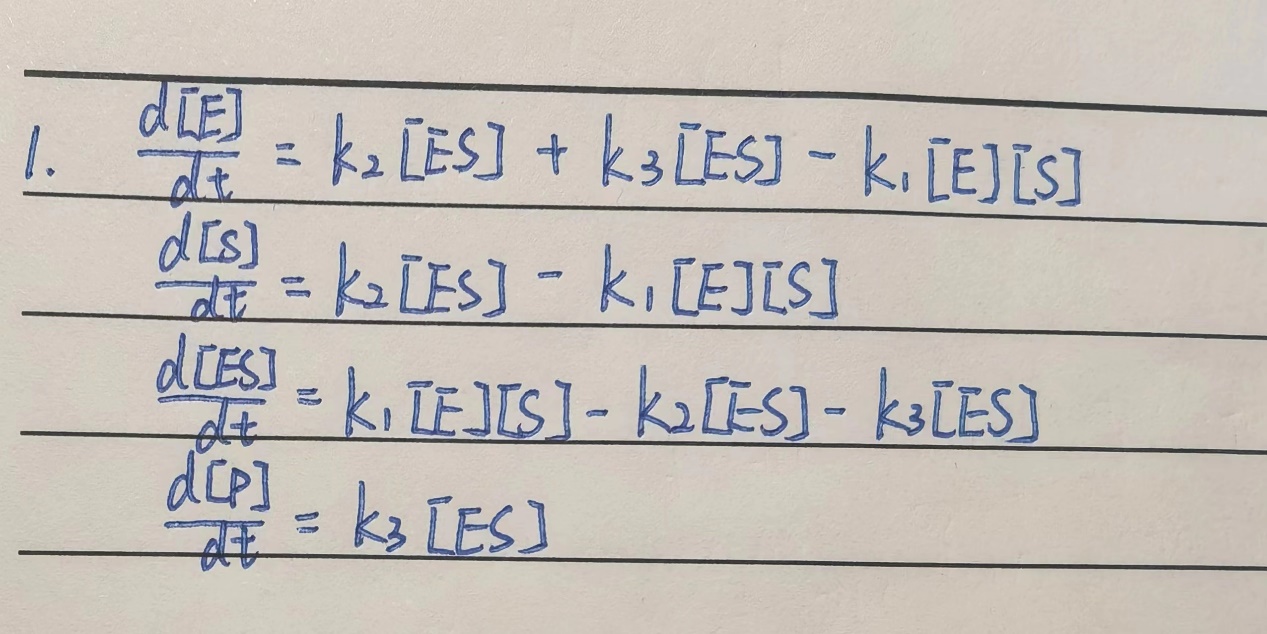
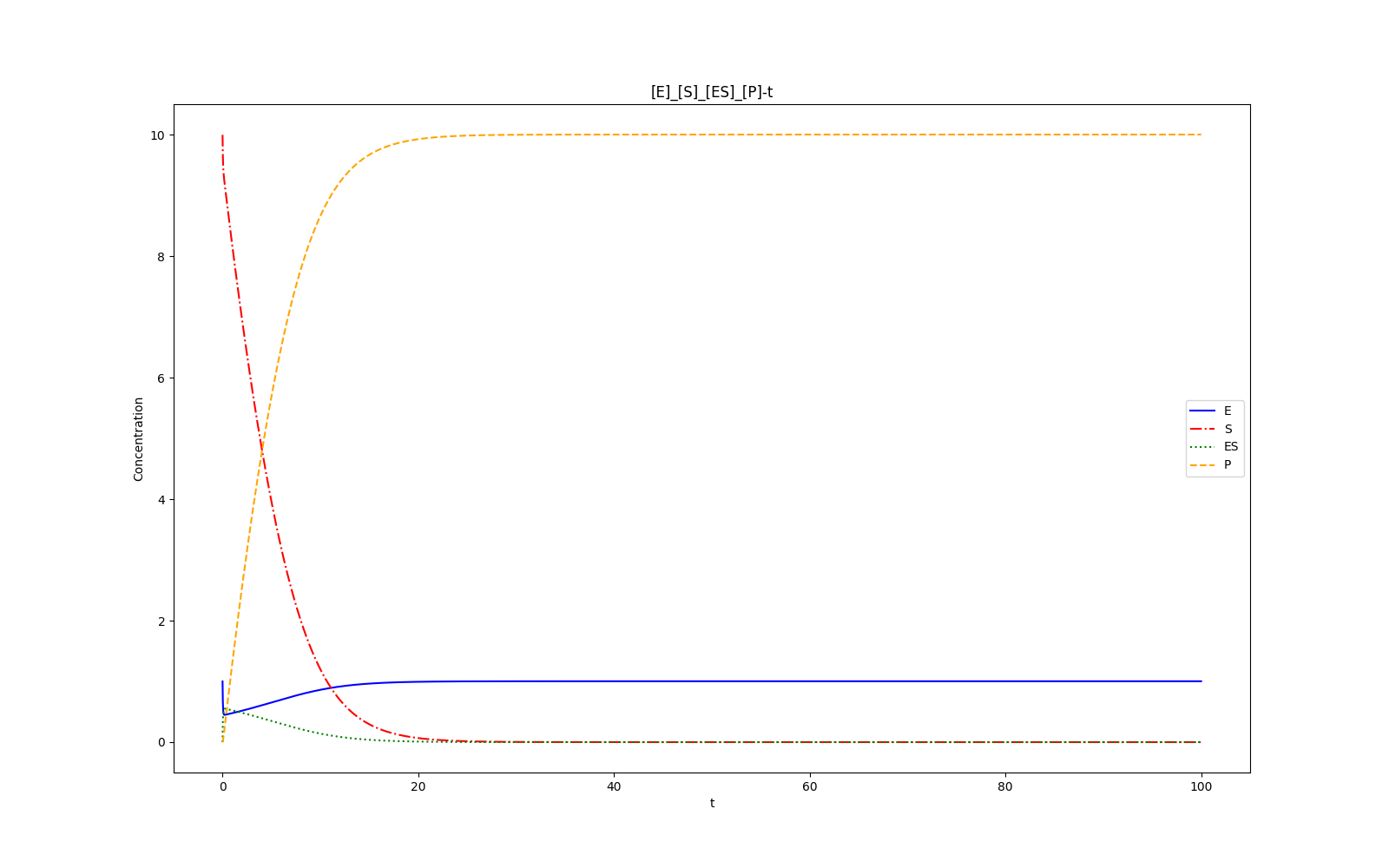
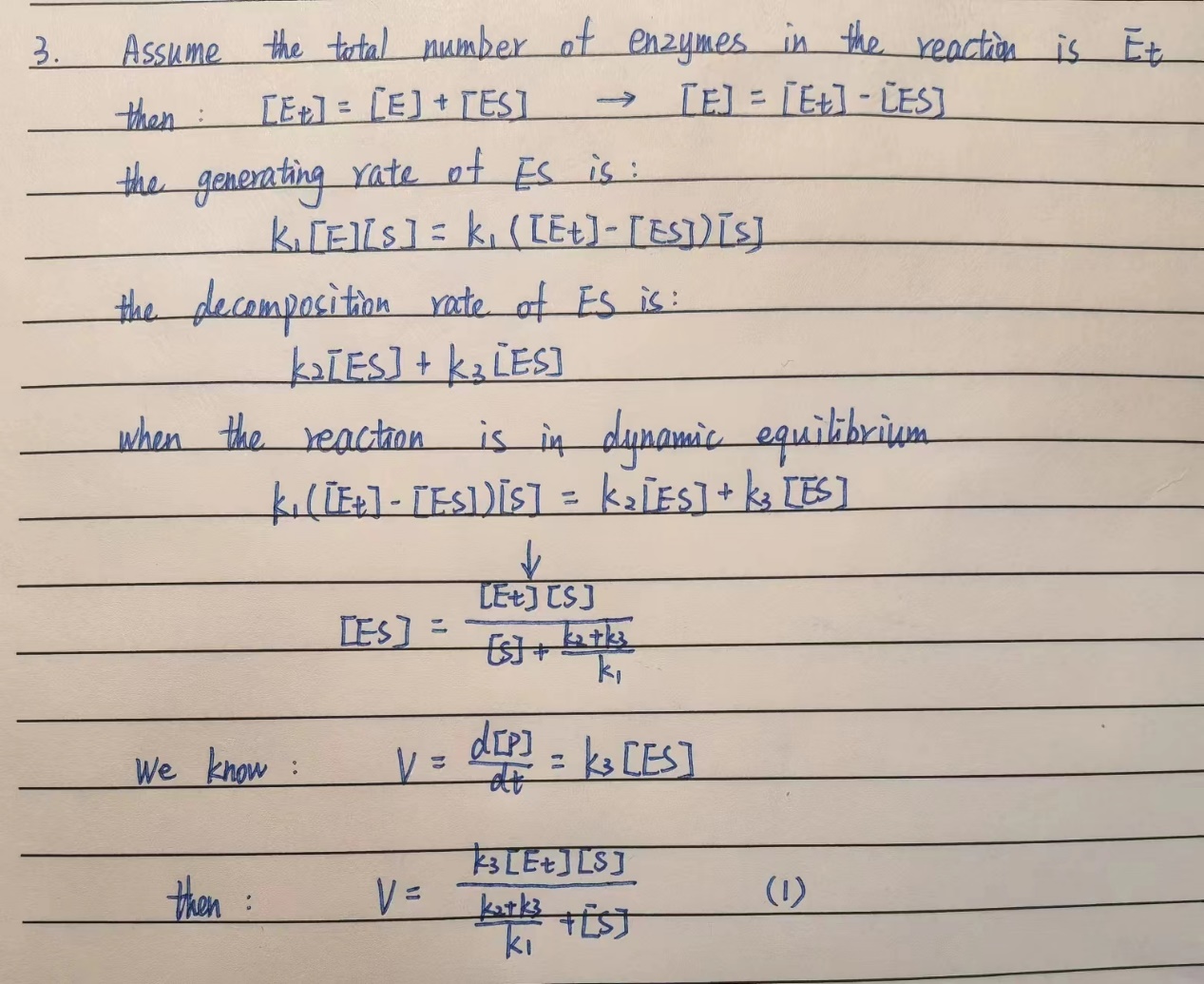
1.



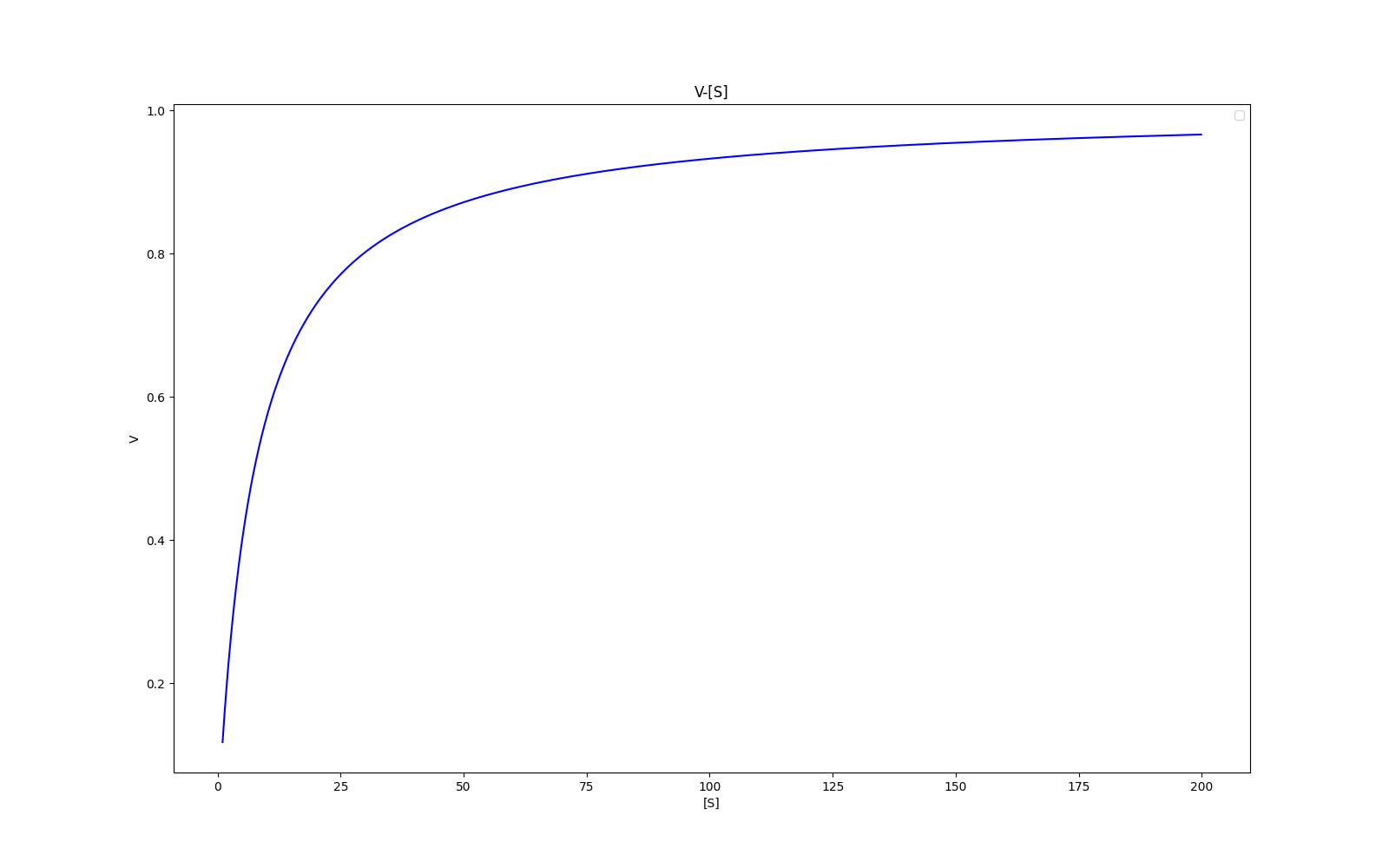
2.Picture 1 shows the concentration of E, S, ES and P at different time. See the attachment in the end for the code.

Picture 1

3.



According to formula(1), I plot the velocity V as a function of the concentration of the substrate S (see Picture 2). See the attachment in the end for the code.



Picture 2

In theory, when all the enzymes (E) formulate intermediates (ES), the velocity V saturates to a maximum value, so Vm=k3\*[Et]. According to picture 2, when the concentrations of S are small, the velocity V increases approximately linearly, and as the increasing of concentrations of S, the velocity V saturates to Vm, which is k3\*[Et] in formula(1).

Attachment

# Question2\_3  
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
  
k1 = 100 / 60  
k2 = 600 / 60  
k3 = 150 / 60  
  
# Four equations  
# x1 represents [E]  
# x2 represents [S]  
# x3 represents [ES]  
# X4 represents [P]  
def funcX1t(x1, x2, x3, x4):  
 return k2 \* x3 + k3 \* x3 - k1 \* x1 \* x2  
  
def funcX2t(x1, x2, x3, x4):  
 return k2 \* x3 - k1 \* x1 \* x2  
  
def funcX3t(x1, x2, x3, x4):  
 return k1 \* x1 \* x2 - k2 \* x3 - k3 \* x3  
  
def funcX4t(x1, x2, x3, x4):  
 return k3 \* x3  
  
# Solve the equations  
t\_ini = 0 # time min  
t\_end = 100.0 # time max  
t\_h = 1e-3 # step value  
  
t = np.linspace(t\_ini, t\_end, int((t\_end - t\_ini) / t\_h + 1))  
x1 = t.copy()  
x2 = t.copy()  
x3 = t.copy()  
x4 = t.copy()  
x1[0] = 1.0  
x2[0] = 10.0  
x3[0] = 0.0  
x4[0] = 0.0  
  
# RK4  
for i in range(t.shape[0] - 1):  
 # print(i)  
 h\_i = t[i + 1] - t[i]  
  
 k1\_x1 = funcX1t(x1[i], x2[i], x3[i], x4[i])  
 k1\_x2 = funcX2t(x1[i], x2[i], x3[i], x4[i])  
 k1\_x3 = funcX3t(x1[i], x2[i], x3[i], x4[i])  
 k1\_x4 = funcX4t(x1[i], x2[i], x3[i], x4[i])  
  
 k2\_x1 = funcX1t(x1[i] + h\_i / 2 \* k1\_x1, x2[i], x3[i], x4[i])  
 k2\_x2 = funcX2t(x1[i], x2[i] + h\_i / 2 \* k1\_x2, x3[i], x4[i])  
 k2\_x3 = funcX3t(x1[i], x2[i], x3[i] + h\_i / 2 \* k1\_x3, x4[i])  
 k2\_x4 = funcX4t(x1[i], x2[i], x3[i], x4[i] + h\_i / 2 \* k1\_x4)  
  
 k3\_x1 = funcX1t(x1[i] + h\_i / 2 \* k2\_x1, x2[i], x3[i], x4[i])  
 k3\_x2 = funcX2t(x1[i], x2[i] + h\_i / 2 \* k2\_x2, x3[i], x4[i])  
 k3\_x3 = funcX3t(x1[i], x2[i], x3[i] + h\_i / 2 \* k2\_x3, x4[i])  
 k3\_x4 = funcX4t(x1[i], x2[i], x3[i], x4[i] + h\_i / 2 \* k2\_x4)  
  
 k4\_x1 = funcX1t(x1[i] + h\_i \* k3\_x1, x2[i], x3[i], x4[i])  
 k4\_x2 = funcX2t(x1[i], x2[i] + h\_i \* k3\_x2, x3[i], x4[i])  
 k4\_x3 = funcX3t(x1[i], x2[i], x3[i] + h\_i \* k3\_x3, x4[i])  
 k4\_x4 = funcX4t(x1[i], x2[i], x3[i], x4[i] + h\_i \* k3\_x4)  
  
 x1[i + 1] = x1[i] + h\_i / 6.0 \* (k1\_x1 + 2.0 \* k2\_x1 + 2.0 \* k3\_x1 + k4\_x1)  
 x2[i + 1] = x2[i] + h\_i / 6.0 \* (k1\_x2 + 2.0 \* k2\_x2 + 2.0 \* k3\_x2 + k4\_x2)  
 x3[i + 1] = x3[i] + h\_i / 6.0 \* (k1\_x3 + 2.0 \* k2\_x3 + 2.0 \* k3\_x3 + k4\_x3)  
 x4[i + 1] = x4[i] + h\_i / 6.0 \* (k1\_x4 + 2.0 \* k2\_x4 + 2.0 \* k3\_x4 + k4\_x4)  
  
filepath = ""  
# Draw  
plt.figure(figsize=(16, 10))  
  
plt.plot(t, x1, 'b', linestyle='-', label='E')  
plt.plot(t, x2, 'r', linestyle='-.', label='S')  
plt.plot(t, x3, 'green', linestyle=':', label='ES')  
plt.plot(t, x4, 'orange', linestyle='--', label='P')  
plt.legend()  
plt.xlabel('t')  
plt.ylabel('Concentration')  
plt.title('[E]\_[S]\_[ES]\_[P]-t')  
  
plt.savefig(filepath + "E\_S\_ES\_P.png")  
  
  
# Question2\_3  
vmax = np.mean(x1+x3)  
k = (k2 + k3) / k1  
  
def funcX4X2(x2):  
 return vmax \* x2 / (k + x2)  
  
x22=np.arange(1,200,0.1)  
dp=[funcX4X2(x22[tt]) for tt in np.arange(len(x22))]  
dp\_E\_S\_ES=pd.DataFrame({'S':x22,'dp':dp})  
  
plt.figure(figsize=(16,10))  
plt.plot(dp\_E\_S\_ES['S'], dp\_E\_S\_ES['dp'], 'b')  
  
plt.legend()  
plt.xlabel('[S]')  
plt.ylabel('V')  
plt.title('V-[S]')  
  
  
plt.savefig(filepath+"V\_S.png")  
  
  
dp\_E\_S\_ES=dp\_E\_S\_ES.sort\_values('dp',ascending=False)  
  
print(f"max\_dp={dp\_E\_S\_ES['dp'].max()}")  
print('Vmax=',vmax)