Национальный исследовательский университет «МИЭТ»

Лабораторная работа №3

ПРО 2016

Выполнил:

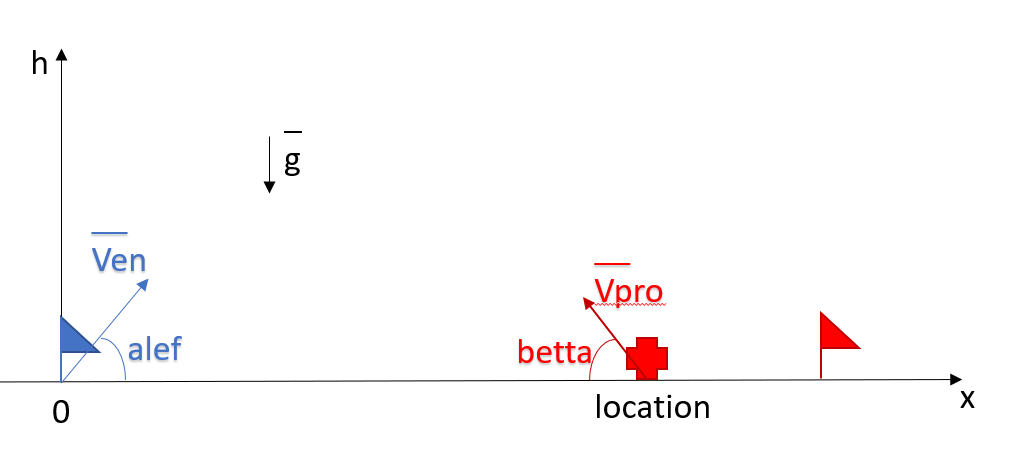
Студент группы ПИН-44

Мясников Максим

Москва, 2021 г.

**Постановка задачи**

"Противник" производит пуск ракеты класса "земля-земля" из т. А со скоростью v0=1000 м/с под углом pi/4 по цели в т. Д, расположенной на Вашей территории (расстояние L между тт. А и Д можно рассчитать). Старт мгновенно фиксируется со спутника и после принятия решения о противодействии начинается телеметрия траектории ракеты. Ежесекундно измеряются (к сожалению с неизбежными ошибками) координаты траектории движения ракеты на пассивном участке траектории с 15-й по 40-ю секунды полета. Полученные данные обрабатываются для получения уравнения движения ракеты. Одна антиракета должна стартовать из точки С, расположенной на растоянии 0.75L от т. А (или 0.25L от т. Д)не позднее, чем ракета пройдет через максимальную по высоте точку своей траектории) и поразить ракету в полете, другая антиракета из этой же точки С должна накрыть пусковую площадку противника (оцененное по измерениям положение т. А). Стартовая Скорость антиракеты u0=2000 м/с, а угол старта и момент старта Вам необходимо определить.

****

Допущения:

* Ускорение свободного падения g = 9.8 м/с2
* Шаг по времени t = 0.01 сек
* Ракеты – материальные точки

Входные данные:

* Начальная скорость ракеты противника ven = 1000 м/с
* Угол запуска ракеты противника alefen = pi/4
* Диапазон высоты съема данных diap ∈ [10000 20000] м
* Погрешность снятия данных eps = 10 м
* Время расчета траекторий ПРО tsolve = 5 сек
* Период получения информации о вражеской ракете titer = 0.1 сек
* Расстояние от вражеской ПУ до нашей ПРО location = 75000 м;
* Скорость наших ракет vpro= 2000 м/с
* Радиус взрыва bomb\_radius = [50 100 150] м;

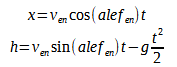
Выходные данные:

* Заключение об уничтожении вражеской ракеты и вражеской ПУ
* Вероятность сбития вражеской ракеты, вражеского ПУ

**Аналитическое исследование**

Будем работать с моделью в режиме реального времени с итерационным шагом titer = 0.01 сек.

Движение вражеской ракеты:



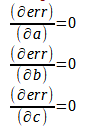
Погрешность получения данных ПРО о цели:



Приближать полученные данные до параболической траектории будем методом наименьших квадратов:



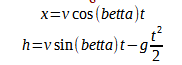
Минимизация будет достигнута при коэффициентах a, b, c полученных решением системы:



Получим некоторое уравнение параболы, по которой по нашим предположениям движется вражеская ракета:



Построив расчетную траекторию противника, рассчитаем углы запуска наших ракет по вражеской ракете и вражеской ПУ. Воспользуемся уравнением движения:



Тогда:

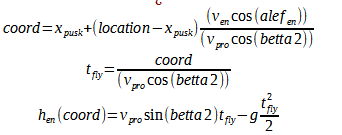
Необходимо поразить две точки:

1. (xen(hen=0), 0)

Для нее угол запуска нашей ракеты рассчитывается по формуле:



1. Второй угол будет находиться из системы



Где xpusk – координата вражеской ракеты в момент пуска нашей.

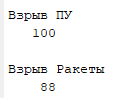
**Численное решение**



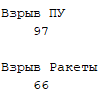
Для моделирования вероятностей сбития вражеской ракеты, вражеского ПУ произведём 100 запусков, для каждого из типов наших ракет противодействия ракет:

Процентная вероятность взрыва противника при

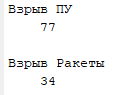
* Радиус взрыва 150 м:



* Радиус взрыва 100 м:



* Радиус взрыва 50 м:



**Основные источники ошибки**

* Аппроксимация траектории движения ракеты по считанным с нормальной погрешностью;
* Вычислительные ошибки при символьном вычислении углов запуска ракеты
* Ошибки вызванные итеративностью модели (итерационный шаг)

**Вывод**

Для успешного поражения ракет противника необходимо запускать ракеты возмездия с радиусом взрыва порядка 150 м

Для успешного поражения ракет противника необходимо запускать ракеты возмездия с радиусом взрыва порядка 100 м

**Приложение 1. Код используемый для модели**

clear

doPlot = false;

v\_en = 1000;

alef\_en = pi/4;

location = 75000;

minDiap = 10000;

maxDiap = 20000;

eps = 10;

t\_solve = 5;

g = 9.8;

t\_iter = 0.01;

t\_rli = 0.1;

v\_pro=2000;

b\_step = 0.001;

bomb\_radius=50;

t=0;

index = 1;

newIndex = 1;

X\_en(index) = 0;

H\_en(index) = 0;

x\_en(newIndex) = 0;

h\_en(newIndex) = 0;

t\_last = 0;

t\_pusk = 0;

X\_pusk =0;

H\_pusk =0;

bombx\_en =0;

bombh\_en =0;

h2=0;

t\_bomb=1000;

syms a

syms b

syms c

C1 = 0;

C2 = 0;

for model=1:1:100

clear X\_en;

clear H\_en;

clear x\_en;

clear h\_en;

t=0;

index = 1;

newIndex = 1;

X\_en(index) = 0;

H\_en(index) = 0;

x\_en(newIndex) = 0;

h\_en(newIndex) = 0;

t\_last = 0;

t\_pusk = 0;

X\_pusk =0;

H\_pusk =0;

bombx\_en =0;

bombh\_en =0;

h2=0;

t\_bomb=1000;

syms a

syms b

syms c

while(true)

t=t+t\_iter;

index=index+1;

X\_en(index) = v\_en\*cos(alef\_en)\*t;

H\_en(index) = v\_en\*sin(alef\_en)\*t-g\*t\*t/2;

if H\_en(index) >= minDiap && H\_en(index) <= maxDiap && X\_en(index) < location/2

if t\_last == 0

t\_last = t;

end

if t\_last - t < t\_rli

x\_en(newIndex) = X\_en(index);

h\_en(newIndex) = normrnd(H\_en(index), eps);

newIndex = newIndex+1;

t\_last=t\_last+1;

end

else

if t\_last ~= 0

if t-t\_last >= t\_solve-1

t\_pusk = t;

X\_pusk = X\_en(index);

H\_pusk = H\_en(index);

t\_last = 0;

Q = 0;

for s = 1:1:newIndex-1

Q = Q + (h\_en(s) - a\*x\_en(s)\*x\_en(s)-b\*x\_en(s)-c)^2;

end

coefs = solve(diff(Q, a),diff(Q, b),diff(Q, c));

syms x

coords1 = solve(double(coefs.a)\*x\*x+double(coefs.b)\*x+double(coefs.c));

coord1 = double(coords1(1));

betta1=(asin( g\*(location-coord1) / v\_pro / v\_pro )) / 2;

syms betta2;

coord2 = X\_pusk + ( location - X\_pusk ) \* (v\_en \* cos(alef\_en)) / (v\_pro \* cos(betta2));

t\_fly = coord2 /v\_pro / cos(betta2);

h\_en\_coord2 = double(coefs.a)\*coord2\*coord2+double(coefs.b)\*coord2+double(coefs.c);

q = v\_pro\*sin(betta2)\*t\_fly-t\_fly\*t\_fly\*g/2-h\_en\_coord2;

qb = 350;

bb = 1;

for b = 0:b\_step:pi/2

qq=double(subs(q, 'betta2', b));

if abs(qq) < qb

qb = abs(qq);

betta2 = b;

end

end

coord2 = subs(coord2, 'betta2', betta2);

h2 = double(coefs.a)\*coord2\*coord2+double(coefs.b)\*coord2+double(coefs.c);

syms b2;

betta2=double(atan(h2/(location-coord2)));

t\_bomb = double((coord2) / (v\_en\* cos(alef\_en)));

end

end

end

if t-t\_bomb <= t\_iter

bombx\_en = X\_en(index);

bombh\_en = H\_en(index);

end

if H\_en(end)<=0 && X\_en(end)>0

break

end

end

if(doPlot)

hold on

grid on

xlim([-1000 120000])

ylim([0 30000])

xlabel('расстоие от базы противника, м')

ylabel('высота, м')

title('полет вражеских ракет')

plot(X\_en, H\_en, 'b')

plot(location, 0, 'k+', 'LineWidth', 2)

plot(100000, 0, 'kp', 'LineWidth', 2)

plot(x\_en, h\_en,'\*')

plot(X\_pusk, H\_pusk, '^', 'LineWidth', 2)

plot(bombx\_en, bombh\_en, '^', 'LineWidth', 2)

x=double(coords1(1)):0.1:100000;

plot(x, double(coefs.a).\*x.\*x+double(coefs.b).\*x+double(coefs.c), 'r');

x=double(coords1(1)):0.1:location;

plot(x, x\*tan(betta1)-g.\*x.\*x./2./v\_pro./v\_pro./cos(betta1)./cos(betta1));

x=double(coord2):0.1:location;

h=(1-(x-double(coord2))./(location-double(coord2))).\*double(h2);

plot(x, h);

legend('траектория врага', 'ПРО', 'штаб', 'снятые данные', 'рак. в мом. пуск. наш.', 'рак. в мом. взрыв', 'мним.траект', 'возмездие', 'возмездие');

c1 = sqrt((0-double(coords1(1)))^2+(0-double(coefs.a).\*double(coords1(1)).\*double(coords1(1))+double(coefs.b).\*double(coords1(1))+double(coefs.c))^2);

if double(c1) <= bomb\_radius

C1 =C1+1;

end

c2 = sqrt((bombx\_en-double(coord2))^2+(bombh\_en-(1-(double(coord2)-double(coord2))./(location-double(coord2))).\*double(h2))^2);

if double(c2) <= bomb\_radius

C2 =C2+1;

end

else

c1 = sqrt((0-double(coords1(1)))^2+(0-double(coefs.a).\*double(coords1(1)).\*double(coords1(1))+double(coefs.b).\*double(coords1(1))+double(coefs.c))^2);

if double(c1) <= bomb\_radius

C1 =C1+1;

end

c2 = sqrt((bombx\_en-double(coord2))^2+(bombh\_en-(1-(double(coord2)-double(coord2))./(location-double(coord2))).\*double(h2))^2);

if double(c2) <= bomb\_radius

C2 =C2+1;

end

end

end

disp('Взрыв ПУ')

disp(C1);

disp('Взрыв Ракеты')

disp(C2);