Behiye Erdemir
240206013

Elevator for 3 Storey Apartment

1. Problem Description

In the scope of the project, an elevator is modelled by considering its functions. Firstly, When the elevator reaches the called floor, the door opens and the door closes after the sensor on it detects there is not an object. For a 3-storey apartment, the button is pressed for the floor wanted to go to and the elevator first accelerates and then moves towards its target at a constant speed. Then, as the elevator approaches the target floor, it slows down and stops. In addition, when the stop button on the elevator is pressed, it causes the elevator to stop in its current position.

In addition, in terms of modeling, the floor selection was made not as the 1st, 2nd, 3rd floor, but as the floor difference between the desired floor and the current floor. The time spent for 1 floor relocation is set to 20 seconds. Besides, the maximum speed of the Elevator is accepted as 20 meters / second and the 1 floor distance is 3 meters.

2. System Design

The model of the system is given in Figure 1. The system starts with *X* signal. It is taken as 1. For the door of the elevator, C1-FFL is used to provide sign-sensitive delay on the bevavior of the door. If some objects obstruct the light from the sensor, the door opens immediately. Conversely, if there is nothing to obstruct the light, the door waits a brief before closing. Beyond that, brief pulses can not affect the system being in fluctuations. It is critical for protection[1]. *H* node represents an object. It can be 0 or 1. The output of the start signal ***and*** *H*, *D*, feeds the FFL. The dynamic evolution of *Y* and *A* is as in below.

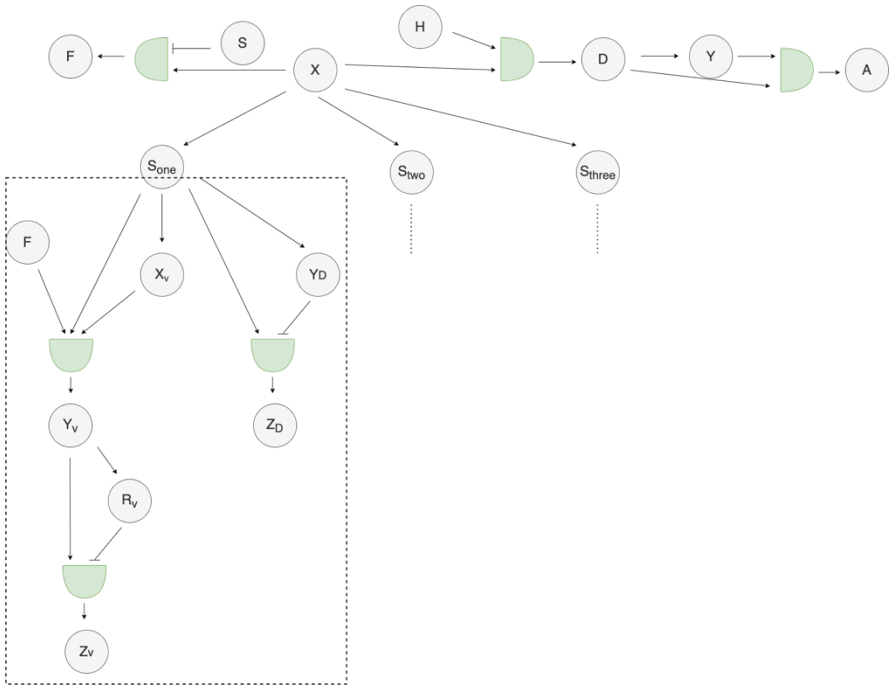


Figure 1 : Model of the elevator

$$\frac{d[Y]}{dt} = \beta_Y * \frac{[D]^{n_{D2Y}}(t)}{{\kappa_{D2Y}}^{n_{D2Y}} + [D]^{n_{D2Y}}(t)} - \alpha_Y([Y])(t), where\ \kappa_{D2Y} = 0.5, n_{D2Y} = 10, \beta_Y = \alpha_Y = 1$$

$$\frac{d[A]}{dt} = \beta_A * \frac{[Y]^{n_{Y2A}}(t)}{{\kappa_{Y2A}}^{n_{Y2A}} + [Y]^{n_{Y2A}}(t)} - \alpha_A([A])(t), where\ \kappa_{D2Y} = 0.5, n_{D2Y} = 10, \beta_Y = \alpha_Y = 1$$

The stop button, *S*, can also be 0 and 1. The X and the NOT S pass through the or gate. And resulting F is to determine whether the system works or not. The truth table of the F is seen in Table 1.

X	S	F
1	1	0
1	0	1
0	1	0
0	0	0

Table 1: Truth Table of the F

And when floor is selected, the S_{floor_change} will be activated. In the model of this system seen in Figure 1, S_{two} and S_{three} are not given, just S_{one} is given. Because they are the same as the dashed rectangular. So, S_{floor_change} represents all of them. The acceleration status with the activation of the elevator is denoted by Yv. Next to C1-FFL, *F* also passes through the and gate. Thus, if the stop button is pressed, the speed will also be 0.

The dynamic evolution of S_{change_floor}, X_v and Y_v is given in below.

$$\frac{d[S_{floor}]}{dt} = \beta_{S_{floor}} * \frac{[X]^{n_{X2S_{floor}}}(t)}{{\kappa_{X2S_{floor}}}^{n_{X2S_{floor}}} + [X]^{n_{X2S_{floor}}}(t)} - \alpha_{S_{floor}}([S_{floor}])(t),$$

$$where\ \kappa_{X2S_{floor}} = 0.5, n_{X2S_{floor}} = 10, \beta_{S_{floor}} = \alpha_{S_{floor}} = 1$$

$$\frac{d[X_v]}{dt} = \beta_{X_v} * \frac{[S_{floor}]^{n_{S_{floor}2X_v}}(t)}{{\kappa_{S_{floor}2X_v}}^{n_{S_{floor}2X_v}} + [S_{floor}]^{n_{S_{floor}2X_v}}(t)} - \alpha_A([X_v])(t),$$

$$where\ \kappa_{S_{floor}2X_v} = 0.5, n_{S_{floor}2X_v} = 10, \beta_{X_v} = 20.0, \alpha_{X_v} = 1$$

$$\frac{d[Y_v]}{dt} = \beta_{Y_v} * \left(\frac{[X_v]^{n_{X_v2Y_v}}(t)}{{\kappa_{X_v2Y_v}}^{n_{X_v2Y_v}} + [X_v]^{n_{X_v2Y_v}}(t)}\right) * \left(\frac{[F]^{n_{F2Y_v}}(t)}{{\kappa_{F2Y_v}}^{n_{F2Y_v}} + [F]^{n_{F2Y_v}}(t)}\right) - \alpha_{Y_v}([Y_v])(t)) *,$$

$$where\ \kappa_{F2Y_v} = \kappa_{X_v2Y_v} = 0.5, n_{F2Y_v} = n_{X_v2Y_v} = 10, \beta_{Y_v} = 20.0, \alpha_{Y_v} = 1$$

In addition, the elevator speed should decrease when approaching the targeted floor. For this reason, a repression node has been added, which activates 5 seconds before reaching the targeted floor. So, Z_v shows the final speed. The dynamic evolution of R_v, X_v and Z_v is given in below.

$$\frac{d[R_v]}{dt} = \beta_{R_v} * \frac{[Y_v]^{n_{Y_v2R_v}}(t)}{{\kappa_{Y_v2R_v}}^{n_{Y_v2R_v}} + [Y_v]^{n_{Y_v2R_v}}(t)} - \alpha_{R_v}([R_v])(t),$$

$$where\ \kappa_{Y_v2R_v} = 0.5, n_{Y_v2R_v} = 10, \beta_{R_v} = 20.0, \alpha_{R_v} = 1, t_{end} - 5 < t < t_{end}$$

$$\frac{d[Z_v]}{dt} = \beta_{Z_v} * \left(\frac{[Y_v]^{n_{Y_v2Z_v}}(t)}{{\kappa_{Y_v2Z_v}}^{n_{Y_v2Z_v}} + [Y_v]^{n_{Y_v2Z_v}}(t)} \right) * \left(\frac{1}{1 + \left(\frac{[R_v]}{\kappa_{R_v2Y_v}} \right)^{n_{R_v2Y_v}}} \right) - \alpha_{Z_v}([Z_v])(t)) *,$$

$$where\ \kappa_{Y_v2Z_v} = \kappa_{R_v2Y_v} = 0.5, n_{Y_v2Z_v} = n_{R_v2Y_v} = 10, \beta_{R_v} = 20.0, \alpha_{R_v} = 1$$

Finally, the variation of the distance of the elevator to the target is tried to be shown. For this, I1-FFL is used. Z_D shows the mentioned information of distance. The dynamic evolution of Y_D and Z_D is given in below. The beta value of Z depends on the floor choice.

$$\frac{d[Y_D]}{dt} = \beta_{Y_D} * \frac{\left[S_{floor}\right]^{n_{S_{floor}2Y_D}}(t)}{{\kappa_{S_{floor}2Y_D}}^{n_{S_{floor}2Y_D}} + \left[S_{floor}\right]^{n_{S_{floor}2Y_D}}(t)} - \alpha_{Y_D}([Y_D])(t),$$

$$where\ \kappa_{S_{floor}2Y_D} = 0.5, n_{S_{floor}2Y_D} = 10, \beta_{Y_D} = 3 * \alpha_{Y_D}, \alpha_{Y_D} = 0.25$$

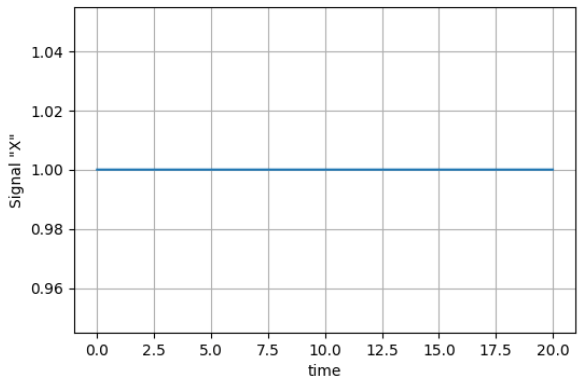
$$\frac{d[Z_D]}{dt} = \beta_{Z_D} * \frac{1}{1 + \left(\frac{[Y_D]}{\kappa_{Y_D2Z_D}} \right)^{n_{Y_D2Z_D}}} - \alpha_{Z_D}([Z_D])(t),$$

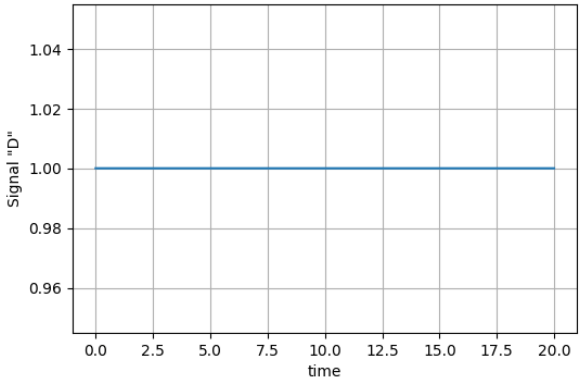
$$where\ \kappa_{Y_D2Z_D} = 0.5, n_{Y_D2Z_D} = 10, \beta_{Y_D} = (one\ of\ them\ 3.3, 4.4, 5.0) * 3 * \alpha_{Y_D}, \alpha_{Y_D} = 0.25$$

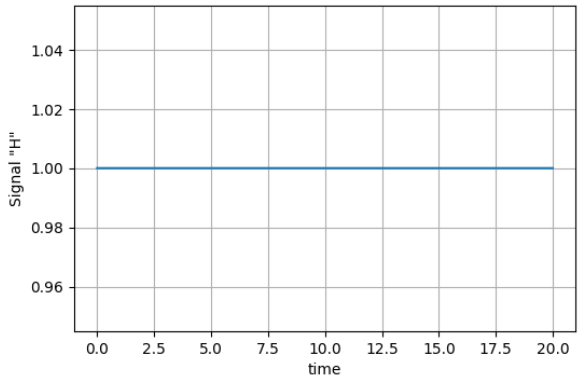
The situation that can not modeled in this model is that the distance remains constant when the stop button is pressed.

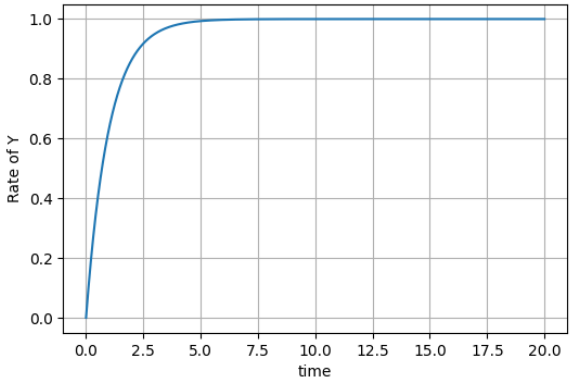
3. Simulations

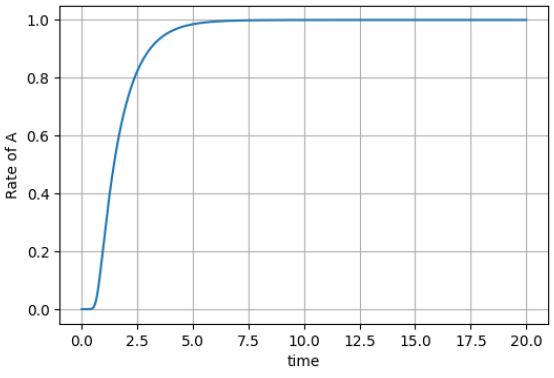
Simulations results are below for $X = 1$, $S = 0$ (stop button is not pressed), *floor_choice* = 1 and $H = 1$.

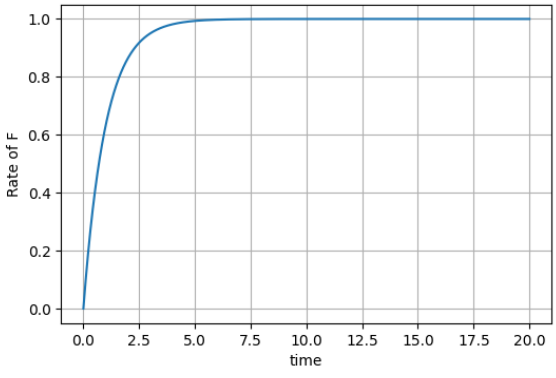


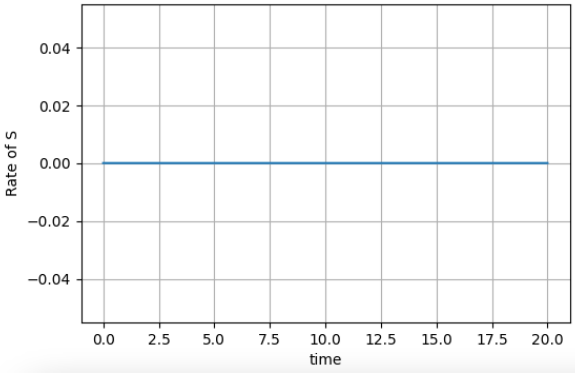


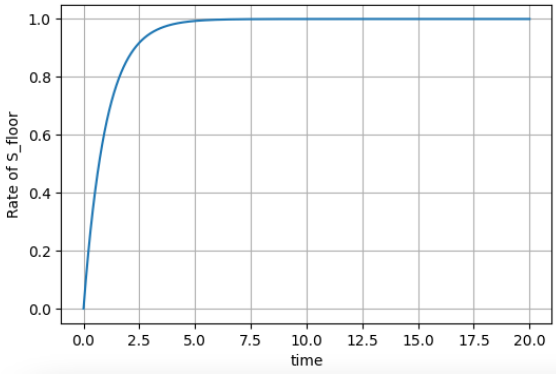


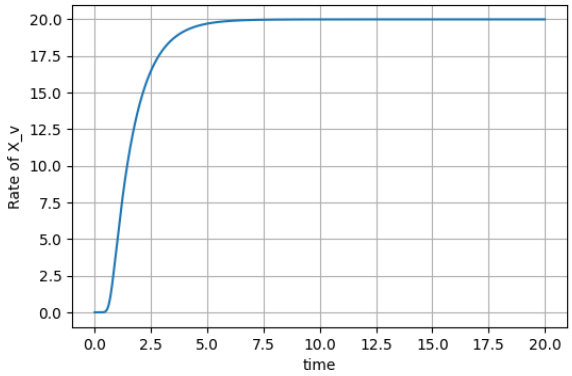


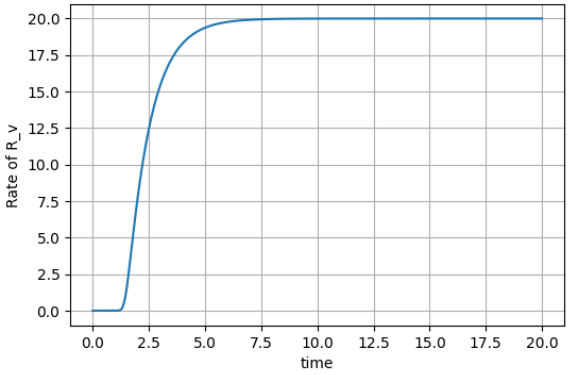


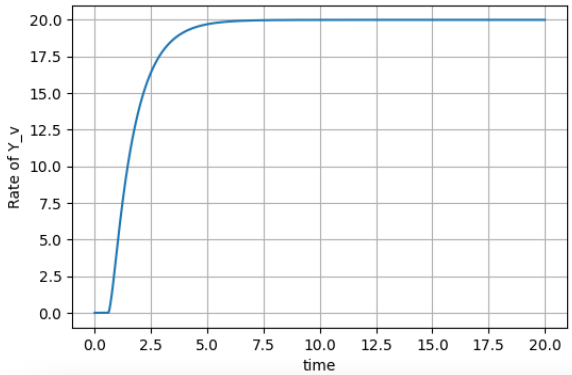


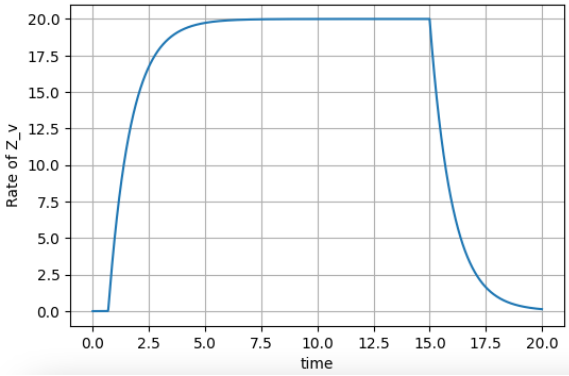


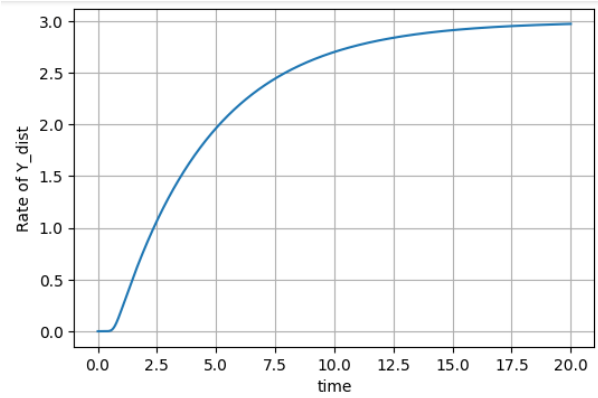


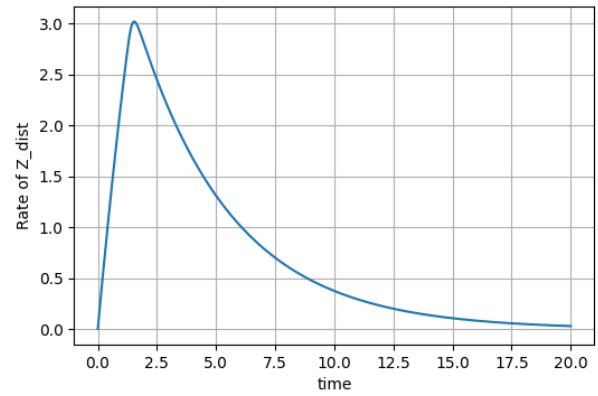

Rate of X_v
time


Rate of R_v
time

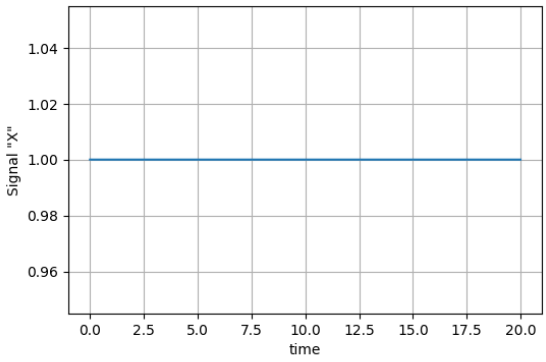

Rate of Y_v
time

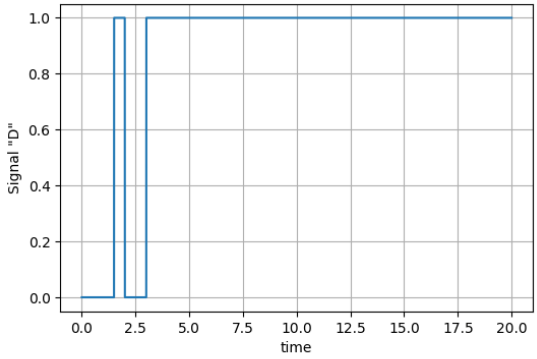

Rate of Z_v
time

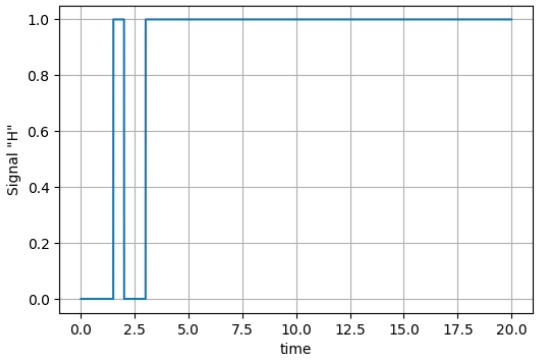

Rate of Y_dist
time

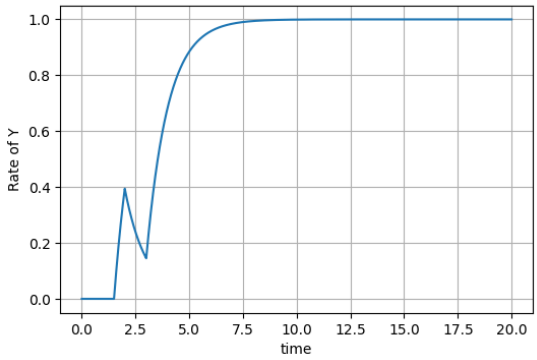

Rate of Z_dist
time

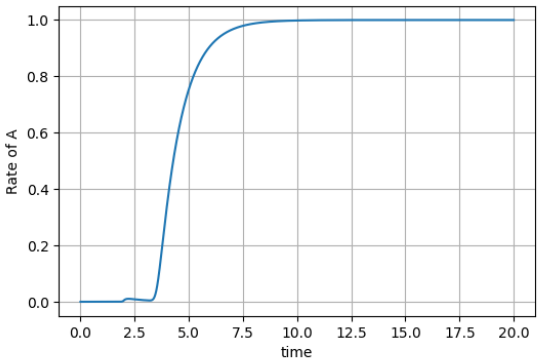
Simulations results are below for $X = 1$, $S = 0$ (stop button is not pressed), *floor_choice* = 1 and *H* in fluctuation (there is some object on door in brief time).

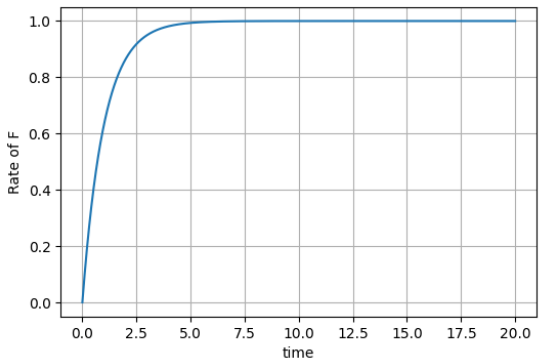


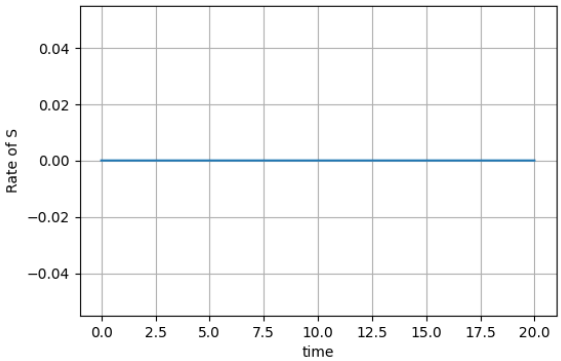


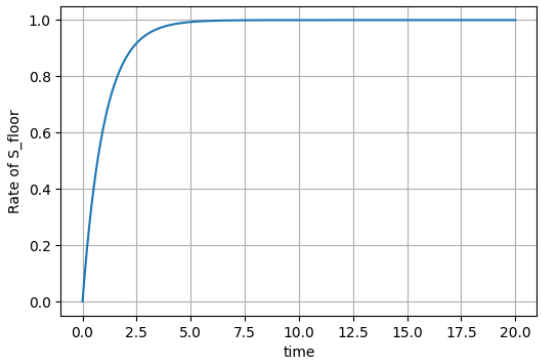


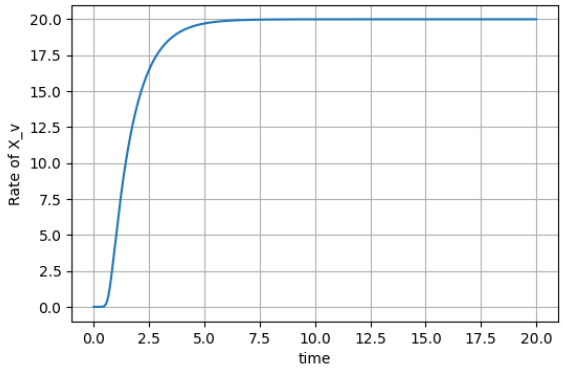


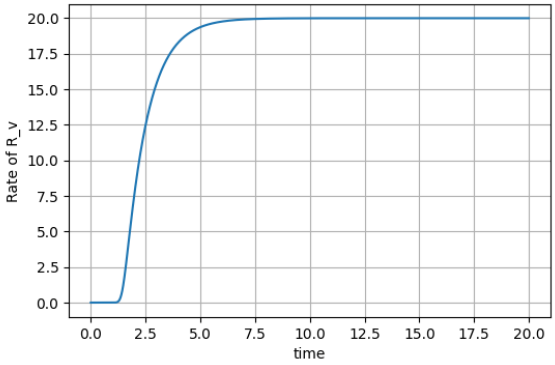


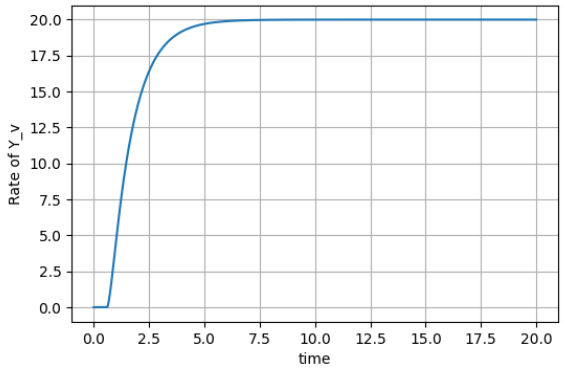


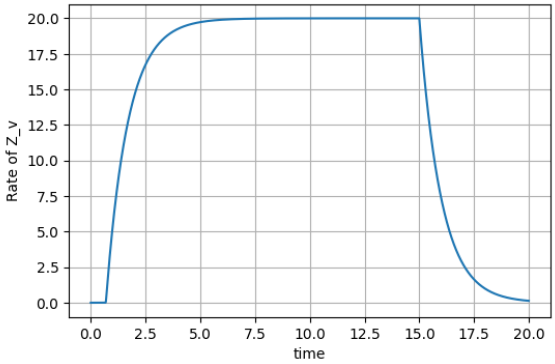


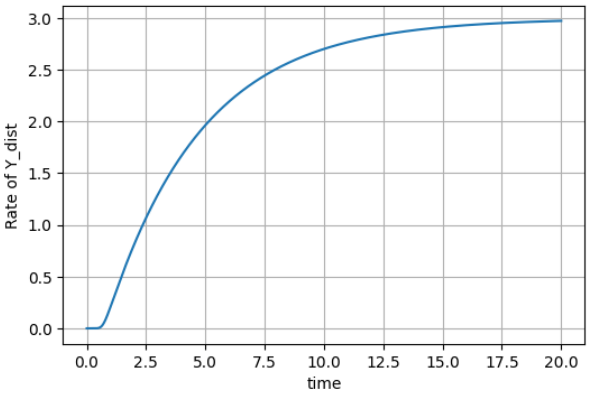


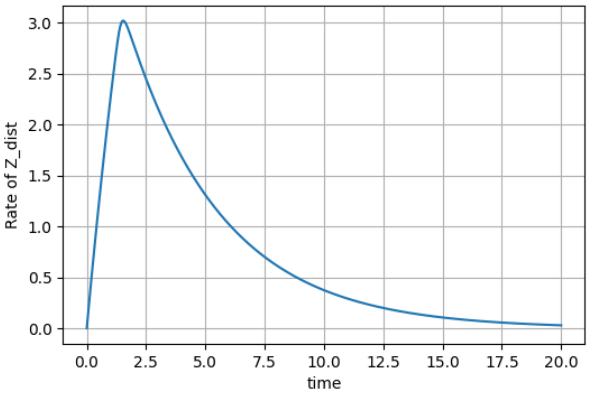

Rate of X_v
time


Rate of R_v
time

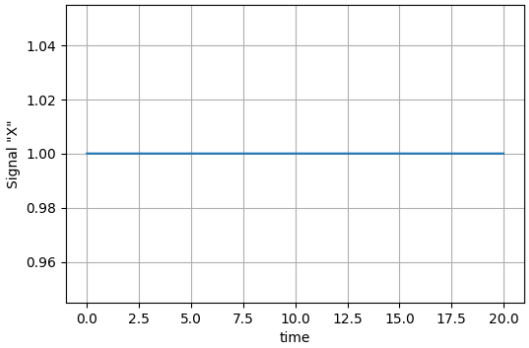

Rate of Y_v
time

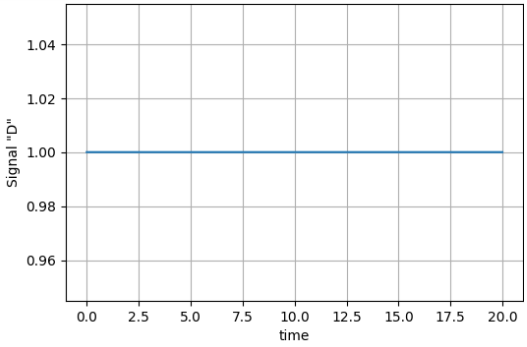

Rate of Z_v
time

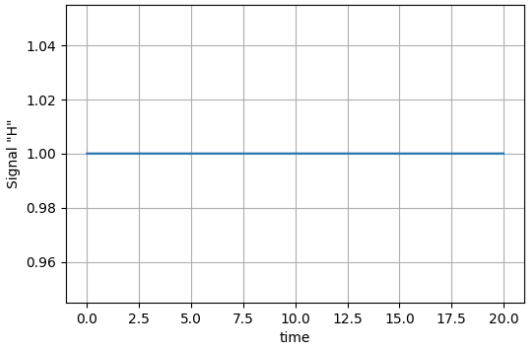

Rate of Y_dist
time

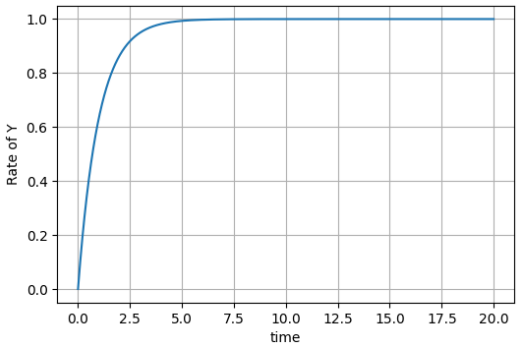

Rate of Z_dist
time

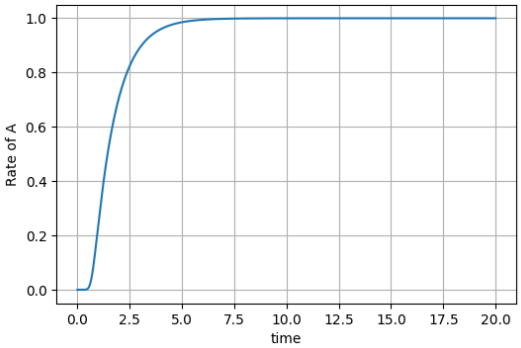
Simulations results are below for *X* = 1, *S* = 1 (stop button is pressed), *floor_choice* = 1 and H = 1.

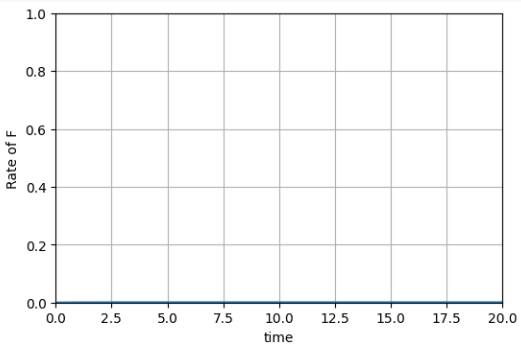


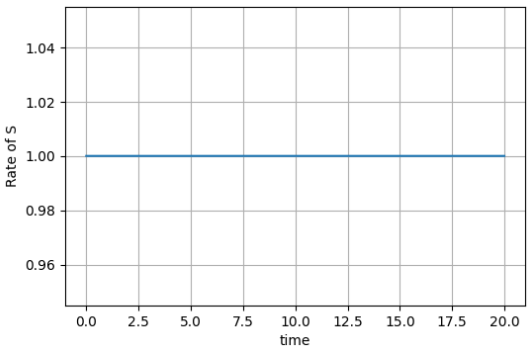


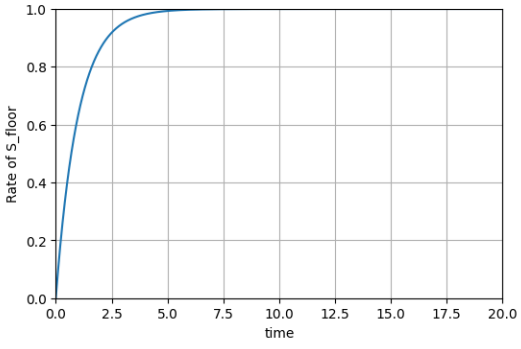


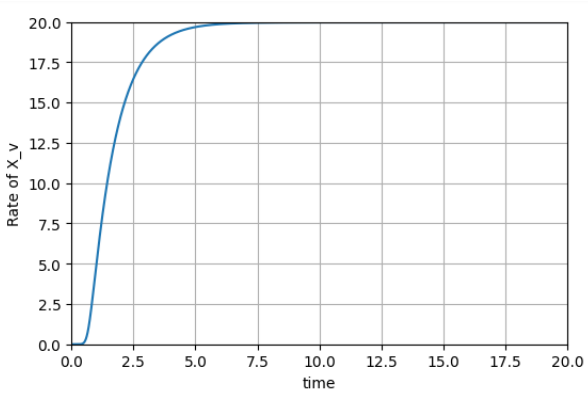


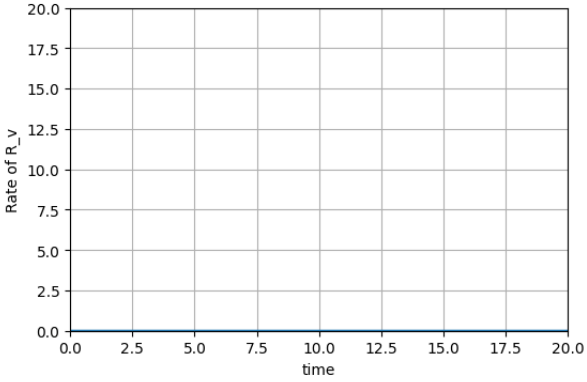


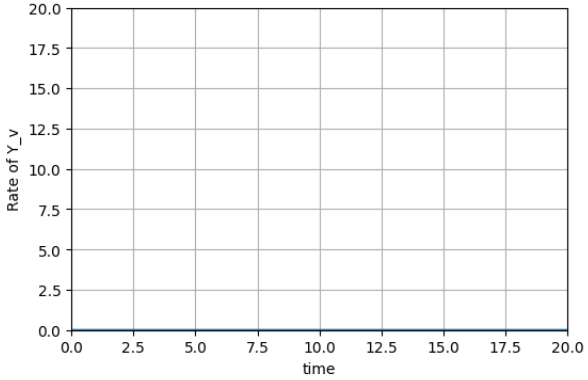


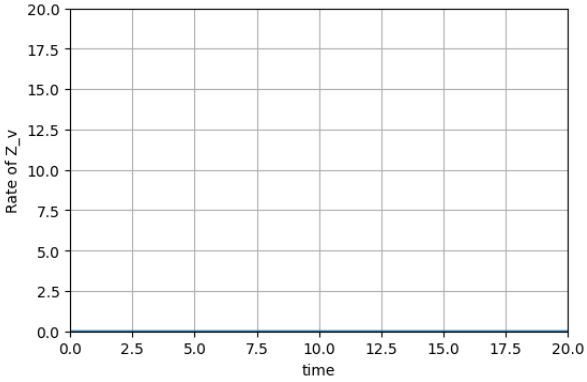


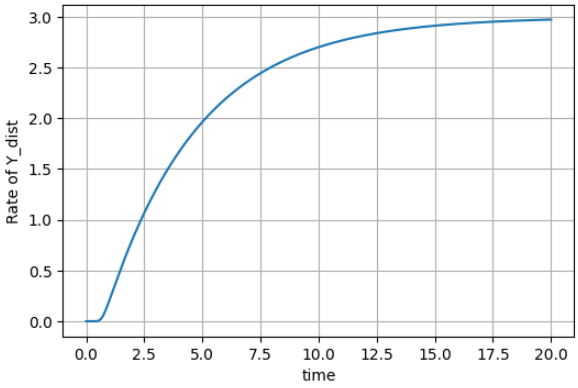


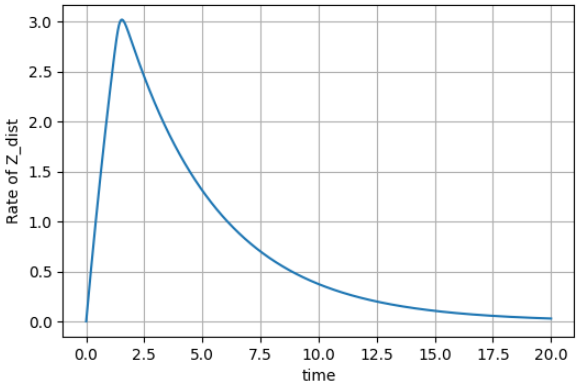

Rate of X_v
time


Rate of R_v
time

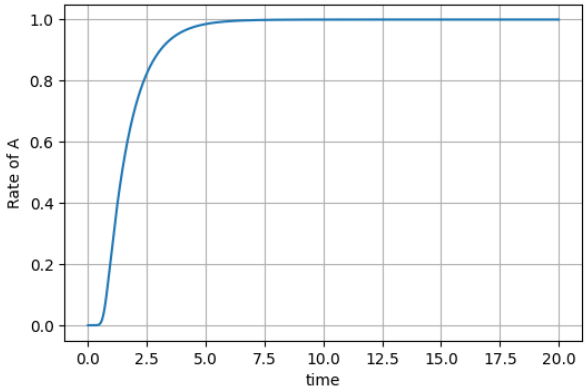

Rate of Y_v
time

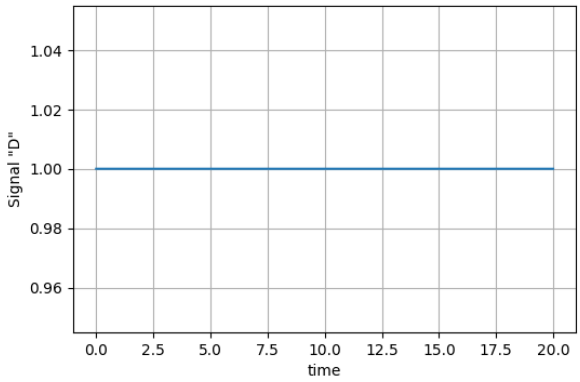

Rate of Z_v
time

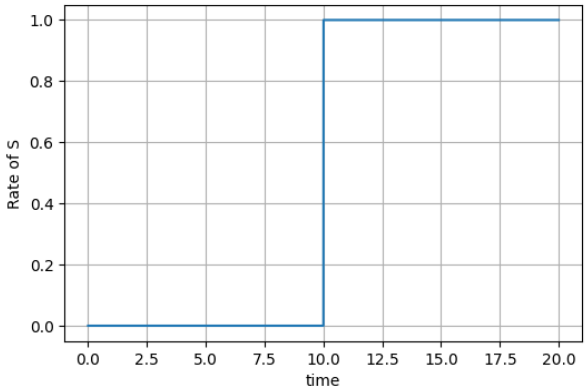

Rate of Y_dist
time

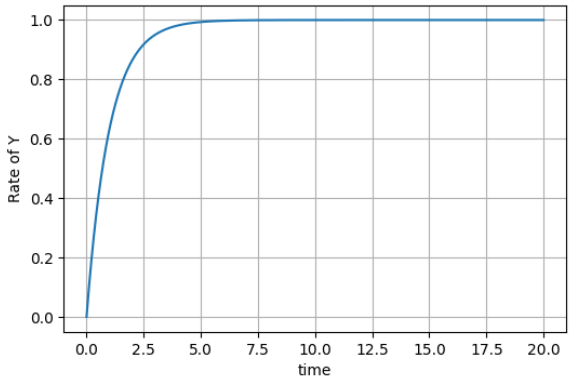

Rate of Z_dist
time

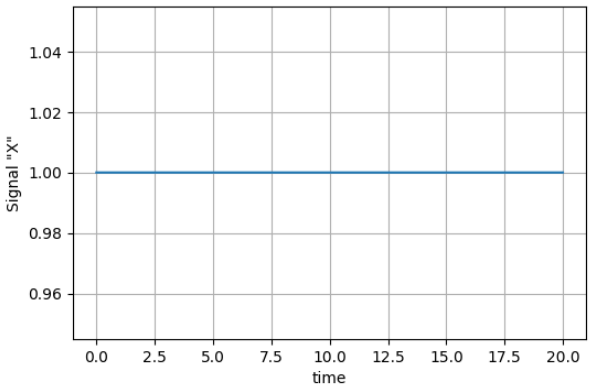
Simulations results are below for *X* = 1, *S* = 1 (stop button is pressed at t_end/2), *floor_choice* = 1 and H = 1.

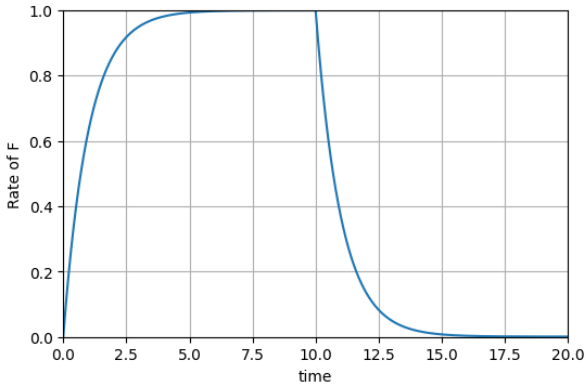


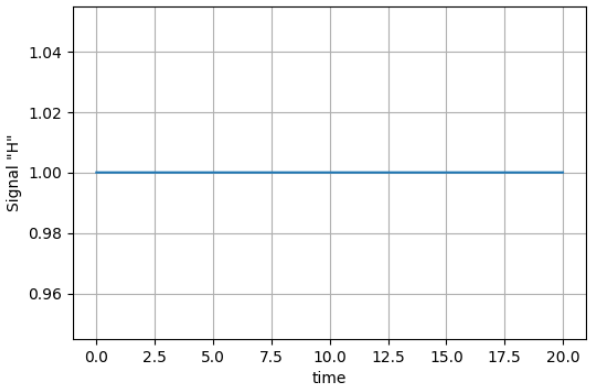


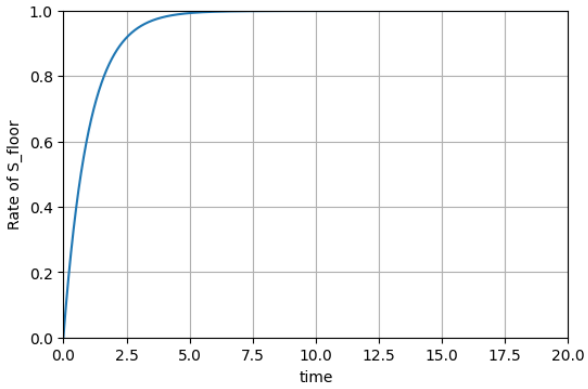


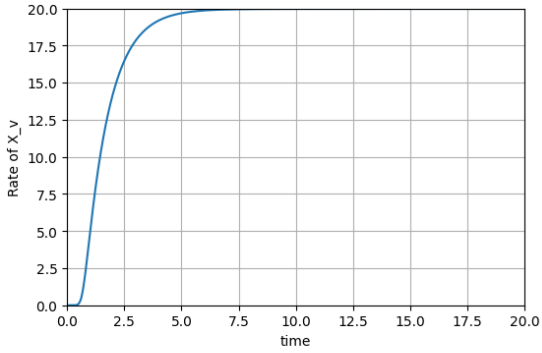


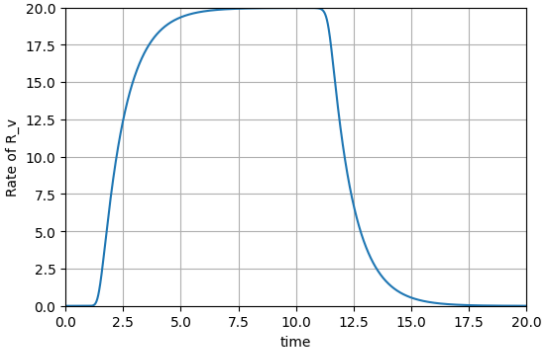


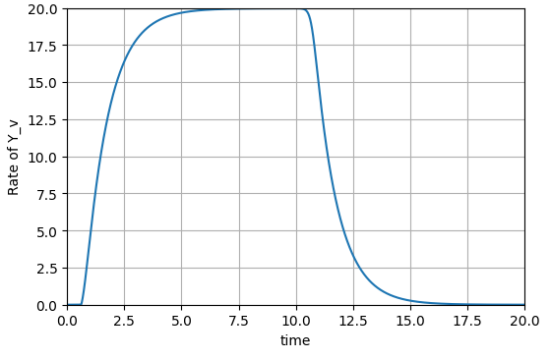


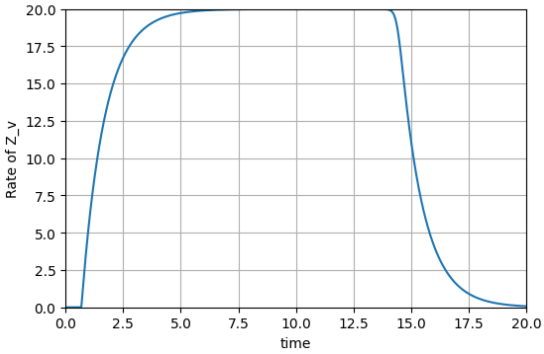


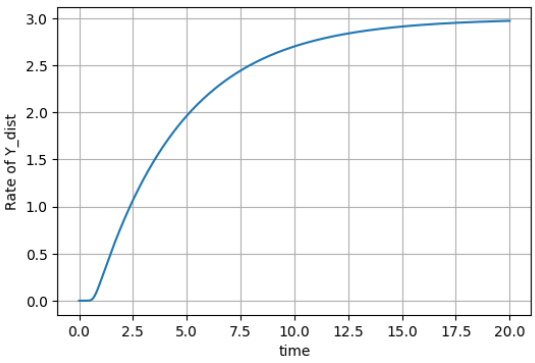


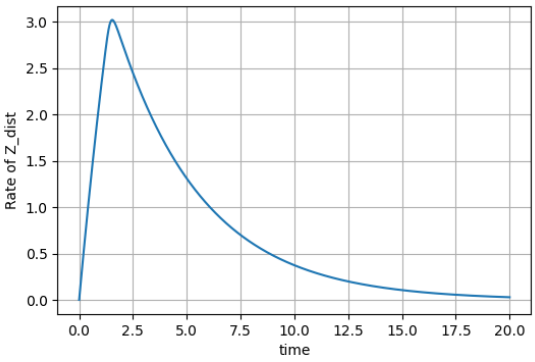

Rate of X_v
time


Rate of R_v
time

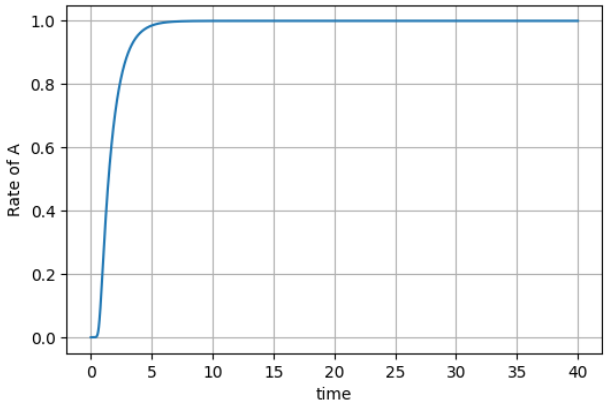

Rate of Y_v
time

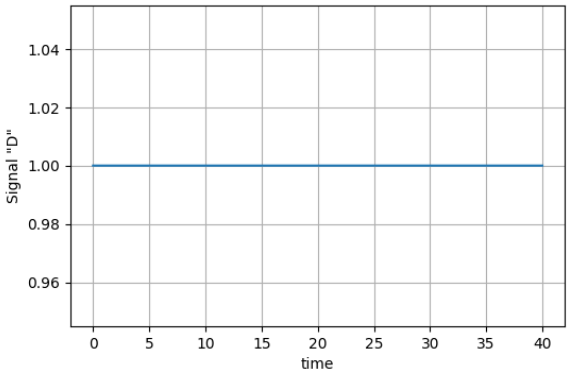

Rate of Z_v
time

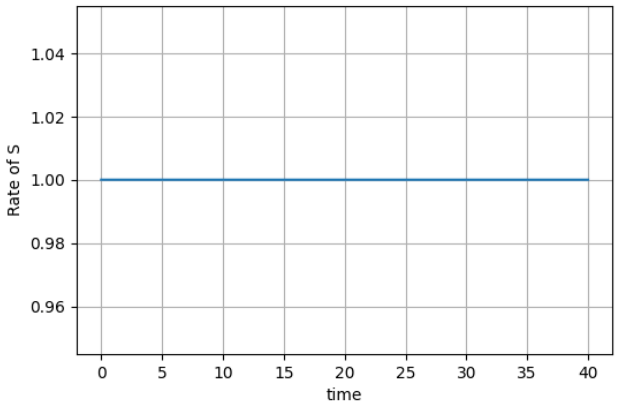

Rate of Y_dist
time

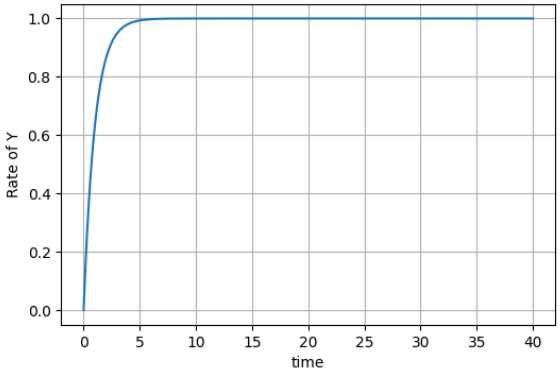

Rate of Z_dist
time

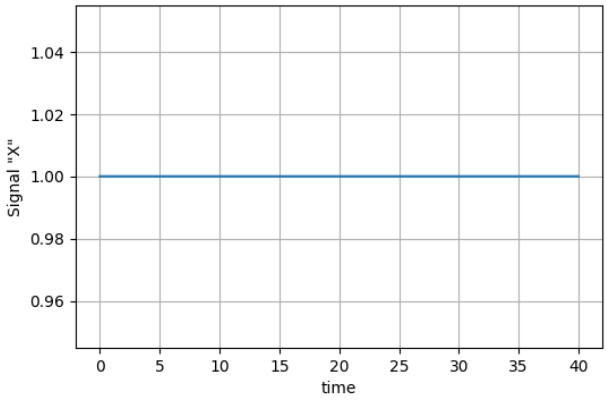
Simulations results are below for $X = 1$, $S = 0$ (stop button is not pressed), *floor_choice* = 2 and $H = 1$.

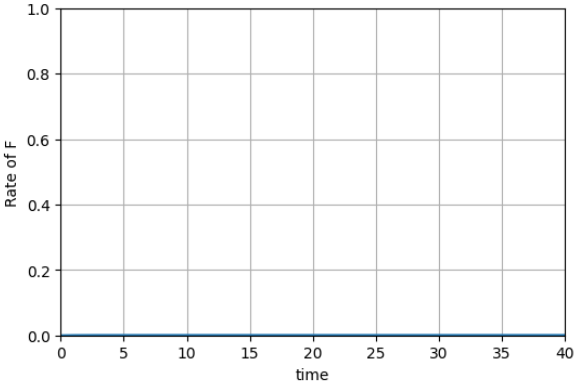


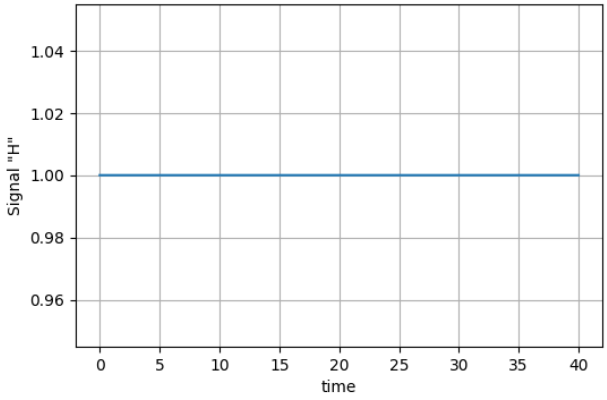


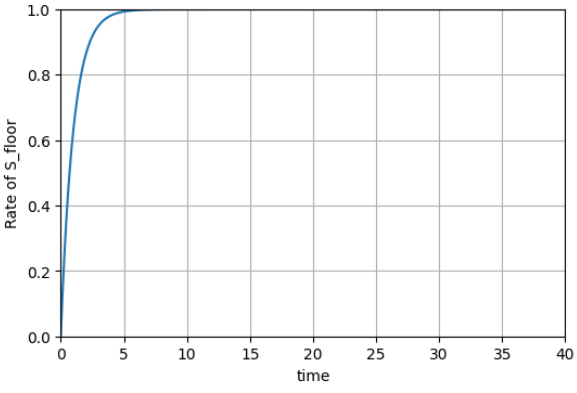


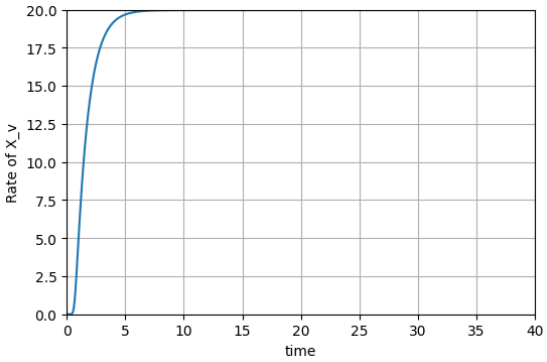


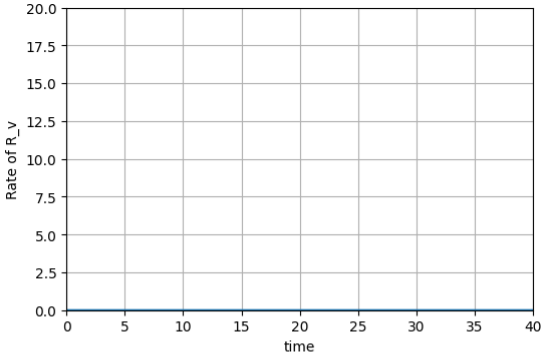


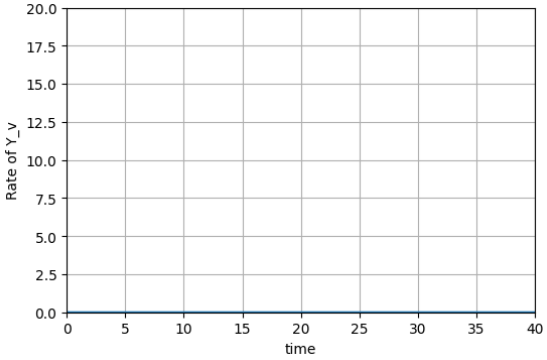


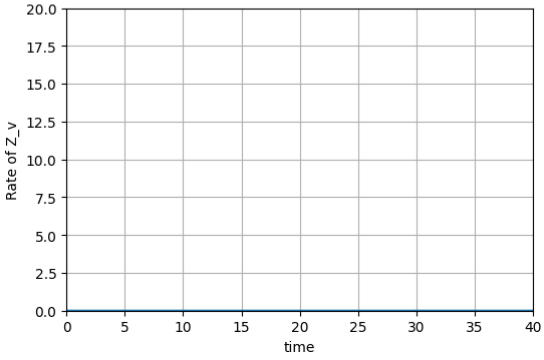


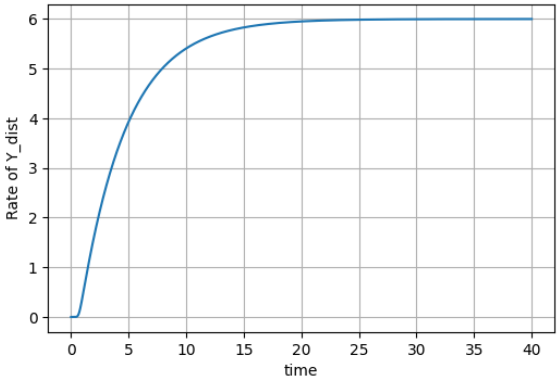


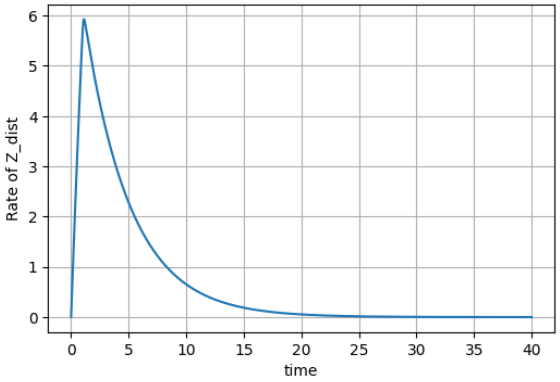

Rate of X_v
time


Rate of R_v
time

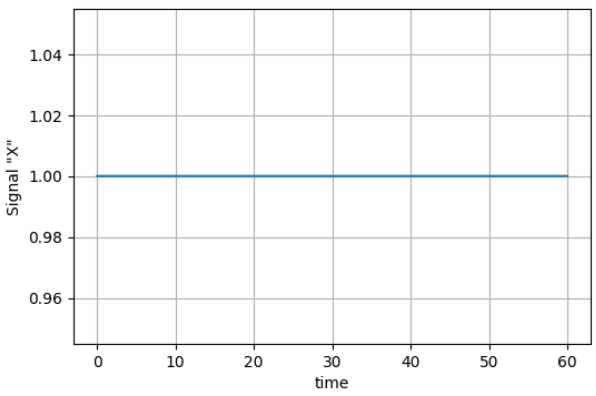

Rate of Y_v
time

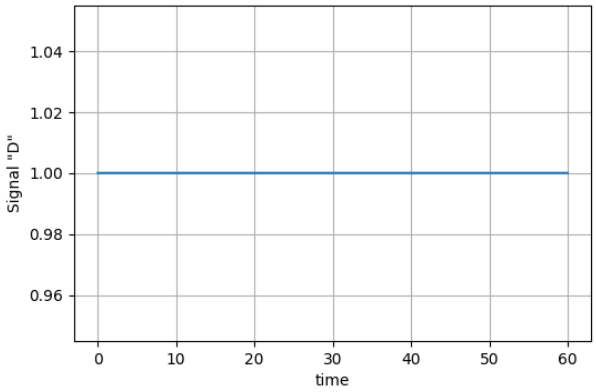

Rate of Z_v
time

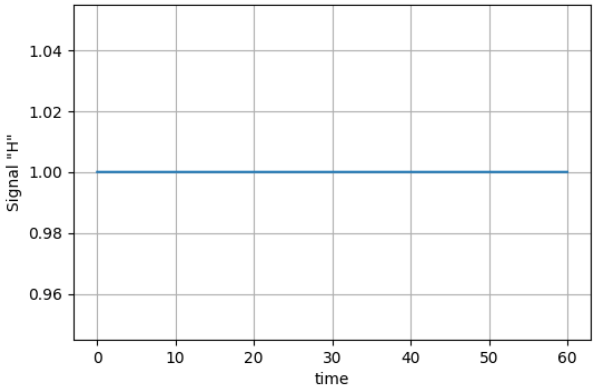

Rate of Y_dist
time

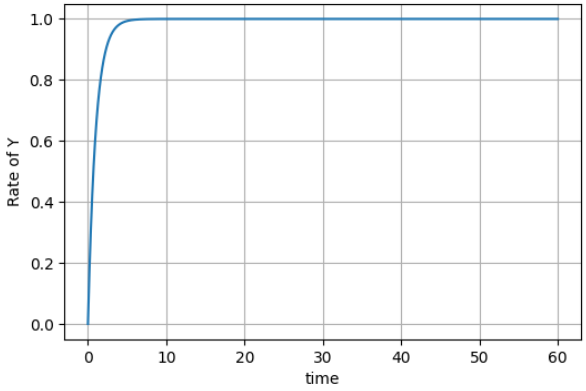

Rate of Z_dist
time

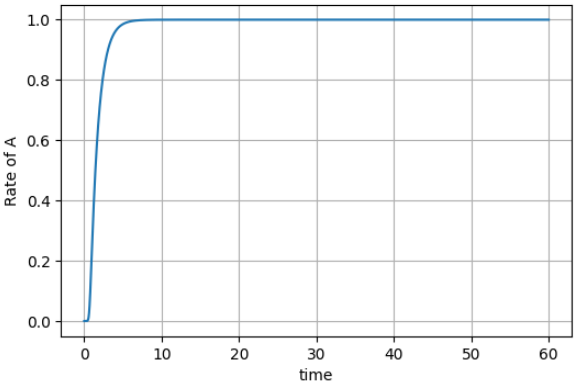
Simulations results are below for $X = 1$, $S = 0$ (stop button is not pressed), *floor_choice* = 3 and $H = 1$.

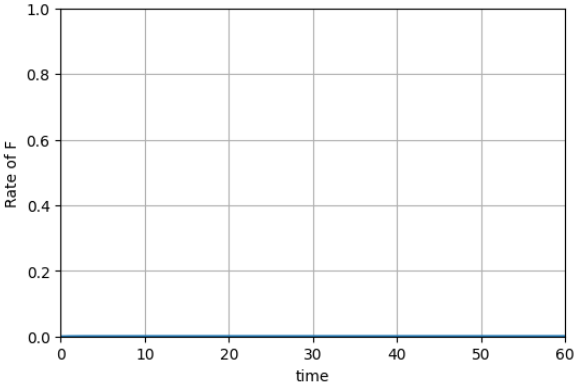


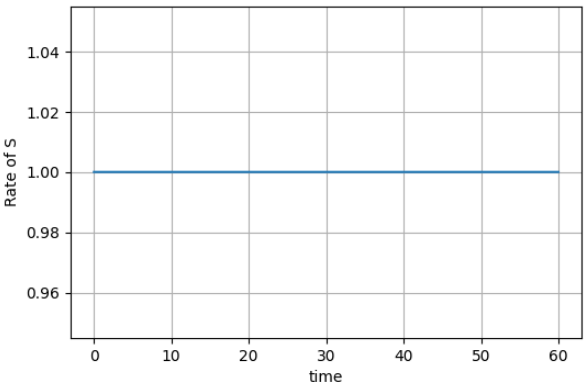


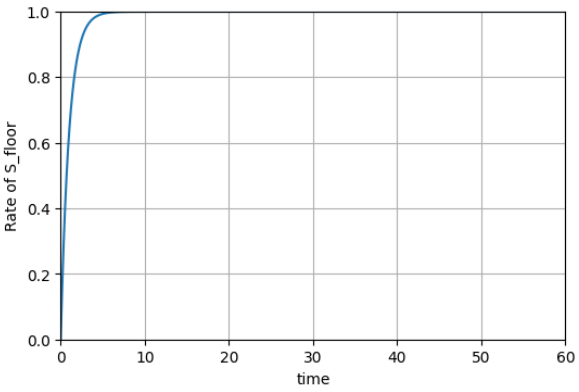


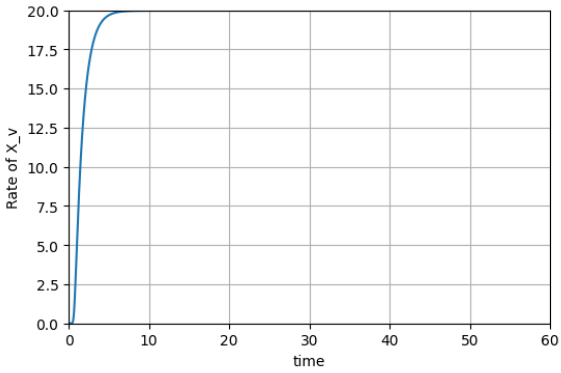


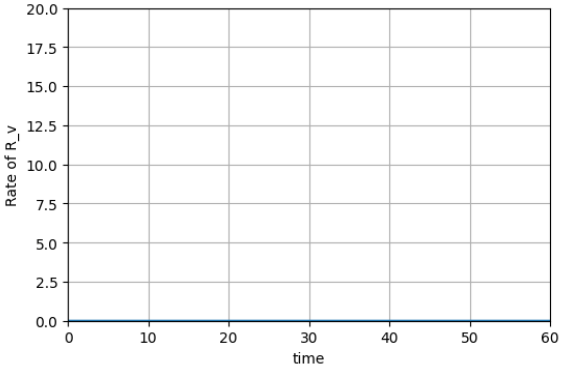


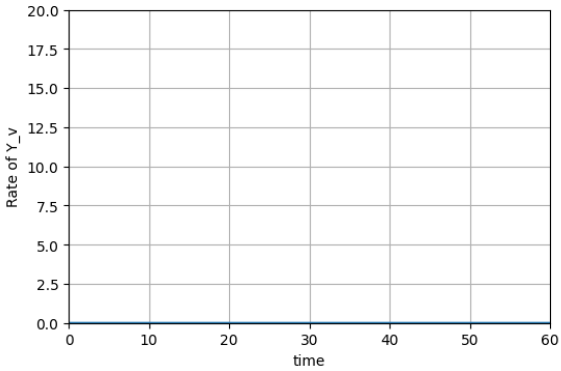


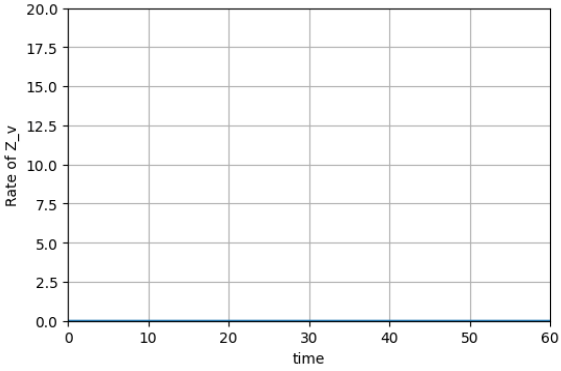


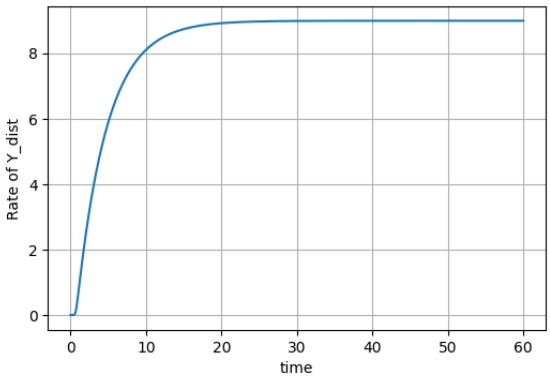


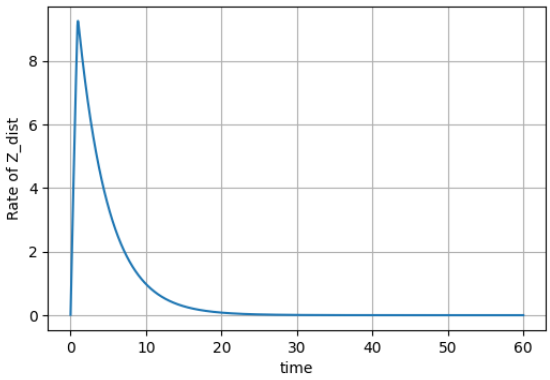

Rate of X_v
time


Rate of R_v
time


Rate of Y_v
time


Rate of Z_v
time


Rate of Y_dist
time


Rate of Z_dist
time

REFERENCES

[1] Alon, Uri. *An introduction to systems biology: design principles of biological circuits*. CRC press, 2019.

APPENDIX I

```
import numpy as np
import matplotlib.pyplot as plt

floor_distance=3.0
floor_choice=3
t_end=floor_choice*20
speed=20.0

dt = 0.001
t = np.arange(0,t_end,dt)

X = np.ndarray((len(t)))
H = np.ndarray((len(t)))
D = np.ndarray((len(t)))
Y = np.ndarray((len(t)))
A = np.ndarray((len(t)))
S = np.ndarray((len(t)))
F = np.ndarray((len(t)))
S_floor=np.ndarray((len(t)))
X_v = np.ndarray((len(t)))
Y_v = np.ndarray((len(t)))
R_v = np.ndarray((len(t)))
Z_v = np.ndarray((len(t)))
Y_dist = np.ndarray((len(t)))
Z_dist = np.ndarray((len(t)))

alpha_X=1.0
alpha_Y=1.0
alpha_A=1.0
alpha_S=1.0
alpha_F=1.0
alpha_S_floor=1.0
alpha_X_v=1.0
alpha_R_v=1.0
alpha_Y_v=1.0
alpha_Z_v=1.0
alpha_Y_dist=0.25
alpha_Z_dist=0.25

beta_X=1.0
beta_Y=1.0
beta_A=1.0
beta_S=1.0
beta_F=1.0
beta_S_floor=1.0
beta_X_v=speed
```

```
beta_Y_v=speed
beta_R_v=speed
beta_Z_v=speed
beta_Y_dist=floor_distance*alpha_Y_dist*floor_choice

if floor_choice==1:
  constant=3.4
elif floor_choice==2:
  constant=4.3
elif floor_choice==3:
  constant= 5.0
beta_Z_dist=constant*floor_distance*alpha_Y_dist*floor_choice

c_X = 0.0
c_Y = 0.0
c_A = 0.0
c_S = 0.0
c_F = 0.0
c_S_floor=0.0
c_X_v = 0.0
c_Y_v = 0.0
c_R_v = 0.0
c_Z_v = 0.0
c_Y_dist = 0.0
c_Y_dist = 0.0
c_Z_dist = 0.0
c_H=0.0

n_F=10
n_F2Y=10
n_YdZd=10
n_Sf_Yd=10
n_X2S_floor=10
n_Y2A=10
n_Rv2Yv=10
n_Rv2Zv=10
n_D2Y=10
n_S_floor2X=10
n_Yv2Zv=10
n_X2Y= 10

k_F=0.5
k_F2Y=0.5
k_YdZd=0.5
k_Sf_Yd=0.5
k_X2S_floor=0.5
k_Y2A=0.5
k_D2Y=0.5
k_S_floor2X=0.5
```

```
k_X2Y = 0.5
k_Rv2Yv=10
k_Rv2Zv=0.5
k_Yv2Zv=0.5

cind = 0
for ct in t:
  c_X = 1.0
  c_H = 1.0
  #c_H = 1.0 * ((ct >= 1.5) & (ct <=2)).astype(float) + 1*((ct >= 3) &
(ct <= t_end)).astype(float) #elevator asymmetric behavior
  c_D=c_X*c_H
  dYdt=beta_Y*(c_D**n_D2Y)/(k_D2Y**n_D2Y+c_D**n_D2Y)- alpha_Y * c_Y
  dAdt = beta_A*(c_Y**n_Y2A)/(k_Y2A**n_Y2A+c_Y**n_Y2A)- alpha_A * c_A
  c_S = 1.0
  #c_S=1.0*((ct >= t_end-10).astype(float)) #stop button
  dFdt= beta_F
*(1/((1+(c_S/k_F)**n_F)))*(c_X**n_F/(k_F**n_F+c_X**n_F))- alpha_F * c_F

dS_floordt=beta_S_floor*(c_X**n_X2S_floor)/(k_X2S_floor**n_X2S_floor+c_
X**n_X2S_floor)- alpha_S_floor * c_S_floor
  dX_vdt =
beta_X_v*(c_S_floor**n_S_floor2X)/(k_S_floor2X**n_S_floor2X+c_S_floor**
n_S_floor2X)- alpha_X_v * c_X_v
  dY_vdt =
beta_Y_v*(c_X_v**n_X2Y)/(k_X2Y**n_X2Y+c_X_v**n_X2Y)*(c_F**n_F2Y/(k_F2Y*
*n_F2Y+c_F**n_F2Y))- alpha_Y_v * c_Y_v
  dR_vdt=beta_R_v*(c_Y_v**n_Rv2Yv)/(k_Rv2Yv**n_Rv2Yv+c_Y_v**n_Rv2Yv)-
alpha_R_v * c_R_v
  dZ_vdt = beta_Z_v * (c_Y_v**n_Yv2Zv/(k_Yv2Zv**n_Yv2Zv +
c_Y_v**n_Yv2Zv))*(1/(1 + ((c_R_v/k_Rv2Zv)*((ct >= t_end-5) & (ct <=
t_end)).astype(float))**n_Rv2Zv)) - alpha_Z_v * c_Z_v
  dY_distdt = beta_Y_dist*(c_S_floor**n_Sf_Yd)/(k_Sf_Yd**n_Sf_Yd +
c_S_floor**n_Sf_Yd)- alpha_Y_dist * c_Y_dist
  dZ_distdt = beta_Z_dist*((1)/(1+(c_Y_dist/k_YdZd)**n_YdZd))-
alpha_Z_dist * c_Z_dist


  nY = c_Y + dt * dYdt
  nA = c_A + dt * dAdt
  nF = c_F + dt * dFdt
  n_S_floor = c_S_floor + dt * dS_floordt
  n_X_v = c_X_v + dt * dX_vdt
  n_Y_v = c_Y_v + dt * dY_vdt
  n_R_v = c_R_v + dt * dR_vdt
  n_Z_v = c_Z_v + dt * dZ_vdt
  n_Y_dist = c_Y_dist + dt * dY_distdt
  n_Z_dist = c_Z_dist + dt * dZ_distdt
```

```
    X[cind] = c_X
    H[cind] = c_H
    D[cind] = c_D
    Y[cind] = nY
    A[cind] = nA
    S[cind] = c_S
    F[cind] = nF
    S_floor[cind] =  n_S_floor
    X_v[cind] = n_X_v
    Y_v[cind] = n_Y_v
    R_v[cind] = n_R_v
    Z_v[cind] = n_Z_v
    Y_dist[cind] = n_Y_dist
    Z_dist[cind] = n_Z_dist

    c_Y = nY
    c_A = nA
    c_F = nF
    c_S_floor =  n_S_floor
    c_X_v = n_X_v
    c_Y_v = n_Y_v
    c_R_v = n_R_v
    c_Z_v = n_Z_v
    c_Y_dist = n_Y_dist
    c_Z_dist = n_Z_dist

    cind +=1

plt.figure(dpi=100)
plt.plot(t,X)
plt.ylabel('Signal "X"')
plt.xlabel('time')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,H)
plt.ylabel('Signal "H"')
plt.xlabel('time')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,D)
plt.xlabel('time')
plt.ylabel('Signal "D"')
plt.grid()
plt.show()
```

```
plt.figure(dpi=100)
plt.plot(t,Y)
plt.xlabel('time')
plt.ylabel('Rate of Y')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,A)
plt.xlabel('time')
plt.ylabel('Rate of A')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,S)
plt.xlabel('time')
plt.ylabel('Rate of S')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 1])
plt.plot(t,F)
plt.xlabel('time')
plt.ylabel('Rate of F')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 1])
plt.plot(t,S_floor)
plt.xlabel('time')
plt.ylabel('Rate of S_floor')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 20])
plt.plot(t,X_v)
plt.xlabel('time')
plt.ylabel('Rate of X_v')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 20])
plt.plot(t,Y_v)
```

```
plt.xlabel('time')
plt.ylabel('Rate of Y_v')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 20])
plt.plot(t,R_v)
plt.xlabel('time')
plt.ylabel('Rate of R_v')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.axis([0, t_end, 0, 20])
plt.plot(t,Z_v)
plt.xlabel('time')
plt.ylabel('Rate of Z_v')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,Y_dist)
plt.xlabel('time')
plt.ylabel('Rate of Y_dist')
plt.grid()
plt.show()

plt.figure(dpi=100)
plt.plot(t,Z_dist)
plt.xlabel('time')
plt.ylabel('Rate of Z_dist')
plt.grid()
plt.show()
```