```
clear
close all
clc
```

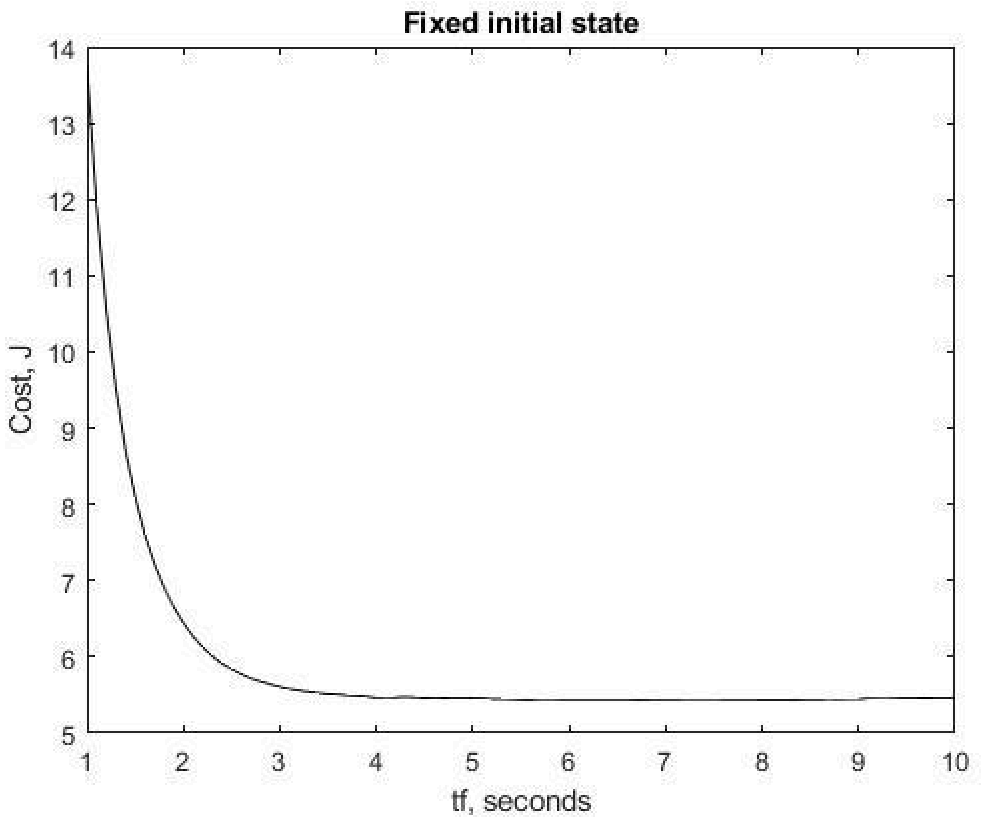
## Numerical solutions

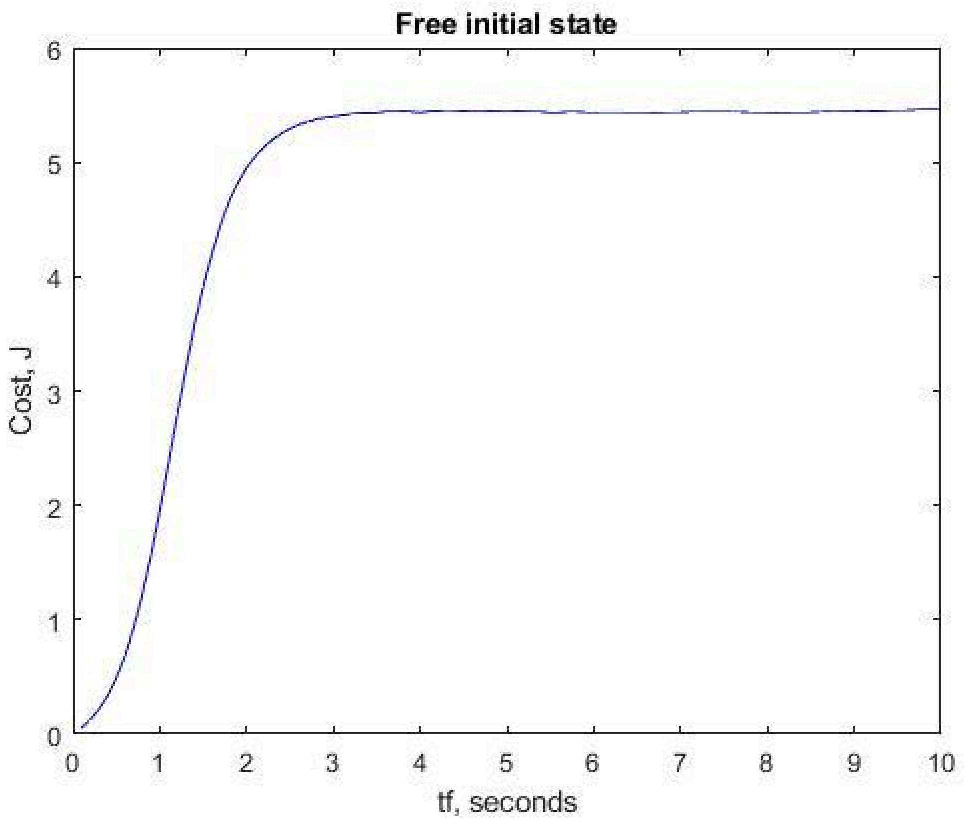
```
tic
for i = (1:100)
    tf = i/10;

    [TX,SX] = ode45(@(tx,sx)eom(tx,sx),linspace(0,tf,tf*10+1),
[10e-5,10e-5]);
    [TL,Lambda] = ode45(@(tx,sx)LagrangeEom(tx,sx,TX,SX),flip(TX),
[10*SX(1,end),0]);
    TL = flip(TL);
    Lambda = flip(Lambda);

    xg = [SX(:,1) SX(:,2) Lambda(:,1) Lambda(:,2)]';
    solinit.x = TX;
    solinit.y = xg;



    %BVP4C calling statement:
    sol = bvp4c(@(tx,sx)ELeom(tx,sx),@bcfunc,solinit);
    sol2 = bvp4c(@(tx,sx)ELeom(tx,sx),@bcfunc2,solinit);

    [TJ,SJ] =
 ode45(@(tj,sj)CostEom(tj,sj,sol.y,sol.x,sol2.x,sol2.y),linspace(0,tf,tf*100+1),
[0,0]);

    L2_int = SJ(end,1);
    L4_int = SJ(end,2);
    J2(i) = 5*sol.y(1,end)^2+0.5*L2_int;
    J4(i) = 5*sol2.y(1,end)^2+0.5*L4_int;

end

figure(1)
tf = (1:100)/10;
plot(tf(10:100),J2(10:100),'-k')
title("Fixed initial state")
xlabel('tf, seconds')
ylabel('Cost, J')
figure(2)
plot(tf,J4,'-b')
Time = toc
title("Free initial state")
xlabel('tf, seconds')
ylabel('Cost, J')
%Integrate states (x1,x2) forward in time
```

```
function ds = eom(~,sx)
% s(1) = x1, s(2) = x2
ds(1) = 1+sx(2);
ds(2) = sx(1)-sx(2);
ds = ds';
end
%Integrate Lagrange Multipliers backward in time
function ds = LagrangeEom(tx,sx,TX,SX)
% s(1) = lambda_x, s(2) = lambda_y
X = interp1(TX,SX,tx);
ds(1) = -X(1)-sx(2);
ds(2) = -X(2)-sx(1)+sx(2);
ds = ds';
end

function ds = CostEom(tj,~,y,x,x2,y2)
% s(1) = lambda_x, s(2) = lambda_y
X = interp1(x',y',tj);
X2 = interp1(x2',y2',tj);
ds(1) = X(1)^2+X(2)^2+X(3)^2;
ds(2) = X2(1)^2+X2(2)^2+X2(3)^2;
ds = ds';
end

% Euler-Lagrange Equations:
function ds = ELeom(~,sx)
% s(1) = x_1, s(2) = x_2, s(3) = Lambda_1, s(4) = Lambda_2
u = -sx(3); % optimal control law
ds(1) = sx(2)+u;
ds(2) = sx(1)-sx(2);
ds(3) = -(sx(1)+sx(4));
ds(4) = -(sx(2)+sx(3)-sx(4));
ds = ds';
end

% Boundary Condition Function
function con = bcfunc(so,sf)
% so is the initial states
% sf is the final states
con(1) = so(1); % x_1(0) = 0
con(2) = so(2); % x_2(0) = 0
con(3) = sf(2)-1; % x_2(tf) = 1
con(4) = sf(3)-10*sf(1); % Lambda_1(tf) = 10*x_1(tf)
end

% The other boundary Condition Function
function con = bcfunc2(so,sf)
% so is the initial states
% sf is the final states
con(1) = so(3); % Lambda_1(0) = 0
con(2) = so(4); % Lambda_2(0) = 0
con(3) = sf(2)-1; % x_2(tf) = 1
con(4) = sf(3)-10*sf(1); % Lambda_1(tf) = 10*x_1(tf)
end
```

```
% Based on the plots shown, both costs (fixed vs free initial states)
% approach an infinum of 5.461 as time approaches infinity. The fixed
% initial state function approaches infinity. The free initial state
% approaches zero as t approaches zero.
```

```
Time =

    7.0009
```





*Published with MATLAB® R2019a*