ECE-597: Optimal Control
Homework #7

Due: Thursday November 1, 2007

1. In this problem we will use the Matlab routine *fopt.m* to solve a minimum time problem. Specifically, consider the problem

$$\begin{aligned} \text{minimize } J &= t_f \\ \text{subject to } & \quad \dot{x}(t) = \cos(u) - y(t) \\ & \quad \dot{y}(t) = \sin(u) \\ & \quad x(0) = 3.659, \ y(0) = -1.864 \\ & \quad x(t_f) = 0, \ y(t_f) = 0 \end{aligned}$$

What makes this routine so much fun is that (1) you need to make an initial guess for the final time, and (2) the routine seems quite sensitive to the initial guess for the control sequence. Initially use the routine with $k = 1e-5$, $told = 5e-5$, $tols = 5e-5$, $mxit = 25$, $N = 100$ time steps, and an initial guess of final time $t_f = 10$ with the initial control sequence all zeros.

However, the initial guess of the optimal final time of $t_f = 10$ is actually too large. The true optimal final time is between 4 and 8 seconds. You are to try to find an initial guess of the control sequence and optimal time to produce a good estimate of the final time. You need to try at least four different sets of initial estimates. You are expected to work *alone* on this part. I don't want everybody to do the same thing.

For each of your simulations, you need to plot $u(t)$ vs. t and $y(t)$ vs. $x(t)$ on the same graph. One of your titles should indicate the estimated final time.

2. From *Bryson*. A rocket is launched with velocity u_0 parallel to the x-axis, v_0 parallel to the y-axis, with a constant thrust specific force a. The system starts from $x(0) = y(0) = 0$. We want to find $\theta(i)$ to minimize the final time and require $x(N) = x_f$ and $y(N) = y_f$. The equations of motion are:

$$\begin{aligned} u(i+1) &= u(i) + a\Delta T \cos(\theta(i)) \\ v(i+1) &= v(i) + a\Delta T \sin(\theta(i)) \\ y(i+1) &= y(i) + \Delta T v(i) + \frac{a}{2}[\Delta T]^2 \sin(\theta(i)) \\ x(i+1) &= x(i) + \Delta T u(i) + \frac{a}{2}[\Delta T]^2 \cos(\theta(i)) \end{aligned}$$

a) Show that dimensionless equations of motion are:

$$\begin{aligned} u(i+1) &= u(i) + \Delta T \cos(\theta(i)) \\ v(i+1) &= v(i) + \Delta T \sin(\theta(i)) \\ x(i+1) &= x(i) + \Delta T u(i) + \frac{1}{2}[\Delta T]^2 \cos(\theta(i)) \\ y(i+1) &= y(i) + \Delta T v(i) + \frac{1}{2}[\Delta T]^2 \sin(\theta(i)) \end{aligned}$$

where time is measured in units of t_f, (u, v) in units of at_f, and (x, y, x_f, y_f) in units of at_f^2.

b) Show that

$$\begin{aligned}
\lambda_x &= \nu_x \\
\lambda_y &= \nu_y \\
\lambda_u(i) &= (N-i)\Delta T \nu_x \\
\lambda_v(i) &= (N-i)\Delta T \nu_y
\end{aligned}$$

c) Using the optimality condition, show that

$$\tan(\theta(i)) = \frac{\nu_y}{\nu_x}$$

which means $\theta(i) = \theta_0$, a constant.

d) By iterating forward (at least through $i = 3$), show that

$$\begin{aligned}
u(i) &= u_0 + i\Delta T \cos(\theta_0) \\
v(i) &= v_0 + i\Delta T \sin(\theta_0) \\
x(i) &= i\Delta T u_0 + \frac{1}{2}[i\Delta T]^2 \cos(\theta_0) \\
y(i) &= i\Delta T v_0 + \frac{1}{2}[i\Delta T]^2 \sin(\theta_0)
\end{aligned}$$

e) Determine the two nonlinear equations that need to be solved for Δ and θ_0 to complete the solution to this problem.

3. From *Bryson.* A bead slides on a wire without friction in a gravitational field. $\gamma(t)$ is the angle with respect to the horizontal. The equations of motion are

$$\begin{aligned}
\dot{V}(t) &= g\sin(\gamma(t)) \\
\dot{x}(t) &= V(t)\cos(\gamma(t)) \\
\dot{y}(t) &= V(t)\sin(\gamma(t))
\end{aligned}$$

We have $x(0) = y(0) = V(0) = 0$ and we want to reach the final point $x(t_f) = x_f$, $y(t_f) = y_f$ in the minimum time (make t_f as small as possible).

a) Show that the dimensionless equations of motion are given by

$$\begin{aligned}
\dot{V}(t) &= \sin(\gamma(t)) \\
\dot{x}(t) &= V(t)\cos(\gamma(t)) \\
\dot{y}(t) &= V(t)\sin(\gamma(t))
\end{aligned}$$

where (x, y) are measured in terms of x_f, t is measured in terms of $\sqrt{\frac{x_f}{g}}$, and V is measured in terms of $\sqrt{gx_f}$.

b) Show that

$$\begin{aligned}\lambda_x &= \nu_x \\ \lambda_y &= \nu_y\end{aligned}$$

c) Show that $H = -1$.

d) Using the optimality condition, show that

$$\lambda_v(t) = V(t)[\nu_x \tan(\gamma(t)) - \nu_y]$$

We have the initial condition $V(0) = 0$. From the optimality condition, this means $\cos(\gamma(0)) = 0$. Let's choose $\gamma(0) = \frac{\pi}{2}$ (This is one of many possible choices.)

e) Now substitute the equation for $\lambda_v(t)$ into your expression for H, and use the fact that $H = -1$ to show

$$V(t)\nu_x \sec(\gamma(t)) = -1$$

f) Taking the derivative of the expression you found in part **e**, and using the fact that $\dot{V}(t) = \sin(\gamma(t))$ (since we are using the dimensionless equations) and $-1 = V(t)\nu_x \sec(\gamma(t))$, show that

$$\gamma(t) = \frac{\pi}{2} + \nu_x t$$

g) Starting from $\dot{V}(t) = \sin(\gamma(t))$, your answer for part **f**, and some simple trigonometric identities, show that

$$V(t) = \frac{1}{\nu_x} \sin(\nu_x t)$$

h) Starting from $\dot{x}(t) = V(t)\cos(\gamma(t))$ and $\dot{y}(t) = V(t)\sin(\gamma(t))$ and your answer for part **f**, show that

$$\begin{aligned}x(t) &= \frac{-1}{2\nu_x}(t - \frac{1}{2\nu_x} \sin(2\nu_x t)) \\ y(t) &= \frac{1}{4\nu_x^2}(1 - cos(2\nu_x t))\end{aligned}$$

i) Determine the two nonlinear equations that must be solved for ν_x and t_f to finally solve this problem.

Appendix

Continuous-Time Problems with Open Final Time using **fopt.m**

We will be utilizing the routine **fopt.m** to solve continuous-time optimization problems of the form:

Find the input $u(t), \; t_0 \leq t \leq t_f$, and final time t_f to minimize

$$J = \phi[x(t_f), t_f]$$

subject to the constraints

$$\begin{aligned} \dot{x}(t) &= f[x(t), u(t), t] \\ \psi[x(t_f), t_f] &= 0 \\ x(0) &= x_0 \; (known) \end{aligned}$$

In order to use the routine **fopt.m**, you need to write a routine that returns one of three things depending on the value of the variable **flg**. The general form of your routine will be as follows:

```
function [f1,f2,f3] = bobs_fopt(u,s,t,flg)
```

Here u is the current input, $u(t)$, and s ($s(t)$) contains the current state, so $\dot{s}(t) = f(s(t), u(t), t)$. Your routine should compute the following:

if **flg** $= 1$ f1 $= \dot{s}(t) = f(s(t), u(t), t)$
if **flg** $= 2$ f1 $= \Phi$, f2 $= \Phi_s$, f3 $= \Phi_t$
if **flg** $= 3$ f1 $= f_s$, f2 $= f_u$

Note: Φ_t is just the partial derivative with respect to t, not the total derivative.
An example of the usage is:

```
[tu,ts,tf,nu,la0] = fopt('bobs_fopt',tu,tf,s0,k,told,tols,mxit,eta)
```

The (input) arguments to **fopt.m** are the following:

- the function you just created (in single quotes).
- tu is an initial guess of times (first column), and control values (subsequent columns) that minimizes J. If there are multiple control signals at a given time, they are all in the same row. Note that these are just the initial time and control values, the times and control values will be modified as the program runs. The initial time should start at zero.
- the initial states, $s0$. Note that you must include and initial guess for the "cumulative" state q also.

- the final time, tf. *This is just and initial guess!*
- k, the step size parameter, $k > 0$ to minimize. Often you need to play around with this one.
- *told*, the tolerance (a stopping parameter) for ode23 (differential equation solver for Matlab)
- *tols*, the tolerance (a stopping parameter), when $|\Delta u| < tols$ between iterations, the programs stops.
- *mxit*, the maximum number of iterations to try.
- *eta*, where 0 < eta ≤ 1, and d(psi) = -eta*psi is the desired change in the terminal constraints on the next iteration. *This is an optional input.*

fopt.m returns the following:

- *tu* the optimal input sequence and corresponding times. The first column is the time, the corresponding columns are the control signals. All entries in one row correspond to one time.
- *ts* the states and corresponding times. The first column is the time, the corresponding columns are the states. All entries in one row correspond to the same time. Note that the times in *tu* and the times in *ts* may not be the same, and they may not be evenly spaced.
- t_f, the optimal final time
- *nu*, the Lagrange multipliers on psi
- *la*0 the Lagrange multipliers

It is usually best to start with a small number of iterations, like 5, and see what happens as you change k. Start with small values of k and gradually increase them. It can be very difficult to make this program converge, especially if your initial guess is far away from the true solution.

Note!! If you are using the **fopt.m** file, and you use the maximum number of allowed iterations, assume that the solution has NOT converged. You must usually change the value of k and/or increase the number of allowed iterations. Do not set tol to less than about 5e-5. Also try to make k as large as possible and still have convergence.

Example A Given a rocket engine with maximum constant thrust T, operating for a time t_f, we want to find the thrust direction history $\theta(t)$ to transfer a spacecraft from a given initial circular orbit to the largest possible circular orbit. r is the radial distance of the space craft from the attracting center, u = radial component of velocity, v = tangential component of velocity, m = mass of spacecraft, $-\dot{m}$ = fuel consumption rate, μ = gravitational constant of attracting center. Assume we measure time in units of $\sqrt{r_o^3/\mu}$, r in units of r_o, u and v in units of $\sqrt{\mu/r_o}$, m in units of m_o, and thrust in units of $\mu m_o/r_o^2$, the problem can be stated as:

Given $r(0) = 1$, $u(0) = 0$, $v(0) = 1$, $\theta(0) = 0$, $u(t_f) = 0$, $v(t_f) = 1/\sqrt{r_f)}$ and $a = \frac{T}{1-|\dot{m}|t}$, find $\beta(t)$ to achieve these results while minimize t_f.

The dimensionless equations of motion are:

$$
\begin{aligned}
\dot{r} &= u \\
\dot{u} &= \frac{v^2}{r} - \frac{1}{r^2} + a\sin(\beta) \\
\dot{v} &= -\frac{uv}{r} + a\cos(\beta) \\
\dot{\theta} &= \frac{v}{r}
\end{aligned}
$$

Here we will assume $T = 0.1405$, $\dot{m} = 0.07489$, $r_f = 1.5237$.

Let's define the state vector as

$$
s = \begin{bmatrix} r \\ u \\ v \\ \theta \end{bmatrix}
$$

and let's define

$$
\Phi[s(t_f), t_f] = \begin{bmatrix} \phi[s(t_f), t_f] \\ \psi[s(t_f), t_f] \end{bmatrix}
$$

For this problem we want to **minimize** t_f, so we have

$$
\phi[s(t_f), t_f] = t_f
$$

We also have the hard terminal constraints

$$
\psi[s(t_f), t_f] = \begin{bmatrix} r - r(t_f) \\ u \\ v - 1/\sqrt{r_f} \end{bmatrix}
$$

so we can write

$$
\Phi[s(t_f), t_f] = \begin{bmatrix} t_f \\ r - r_f \\ u \\ v - 1/\sqrt{r_f} \end{bmatrix}
$$

We can then write

$$
f = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} u \\ \frac{v^2}{r} - \frac{1}{r^2} + a\sin(\theta) \\ -\frac{uv}{r} + a\cos(\theta) \\ \frac{v}{r} \end{bmatrix}
$$

Next we need

$$
\begin{aligned}
\Phi_{s(t_f)}[s(t_f), t_f] &= \begin{bmatrix} \phi_{s(t_f)}[s(t_f), t_f] \\ \psi_{s(t_f)}[s(t_f), t_f] \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial \phi[s(t_f),t_f]}{\partial r(t_f)} & \frac{\partial \phi[s(t_f),t_f]}{\partial u(t_f)} & \frac{\partial \phi[s(t_f),t_f]}{\partial v(t_f)} & \frac{\partial \phi[s(t_f),t_f]}{\partial \theta(t_f)} \\ \frac{\partial \psi[s(t_f,t_f)]}{\partial r(t_f)} & \frac{\partial \psi[s(t_f),t_f]}{\partial u(t_f)} & \frac{\partial \psi[s(t_f),t_f]}{\partial v(t_f)} & \frac{\partial \psi[s(t_f),t_f]}{\partial \theta(t_f)} \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\end{aligned}
$$

and

$$
\begin{aligned}
\Phi_{t_f}[s(t_f), t_f] &= \begin{bmatrix} \frac{\partial \phi[s(t_f),t_f]}{\partial t_f} \\ \frac{\partial \psi[s(t_f),t_f]}{\partial t_f} \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\end{aligned}
$$

Then we need

$$
\begin{aligned}
f_s &= \begin{bmatrix} \frac{\partial f}{\partial r(t)} & \frac{\partial f}{\partial u(t)} & \frac{\partial f}{\partial u(t)} & \frac{\partial f}{\partial \theta(t)} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial r(t)} & \frac{\partial f_1}{\partial u(t)} & \frac{\partial f_1}{\partial v(t)} & \frac{\partial f_1}{\partial \theta(t)} \\ \frac{\partial f_2}{\partial r(t)} & \frac{\partial f_2}{\partial u(t)} & \frac{\partial f_2}{\partial v(t)} & \frac{\partial f_2}{\partial \theta(t)} \\ \frac{\partial f_3}{\partial r(t)} & \frac{\partial f_3}{\partial u(t)} & \frac{\partial f_3}{\partial v(t)} & \frac{\partial f_3}{\partial \theta(t)} \\ \frac{\partial f_4}{\partial r(t)} & \frac{\partial f_4}{\partial u(t)} & \frac{\partial f_4}{\partial v(t)} & \frac{\partial f_4}{\partial \theta(t)} \end{bmatrix} \\
&= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{v^2}{r^2} + \frac{2}{r^3} & 0 & \frac{2v}{r} & 0 \\ \frac{uv}{r^2} & \frac{-v}{r} & \frac{-u}{r} & 0 \end{bmatrix}
\end{aligned}
$$

and finally

$$
f_u = f_\beta = \begin{bmatrix} \frac{\partial f_1}{\partial \theta(t)} \\ \frac{\partial f_2}{\partial \theta(t)} \\ \frac{\partial f_3}{\partial \theta(t)} \\ \frac{\partial f_4}{\partial \theta(t)} \end{bmatrix} = \begin{bmatrix} 0 \\ a\cos(\beta(t)) \\ -a\sin(\beta(t)) \\ 0 \end{bmatrix}
$$

This is implemented in the routine **bobs_fopt_a.m** on the class web site, and it is run using the driver file **fopt_example_a.m**.