MA5702 Laboratorio de Control Óptimo. Octubre 2009-2
Profesor: Héctor Ramírez C. **Auxiliares:** Julio Backhoff, Oscar Peredo.

Laboratorio #6
Pontryagin

23 Octubre de 2009

Descripción

El objetivo de este laboratorio es utilizar el principio del máximo de Pontryagin para un problema aplicado y resolverlo numericamente utilizando una aplicación de Matlab apropiada.

Parte A. Modelo

Supongamos que poseemos una empresa que negocia comerciando una única materia prima (producto). La empresa puede comprar o vender este producto, de esta manera su riqueza proviene de almacenarlo y/o venderlo. Sean $x(t)$ el dinero e $y(t)$ la cantidad del producto que posee la empresa en el tiempo t, respectivamente. Supongamos que dentro del horizonte finito T conocemos el precio $q(t)$ del producto en el tiempo t además del costo unitario λ de almacenar una unidad del producto.

La empresa puede controlar la tasa a la que vende o compra el bien, $u(t)$, siendo esta negativa si se vende y positiva de lo contrario.

Con todo esto el problema de la empresa es maximizar la cantidad de dinero en el tiempo T, es decir, la suma del dinero que posee con el que tiene almacenado en forma de producto en el tiempo T.

Podemos suponer que la tasa de compra/venta está acotada, digamos, $|u(t)| \leq C$.

De lo anterior se desprende entonces que el problema de Control Óptimo planteado es:

$$\begin{aligned} \text{Maximizar} \quad & x(T) + q(T)y(T) && (1)\\ \text{Sujeto a:} \quad & \dot{x}(t) = -\lambda y(t) - q(t)u(t) && (2)\\ & \dot{y}(t) = u(t) && (3)\\ & x(0) = x_0, y(0) = y_0, |u(t)| \leq C && (4) \end{aligned}$$

Ejercicio 1. Justifique el modelo planteado.

Parte B. Principio del Máximo de Pontryagin (PMP)

Ejercicio 2. Utilizando las ecuaciones asociadas a las variables duales (p_1, p_2) del PMP, demuestre que las variables duales satisfacen:

$$\begin{cases} p_1(t) = 1 \\ p_2(t) = \lambda(t - T) + q(T) \end{cases} \tag{5}$$

Observe que en este caso se trata del principio del Máximo de Pontryagin, así que debe ajustar las condiciones estudiadas en el curso de control óptimo.

Ejercicio 3. Utilizando la condición de optimalidad del Hamiltoniano demuestre que el control $u(t)$ es de tipo bang-bang, es decir:

$$u(t) = \begin{cases} C & \text{si } q(t) < p_2(t) \\ -C & \text{si } q(t) > p_2(t) \end{cases} \tag{6}$$

Parte C. DYNOPT

Cabe destacar que en la red no existen muchas implementaciones de métodos numéricos para la resolución de problemas de control óptimo, y de las que están disponibles, la mayoría requieren un pago por licencia. La implementación que se utilizará se llama DYNOPT [1] y se puede descargar de manera gratuita desde el link:

`http://www.kirp.chtf.stuba.sk/publication_info.php?id_pub=271`

Para utilizarla, debe tener MATLAB instalado con el Toolbox de Optimización.

Discretización de un problema de control óptimo

Existen varios métodos númericos para resolver un problema de control óptimo. En particular se utilizará el *método simultáneo directo* desarrollado en DYNOPT. Este método resuelve problemas de tipo Mayer, donde el funcional a minimizar viene dado por

$$C[u(\cdot)] = \phi(x(T), \mathbf{p}, T) \tag{7}$$

donde la variable $\mathbf{p}$ representa los parámetros independientes del tiempo a considerar. Suponemos que la dinámica del estado:

$$M\dot{x}(t) = f(x(t), u(t), \mathbf{p}, t) \tag{8}$$
$$x(t_0) = x_0(\mathbf{p}) \tag{9}$$

La matríz M se denomina matríz de masa y las condiciones iniciales dependen del vector de parámetros $\mathbf{p}$. Además de la dinámica, podemos incluir restricciones de igualdad y desigualdad sobre el estado y el control:

$$h(x(t), u(t), \mathbf{p}, t) = 0 \tag{10}$$
$$g(x(t), u(t), \mathbf{p}, t) \leq 0 \tag{11}$$

Las funciones f, g y h deben ser suficientemente regulares. Con esto, el problema a resolver es el siguiente:

$$\begin{array}{cc} \mathbf{min}_{\mathbf{u}(\cdot),\mathbf{p}} & \phi(\mathbf{x}(T), \mathbf{p}, T) \\ \mathbf{s.a} & M\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t) \\ & \mathbf{x}(t_0) = \mathbf{x}_0(\mathbf{p}) \\ & h(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t) = \mathbf{0} \\ & g(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t) \leq \mathbf{0} \\ & \mathbf{x}(t)^L \leq \mathbf{x}(t) \leq \mathbf{x}(t)^U \\ & \mathbf{u}(t)^L \leq \mathbf{u}(t) \leq \mathbf{u}(t)^U \\ & \mathbf{p}^L \leq \mathbf{p} \leq \mathbf{p}^U \end{array}$$

Para discretizar, se utilizan polinomios de Lagrange de grado $K_x + 1$ aproximando al estado y K_u aproximando al control sobre una malla unidimensional $t_0 = \zeta_1 < \zeta_2 < ... < \zeta_{NE} < \zeta_{NE+1} = T$:

$$x_{K_x}(t) = \sum_{j=0}^{K_x} x_{ij} l_j(t), \qquad \text{sobre el segmento } \zeta_i \leq t \leq \zeta_{i+1}, i = 1, ..., NE \tag{12}$$

$$u_{K_u}(t) = \sum_{j=1}^{K_u} u_{ij} \theta_j(t), \qquad \text{sobre el segmento } \zeta_i \leq t \leq \zeta_{i+1}, i = 1, ..., NE \tag{13}$$

con

$$l_j(t) = \prod_{k=0, k\neq j}^{K_x} \frac{t - t_{ik}}{t_{ij} - t_{ik}} \tag{14}$$

$$\theta_j(t) = \prod_{k=1, k\neq j}^{K_u} \frac{t - t_{ik}}{t_{ij} - t_{ik}} \tag{15}$$

Para revisar la discretización de la dinámica y de las otras restricciones, ver [1]. Finalmente se contruye un gran problema de optimización no lineal en las variables $x_{ij}, u_{ij}, \Delta\zeta_i$ y $\mathbf{p}$, el cual se puede resolver utilizando el Toolbox de Optimización de MATLAB, utilizando especificamente la función `fmincon`, que utiliza el método SQP (*Sequentially Quadratic Programming*) visto en el curso de Optimización No Lineal, para problemas medianos y pequeños.

El uso del toolbox **DYNOPT** se explica en el apéndice.

Ejercicio 4. Escriba los archivos `objfun.m`, `process.m` y `confun.m`, necesarios para el uso de **DYNOPT**, para el problema (1)-(2)-(3)-(4).

Parte D. Resolución numérica

En los siguientes ejercicios, considere los siguientes parámetros:

$$T = 1,\ x_0 = 0{,}6,\ y_0 = 0{,}4,\ C = 2,\ q(t) = \sin(\pi t/2)$$

Con lo anterior, el único parámetro libre es λ, el costo unitario de almacenamiento del producto.

Ejercicio 5. Calcule los valores de $x(t)$, $y(t)$ y $u(t)$ para distintos valores de λ, por ejemplo, λ =0.1,0.2,...,1.0. Grafique las soluciones (como se muestra en la figura 1) e identifique los casos donde el control obtenido corresponde al control teórico calculado en el ejercicio 3.

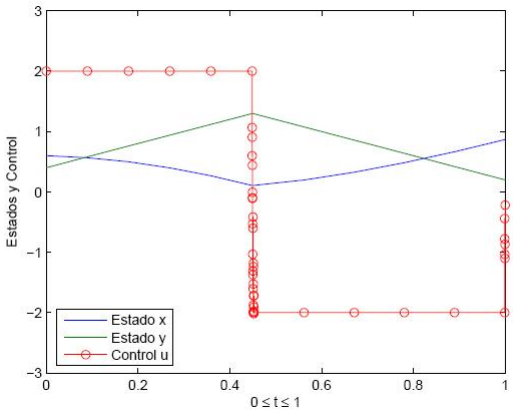

Figura 1: Evolución de Estados y Control con respecto al tiempo

Ejercicio 6. Encuentre otra función $q(t)$ que entregue como resultado un control $u(t)$ de tipo bang-bang utilizando los mismos valores de λ del ejercicio anterior (observe que se debe cumplir $p_2(t^*) = q(t^*)$ para algún $t^* \in (0,1)$). Calcule los valores de $x(t)$, $y(t)$ y $u(t)$. Grafique las soluciones (como se muestra en la figura 1) e identifique los casos donde el control obtenido corresponde al control teórico calculado en el ejercicio 3.

Ejercicio 7. Interprete los resultados obtenidos en los ejercicios 5 y 6, y deduzca cual es el valor de λ más conveniente para el comerciante en cada ejercicio, dada la configuración de parámetros enunciada. Compare los resultados con los obtenidos teóricamente.

Referencias

[1] M. Cizniar, M. Fikar, and M.A. Latifi: *MATLAB Dynamic Optimisation Code DYNOPT, User's guide*, Technical Report, KIRP FCHPT STU, Bratislava, 2006. `http://www.kirp.chtf.stuba.sk/publication_info.php?id_pub=271`

Utilización de DYNOPT

La interfaz de uso consiste en 3 funciones llamadas `process.m`, `objfun.m` y `confun.m`, donde se especifican la dinámica, la función objetivo y las restricciones.

Consideremos el siguiente problema sin restricciones g y h, donde se desea minimizar el estado final encontrando un control :

$$\begin{array}{lll} \textbf{min} & x_2(T) & \\ \textbf{s.a} & \dot{x_1} = u & x_1(0) = 1 \\ & \dot{x_2} = x_1^2 + u^2 & x_2(0) = 0 \end{array}$$

El archivo `process.m` describe la dinámica y se define de la siguiente manera:

```
function sys = process(t,x,flag,u,p)
switch flag,
    case 0 % f(x,u,p,t)
        sys = [u;x(1)^2+u^2];
    case 1 % df/dx
        sys = [];
    case 2 % df/du
        sys = [];
    case 3 % df/dp
        sys = [];
    case 4 % df/dt
        sys = [];
    case 5 % x0
        sys = [1;0];
    case 6 % dx0/dp
        sys = [];
    case 7 % M
        sys = [];
    case 8 % unused flag
        sys = [];
    otherwise
        error(['unhandled flag = ',num2str(flag)]);
end
```

En `case 0` se ingresa la función por filas:

$$f(\mathbf{x},\mathbf{u},\mathbf{p},t) = \begin{bmatrix} f_1(\mathbf{x},\mathbf{u},\mathbf{p},t) \\ \vdots \\ f_n(\mathbf{x},\mathbf{u},\mathbf{p},t) \end{bmatrix}$$

En `case 1` se ingresa la derivada con respecto a $\mathbf{x}$, de la siguente manera:

$$\begin{bmatrix} \frac{df_1}{dx_1} & \frac{df_2}{dx_1} & \cdots & \frac{df_n}{dx_1} \\ \vdots & \ddots & & \vdots \\ \frac{df_1}{dx_n} & \frac{df_2}{dx_n} & \cdots & \frac{df_n}{dx_n} \end{bmatrix}$$

Los otros `case` se ingresan de manera similar.

El archivo `objfun.m` define la función objetivo y se define de la siguiente manera:

```
function f = objfun(t,x,u,p)
f = [x(2)];
```

Recuerde que la función objetivo por defecto se considera de tipo Mayer.

Para ejecutar el programa, se ingresa lo siguiente en la línea de comandos de MATLAB:

```
options = optimset('LargeScale','off','Display','iter');
options = optimset(options,'MaxFunEvals',1e5);
options = optimset(options,'TolFun',1e-7);
options = optimset(options,'TolCon',1e-7);
options = optimset(options,'TolX',1e-7);

optimparam.optvar = 3; % numero de variables, en este caso 3, u, x(1) y x(2)
optimparam.objtype = []; % tipo de funcion objetivo, por defecto es de tipo Mayer
optimparam.ncolx = 3; % numero de segmentos (NE) para x
optimparam.ncolu = 2; % numero de segmentos (NE) para u
optimparam.li = ones(3,1)*(1/3); % se debe dejar con el mismo numero de segmentos para x
optimparam.tf = 1; % tiempo final T
optimparam.ui = zeros(1,3); % se debe dejar con el mismo numero de segmentos para x
optimparam.par = []; % parametros
optimparam.bdu = []; % cotas de x
optimparam.bdx = []; % cotas de u
optimparam.bdp =[]; % cotas de p
optimparam.objfun = @objfun;
optimparam.confun = [];  % si se definene restricciones, aca debe ir @confun
optimparam.process = @process;
optimparam.options = options;

[optimout,optimparam]=dynopt(optimparam)
```

En la variable `optimout` queda guardada la solución de problema para los inputs definidos en `optimparam`. Puede serle útil utilizar la función `profiles` que entrega 3 vectores o matrices:

```
[tplot,uplot,xplot] = profiles(optimout,optimparam,ntimes);
```

En `tplot` se guardan los tiempos donde se han evaluado el control y el estado, en `uplot` se guarda la matriz de los controles, con el control u_i en la columna i-ésima de `u`, y lo mismo para `xplot`, donde se guarda la matríz de los estados.

Para el ejemplo, se realiza lo siguiente:

```
>> [tplot,uplot,xplot] = profiles(optimout,optimparam,10);
>> plot(tplot,uplot)
>> plot(tplot,xplot(:,1))
>> plot(tplot,xplot(:,2))
```

Y se obtienen los gráficos de las figuras 2 y 3.

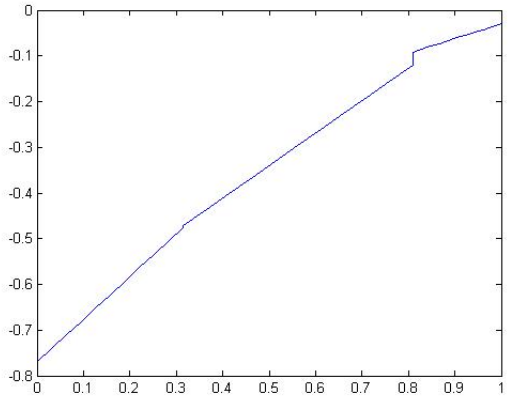

Figura 2: Gráfico del control vs. tiempo

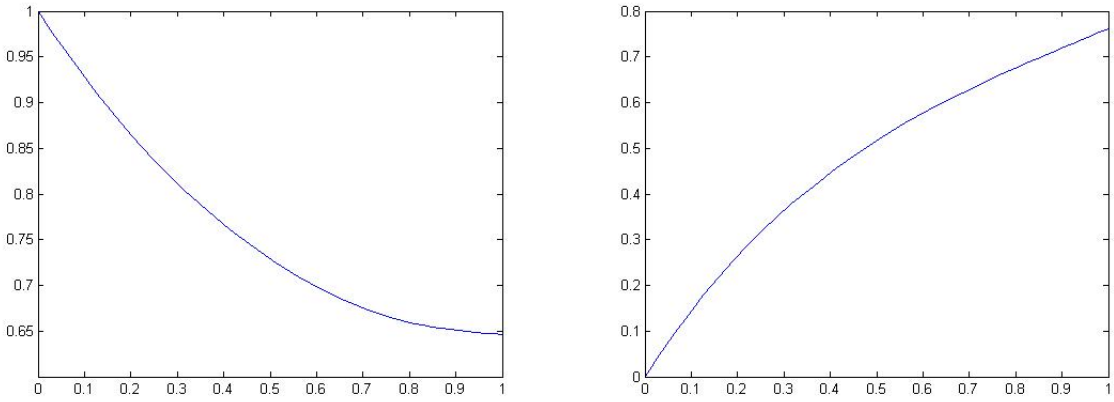

Figura 3: Gráficos del estado 1 vs. tiempo y estado 2 vs. tiempo