2.1

Four equations for the rate of changes of the four species, E, S, ES, and P:

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

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

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

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

[E], [S], [ES], [P] are the concentration of E, S, ES, P

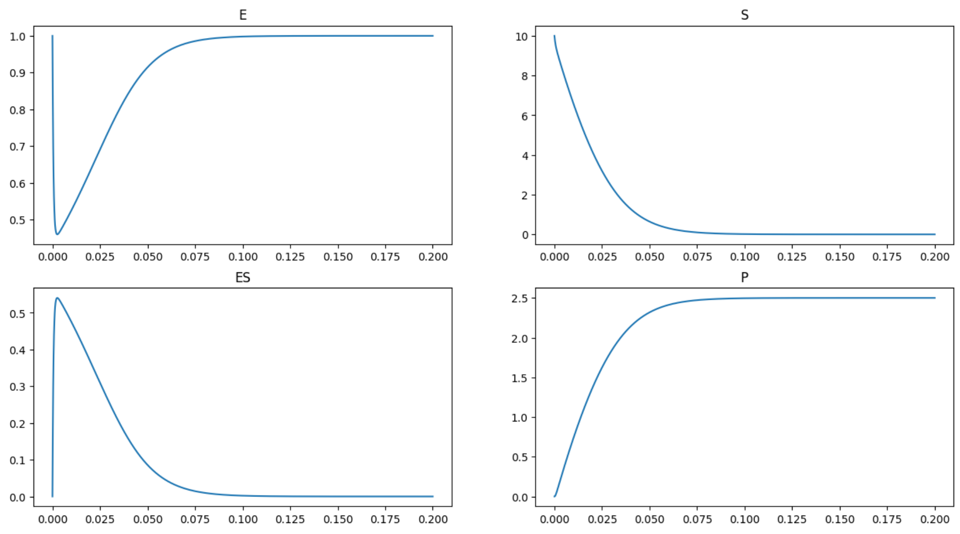
2.2

fourth-order Runge-Kutta method : h = 0.0001, n = 2000

Code:

```
import matplotlib.pyplot as plt

def f_E(E: float, S: float, ES: float, P: float):
    return -100 * E * S + (600 + 150) * ES


def f_S(E: float, S: float, ES: float, P: float):
    return -100 * E * S + 150 * ES


def f_ES(E: float, S: float, ES: float, P: float):
    return 100 * E * S - (600 + 150) * ES


def f_P(E: float, S: float, ES: float, P: float):
    return 150 * ES


def Runge_Kutta(type: int, E: float, S: float, ES: float, P: float, h: float,
func):
    k1 = h * func(E, S, ES, P)
    if type == 0:
        k2 = h * func(E + k1 * h / 2, S, ES, P)
        k3 = h * func(E + k2 * h / 2, S, ES, P)
        k4 = h * func(E + k3 * h / 2, S, ES, P)
        return E + (k1 + 2 * k2 + 2 * k3 + k4) / 6
    elif type == 1:
        k2 = h * func(E, S + k1 * h / 2, ES, P)
        k3 = h * func(E, S + k2 * h / 2, ES, P)
        k4 = h * func(E, S + k3 * h / 2, ES, P)
        return S + (k1 + 2 * k2 + 2 * k3 + k4) / 6
    elif type == 2:
        k2 = h * func(E, S, ES + k1 * h / 2, P)
        k3 = h * func(E, S, ES + k2 * h / 2, P)
        k4 = h * func(E, S, ES + k3 * h / 2, P)
        return ES + (k1 + 2 * k2 + 2 * k3 + k4) / 6
    elif type == 3:
        k2 = h * func(E, S, ES, P + k1 * h / 2)
        k3 = h * func(E, S, ES, P + k2 * h / 2)
        k4 = h * func(E, S, ES, P + k3 * h / 2)
        return P + (k1 + 2 * k2 + 2 * k3 + k4) /6
    return 0


def main():

    #2.2
    x = [0]
    E = [1.0]
    S = [10.0]
    ES = [0.0]
    P = [0.0]
    h = 0.0001

    for i in range(2000):
        e = E[i]
        s = S[i]
        es = ES[i]
        p = P[i]
        E.append(Runge_Kutta(0, e, s, es, p, h, f_E))
        S.append(Runge_Kutta(1, e, s, es, p, h, f_S))
        ES.append(Runge_Kutta(2, e, s, es, p, h, f_ES))
        P.append(Runge_Kutta(3, e, s, es, p, h, f_P))
        x.append(x[i]+h)

    plt.figure(figsize=(20, 10), dpi=100)

    plt.subplot(221)
    plt.title('E')
    plt.plot(x, E)

    plt.subplot(222)
    plt.title('S')
    plt.plot(x, S)

    plt.subplot(223)
    plt.title('ES')
    plt.plot(x, ES)

    plt.subplot(224)
    plt.title('P')
    plt.plot(x, P)

    plt.show()
    #2.3
    V = [0]
    v_max = 0
    for i in range(1999):
        v = P[i+1] - P[i]
        if v > v_max:
            v_max = v
        V.append(v)
    V.append(0)

    plt.figure(figsize=(20, 10), dpi=100)
    plt.ylabel('V')
    plt.xlabel('S')
    plt.plot(S, V)
    plt.show()
    print(v_max)

if __name__ == '__main__':
    main()
```

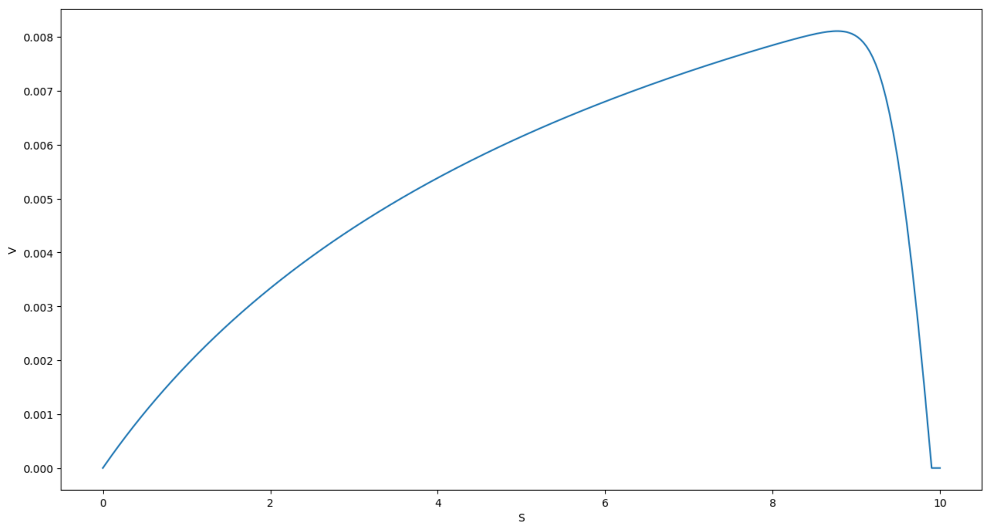
Result:



2.3

$$V = \frac{\Delta P}{\Delta t} = \frac{P[i+1] - P[i]}{h} \qquad (0 \le i \le 1999)$$

Result:



From the plot, we could find the value of Vm is around 0.008.