Knowing that I need to numerically solve some equations using the fourth-order Runge-Kutta method, so I still prefer to use my adept mathematics software (MATLAB) to solve this problem.

Because MATLAB has a differential function, the ode45() function, specifically for solving the fourth-order Runge-Kutta method, it will be much easier to solve the second question.

1. Four Equations

In Wikipedia, I found the differential equation of the law of mass action. As shown in the figure below

Dynamic approach to the equilibrium state [edit]

The third paper of 1864[8] was concerned with the kinetics of the same equilibrium system. Writing the dissociated active mass at some point in time as x, the rate of reaction was given as

$$\left(\frac{dx}{dt}\right)_{\text{forward}} = k(p-x)^a(q-x)^b$$

Likewise the reverse reaction of A' with B' proceeded at a rate given by

$$\left(\frac{dx}{dt}\right)_{\text{reverse}} = k'(p'+x)^{a'}(q'+x)^{b'}$$

Figure 1

However, from my perspective, I'm not sure if the above formula is suitable for this question. Because the question required is equations for the rate of changes, it should be in the following shape

$$\frac{d[E]}{dt}, \frac{d[S]}{dt}, \frac{d[ES]}{dt}, \frac{d[P]}{dt}$$

Finally, I find the most appropriate explanation for the law of mass action in Exploring Mathematical Modeling in Biology Through Case Studies and Experimental Activities. As shown in the figure below

$$A + B \underset{k_-}{\overset{k_+}{\rightleftharpoons}} C$$

. Here the rate of change of [C] is equal to the rate of production of C minus the rate of consumption of C, that is,

$$\frac{d[C]}{dt} = k_+[A][B] - k_-[C],$$

and

$$\frac{d[A]}{dt} = \frac{d[B]}{dt} = -k_+[A][B] + k_-[C].$$

Figure 2

So, I write four equations for the rate of changes of the four species (E, S, ES, and P). As shown below

$$\frac{d[E]}{dt} = -k_1 \cdot [E] \cdot [S] + k_2 \cdot [ES] + k_3 \cdot [ES]$$

$$\frac{d[S]}{dt} = -k_1 \cdot [E] \cdot [S] + k_2 \cdot [ES]$$

$$\frac{d[ES]}{dt} = k_1 \cdot [E] \cdot [S] - k_2 \cdot [ES] - k_3 \cdot [E] \cdot [P]$$

$$\frac{d[P]}{dt} = k_3 \cdot [ES]$$

2. Fourth-order Runge-Kutta Method

MATLAB has a differential function, the ode45() function, specifically for solving the fourth-order Runge-Kutta method, so it will be much easier to solve the second question.

The first step is to establish a function containing four differential equations. Using the code to implement, we can get the following code

For code details, please kindly find the attached file " Equations.m " (Open with MATLAB). For the convenience of review, I have copied all the code of the document as follows.

```
function [f] = Equations(t,x)

k1 = 100; k2 = 600; k3 = 150;
% The following sentence will be used

% Set [E] is x(1)
% Set [S] is x(2)
% Set [ES] is x(3)
% Set [P] is x(4)

Eq1 = -k1 * x(1) * x(2) + ( k2 + k3 ) * x(3);
Eq2 = -k1 * x(1) * x(2) + k2 * x(3);
Eq3 = k1 * x(1) * x(2) -( k2 + k3 ) * x(3);
Eq4 = k3 * x(3);
% Corresponding to the four equations mentioned in the first section

f = [Eq1; Eq2; Eq3; Eq4];
% Four differential equations

end
```

The second step is to solve these four differential equations by using the ode45() function. Also, we can draw the numerical solution of these four differential functions by using the plot() function. The results are shown in the figure below.

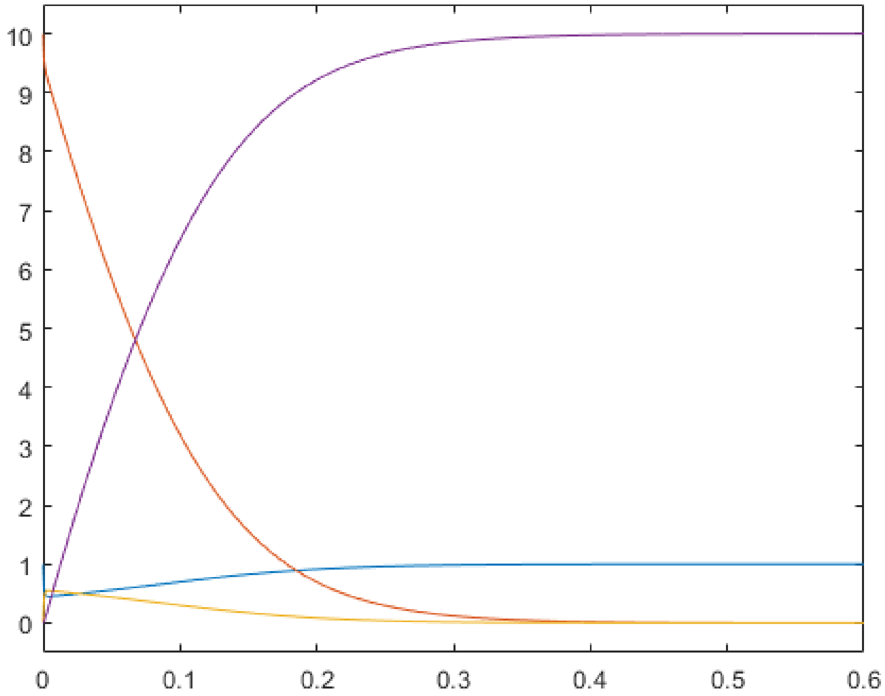

Figure 3

For figure details, please kindly find the attached file " Numerical Solution.png ".

Using the code to implement, we can get the following code

For output data, please kindly find the attached file " output_question_8_2.txt ".
For code details, please kindly find the attached file " Runge_Kutta.m " (Open with MATLAB). For the convenience of review, I have copied all the code of the document as follows.

```
[t,x]=ode45('Equations',[0,1],[1;10;0;0]);
% Use the ode45() function to solve these four differential functions

plot(t,x);
axis([0,0.6,-0.5,10.5]);
% Draw the numerical solution of these four differential functions
fileID = fopen('output_question_8_2.txt','wt'); % Control the output

fprintf(fileID,'%s %s %s %s %s\n','x1','x2','x3','x4','t');
% Control the output (Title)
fprintf(fileID,'%7.5f %7.5f %7.5f %7.5f %7.5f\n',[t,x]');
% Write the results into the document

fclose(fileID); % Finish the control
```

3. Velocity Maximum Value

According to the question requirement, V is equal to the following equation

$$V = \frac{d[P]}{dt} = k_3 \cdot [ES]$$

Using the code to implement, we can get the following code

For code details, please kindly find the attached file " Maximum_Value.m " (Open with MATLAB). For the convenience of review, I have copied all the code of the document as follows.

```
[t,x]=ode45('Equations',[0,1],[1;10;0;0]);
% Use the ode45() function to solve these four differential functions

k1 = 100; k2 = 600; k3 = 150;
% The following sentence will be used

S = x(:,2);
V = k3 .* x(:,3);
% According to Eq4, d[P]/dt = V = k3 * x(3);

plot(S,V);
% Draw the numerical solution of S and V

% The following sentence is to find the velocity maximum value
% And its corresponding value of S
find = [S,V];
for i = 1:size(find,1)
    if find(i,2) == max(V)
        disp(find(i,1))
        disp(find(i,2))
    end
end
```

Through traversing, I find that the maximum value of V is 82.6478, and its corresponding value of S is 9.2075. (Keep 4 decimal places)

The plot results are shown in the figure below.

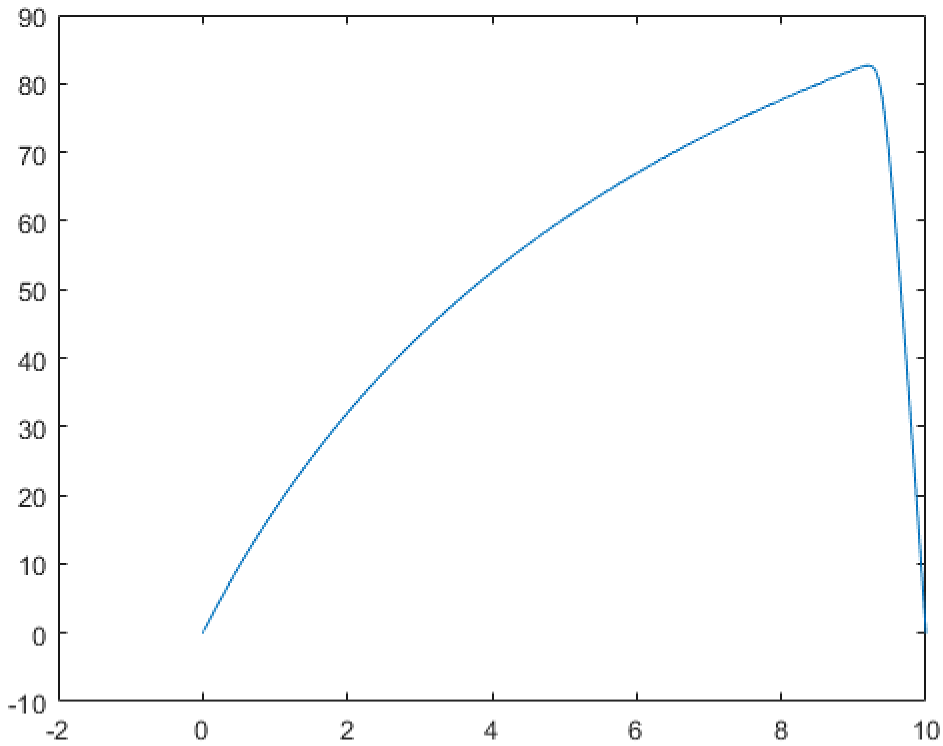

Figure 4

For figure details, please kindly find the attached file " Maximum_Value.png ".