Report for Project 2

November 24, 2022

1 Background

This report is about a MATLAB model for a fox chasing a rabbit. Using the model to find the results of the event under different speed conditions.

1.1 Position information

Put this question in a coordinate system, there are some key points and spaces that show in Figure 1:

1. The fox is started at $F(-250, -550)$.
2. The rabbit is initially located at $R(0, 0)$.
3. There is a warehouse that indefinitely extends to the west with its southeast corner located at point $S(-200, -400)$ and its northeast corner located at point $N(-200, 0)$.
4. The burrow is located at point $B(-600, 600)$.
5. The rabbit is deemed to be captured when the distance between them is smaller or equal to 0.1 meters, i.e. $|F_t R_t| \leq 0.1$.

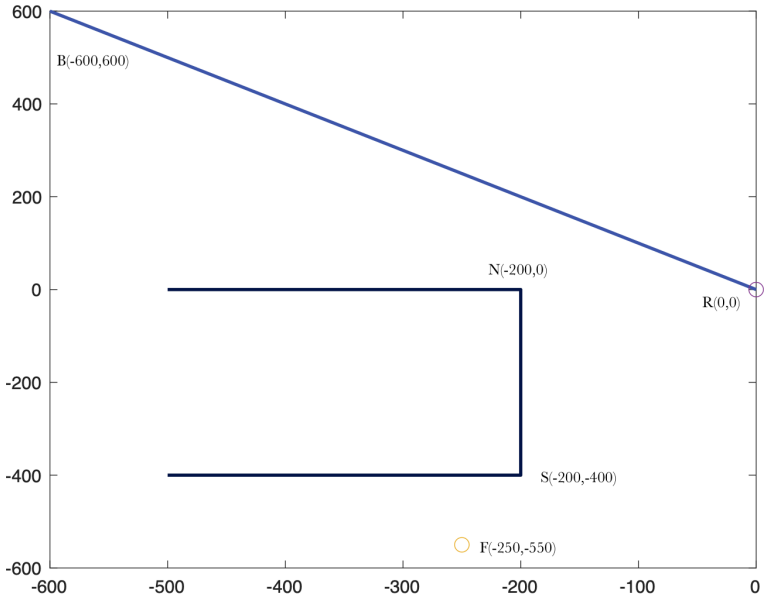


Figure 1: The initial positions.

1.2 Movement rules and information

1. The rabbit always moves towards the burrow.
2. If the fox can see the rabbit, it moves straight toward the rabbit, which means the direction of the velocity vector of the fox is the same as the direction of the vector $\overrightarrow{F_tR_t}$.
3. If the fox cannot see the rabbit, there are two cases:
   · If the sight is blocked by corner S, the fox moves directly to corner S.
   · If the sight is blocked by corner N, the fox moves parallel to the N-S perimeter until it can see the rabbit.
4. The rabbit's initial speed is $s_{r0} = 13m/s$.
5. The fox's initial speed is $s_{f0} = 16m/s$.

2 Required results

Before the rabbit reaches its burrow, determine whether the fox can catch it. Show the corresponding fox's coordinates, time of occurrence T and the length fox travelled d_f.

2.1 Constant speeds

In this case, the fox and the rabbit move at a constant speed which equal to the initial speeds.

2.2 Diminishing speeds

Consider if the fox and the rabbit will get tired after the movements, the longer they run, the more tired they get, then there are two parameters, diminishing rate of speeds (μ) and length of the path travelled ($d(t)$) added to determine their speeds.
The speeds of the rabbit and the fox could be calculated by following formulae, where $\mu_r = 0.0008$ and $\mu_f = 0.0002$, and $d_r(t)$, $d_f(t)$ is the length of the path they travelled at time t.

$$s_f(t) = s_{f0}e^{-\mu_f d_f(t)},\ s_r(t) = s_{r0}e^{-\mu_r d_r(t)}.$$

It is not hard to see question 1 is a special case in question 2 with $\mu_f = \mu_r = 0$. Once question 2 has been solved, the first one can be solved.

3 Solution framework

According to the previous statement, firstly, focus on the second question. The main idea is to solve the problem by constructing the ODE function set, then calculating using ODE45.

3.1 Stop conditions

Regardless of the intermediate movement trajectory, according to the question, there are only two possibilities for judging the event ends: the rabbit is captured or it escapes into the burrow. Hence, when the event ends, either the distance between the fox and rabbit (i.e. $|F_tR_t|$) is less or equal to 0.1 meters or the difference between the y-coordinates of the rabbit and the burrow (i.e. $|y_R - y_B|$) is zero.

3.2 ODE set functions

When constructing the ODE functions, the first-order derivative of the left-hand side should be equal to the right-hand side. For all values needed in the calculation, if no given data is available, they are regarded as variables.

3.2.1 Variables and analysis

Then there are six variables : d_f, x_f, y_f, d_r, x_r, and y_r (d for distance, x for the x-axis coordinate, y for the y-axis coordinate, f for the fox, and r for the rabbit).
The first derivative of the distance against time is always equal to the value of speed, the rabbit's velocity is always along the negative direction of the x-axis in the horizontal direction and along the positive direction of the y-axis in the vertical direction. So no matter how the fox's view changes, four equations remain unchanged during the whole period, only the velocity vector of the fox will change in different situations.

3.2.2 Four unchanged ODEs

Based on the previous analysis, it is easy to find the derivative of d_f and d_r which is separately equal to s_f and s_r. Hence,

$$\begin{cases} \frac{d(d_f(t))}{dt} = s_{f0}e^{-\mu_f d_f(t)} = s_f, \\ \frac{d(d_r(t))}{dt} = s_{r0}e^{-\mu_r d_r(t)} = s_r. \end{cases} \tag{1}$$

The equations for the x_r and y_r is also not hard, the rabbit moves from $(0, 0)$ to the burrow $(-600, 600)$, which is on the line $y = -x$. Decompose the rabbit's velocity into the x and y axes is $-\frac{1}{\sqrt{2}}s_r$ on the x-axis and $\frac{1}{\sqrt{2}}s_r$ on the y-axis. Thus,

$$\begin{cases} \frac{dx_r}{dt} = -\frac{1}{\sqrt{2}}s_r, \\ \frac{dy_r}{dt} = \frac{1}{\sqrt{2}}s_r. \end{cases} \tag{2}$$

3.2.3 Fox's velocity in three cases

Then remains the fox's velocity to be determined and decomposed, there are three situations.
1) The fox can see the rabbit
Then the direction of the fox's velocity is towards the rabbit, the fox's velocity is decomposed in the following formulae,

$$\begin{cases} \frac{dx_f}{dt} = \frac{s_f(x_r - x_f)}{\sqrt{(x_r - x_f)^2 + (y_r - y_f)^2}}, \\ \frac{dy_f}{dt} = \frac{s_f(y_r - y_f)}{\sqrt{(x_r - x_f)^2 + (y_r - y_f)^2}}, \end{cases} \tag{3}$$

where the denominator in equations is the distance between the fox and the rabbit.
2) The fox cannot see the rabbit when its position is before the point S
In this situation, the fox will run towards point S until it re-seeing the rabbit. Hence, using the coordinates of the point S to substitute the rabbit's coordinates in (3),

$$\begin{cases} \frac{dx_f}{dt} = \frac{s_f(x_s - x_f)}{\sqrt{(x_s - x_f)^2 + (y_s - y_f)^2}}, \\ \frac{dy_f}{dt} = \frac{s_f(y_s - y_f)}{\sqrt{(x_s - x_f)^2 + (y_s - y_f)^2}}, \end{cases} \tag{4}$$

where the denominator in equations is the distance between the fox and point S.

3) The fox cannot see the rabbit when its position is between S and N

The fox will move parallel to the N-S perimeter until it sees the rabbit again, so the x-axis velocity is zero while the y-axis velocity is s_f,

$$\begin{cases} \frac{dx_f}{dt} = 0, \\ \frac{dy_f}{dt} = s_f. \end{cases} \tag{5}$$

Then combine the $(1), (2)$ with (3) or (4), or (5) according to the fox's position, and all six ODE equations are constructed. The coming difficulty is to determine whether the fox could see the rabbit or not.

3.2.4 The fox's sight

Assuming the line FR and the line NS intersect at point P, for example, shown in the figure 2, then the sight of the fox could be determine by the y-coordinate of the point P.

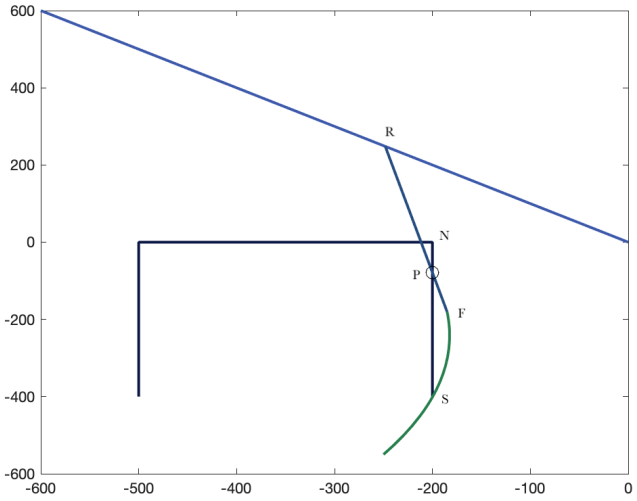


Figure 2: The line FR and the line NS intersect at point P.

When the point P is located on the NS line segment, the fox's sight is blocked by the warehouse either N or S, in other words, when $y_s \leq y_p \leq y_n$ the fox could not see the rabbit. The next problem is to determine which corner of the warehouse has blocked the fox, different corners affect the direction of movement of the fox.

The corner S could block the fox only if it has not yet reached point S, hence if $x_f < x_s$ and $y_f < y_s$ the fox is blocked by corner S, otherwise, the fox is blocked by corner N. The final piece of the puzzle to solve the problem is the coordinates of point P.

3.2.5 The coordinates of point P

Using vectors to find the coordinates of point P. Since points R, P and F are located on the same line, they have the relation $|\overrightarrow{PF}| = k|\overrightarrow{RP}|$, which expressed in coordinates is $(x_p - x_f, y_p -$

$y_f) = k(x_r - x_p, y_r - y_p)$, implies that the y-coordinate of the point P is

$$y_p = \frac{ky_r + y_f}{1 + k}.$$

Note the special case when k is equal to -1, the denominator is undefined. Since $x_p - x_f = k(x_r - x_p)$, when k is equal to -1, $x_p - x_f = x_p - x_r$, which implies that $x_f = x_r$. In this case, the fox and the rabbit have the same x-coordinate, the fox can see the rabbit and move toward the rabbit.

Then summarise the above ideas and allocate into three functions (primary function, event function and ODE set function), the problem could be solved with code.

4 Summary and organization

According to the above thinking, the flowchart to solve the question is shown in the figure 3.

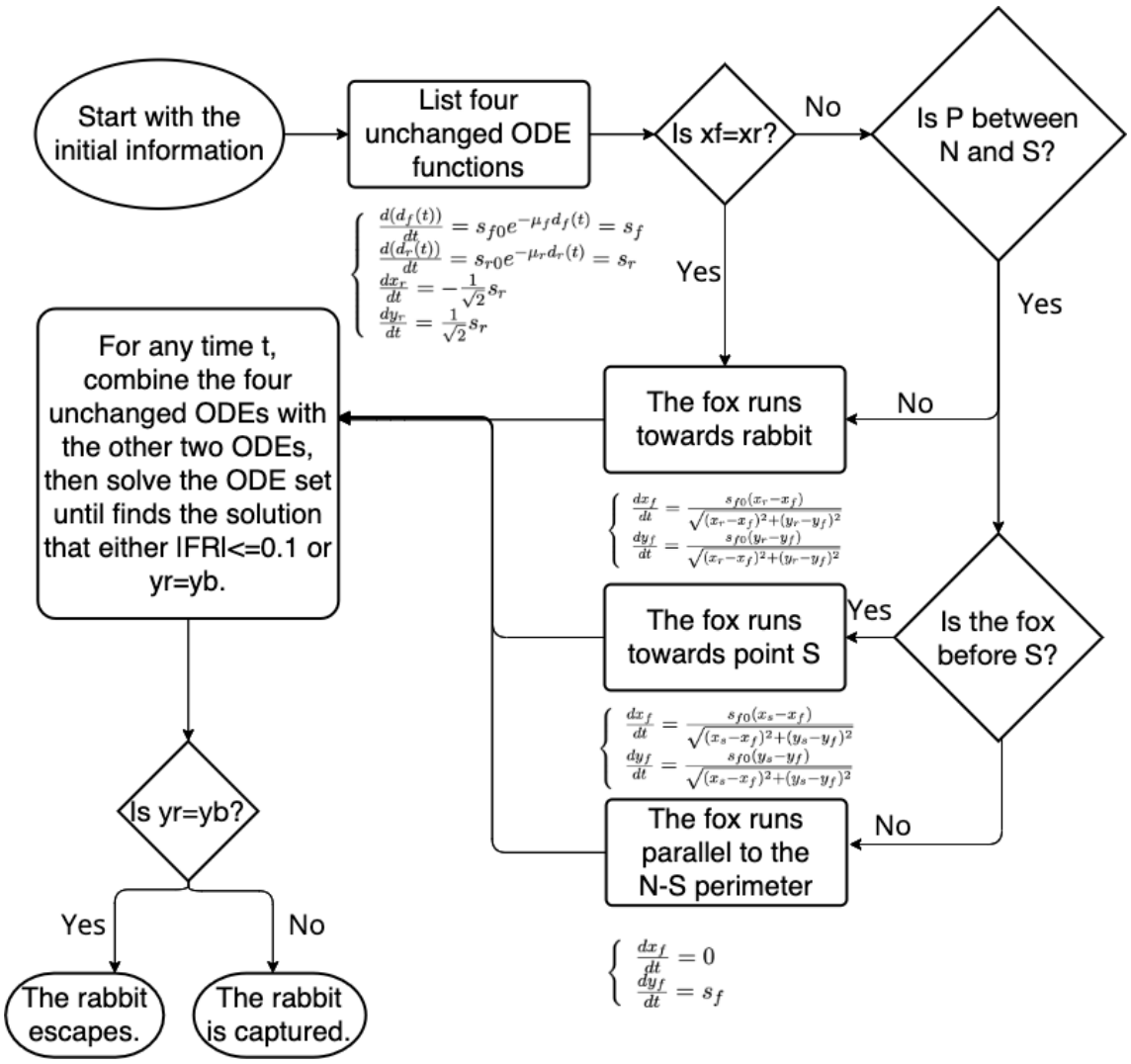


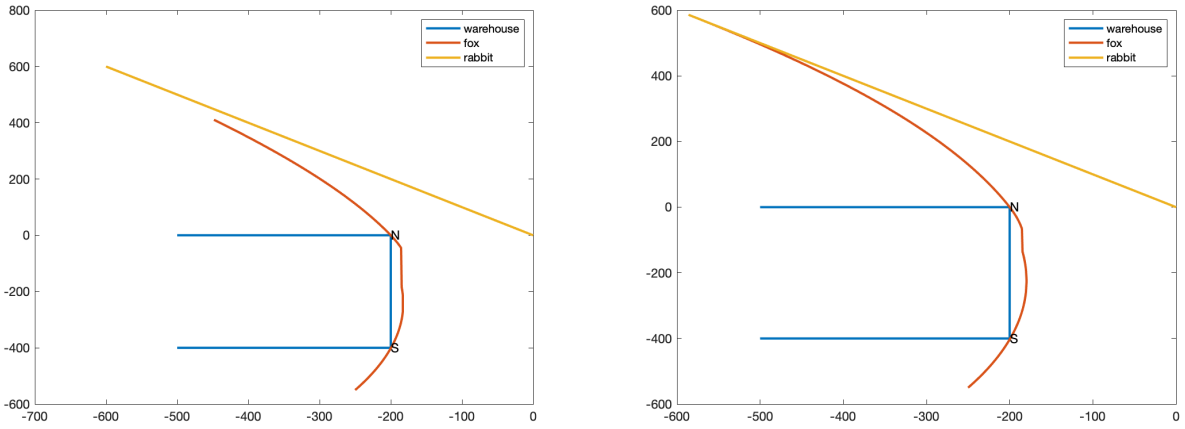
Figure 3: The flowchart of the solving ideas.

4.1 Allocation in the three functions

The primary function
The primary function contains the codes for calling the ODE45 function and using the difference between the rabbit's and burrow's x-coordinates to determine whether the rabbit escapes into the burrow or is chased by the fox.
The event function
The event function decides when to stop the integral. Using the distance between the rabbit and the fox to multiply the distance between the rabbit and the burrow, either result equals zero, the integral would stop.
The ODE set function
The ODE function should first list the four unchanged ODE functions, then determine if x_f is equal to x_r (i.e. $k = -1$). When they are not equal, use the formula to calculate the y-coordinate of the point P, depending on its value, using different ODE functions for the velocity of the fox.

5 Results

By running the codes, getting the results that the rabbit could escape into the burrow under the constant speeds, and the rabbit is captured under the diminishing speeds.
Figure 4(a) shows the movements under constant speeds, and figure 4(b) shows the movements under diminishing speeds. More details about the results show in figure 5.



(a) constant speed (b) diminishing speed

Figure 4: movements of the fox and the rabbit

```
>> Q1
Rabbit escapes into burrow.
At time T=65.2714s the fox travelled 1044.3423 meters and located at(-448.3933,410.7332), the rabbit escapes.
```

(a) constant speeds

```
>> Q2
Rabbit captured by fox.
At time T=90.463s the fox travelled 1271.2019 meters and located at(-586.0398,586.0398), the rabbit is captured.
```

(b) diminishing speeds

Figure 5: result details

6 Appendix - MATLAB code

01-Primary function part for Question 2

```
% Q2, change mu to 0 for Q1
mur=0.0008;muf=0.0002;
sr0=13;sf0=16;
s=[-200 -400];n=[-200 0];
z0=[0 -250 -550 0 0 0];
tspan=[0 100];
x=[-500,-200,-200,-500];
y=[-400,-400,0,0];% plot for a warehouse
options = odeset('Events',@(t,z) stop(t,z),'MaxStep',0.002);
[t,z,te,ze,yi] = ode45(@(t,z)ode(t,z,sr0,sf0,mur,muf),tspan,z0,options);
plot(x,y,z(:,2),z(:,3),z(:,5),z(:,6),'LineWidth',2)
legend('warehouse','fox','rabbit')
hold on
text(-200,0,'N')
text(-200,-400,'S')
if abs(ze(6)-600)<10^(-10)% adjust for float number
    a='the rabbit escapes.';
elseif sqrt((ze(5)-ze(2))^2+(ze(6)-ze(3))^2)<=0.1
    a='the rabbit is captured.';
end
disp(['At time T=' num2str(te) 's the fox travelled ' num2str(ze(1)) ...
    ' meters and located at(' num2str(ze(2)) ',' num2str(ze(3)) '), ' a ])
```

02-Primary function part for the Question 1
Only the first line of code needs to be changed compared to the question 2.

```
mur=0;muf=0;
```

03-ODE set function

```
function dzdt = ode(t,z,sr0,sf0,mur,muf)
%1.df 2.xf 3.yf 4.dr 5.xr 6.yr
% sight=1 means can see the rabbit, while sight=0 means cannot see
dzdt=0*z;% make sure the shape of dzdt is same to the z
```

```
sight=1;% move towards rabbit
s=[-200 -400];
n=[-200 0];
dist=sqrt((z(5)-z(2))^2+(z(6)-z(3))^2); % |FR|
dist1=sqrt((s(1)-z(2))^2+(s(2)-z(3))^2);% |SF|
dzdt(1)=sf0*exp(-muf*z(1));% sf
dzdt(4)=sr0*exp(-mur*z(4));% sr
dzdt(5)=-dzdt(4)/sqrt(2);% vr x-axis
dzdt(6)=dzdt(4)/sqrt(2);% vr y-axis
if abs(z(2)-z(5))>eps% k ~= -1
    k=(-200-z(2))/(z(5)+200);
    yp=(k*z(6)+z(3))/(1+k);
    if yp>=s(2) && yp<=n(2)% fox's sight is blocked
        sight=0;
    end
end
if sight==0
    if z(3)<s(2) && z(2)<s(1)% towards s
        dzdt(2)=dzdt(1)*(s(1)-z(2))/dist1;% vf x-axis
        dzdt(3)=dzdt(1)*(s(2)-z(3))/dist1;% vf y-axis
    else% straight
        dzdt(2)=0;% vf x-axis
        dzdt(3)=dzdt(1);% vf y-axis
    end
else
    dzdt(2)=dzdt(1)*(z(5)-z(2))/dist;% vf x-axis
    dzdt(3)=dzdt(1)*(z(6)-z(3))/dist;% vf y-axis, towards rabbit
end
end
```

04-event function

```
function [value,isterminal,direction] = stop(t,z)
dist=sqrt((z(5)-z(2))^2+(z(6)-z(3))^2);% |FR|
value(1)=(600-z(6))*(dist-0.1);% captured or escape
isterminal(1)=1;
direction(1)=0;
end
```