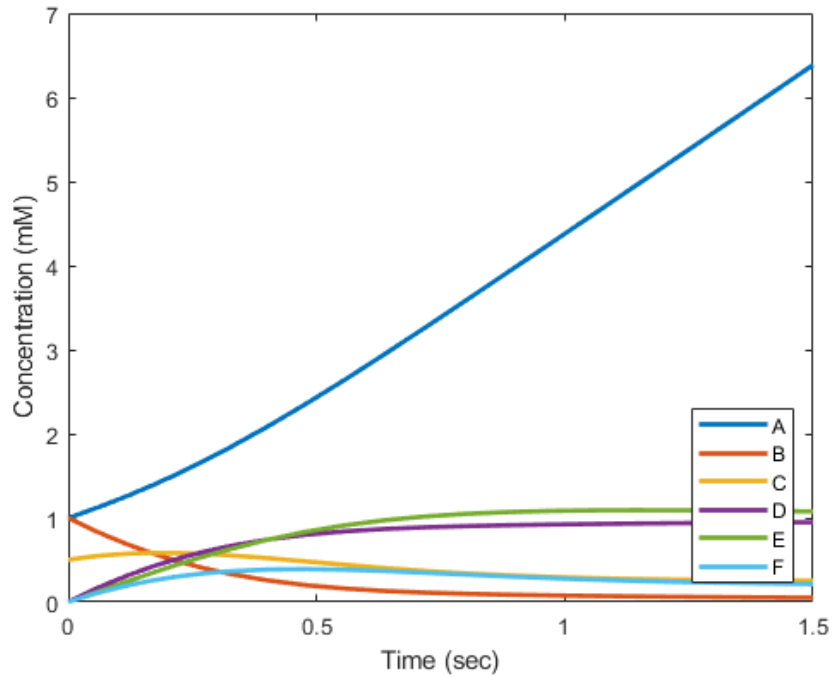
ii. Conservation statements

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

(d) Simulation



(e) Explain why is there no steady state if we choose $k_0 = 5 \text{ mM sec}^{-1}$? (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.

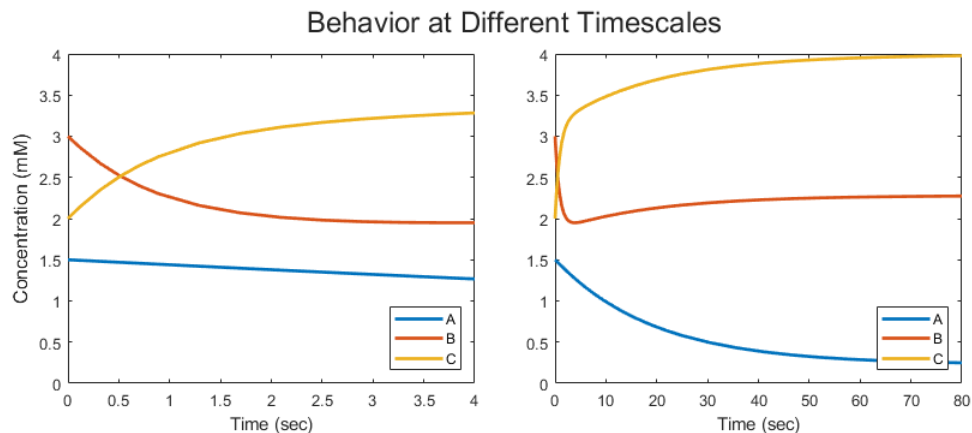
i. Differential equations

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

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

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

ii. Simulation



(b) Reduced model

i. Differential equations

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

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

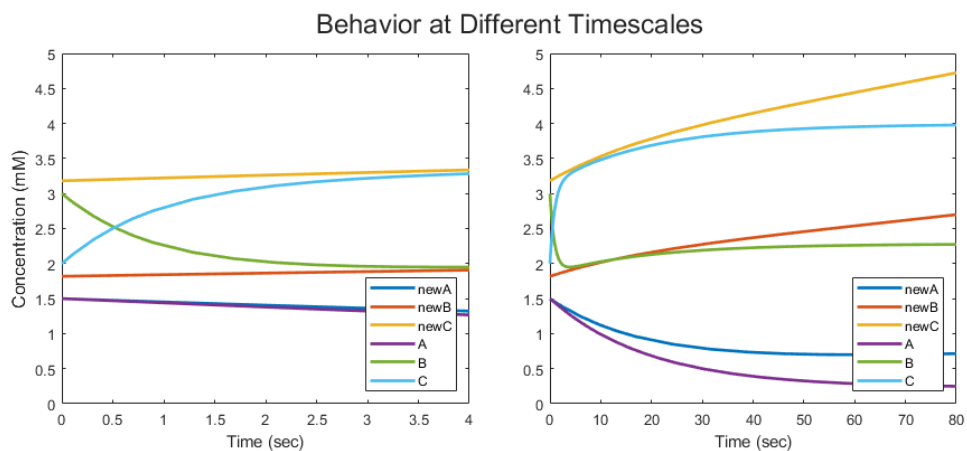
ii. Algebraic expressions

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

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

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

(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

```
1 function dYdt = q1e_model(t,Y) % TODO - Write the function declaration.
2     % Name of the function is lab1_model
3     % TODO - Extract a, b, c, and d from input vector Y
4     a = Y(1); %a
5     b = Y(2); %b
6     c = Y(3); %c
7     d = Y(4); %d
8     e = Y(5); %e
9     f = Y(6); %f
10
11     % TODO - Define the constants k1 through k5
12     k0 = 0.5; %mM/s
13     k1 = 3; %1/(mM*s)
14     k2 = 1; %1/s
15     k3 = 4; %1/s
16     k4 = 1; %1/s
17     k5 = 5; %1/s
18
19     % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
20     dadt = -(k1*(a*b)) + k0;
21     dbdt = (k2*d) - (k1*(a*b));
22     dcdt = (k1*(a*b)) - (k3*c);
23     dddt = (k1*(a*b)) - (k2*d);
24     dedt = (k3*c) - (k4*e);
25     dfdt = (k3*c) - (k5*f);
26
27
28     % Create output column vector dYdt
29     dYdt = [dadt; dbdt; dcdt; dddt; dedt; dfdt];
30 end
```

4. Driver Code for Question 1

```
1 clear all
2
3 % TODO define the timespan to simulation
4 tRange = [0 1.5];
5
6 % TODO define the initial conditions
7 Y0 = [1,1,0.5,0,0,0];
8
9 % call the solver of choice (ode45 is fine)
10 [tSol,YSol] = ode15s(@q1e_model,tRange,Y0);
11
12 % plot solutions to look like figure in lab
13
14 plot(tSol,YSol(:,1),'LineWidth',2)
15 hold on
16 plot(tSol,YSol(:,2),'LineWidth',2)
17 plot(tSol,YSol(:,3),'LineWidth',2)
18 plot(tSol,YSol(:,4),'LineWidth',2)
19 plot(tSol,YSol(:,5),'LineWidth',2)
20 plot(tSol,YSol(:,6),'LineWidth',2)
21 legend('A','B','C','D','E','F','Location','southeast')
22 xlabel('Time (sec)')
23 ylabel('Concentration (mM)')
```

5. Model Code for Question 2 Before Rapid Equilibrium

```

1 function dYdt = q2a_model(t,Y) % TODO - Write the function declaration.
2     % Name of the function is lab1_model
3     % TODO - Extract a, b, c, and d from input vector Y
4     a = Y(1); %a
5     b = Y(2); %b
6     c = Y(3); %c
7
8     % TODO - Define the constants k1 through k5
9     k1 = 0.05; %mM/s
10    k_1 = 0.005; %1/(mM*s)
11    k2 = 0.7; %1/s
12    k_2 = 0.4; %1/s
13
14    % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
15    dadt = (k_1*b) - (k1*a);
16    dbdt = -(k_1*b) + (k1*a) - (k2*b) + (k_2*c);
17    dcdt = (k2*b) - (k_2*c);
18
19
20    % Create output column vector dYdt
21    dYdt = [dadt; dbdt; dcdt];
22 end

```

6. Driver Code for Question 2 Before Rapid Equilibrium

```

1 clear all
2
3 % TODO define the timespan to simulation
4 tRange = [0 4];
5 tRange1 = [0 80];
6
7 % TODO define the initial conditions
8 Y0 = [1.5,3,2];
9
10 % call the solver of choice (ode45 is fine)
11 [tSol,YSol] = ode15s(@q2a_model,tRange,Y0);
12 [tSol1,YSol1] = ode15s(@q2a_model,tRange1,Y0);
13
14 % plot solutions to look like figure in lab
15
16 t = tiledlayout(1,2);
17
18 t1 = nexttile;
19 plot(tSol,YSol(:,1),'LineWidth',2)
20 hold on
21 plot(tSol,YSol(:,2),'LineWidth',2)
22 plot(tSol,YSol(:,3),'LineWidth',2)
23 legend('A','B','C','Location','southeast')
24 xlabel('Time (sec)')
25 hold off
26
27 t2 = nexttile;
28 plot(tSol1,YSol1(:,1),'LineWidth',2)
29 hold on
30 plot(tSol1,YSol1(:,2),'LineWidth',2)
31 plot(tSol1,YSol1(:,3),'LineWidth',2)
32 legend('A','B','C','Location','southeast')
33 xlabel('Time (sec)')
34 hold off
35
36 linkaxes([t1,t2],'y');

```

```

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

```

7. Model Code for Question 2 After Rapid Equilibrium

```

1 function dYdt = q2a_reducedmodel(t,Y) % TODO - Write the function declaration.
2     % Name of the function is lab1_model
3     % TODO - Extract a, b, c, and d from input vector Y
4     a = Y(1); %a
5     d = Y(2); %d
6
7     % TODO - Define the constants k1 through k5
8     k1 = 0.05; %mM/s
9     k_1 = 0.005; %1/(mM*s)
10    k2 = 0.7; %1/s
11    k_2 = 0.4; %1/s
12
13    % TODO - Define dadt, dbdt, dcdt, dddt from the ODEs
14    dadt = (k_1*d) - (k1*a);
15    dddt = (k1*a) - k_1*((k_2*d)/(k_2+k2));
16
17    % Create output column vector dYdt
18    dYdt = [dadt; dddt];
19 end

```

8. Driver Code for Question 2 After Rapid Equilibrium

```

1 clear all
2
3 % TODO define the timespan to simulation
4 tRange = [0 4];
5 tRange1 = [0 80];
6
7 % TODO define the initial conditions
8 Y0 = [1.5,5];
9 Y1 = [1.5,3,2];
10
11 % TODO - Define the constants k1 through k5
12 k1 = 0.05; %mM/s
13 k_1 = 0.005; %1/(mM*s)
14 k2 = 0.7; %1/s
15 k_2 = 0.4; %1/s
16
17 % call the solver of choice (ode45 is fine)
18 [tSol,YSol] = ode15s(@q2a_reducedmodel,tRange,Y0);
19 [tSol1,YSol1] = ode15s(@q2a_reducedmodel,tRange1,Y0);
20
21 [tSol2,YSol2] = ode15s(@q2a_model,tRange,Y1);
22 [tSol3,YSol3] = ode15s(@q2a_model,tRange1,Y1);
23
24 % back solve for approximated b and c concentrations
25 newB = (k_2/(k_2+k2))*YSol(:,2);
26 newC = (k2/(k_2+k2))*YSol(:,2);
27 newnewB = (k_2/(k_2+k2))*YSol1(:,2);
28 newnewC = (k2/(k_2+k2))*YSol1(:,2);
29
30 % plot solutions to look like figure in lab
31

```

```

32 t = tiledlayout(1,2);
33
34 t1 = nexttile;
35 plot(tSol,YSol(:,1),'LineWidth',2)
36 hold on
37 plot(tSol,newB,'LineWidth',2)
38 plot(tSol,newC,'LineWidth',2)
39 plot(tSol2,YSol2(:,1),'LineWidth',2)
40 plot(tSol2,YSol2(:,2),'LineWidth',2)
41 plot(tSol2,YSol2(:,3),'LineWidth',2)
42 legend('newA','newB','newC','A','B','C','Location','southeast')
43 xlabel('Time (sec)')
44 hold off
45
46 t2 = nexttile;
47 plot(tSol1,YSol1(:,1),'LineWidth',2)
48 hold on
49 plot(tSol1,newnewB,'LineWidth',2)
50 plot(tSol1,newnewC,'LineWidth',2)
51 plot(tSol3,YSol3(:,1),'LineWidth',2)
52 plot(tSol3,YSol3(:,2),'LineWidth',2)
53 plot(tSol3,YSol3(:,3),'LineWidth',2)
54 legend('newA','newB','newC','A','B','C','Location','southeast')
55 xlabel('Time (sec)')
56 hold off
57
58 linkaxes([t1,t2],'y');
59 title(t, 'Behavior at Different Timescales','FontSize',18);
60 ylabel(t,'Concentration (mM)', 'FontSize',12);
61 t.TileSpacing = 'compact';

```