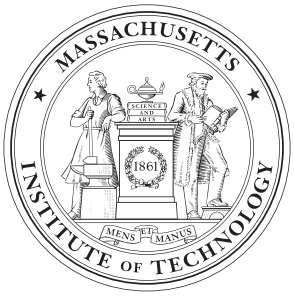
16.323 Optimal Control

PROBLEM SET 2

Due: Thursday, March 6, 2014



Author:
Daniel WIESE

Instructor:
Prof. Steven HALL

(1) Automatic Differentiation

The problem is to find the optimal controller for the inverted pendulum, with dynamics given by

$$\ddot{\theta}(t) = \sin(\theta(t)) + u$$

$\theta = 0$ corresponds to vertical. The cost functional to minimize is

$$J = \int_0^{t_f} (\theta^2 + \rho u^2) dt$$

For the purposes of this problem, we will take $t_f = 10$, $\rho = 10$.

1. Adjoint Method

The code was provided that integrates the given state equation using a second order Runge-Kutta solver. This function took as an input a control vector u over the specified simulation time, and integrated the equations of motion using this control input. The cost was evaluated and outputted from the function, as was the state trajectory x. The reverse accumulation adjoint method was used to write the function `pendulum_b.m` which contains both the forward code described above and the reverse code. The reverse code used automatic differentiation to return the derivative of the cost function with respect to the input u, in addition to the other outputs already described. The code can be found in the appendix.

2. Finding Optimal Control History

Using the function `pendulum_b.m` above, the gradient, which was obtained using automatic differentiation, can be passed to `fminunc.m` to determine the optimal control history u^*. This control history was then given to the pendulum to bring the pendulum to the upright position. Plots of the optimal control history and state history are shown below.

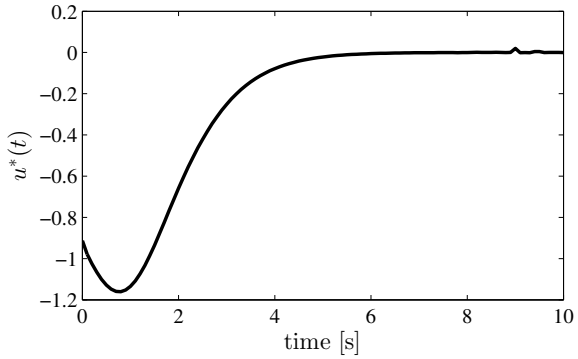


Figure 1: Optimal control input $u^*(t)$

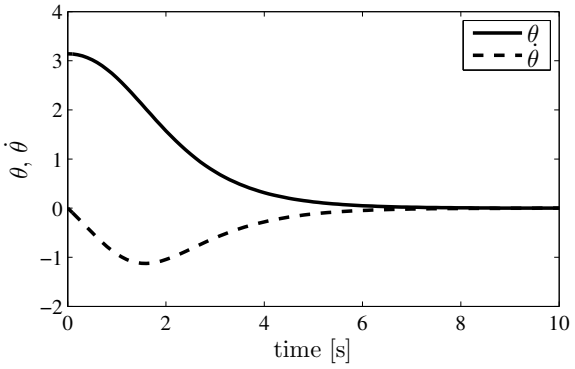


Figure 2: State history with optimal control input

(2) Discrete Time LQ-Tracking Problem

The discrete-time linear quadratic tracking problem is to minimize the cost

$$J = \sum_{k=0}^{N} \underbrace{\frac{1}{2}\left[(r_k - C_k x_k)^\top (r_k - C_k x_k) + u_k^\top R_k u_k\right]}_{g_k(x_k, u_k)}$$

subject to the constraint that the dynamics are

$$x_{k+1} = A_k x_k + B_k u_k$$

The goal is for the performance variables $z_k = C_k x_k$ to track the reference input r_k as closely as possible, using minimum control effort.

1. Use dynamic programming to find the control that minimizes the cost.

Define the terminal cost as

$$V_N(x_N) = \min_{u_N} g_N(x_N, u_N)$$

which is the cost associated with the the final state of the system. From the dynamics we can see that a control input u_N has no influence on the state x_N, and thus no influence on the terminal cost through the dynamics. Because of this, the optimal control $u_N = 0$, and the terminal cost can be written as

$$V_N(x_N) = \frac{1}{2}\left[(r_N - C_N x_N)^\top (r_N - C_N x_N)\right]$$

The cost to go at $N-1$ is the cost associated with going from $N-1$ to N, as well as the cost associated with state N. This can be written as

$$V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \left\{g_{N-1}(x_{N-1}, u_{N-1}) + V_N(x_N)\right\}$$

Inserting the expression for g_{N-1} and V_N this becomes

$$V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \left\{ \frac{1}{2}\left[(r_{N-1} - C_{N-1} x_{N-1})^\top (r_{N-1} - C_{N-1} x_{N-1}) + u_{N-1}^\top R_{N-1} u_{N-1}\right] + \frac{1}{2}\left[(r_N - C_N x_N)^\top (r_N - C_N x_N)\right] \right\}$$

Using the plant dynamics $x_N = A_{N-1} x_{N-1} + B_{N-1} u_{N-1}$ and simplifying the subscript notation, this becomes

$$V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \Bigg\{ \frac{1}{2} \Big[(r - Cx)^\top (r - Cx) + u^\top R u \Big]_{N-1} + \frac{1}{2} \Big[\big(r_N - C_N (A_{N-1} x_{N-1} + B_{N-1} u_{N-1}) \big)^\top \big(r_N - C_N (A_{N-1} x_{N-1} + B_{N-1} u_{N-1}) \big) \Big] \Bigg\}$$

Simplifying

$$V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \Bigg\{ \frac{1}{2} \Big[r^\top r - x^\top C^\top r - r^\top C x + x^\top C^\top C x + u^\top R u \Big]_{N-1} + \frac{1}{2} \Big[\big(r_N^\top - (Ax + Bu)_{N-1}^\top C_N^\top \big) \big(r_N - C_N (Ax + Bu)_{N-1} \big) \Big] \Bigg\}$$

Simplifying

$$V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \Bigg\{ \frac{1}{2} \Big[r^\top r - 2 r^\top C x + x^\top C^\top C x + u^\top R u \Big]_{N-1} + \frac{1}{2} \Big[r_N^\top r_N - (Ax + Bu)_{N-1}^\top C_N^\top r_N - r_N^\top C_N (Ax + Bu)_{N-1} + (Ax + Bu)_{N-1}^\top C_N^\top C_N (Ax + Bu)_{N-1} \Big] \Bigg\}$$

Simplifying

$$\begin{aligned} V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \Bigg\{ & \frac{1}{2} \Big[r^\top r - 2 r^\top C x + x^\top C^\top C x + u^\top R u \Big]_{N-1} \\ & + \frac{1}{2} \big[r_N^\top r_N - 2 (Ax + Bu)_{N-1}^\top C_N^\top r_N \\ & + x_{N-1}^\top A_{N-1} C_N^\top C_N A_{N-1} x_{N-1} + u_{N-1}^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\ & + x_{N-1}^\top A_{N-1} C_N^\top C_N B_{N-1} u_{N-1} + u_{N-1}^\top B_{N-1}^\top C_N^\top C_N B_{N-1} u_{N-1} \big] \Bigg\} \end{aligned}$$

Simplifying

$$\begin{aligned} V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \Bigg\{ & \frac{1}{2} \Big[r^\top r - 2 r^\top C x + x^\top C^\top C x + u^\top R u \Big]_{N-1} \\ & + \frac{1}{2} \big[r_N^\top r_N - 2 x_{N-1}^\top A_{N-1}^\top C_N^\top r_N - 2 u_{N-1}^\top B_{N-1}^\top C_N^\top r_N \\ & + x_{N-1}^\top A_{N-1} C_N^\top C_N A_{N-1} x_{N-1} + 2 u_{N-1}^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\ & + u_{N-1}^\top B_{N-1}^\top C_N^\top C_N B_{N-1} u_{N-1} \big] \Bigg\} \end{aligned}$$

And to minimize we evaluate $\frac{\partial}{\partial u_{N-1}} V_{N-1}(x_{N-1})$ as follows

$$\frac{\partial}{\partial u_{N-1}} V_{N-1}(x_{N-1}) = \frac{\partial}{\partial u_{N-1}} \frac{1}{2} \Big\{ u_{N-1}^\top R_{N-1} u_{N-1} - 2u_{N-1}^\top B_{N-1}^\top C_N^\top r_N \\ + 2u_{N-1}^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} + u_{N-1}^\top B_{N-1}^\top C_N^\top C_N B_{N-1} u_{N-1} \Big\}$$

Evaluating the derivative

$$\frac{\partial}{\partial u_{N-1}} V_{N-1}(x_{N-1}) = \Big\{ R_{N-1} u_{N-1} - B_{N-1}^\top C_N^\top r_N + B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} + B_{N-1}^\top C_N^\top C_N B_{N-1} u_{N-1} \Big\}$$

Simplifying

$$\frac{\partial}{\partial u_{N-1}} V_{N-1}(x_{N-1}) = (R_{N-1} + B_{N-1}^\top C_N^\top C_N B_{N-1}) u_{N-1} - (B_{N-1}^\top C_N^\top r_N - B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1})$$

And the optimal control input u_{N-1}^* is the control such that $\frac{\partial V_{N-1}}{\partial u_{N-1}} =$, with the second derivative positive. This gives

$$\boxed{u_{N-1}^* = (R_{N-1} + B_{N-1}^\top C_N^\top C_N B_{N-1})^{-1} (B_{N-1}^\top C_N^\top r_N - B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1})}$$

Which can be written as

$$\boxed{\begin{aligned} u_{N-1}^* &= G_N r_N - F_N x_{N-1} \\ G_N &= (R_{N-1} + B_{N-1}^\top C_N^\top C_N B_{N-1})^{-1} B_{N-1}^\top C_N^\top \\ F_N &= (R_{N-1} + B_{N-1}^\top C_N^\top C_N B_{N-1})^{-1} B_{N-1}^\top C_N^\top C_N A_{N-1} \end{aligned}}$$

Plugging this expression for the control law into $V_{N-1}(x_{N-1})$ we have

$$\begin{aligned} V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2} \Big\{ & \Big[r^\top r - 2r^\top C x + x^\top C^\top C x \Big]_{N-1} + (G_N r_N - F_N x_{N-1})^\top R_{N-1} (G_N r_N - F_N x_{N-1}) \\ & + r_N^\top r_N - 2x_{N-1}^\top A_{N-1}^\top C_N^\top r_N - 2(G_N r_N - F_N x_{N-1})^\top B_{N-1}^\top C_N^\top r_N \\ & + x_{N-1}^\top A_{N-1} C_N^\top C_N A_{N-1} x_{N-1} + 2(G_N r_N - F_N x_{N-1})^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\ & + (G_N r_N - F_N x_{N-1})^\top B_{N-1}^\top C_N^\top C_N B_{N-1} (G_N r_N - F_N x_{N-1}) \Big\} \end{aligned}$$

Simplifying

$$
\begin{aligned}
V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2}\Big\{ & \left[r^\top r - 2r^\top Cx + x^\top C^\top Cx\right]_{N-1} + (r_N^\top G_N^\top - x_{N-1}^\top F_N^\top) R_{N-1}(G_N r_N - F_N x_{N-1}) \\
& + r_N^\top r_N - 2x_{N-1}^\top A_{N-1}^\top C_N^\top r_N - 2(r_N^\top G_N^\top - x_{N-1}^\top F_N^\top) B_{N-1}^\top C_N^\top r_N \\
& + x_{N-1}^\top A_{N-1} C_N^\top C_N A_{N-1} x_{N-1} + 2(r_N^\top G_N^\top - x_{N-1}^\top F_N^\top) B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\
& + (r_N^\top G_N^\top - x_{N-1}^\top F_N^\top) B_{N-1}^\top C_N^\top C_N B_{N-1}(G_N r_N - F_N x_{N-1}) \Big\}
\end{aligned}
$$

Simplifying

$$
\begin{aligned}
V_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2}\Big\{ & \left[r^\top r - 2r^\top Cx + x^\top C^\top Cx\right]_{N-1} + r_N^\top G_N^\top R_{N-1} G_N r_N + x_{N-1}^\top F_N^\top R_{N-1} F_N x_{N-1} \\
& - 2r_N^\top G_N^\top R_{N-1} F_N x_{N-1} \\
& + r_N^\top r_N - 2r_N^\top C_N A_{N-1} x_{N-1} - 2r_N^\top G_N^\top B_{N-1}^\top C_N^\top r_N + 2x_{N-1}^\top F_N^\top B_{N-1}^\top C_N^\top r_N \\
& + x_{N-1}^\top A_{N-1} C_N^\top C_N A_{N-1} x_{N-1} + 2r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\
& - 2x_{N-1}^\top F_N^\top B_{N-1}^\top C_N^\top C_N A_{N-1} x_{N-1} \\
& + r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} G_N r_N - 2r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} F_N x_{N-1} \\
& + x_{N-1}^\top F_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} F_N x_{N-1} \Big\}
\end{aligned}
$$

This cost-to-go function can be expressed as

$$
\begin{aligned}
V_{N-1}(x_{N-1}) &= \frac{1}{2} x_{N-1}^\top P_{N-1} x_{N-1} + q_{N-1} x_{N-1} + s_{N-1} \\
P_{N-1} &= \frac{1}{2}(C_{N-1}^\top C_{N-1} + F_N^\top R_{N-1} F_N + A_{N-1}^\top C_N^\top C_N A_{N-1} - 2F_N^\top B_{N-1}^\top C_N^\top C_N A_{N-1} + F_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} F_N) \\
q_{N-1} &= -r_{N-1}^\top C_{N-1} - r_N^\top G_N^\top R_{N-1} F_N - r_N^\top C_N A_{N-1} + r_N^\top C_N B_{N-1} F_N + r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N A_{N-1} - r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} F_N \\
s_{N-1} &= \frac{1}{2}(r_{N-1}^\top r_{N-1} + r_N^\top G_N^\top R_{N-1} G_N r_N + r_N^\top r_N - 2r_N^\top G_N^\top B_{N-1}^\top C_N^\top r_N + r_N^\top G_N^\top B_{N-1}^\top C_N^\top C_N B_{N-1} G_N r_N)
\end{aligned}
$$

Using induction, this cost-to-go can be applied to any time k, replacing the index N with k. That is, from $V_N(x_N)$ we have

$$
\begin{aligned}
P_N &= \frac{1}{2} C_N^\top C_N \\
q_N &= -r_N^\top C_N \\
s_N &= \frac{1}{2} r_N^\top r_N
\end{aligned}
$$

And P_k, G_k, and F_k are independent of the state and can be computed offline as

$$
\begin{aligned}
V_k(x_k) &= \frac{1}{2} x_k^\top P_k x_k + q_k x_k + s_k \\
P_k &= \frac{1}{2}\left(C_k^\top C_k + F_{k+1}^\top R_k F_{k+1} + \left(A_k^\top - F_{k+1}^\top B_k^\top\right) P_{k+1} \left(A_k - B_k F_{k+1}\right)\right) \\
q_k &= -r_k^\top C_k - r_{k+1}^\top G_{k+1}^\top R_k F_{k+1} - q_{k+1}(-A_k + B_k F_{k+1}) + r_{k+1}^\top G_{k+1}^\top B_k^\top C_{k+1}^\top C_{k+1} A_k - r_{k+1}^\top G_{k+1}^\top B_k^\top C_{k+1}^\top C_{k+1} B_k F_{k+1} \\
s_k &= \frac{1}{2}(r_k^\top r_k + r_{k+1}^\top G_{k+1}^\top R_k G_{k+1} r_{k+1} + 2 s_{k+1} - 2 r_{k+1}^\top G_{k+1}^\top B_k^\top C_{k+1}^\top r_{k+1} + r_{k+1}^\top G_{k+1}^\top B_k^\top C_{k+1}^\top C_{k+1} B_k G_{k+1} r_{k+1})
\end{aligned}
$$

$$
\boxed{u_k^* = (R_k + B_k^\top C_{k+1}^\top C_{k+1} B_k)^{-1}(B_k^\top C_{k+1}^\top r_{k+1} - B_k^\top C_{k+1}^\top C_{k+1} A_k x_k)}
$$

2. Describe how this control law would be implemented

This control law would be implemented by calculating the gains F_k and G_k offline and storing them. The control law would then be applied as shown above, using state feedback and the feedforward component on the reference input.

3. Explain why this control law is not practical

This control law is impractical because calculating the optimal control gains depends on knowledge of the future reference signal. That is, if the reference input changes, the current calculated gains F_k and G_k will no longer be optimal.

(3) Dynamic Programming Distance Problem

I did this problem by hand on the given problem set assignment. Working backwards from M to A, and taking the shortest path at each segment, found a shortest path of 406.

(6) Riccati

Jacopo Francesco Riccati was an Italian mathematician who was born in Venice in 1676. Most of his work dealt with differential equations, and there is one form of a differential equation which bears his name. The so called Riccati equation is a first order ODE which is quadratic in the unknown function. More generally, this name has been used to refer to matrix equations with such a quadratic term. Such an equation appears frequently in optimal control and estimation problems, where the differential equation is quadratic in a positive definite matrix P. Few problems have analytic solutions to this differential equation, but in control applications the steady-state version of the problem can be considered and solved more easily. Riccati's contributions are greatly important to modern control, with optimal control methods such as the linear quadratic regulator being very pervasive in many applications today.

Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Dan Wiese
% 16.323 - HW#2
% Problem_1.m
%-------------------------------------------------------------------------------
% Pendulum master script
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
close all;
clc;

tf=10;
dt=0.1;
rho=10;
x0=[pi; 0];
u=ones(tf/dt+1,1);
t=0:dt:tf;

options=optimset('GradObj', 'on','Hessian', 'off');

[U,Jmin,exitflag,output,grad,hessian]=fminunc(@pendulum_b,u,options);

[¬,¬,xcl]=pendulum_b(U);

%%

sizefont=14;
ticklabelfontsize=12;

figure(1)
plot(t,U,'color',[0 0 0],'linewidth',2)
set(gcf,'Units','pixels');
set(gcf,'PaperUnits','inches','PaperPosition',[0 0 5 3]);
set(gcf,'PaperPositionMode','manual')
set(gcf,'InvertHardCopy','off');
set(gcf,'color',[1 1 1])
xlabel('time [s]','interpreter','latex','FontSize',sizefont)
ylabel('$u^{*}(t)$','interpreter','latex','FontSize',sizefont)
set(gca,'fontsize',ticklabelfontsize);
set(gca,'fontname','times');
print('-depsc','../Figures/prob1_control.eps');

figure(2)
plot(t,xcl(:,1),'color',[0 0 0],'linewidth',2)
hold on
plot(t,xcl(:,2),'color',[0 0 0],'linewidth',2,'linestyle','--')
legend('$\theta$','$\dot{\theta}$','Location','NorthEast')
temphandle=legend;
set(temphandle,'interpreter','latex','FontSize',sizefont,'box','on')
set(gcf,'Units','pixels');
set(gcf,'PaperUnits','inches','PaperPosition',[0 0 5 3]);
set(gcf,'PaperPositionMode','manual')
set(gcf,'InvertHardCopy','off');
```

```
set(gcf,'color',[1 1 1])
xlabel('time [s]','interpreter','latex','FontSize',sizefont)
ylabel('$\theta$, $\dot{\theta}$','interpreter','latex','FontSize',sizefont)
set(gca,'fontsize',ticklabelfontsize);
set(gca,'fontname','times');
print('-depsc','../Figures/prob1_state.eps');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Dan Wiese
% 16.323 - HW#2
% pendulum_b.m
%-------------------------------------------------------------------------------
% Pendulum dynamics forward code and adjoint method reverse code. This function
% takes as an input the vector u, the control input for the entire time
% interval.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [cost,ub,x]=pendulum_b(u)
%Read in initial condition, Δ t, and weight rho from workspace
x0=evalin('base','x0');
dt=evalin('base','dt');
rho=evalin('base','rho');
%Total length of time vector
n=length(u);
%Preallocate the vector x
x1=zeros(n,1);
x2=zeros(n,1);
%Overwrite the first entry of x1 and x2 vector with initial condition
x1(1)=x0(1);
x2(1)=x0(2);
J=zeros(n,1);
%preallocate ks
k11=0;
k12=0;
k21=0;
k22=0;
%Preallocate kbs
k11b=0;
k12b=0;
k21b=0;
k22b=0;
%Preallocate more stuff
Jb=ones(n,1);
ub=zeros(n,1);
x1b=zeros(n,1);
x2b=zeros(n,1);
%-------------------------------------------------------------------------------
% FORWARD CODE
% This code uses RK4 to integrate forward the state x in time, as well as the
% cost at each moment in time.
for ii=1:n-1;
    k11=dt*x2(ii);
    k12=dt*(sin(x1(ii))+u(ii));
```

```
    k21=dt*(x2(ii)+0.5*k12);
    k22=dt*(sin(x1(ii)+0.5* k11)+(u(ii)+u(ii+1))/2);
    x1(ii+1)=x1(ii)+k21;
    x2(ii+1)=x2(ii)+k22;
    J(ii+1)=J(ii)+(x1(ii)^2+x1(ii+1)^2)*dt/2+rho*(u(ii)^2+u(ii+1)^2)*dt/2;
end
x=[x1, x2];
%-------------------------------------------------------------------------------
% REVERSE CODE
for ii=n-1:-1:1
%LINE 7: J(ii+1)=J(ii)+(x1(ii)^2+x1(ii+1)^2)*dt/2+rho*(u(ii)^2+u(ii+1)^2)*dt/2;
x1b(ii)=x1b(ii)+x1(ii)*dt*Jb(ii+1);
x1b(ii+1)=x1b(ii+1)+x1(ii+1)*dt*Jb(ii+1);
ub(ii)=ub(ii)+rho*u(ii)*dt*Jb(ii+1);
ub(ii+1)=ub(ii+1)+rho*u(ii+1)*dt*Jb(ii+1);
Jb(ii)=1;

%LINE 6: x2(ii+1)=x2(ii)+k22;
x2b(ii)=x2b(ii)+x2b(ii+1);
k22b=k22b+x2b(ii+1);

%LINE 5: x1(ii+1)=x1(ii)+k21;
x1b(ii)=x1b(ii)+x1b(ii+1);
k21b=k21b+x1b(ii+1);
%clear variables
x2b(ii+1)=0;
x1b(ii+1)=0;

%LINE 4: k22=dt*(sin(x1(ii)+0.5* k11)+(u(ii)+u(ii+1))/2);
x1b(ii)=x1b(ii)+cos(x1(ii)+0.5*k11)*dt*k22b;
k11b=k11b + cos(x1(ii)+0.5*k11)*0.5*dt*k22b;
ub(ii)=ub(ii)+0.5*dt*k22b;
ub(ii+1)=ub(ii+1)+0.5*dt*k22b;
%clear variables
k22b=0;

%LINE 3: k21=dt*(x2(ii)+0.5*k12);
x2b(ii)=x2b(ii)+dt*k21b;
k12b=k12b+0.5*dt*k21b;
%clear variables
k21b=0;

%LINE 2: k12=dt*(sin(x1(ii))+u(ii));
x1b(ii)=x1b(ii)+cos(x1(ii))*dt*k12b;
ub(ii)=ub(ii)+dt*k12b;
%clear variables
k12b=0;

%LINE 1: k11=dt*x2(ii);
x2b(ii)=x2b(ii)+dt*k11b;
%clear variables
k11b=0;
end
cost=J(end);
end
```