Project 2

Background

In this project, we are going to investigate the dynamics between a fox and a rabbit by solving sets of ODEs which model their positions at different times. In this setting there is a rectangular impenetrable warehouse. One of the shorter edge is SW(200,−400)-NW(200,0) and we can assume it longer edges is extended indefinitely to the east.

The initial position of the fox is (250,−550)with speed $s_{f0} = 16m/s$ .If the fox can see the rabbit, the tangent of the fox's path points toward the rabbit; if the view of the rabbit is blocked by the corner SW then the fox runs to SW directly. If the rabbit is still not in sight when the corner is reached, then the fox moves parallel to the NW-SW perimeter of the warehouse until it sees the rabbit.

The initial position of the rabbit is (0,0) and runs towards its burrow at (600,600) in a straight line with initial speed $s_{r0} = 13m/s$.

In question one the speed of the rabbit $s_r(t)$ and the speed of the fox $s_f(t)$ are constants.

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

$\mu_r = -0.0008$ and $\mu_f = -0.0002$ are the rates of diminishing of rabbit and fox.
$d_r(t)$ and $d_f(t)$ are the distance traveled by rabbit and fox with respect to time.

And in both question we need to determine whether the rabbit can be captured before it reaches its burrow.

A useful function

First, we introduce **function** *inter* **= intersection**(*x1,y1,x2,y2,a*).The first four input arguments are the coordinates of two points $(x1, y1)$and $(x2, y2)$.The final input argument *a* is the constant of the constant function $y = a$.The output argument *inter* is the x coordinate of the intersection of the line through the two points and the line $y = a$.The following is the Matlab code for this function:

```
function  inter = intersection(x1,y1,x2,y2,a)
%x coordinate of the intersection of the line through the
% two points (x1,y1),(x2,y2)and the y=a
inter = x1+(((a)-(y1))/((y2-y1)/(x2-x1)));
end
```

Blocking classification and formulation

The fox's view is blocked by the warehouse when the line segment through fox and rabbit intersects with the warehouse. We need to consider the positions of fox and rabbit related to the vertical line $x = 200$.Let $z_1(t), z_2(t)$are the x as well as y coordinates of rabbit respect to time t and $z_3(t), z_4(t)$ are the x as well as y coordinates of fox respect to time$t$,It is clear that $z_1(t) < 200$ and $z_3(t) < 200$ the fox is able to see the rabbit. So there are potentially 3 blocking situations we need to consider:

$\langle 1 \rangle$: $z_1(t) < 200$ and $z_3(t) > 200$ and **intersection** $(z_1, z_2, z_3, z_4, -400) \geq 200$.

$\langle 2 \rangle$: $z_1(t) > 200$ and $z_3(t) < 200$ and **intersection** $(z_1, z_2, z_3, z_4, 0) \geq 200$.

$\langle 3 \rangle$: $z_1(t) > 200$ and $z_3(t) > 200$ and $z_4(t) \leq -400$

(The Figure 1 shows the potential 3 blocking situations. The **red** line is the rabbit's track, the **blue** line is the situation $\langle 1 \rangle$, the green line is the situation $\langle 2 \rangle$ and the **black** line is the situation $\langle 3 \rangle$.)

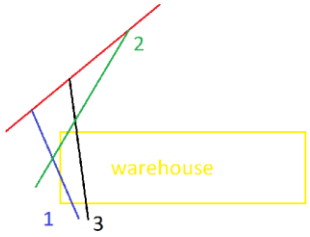


Figure 1

However, $\langle 3 \rangle$ does not happen in both 2 questions. In order to proof this result, it is useful to introduce the following:

Theorem

Let f be a continuous, monotonic function defined on the closed interval $[a, b]$.
Then f has the maximum arclength $(b - a) + |f(b) - f(a)|$.

(Proof of the theorem and more detail see [1])

In order to use the theorem, I define a new coordinate system, with origin located at the initial position of the fox and the directions of the x and y axes are pointing up and left, respectively.

Assume $\langle 3 \rangle$ does appear which means that the dynamic of the fox is chase the rabbit first and then runs towards SW in the closed interval $x \in [0, 150]$ and let t_* is the time when the fox reach SW. (See figure 2)

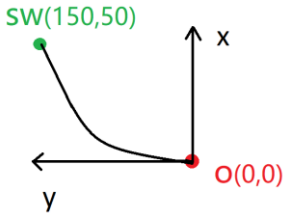


Figure 2

Let $f(x)$ denotes the trajectory of the fox. First $f(x)$ is continuous. Second, the rabbit is on the top left of the fox when fox can see it and SW is also on the top left of the fox when it runs towards SW, so the fox can only be moves up and left ,thus $f(x)$ is a continuous increasing function.

So the theorem gives us the length of the trajectory(the distance travel by fox until t_*) $d_f(t_*)$ is bounded above by 200.(i.e $200 = (150 - 0) + |50 - 0|$).

In question one the speed for the fox and rabbit are constants and $s_f > s_r$ for all time.

So $d_r(t_*) < d_f(t_*) \leq 200$.Let $d_{\text{rh}}(t_*)$ is the horizontal distance for rabbit travel at time t_*.

$d_{\text{rh}}(t_*) < 200 \times \cos\left(\frac{\pi}{4}\right) < 200$. The rabbit is on the left side of the warehouse, (i.e $z_1(t) < 200$ for the original coordinate system) thus $\langle 3 \rangle$ will not appear.

In question two, the speed for the fox and rabbit are both not constants. By solving the inequality $s_r < s_f$ for d_f.

$$s_r < s_f \Leftrightarrow s_{\text{r0}} e^{-\mu_r d_r} < s_{\text{f0}} e^{-\mu_f d_f} \Leftrightarrow d_f < 4 \times (d_r + 259.549).$$

The RHS of the inequality is bounded below by $4 \times 250 = 1000$. $d_f(t)$ is less than 1000 for all $t \in [0, t_*]$,which means $s_r(t) < s_f(t)$ for all $t \in [0, t_*]$,Thus $d_r(t_*) < d_f(t_*) \leq 200$.Thus $\langle 3 \rangle$ will not appear.(proof similar to the first question)

ODEs for Q1

The dynamic system for rabbit and fox is the following:

$$\dot{z_1} = s_{\text{r0}} \times \cos\left(\frac{\pi}{4}\right), \dot{z_2} = s_{\text{r0}} \times \sin\left(\frac{\pi}{4}\right)$$

When the fox can see the rabbit, the attack path points directly towards the rabbit.

$$\dot{z_3} = s_{\text{f0}} \times \frac{z_1 - z_3}{\sqrt{(z_1 - z_3)^2 + (z_2 - z_4)^2}}, \dot{z_4} = s_{\text{f0}} \times \frac{z_2 - z_4}{\sqrt{(z_1 - z_3)^2 + (z_2 - z_4)^2}}$$

when $\langle 1 \rangle$ happens, the attack path points directly towards $\text{SW}(200, -400)$.

$$\dot{z_3} = s_{\text{f0}} \times \frac{200 - z_3}{\sqrt{(200 - z_3)^2 + (-400 - z_4)^2}}, \dot{z_4} = s_{\text{f0}} \times \frac{-400 - z_4}{\sqrt{(200 - z_3)^2 + (-400 - z_4)^2}}$$

when $\langle 2 \rangle$ happens, the attack path only points upward.(i.e parallel to NW-SW perimeter of the warehouse)

$$\dot{z_3} = 0, \dot{z_4} = s_{\text{f0}}$$

Code for the ODE

Define **function** *dzdt* = **foxrab_ode1**(*z,s_r ,s_f*).z is a column vector defined as $\vec{z} = (z_1, z_2, z_3, z_4)^T$,*s_r and s_f* are the initial speed for the rabbit and the fox. The function return $\frac{d}{dt}\vec{z}$ .In this function we can use **If-elseif-else** to change the dynamic of the fox. The following is the code:

```
function  dzdt = foxrab_ode1(z,s_r ,s_f)
%z=[z1 z2 z3 z4],s_r is the initial speed of the rabbit,s_f is for fox
```

```
dzdt = zeros (4,1); %make  sure  the  output  is a 4*1 column  vector
%the dynamic system for rabbit
dzdt(1) = s_r*cos(pi/4);
dzdt(2) = s_r*sin(pi/4);
%the dynamic system for fox
%situation 1 happen
if z(3) > 200 && z(1) <200 &&intersection(z(1),z(2),z(3),z(4),400)>=200
       dist1 = sqrt((200-z(3))^2+(-400-z(4))^2); % distance between fox and SW
       dzdt (3) = s_f*(200-z(3))/dist1;
       dzdt (4) = s_f*(-400-z(4))/dist1;
%situation 2 happen
elseif z(3) < 200 && z(1) >200 && intersection(z(1),z(2),z(3),z(4),0)>=200
       dzdt (3) = 0;
       dzdt (4) = s_f;
else%the fox can see the rabbit
dist = sqrt((z(1)-z(3))^2+(z(2)-z(4))^2); % distance between fox and rabbit
       dzdt (3) = s_f*(z(1)-z(3))/dist; % horizontal  velocity
       dzdt (4) = s_f*(z(2)-z(4))/dist; % vertical  velocity
end
end
```

Event function

In order to stop the ode solver, we need to introduce the event function. There are two events in the function, namely the fox catches the rabbit and the rabbit reaches the burrow.

1:When the fox catches the rabbit, which means the distance between fox and rabbit is less or equal to minimum distance *mindist=0.1.*The value of $\sqrt{(z_1-z_3)^2+(z_2-z_4)^2}-\text{mindist}$ must pass through 0 from positive to negative so the *direction* is -1 in the event function for this event.

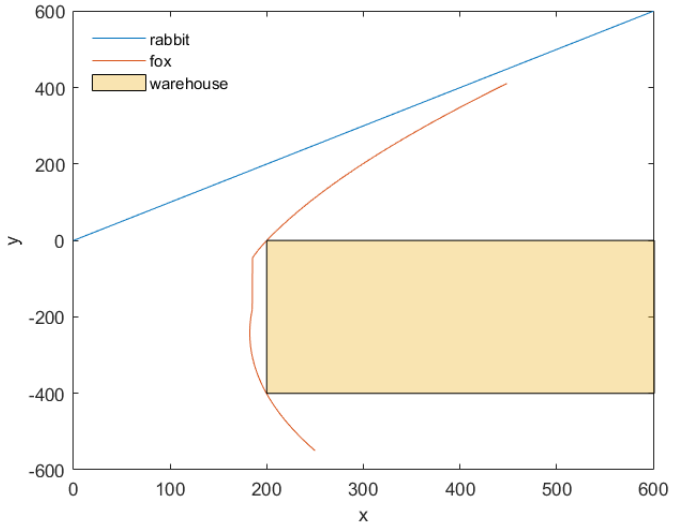
2:The rabbit reaches the burrow, and let the position of burrow is (b_1,b_2).So when the rabbit reach the burrow the value of (b_1-z_1)must pass through 0 from positive to negative so the *direction* is *also* -1 in the event function for this event. The following is the code for the event function:

```
function [value , isterminal , direction] = foxrab_events(z,mindist ,burrow)
%stop the solver when rabbit is caught
value (1) = sqrt((z(1)-z(3))^2+(z(2)-z(4))^2) - mindist;
isterminal (1) = 1;
direction (1) = -1;% the event is triggered when "value(1)"changes from positive
to negative
%stop the solver when rabbit reaches the burrow
value (2) = (burrow(1)-z(1));
isterminal (2) = 1;
direction (2) = -1;% the event is triggered when "value(2)"changes from positive
to negative
end
```

Solution for Q1

In order to solve the system of ODEs numerically, we can use the built in function **ode45.**Moreover,I will use the event function to stop the solver when any event is triggered and setting the absolute tolerance and relative tolerance are 10^-7.(See comment of the code for more details.)

```
s_r = 13;%The initial speed of rabbit
s_f = 16;%The initial speed of fox
mindist = 0.1;%The distance the fox can just catch a rabbit
burrow = [600 600];%The position of the burrow
z0 = [0 0 250 -550];%The initial condition
ts = [0 norm([z0(3) z0(4)])/(s_f - s_r)];%The timespan
options=odeset('RelTol',1e-7,'AbsTol',1e-7,...
'Events',@(t,z) foxrab_events(z,mindist ,burrow));
%use the ode45 to solve the system of ODEs
[t,z,te,ze,zi] = ode45(@(t,z) foxrab_ode1(z,s_r ,s_f),ts,z0,options);
plot(z(:,1),z(:,2),z(:,3),z(:,4))%plot the track of Fox and rabbit
xlabel('x'),ylabel('y')
hold on
plot(polyshape([200 200 601 601],[-400 0 0 -400]))%plot the warehouse
%add the legend
legend({'rabbit','fox','warehouse'},'Location','northwest')
legend('boxoff')
hold off
%set the size of the plotting window
xlim([-1 601])
ylim([-600 601])
```




```
zi%Check which event was triggered
```

```
zi = 2
```

Event 2 is triggered first, which means the rabbit reaches the burrow before it is caught by the fox, so the fox can’t catch the rabbit.

ODEs for Q2

Let $z_5(t)$ and $z_6(t)$ are the distance of rabbit and fox move respect to time t.The speed of rabbit is $s_r(t) = s_{r0}e^{-\mu_r z_5(t)}$ ($\mu_r = 0.0008$), the speed of fox is $s_f(t) = s_{f0}e^{-\mu_f z_6(t)}$ ($\mu_f = 0.0002$).

The dynamic system for rabbit and fox is the following:

$$\dot{z_1} = s_r(t) \times \cos\left(\frac{\pi}{4}\right), \dot{z_2} = s_r(t) \times \sin\left(\frac{\pi}{4}\right)$$

$\mathrm{dz}_5 = \sqrt{(\mathrm{dz}_1)^2 + (\mathrm{dz}_2)^2}$ and $\mathrm{dz}_6 = \sqrt{(\mathrm{dz}_3)^2 + (\mathrm{dz}_4)^2}$ by Pythagoras' Theorem.

Thus:

$$\dot{z_5} = \sqrt{(\dot{z_1})^2 + (\dot{z_2})^2}, \dot{z_6} = \sqrt{(\dot{z_3})^2 + (\dot{z_4})^2}$$

When the fox can see the rabbit, the attack path points directly towards the rabbit.

$$\dot{z_3} = s_f(t) \times \frac{z_1 - z_3}{\sqrt{(z_1 - z_3)^2 + (z_2 - z_4)^2}}, \dot{z_4} = s_f(t) \times \frac{z_2 - z_4}{\sqrt{(z_1 - z_3)^2 + (z_2 - z_4)^2}}$$

When $\langle 1 \rangle$ happens, the attack path points directly towards $\mathrm{SW}(200, -400)$.

$$\dot{z_3} = s_f(t) \times \frac{200 - z_3}{\sqrt{(200 - z_3)^2 + (-400 - z_4)^2}}, \dot{z_4} = s_f(t) \times \frac{-400 - z_4}{\sqrt{(200 - z_3)^2 + (-400 - z_4)^2}}$$

When $\langle 2 \rangle$ happens, the attack path only points upward.(i.e parallel to NW-SW perimeter of the warehouse)

$$\dot{z_3} = 0, \dot{z_4} = s_f(t)$$

Define **function** *dzdt* = **foxrab_ode2**(*z,s_r ,s_f*),the definition of *s_r ,s_f* are the same as question one. Redefine $\vec{z} = (z_1, z_2, z_3, z_4, z_5, z_6)^T$ and the function is very similar to question one, please see the comment of the code for more details. The following is my code for the function:

```
function  dzdt = foxrab_ode2(z,s_r,s_f)
%z=[z1 z2 z3 z4 z5 z6],s_r is the initial speed of the rabbit,s_f is for fox
dzdt = zeros (6,1);%make  sure  the  output  is a 6*1 column  vector
sr = s_r*exp(-0.0008*z(5));%the speed of the rabbit
%the dynamic for rabbit
```

```
dzdt(1) = sr*cos(pi/4);
dzdt(2) = sr*sin(pi/4);
dzdt(5) = sqrt(dzdt(1)^2+dzdt(2)^2);
%the dynamic for fox
sf = s_f*exp(-0.0002*z(6));%the speed of the fox
%the dynamic for fox
if z(3) > 200 && z(1) <200 && intersection(z(1),z(2),z(3),z(4),-400)>=200
    dist1 = sqrt((200-z(3))^2+(-400-z(4))^2); % distance between fox and SW
    dzdt (3) = sf*(200-z(3))/dist1;
    dzdt (4) = sf*(-400-z(4))/dist1;
elseif z(3) < 200 && z(1) >200 && intersection(z(1),z(2),z(3),z(4),0)>=200
    dzdt (3) = 0;
    dzdt (4) = sf;
else
    dist = sqrt((z(1)-z(3))^2+(z(2)-z(4))^2); % distance between fox and rabbit
    dzdt (3) = sf*(z(1)-z(3))/dist; % horizontal  velocity
    dzdt (4) = sf*(z(2)-z(4))/dist; % vertical  velocity
end
dzdt(6) = sqrt(dzdt(3)^2+dzdt(4)^2);
end
```

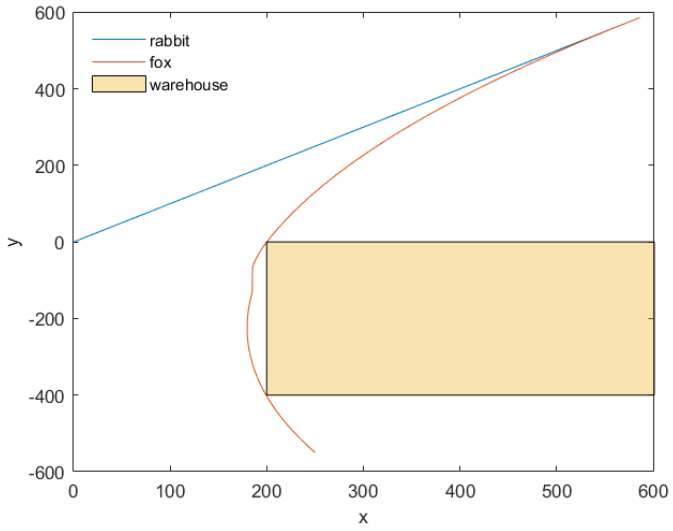
Solution for Q2

We use the same event function and absolute tolerance and relative tolerance as question one.

Notice: To reduce the length of the code, I decided to use some of the initial values in question 1 and rewrite the initial condition.

Please run solution code for Q1 before running the following code:

```
%please run Q1 first
%The last 2 elements is the initial distances traveled by fox and rabbit
z0 = [0 0 250 -550 0 0];
options=odeset(...
'RelTol',1e-7,'AbsTol',1e-7,'Events',@(t,z) foxrab_events(z,mindist ,burrow));
[t,z,te,ze,zi] = ode45(@(t,z) foxrab_ode2(z,s_r,s_f),ts,z0,options);
plot(z(:,1),z(:,2),z(:,3),z(:,4))
xlabel('x'),ylabel('y')
hold on
plot(polyshape([200 200 601 601],[-400 0 0 -400]))
%add the legend
legend({'rabbit','fox','warehouse'},'Location','northwest')
legend('boxoff')
hold off
%set the size of the plotting window
xlim([-1 601])
ylim([-600 601])
```




```
zi%Check which event was triggered
```

```
zi = 1
```

Event 1 is triggered first, which means the fox catches the rabbit before it reaches the borrow, so the fox can catch the rabbit.

References

[1]Marc Deléglise and Andrew Markoe, On the Maximum Arc Length of Monotonic Functions, May 29th 2013.