Solution: Exercise 1

1. **Review Material from the Lecture**

   a) *Describe the concept of mass action!* In chemistry, the law of mass action is the proposition that the rate of a chemical reaction is directly proportional to the product of the activities or concentrations of the reactants. It explains and predicts behaviours of solutions in dynamic equilibrium.

   b) *What are Michaelis-Menten and Hill kinetics?* In biochemistry, Michaelis-Menten kinetics is one of the best-known models of enzyme kinetics. It is named after German biochemist Leonor Michaelis and Canadian physician Maud Menten. The model takes the form of an equation describing the rate of enzymatic reactions, by relating reaction rate v (rate of formation of product, $[P]$) to $[S]$, the concentration of a substrate S. Its formula is given by the so-called Michaelis-Menten equation,

$$v = \frac{d[P]}{dt} = V_{max}\frac{[S]}{[S]+K_M}. \tag{1}$$

   Here, V_{max} represents the maximum rate achieved by the system(which happens at saturating substrate concentration). The value of the Michaelis constant K_M is numerically equal to the substrate concentration at which the reaction rate is half of V_{max}. Biochemical reactions involving a single substrate are often assumed to follow Michaelis-Menten kinetics, without regard to the model's underlying assumptions. It is, however, important to keep in mind that a key assumption in its derivation is the quasi-steady state of the complex of substrate and enzyme, as will be the case when the substrate concentration is much higher than the enzyme concentration. This is typically the case in metabolic reactions, but not in signalling networks, where the Michaelis-Menten kinetics are also used. (Text based on `https://en.wikipedia.org/wiki/Michaelis%E2%80%93Menten_kinetics|`)

   Hill kinetics: The Hill equation, originally devised by Archibald Vivian Hill in 1910 to describe the kinetics of oxygen binding to hemoglobin, provides an extension of the Michaelis-Menten kinetics to cooperative binding reactions. We speak of cooperativity if the binding of the first ligand to a macromolecule with several ligand binding sites alters the binding behaviour and affinity of the subsequent binding ligands. If the binding of subsequent ligands becomes easier, then we speak of positive cooperativity, otherwise of negative cooperativity. The Hill equation reads

$$v = \frac{d[P]}{dt} = v_{max}\frac{[S]^n}{[S]^n+K^n}. \tag{2}$$

   K is the Hill constant, while n is referred to as Hill coefficient and quantifies the extent of the cooperativity. For $n = 1$, ligand binding events are completely independent of each other and we obtain the Michaelis-Menten kinetics. $n > 1$ corresponds to positive cooperativity, $n < 1$ to negative cooperativity. The Hill coefficient was originally devised to explain the cooperative binding of oxygen to haemoglobin (a system which has a Hill coefficient of 2.8-3.0).

   c) *How can activator and inhibitory action be represented in mathematical models of biological regulatory networks? Activators*, A, are modelled to speed up a reaction like the substrate in the Michaelis-Menten and Hill equations,

$$v = v_{max}\frac{[A]^n}{[A]^n+K^n}. \tag{3}$$

Inhibitors of a chemical reaction either fully prevent a reaction or reduce the reaction rate. When the effect of an inhibitor is reversible, the steady state of the inhibited species is reduced, whereas in the case of irreversible inhibition the steady state is zero. Here we will only focus on reversible inhibitions. An important regulatory paradigm is the use of inhibitors and activators to modulate the speed of reactions. Inhibitors can either compete with the substrate for the catalytic cleft (competitive inhibition) or alternatively inhibitors can induce a conformational change that alters the activity of the enzyme (allosteric inhibition).

Competitive Inhibition

$$v = v_{max} \frac{[X]}{[X] + K_M(1 + \frac{[I]}{K_I})} \tag{4}$$

Allosteric Inhibition

$$v = \frac{v_{max}}{1 + \frac{[I]}{K_I}} \frac{[X]}{K_M + [X]}. \tag{5}$$

Here, K_I is the affinity with which the inhibitor binds to the enzyme.

2. **Basic biochemical reaction mechanisms**.

a) *Complete Table 1. Assume mass action kinetics for all six reaction mechanisms.*

	Interaction Graph	Rate equation scheme	ODE
a)	$\begin{tikzcd} {} \arrow[r] & A \arrow[r] & \emptyset \end{tikzcd}$	$\begin{aligned} &\xrightarrow{k_1} A \\ A &\xrightarrow{k_2} \emptyset \end{aligned}$	$[\dot{A}] = k_1 - k_2[A]$
b)	$\begin{tikzcd} A \arrow[dr, <->] & & B \arrow[dl, <->] \\ & C & \end{tikzcd}$	$A + B \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C$	$\begin{aligned} [\dot{A}] &= -k_1[A][B] + k_{-1}[C] \\ [\dot{B}] &= -k_1[A][B] + k_{-1}[C] \\ [\dot{C}] &= k_1[A][B] - k_{-1}[C] \end{aligned}$
c)	$\begin{tikzcd} A \arrow[dr] & & A \arrow[dl] \\ & C & \end{tikzcd}$	$A + A \xrightarrow{k} C$	$\begin{aligned} [\dot{A}] &= -2k[A][A] \\ [\dot{C}] &= k[A][A] \end{aligned}$
d)	$\begin{tikzcd} & A \arrow[d] & \\ B \arrow[r, dash] & {} \arrow[r] & C \end{tikzcd}$	$[B] + [A] \xrightarrow{k_1} [C] + [A]$ or $[B] \xrightarrow{k_1[A]} [C]$	$\begin{aligned} [\dot{A}] &= 0 \\ [\dot{B}] &= -k_1[A][B] \\ [\dot{C}] &= k_1[A][B] \end{aligned}$
e)	$\begin{tikzcd} A \arrow[dr, <->] & & A \arrow[dl, <->] & \\ & AA \arrow[rr, <->] & & AAA \arrow[ul, bend right] \end{tikzcd}$	$\begin{aligned} A + A &\underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} AA \\ A + AA &\underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} AAA \end{aligned}$	$\begin{aligned} [\dot{A}] &= -k_1[A](2[A] + [AA]) \\ &\quad +k_{-1}(2[AA] + [AAA]) \\ [\dot{AA}] &= k_1[A][A] + k_{-1}[AAA] \\ &\quad -[AA](k_{-1} + k_1[A]) \\ [\dot{AAA}] &= k_1[A][AA] - k_{-1}[AAA] \end{aligned}$
f)	$\begin{tikzcd} & S \arrow[d] & & \\ A \arrow[r, dash] & {} \arrow[r] & A^* \arrow[d] & \\ & B \arrow[r, dash] & {} \arrow[r] & B^* \end{tikzcd}$	$\begin{aligned} [A] &\xrightarrow{k_1[S]} [A^*] \\ [B] &\xrightarrow{k_2[A^*]} [B^*] \end{aligned}$	$\begin{aligned} [\dot{A}] &= -k_1[S][A] \\ [\dot{A^*}] &= k_1[S][A] \\ [\dot{B}] &= -k_2[A^*][B] \\ [\dot{B^*}] &= k_2[A^*][B] \end{aligned}$

b) *For each mechanism of 2.a implement in Matlab the corresponding ordinary differential equations (ODEs). As rate constants set all to the value 1 and initial conditions to 0.1. Set the integration time to 10 and plot all the state variables over that time span for each mechanism.*

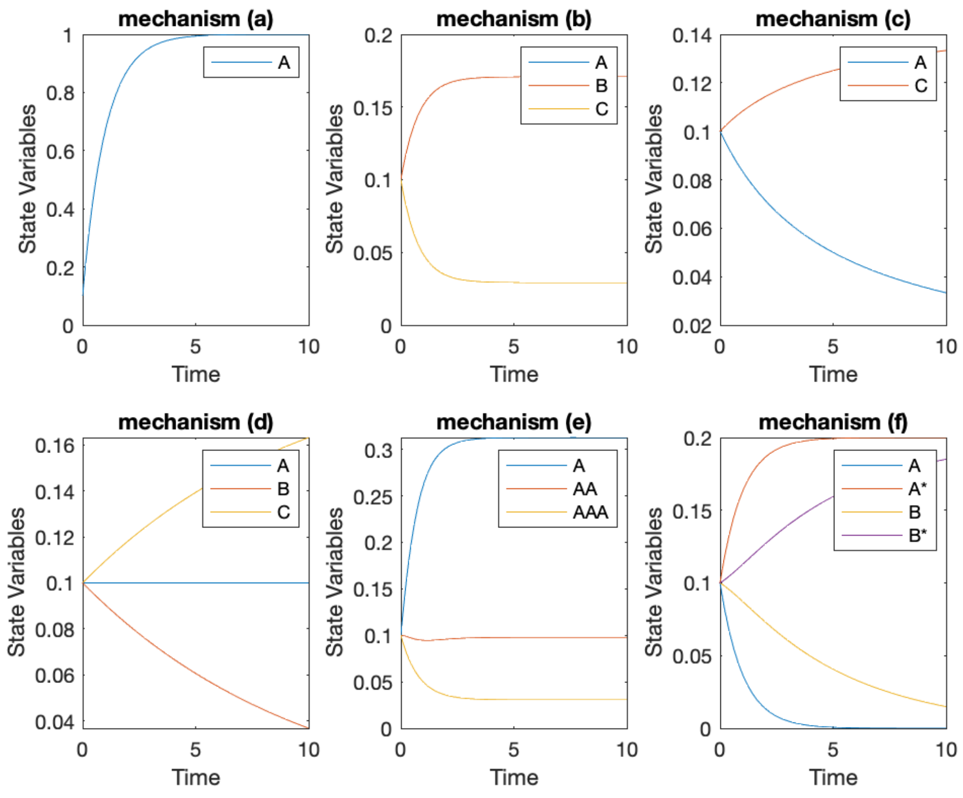

Figure 1: **Numerical Solutions.** The values of the state variables are plotted against time for the different reaction networks in Exercise 2a.

Matlab Code

```
function Ex1_2b

close all % close all previous graphics
clear all % clear the values of all variables

%%%% Mechanism (a)
p(1) = 1; % k1: production rate constant
p(2) = 1; % k2: degradation rate constant
x0   = 0.1; % set initial condition for A
tspan = [0 10]; % define time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_a,x_a] = ode45(@mech_a,tspan,x0,[],p);

clear p x0
%%%% Mechanism (b)
p(1) = 1; % Complex formation rate constant
p(2) = 1; % Complex disossiation rate constant
x0   = [0.1 0.1 0.1]; % initial conditions
tspan = [0 10]; % time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_b,x_b] = ode45(@mech_b,tspan,x0,[],p);

```

```matlab
clear p x0
%%%% Mechanism (c)
p(1) = 1;      % Dimer-association rate constant
x0   = [0.1 0.1]; % initial conditions
tspan = [0 10]; % time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_c,x_c] = ode45(@mech_c,tspan,x0,[],p);

clear p x0
%%%% Mechanism (d)
p(1) = 1; % Modification rate constant
x0 = [0.1 0.1 0.1]; % initial conditions
tspan = [0 10]; % time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_d,x_d] = ode45(@mech_d,tspan,x0,[],p);

clear p x0
%%%% Mechanism (e)
p(1) = 1; % Dimer/Trimer-association rate constant
p(2) = 1; % Dimer/Trimer-dissociation rate constant
x0 = [0.1 0.1 0.1]; % initial conditions
tspan = [0 10]; % time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_e,x_e] = ode45(@mech_e,tspan,x0,[],p);

clear p x0
%%%% Mechanism (f)
p(1) = 1; % Modification rate constant of A
p(2) = 1; % Modification rate constant of B
p(3) = 1; % Signaling rate constant
x0 = [0.1 0.1 0.1 0.1]; % initial conditions
tspan = [0 10]; % time span of integration
% Call matlab integrator ode45 to solve the ODEs for (a)
[t_f,x_f] = ode45(@mech_f,tspan,x0,[],p);


% evoke a graphics object
figure(1)
% create a subplot in the 2 by 3 subplot matrix, in position 1
hold on

subplot(2,3,1)
plot(t_a,x_a)
title('mechanism (a)')
legend({'A'})
xlabel('Time')
ylabel('State Variables')

subplot(2,3,2)
plot(t_b,x_b)
title('mechanism (b)')
legend({'A','B','C'})
xlabel('Time')
ylabel('State Variables')

subplot(2,3,3)
plot(t_c,x_c)
title('mechanism (c)')
legend({'A','C'})
xlabel('Time')
ylabel('State Variables')

subplot(2,3,4)
plot(t_d,x_d)
title('mechanism (d)')
legend({'A','B','C'})
xlabel('Time')
ylabel('State Variables')

subplot(2,3,5)
```

```matlab
plot(t_e,x_e)
title('mechanism (e)')
legend({'A','AA','AAA'})
xlabel('Time')
ylabel('State Variables')

subplot(2,3,6)
plot(t_f,x_f)
title('mechanism (f)')
legend({'A','A*','B','B*'})
xlabel('Time')
ylabel('State Variables')

hold off

end

function dxdt_a = mech_a(t,x,p)
k1  = p(1);
k2  = p(2);
dxdt_a = zeros(1,1);
dxdt_a(1) = k1-k2*x(1);
end

function dxdt_b = mech_b(t,x,p)
k1  = p(1);
k_1 = p(2);
dxdt_b = zeros(3,1);
dxdt_b(1) = -k1*x(1)*x(2) + k_1*x(3);
dxdt_b(2) = -k1*x(1)*x(2) + k_1*x(3);
dxdt_b(3) = k1*x(1)*x(2) - k_1*x(3);
end

function dxdt_c = mech_c(t,x,p)
k  = p(1);
dxdt_c = zeros(2,1);
dxdt_c(1) = -2*k*x(1)*x(1);
dxdt_c(2) = k*x(1)*x(1);
end

function dxdt_d = mech_d(t,x,p)
k1  = p(1);
dxdt_d = zeros(3,1);
dxdt_d(1) = 0;
dxdt_d(2) = -k1*x(1)*x(2);
dxdt_d(3) = k1*x(1)*x(2);
end

function dxdt_e = mech_e(t,x,p)
k1  = p(1);
k_1 = p(2);
dxdt_e = zeros(3,1);
dxdt_e(1) = -k1*x(1)*(2*x(1)+x(2)) + k_1*(2*x(2)+x(3));
dxdt_e(2) = k1*x(1)*x(1) + k_1*x(3) - x(2)*(k_1 + k1*x(1));
dxdt_e(3) = k1*x(1)*x(2) - k_1*x(3);
end

function dxdt_f = mech_f(t,x,p)
k1  = p(1);
k2  = p(2);
S   = p(3);
dxdt_f = zeros(4,1);
dxdt_f(1) = -k1*S*x(1);
dxdt_f(2) = k1*S*x(1);
dxdt_f(3) = -k2*x(2)*x(3);
dxdt_f(4) = k2*x(2)*x(3);
end
```