PyCon: The fastest optimal control package

Jesús Oroya
Chair of Computational Mathematics

November 19, 2020

Abstract

En este documento presentaremos el paquete de control óptimo no lineal: PyCon. Esta es una librería para python que utiliza la programación orientada a objetos para la definición del sistema dinámico y el problema de control. Debido a esto PyCon tiene una sintaxis muy simple para la definicion de problemas y una gran velocidad de computo gracias a la diferenciación automática heredada de CasADi.

Contents

1 Introducción

Al introducirnos al cálculo numérico del control óptimo no lineal con restricciones podemos ver distintas opciones de software. Por lo general, los distintos software estan espacializados en la ingeniría de control donde los detalles técnicos son importantes en la implementación pero que oscurecen el problema matemático. Por el contrario también existen paquetes de software cuyo objetivo final es el resolver un problema de programación no lineal, de manera que el usuario debe discretizar su problema de control óptimo y convertirlo en un NLP, como por ejemplo *AMPL-IPOPT*. Este tipo de problemas han sido intensamente estudiados y con ayuda de la diferenciación automática hoy en día se tiene metodolgías con mucha eficacia.

2 Trabajos relacionados

2.1 CasADi

2.2 IPOPT

2.3 Gekko

3 Diagrama de Clases

A continuación decribiremos de manera genral los problemas de control óptimo, para luego centrarnos en sus principales caráteristicas con el fin de mostrar como esta estrucurada el diagrama de clases diseñado en el paquete PyCon.

3.1 *dynacmis class*: Caracterización de un sistema dinámico

Un sistema dinámico queda definido dada una variable de estado $x \in \mathbb{R}^n$ y una variable de control $u \in \mathbb{R}^m$, donde n y m son las dimensiones de la variable de estado y de control respectivamente además de su ecuación de evolución:

$$\begin{cases} \dot{x}(t) = f(t, x(t), u(t)) & t \in (0, T] \\ x(0) = x_0 \end{cases} \tag{1}$$

En el cálculo numérico es necesario elegir una metodología de discretización temporal por lo que el problema anterior se puede escribir dado una partición $\mathcal{P} = \{t_0, t_1, \ldots, t_{N_t}\}$ del intervalo $[0, T]$ podemos escribir (1) de la siguiente manera:

$$\begin{cases} x_{t+1} = x_t + \end{cases} \tag{2}$$

Entonces los caráteristicas que lo define son:

Name	Notation
Intervalo temporal	[0,T]
Variable de estado	$x \in \mathcal{X}$
Variable de control	$u \in \mathcal{U}$

Table 1: Caracteristicas de un sistema dinámico

1. Un intervalo temporal $(0, T]$
2. La variable de estado $x \in \mathcal{X}$
3. La variable de Control $u \in \mathcal{U}$
4. Ecuación dinámica $f : [0, T] \times \mathcal{X} \times \mathcal{U} \to \mathcal{X}$
5. Condición inicial del problema $x_0 \in \mathcal{X}$

1. Características Obligatorias
   - variable de estado
   - variable de control
   - ecuación dinámica

2. Características Opcionales

    - intervalo de integración: Por defecto se toma 100 punto uniformemente distribuidos en el intervalo $[0, 1]$
    - Metodología de integración: Por defecto se toma el método de euler.
    - Condición inicial del estado

Algorithm 1 Ejemplo de declaración de una sistema dinámico

```
# Declare model variables
x = MX.sym('x',2,1)
u = MX.sym('u',1,1)
t = MX.sym('t',1,1)

# dynamic equations
xdot = vcat([           -x[1]*sin(t)              ,
                   -0.25*x[1]**3-x[0] + u[0]        ])

# Build dynamics obj
idyn = dynamics(t,x,u,xdot)
idyn.SetIntegrator()
```

3.2 Definición de un problema de control óptimo

3.2.1 Caracterización de un problema de control

Un problema de control óptimo general se puede escribir de la siguiente manera:

$$\min_{u(t)\in\mathcal{U}} \left[\Psi(x(T)) + \int_0^T L(t, x(t), u(t)) dt \right] \tag{3}$$

$$g_{low} \leq G(u(t)) \leq g_{up} \tag{4}$$

subject to:

$$\begin{cases} \dot{x}(t) = f(t, x(t), u(t)) & t \in (0, T] \\ x(0) = x_0 \end{cases} \tag{5}$$

A continuación nombraremos las principales características que define este el problema

1. Un sistema dinámico

$$\begin{cases} \dot{x}(t) = f(t, x(t), u(t)) & t \in (0, T] \\ x(0) = x_0 \end{cases} \tag{6}$$

2. Coste Final $\Psi : \mathcal{X} \to \mathbb{R}$

3. Coste a través del camino $L : \mathcal{X} \times (0, T] \to \mathbb{R}$

4. Restricciones del control $G : \mathcal{U} \to \mathbb{R}^K$ donde K es el número de restricciones considerardas.

Algorithm 2 Ejemplo de declaración de una sistema dinámico

```
## Build OCP obj
PathCost   = 1e-3*dot(u,u)
FinalCost   = dot(x,x)
##
iocp = ocp(idyn,PathCost,FinalCost)
iocp.functional.SetIntegrator()
# Set initial Condition
x0_num = [1,0]
# Compile the NLP, where you fix the initial condition of your problem
iocp.BuildNLP(x0_num)
```

4 Tipos de problemas

5 NLP

$$\min_{u} \tag{7}$$

6 Ejemplos