

How to use IpOpt and AMPL to solve time optimal control problems

In this short tutorial we explain how to use IpOpt in order to solve time optimal control problems. We refer to [1, 4, 5] for a survey on numerical methods in optimal control and how to implement them efficiently according to the context.

The tools

- IpOpt is an interior-point optimization routine (see [6]). Basically, it numerically solves optimisation problems of the form:

$$\begin{array}{l} \min \quad f(x) \\ \quad \quad x \in \mathbb{R}^n, \\ \quad \quad g(x) \leq 0, \\ \quad \quad h(x) = 0. \end{array} \quad (\text{P0})$$

IpOpt can be used together with Matlab, FreeFem++... and can be used directly in cpp codes.

- AMPL is an automatic differentiation and the modelling language (see [2]). The interest of using AMPL together with IpOpt is that, the gradient of the cost function and the constraints is automatically generated. Solving problems with IpOpt and AMPL can be made online through the NEOS solvers.

Solving problem (P0)

Let us write in AMPL language the problem (P0).

```
1 var x {i in 1..n} # The variable of the problem
2 minimize cost : f(x); # The cost function to be minimized
3 subject to inequality_constraints : g(x)<=0; # The inequality constraints
4 subject to equality_constraints : h(x)=0; # The equality constraints
5 option solver ipopt; # Set IpOpt as solver
6 # Set options for IpOpt, such as the maximal number of iterations...
7 option ipopt_options "max_iter=20000 linear_solver=mumps halt_on_ampl_error yes";
8 solve; # Solve the problem
9 printf : "# cost = %24.16e\n", cost; # Display the cost value
10 printf : "# Data\n";
11 printf {i in 0..n} : "%24.16e\n", x[i]; # Display the optimal values of x
12 end; # Quit AMPL
```

Solving a time optimal control problem with constraints

Let us now turn to a time optimal control problem.

Given a dynamical system,

$$\dot{y} = f(y, u), \quad (1)$$

some initial condition $y^0 \in \mathbb{R}^n$ and some terminal condition $y^1 \in \mathbb{R}^n$ and some bound $M > 0$, the aim is to find the minimal time $T \geq 0$ such that there exist a control $u \in L^\infty(0, T)^m$ such that,

$$|u_i(t)| \leq M \quad (t \in (0, T) \text{ a.e.}, i \in \{1, \dots, m\}), \quad (2)$$

and the solution y of (1) satisfies the constraint,

$$g(y(t)) \leq 0 \quad (t \in (0, T)), \quad (3)$$

and the terminal condition,

$$y(T) = y^1, \quad (4)$$

where g is given and define constraints of the state variable y . Of course to be able to solve this system, one needs to have $g(y^0) \leq 0$ and $g(y^1) \leq 0$.

The above optimal control problem, can be recast as an optimisation problem under constraints similar to (P0). More precisely, it is

$$\begin{array}{l} \min \quad T \\ \left| \begin{array}{l} u \in L^\infty(0, T)^M, \\ \dot{y} = f(y, u), \quad y(0) = y^0, \\ y(T) = y^1, \\ g(y(t)) \leq 0 \quad (t \in (0, T)). \end{array} \right. \end{array}$$

In order to handle numerically, this problem, we will use a time discretization. Let us explain it with the explicit Euler method. But any other time discretization can be used.

Fix some parameter $N_t \in \mathbb{N}^*$ (the number of time steps) and for $T > 0$ given, define y_i the estimation of $y(iT/N_t)$ for $i \in \{0, \dots, N_t\}$. The explicit Euler scheme, gives the relation,

$$y_{i+1} = y_i + \frac{T}{N_t} f(y_i, u_i),$$

with $u_i \simeq u(iT/N_t)$. Then the state and control constraints are replaced by:

$$g(y_i) \leq 0 \quad \text{and} \quad |u_i| \leq M.$$

Consequently, in discretized version, we end up with a finite dimensional control problem under constraints, whose AMPL version is:

```

1 # Define the parameters of the problem
2 param Nt=100; # number of time step discretization points
3 param M =1; # bound on the control
4 param y0=1; # initial condition
5 param y1=0; # final condition
6
7 # Define variables of the problem
8 var y {i in 0..Nt};
9 var u {i in 0..Nt} >=M, <=M; # The control shall be in [-M,M]
10 var T >=0; # The time T shall be nonnegative
11
12 # The cost function is the time T
13 minimize cost: T;
14 # Set the constraints
15 subject to y_dyn {i in 0..Nt-1} : # y is solution of (1)
16     y[i+1]=y[i]+T/Nt*f(y[i],u[i]);
17 subject to y_init : y[0] =y0; # y(0)=y0
18 subject to y_end : y[Nt]=y1; # y(T)=y1
19 subject to state_constraint {i in 1..Nt-1} : g(y[i])<=0; # g(y(t))<=0
20
21 # Solve with IpOpt
22 option solver ipopt;
```

```

23 option ipopt_options "max_iter=20000 linear_solver=mumps halt_on_ampl_error yes";
24 solve;
25 # Display solution
26 printf : "# T = %24.16e\n", T;
27 printf : "# Nt = %d\n", Nt;
28 printf : "# Data\n";
29 printf {i in 0..Nt} : " %24.16e\n", u[i];
30 printf {i in 0..Nt} : " %24.16e\n", y[i];
31 end;

```

Application to the constrained heat equation

Now we can turn to the control of the heat equation with nonnegative state constraint.

To this end, we consider the controlled 1D heat equation with Neumann boundary control.

$$\begin{cases} \dot{y}(t, x) = \partial_x^2 y(t, x) & (t > 0, x \in (0, 1)), \\ \partial_x y(t, 0) = v_0(t) & (t > 0), \\ \partial_x y(t, 1) = v_1(t) & (t > 0), \end{cases} \quad (5)$$

To this problem, we add the control constraints,

$$|v_0(t)| \leq M \quad \text{and} \quad |v_1(t)| \leq M \quad (t > 0 \text{ a.e.}), \quad (6)$$

with some $M > 0$ given and we add the state constraint,

$$y(t, x) \geq 0 \quad ((t, x) \in \mathbb{R}_+^* \times (0, 1) \text{ a.e.}). \quad (7)$$

To the system (5) we add the initial condition

$$y(0, x) = y^0(x) \quad (x \in (0, 1)), \quad (8)$$

and the terminal condition

$$y(T, x) = y^1(x) \quad (x \in (0, 1)). \quad (9)$$

In [3], it has been proved that every positive constant state y^0 can be steered to some over positive constant state y^1 in a large enough time T . Our goal here is to find numerically the minimal time T such there exist controls satisfying (6) such that the corresponding solution y of (5) and (8) satisfies the constraint (7) and the terminal condition (9), i.e. we aim to minimize

$$\begin{array}{l} \min T \\ \left| \begin{array}{l} T > 0, \\ v_0, v_1 \in L^\infty(0, T), y \text{ solution of (5) and (8) satisfies} \\ y(T, \cdot) = y^1 \quad \text{and} \quad y(t, x) \geq 0 \quad (t > 0, x \in (0, 1) \text{ a.e.}). \end{array} \right. \end{array}$$

Firstly, we will use a space discretization to reduce the system (5). To this end, we define $N \in \mathbb{N}^*$ and for every $n \in \{0, \dots, N\}$, $x_n = \frac{n}{N}$. Based on this discretization of $[0, 1]$, we will discretized (5) using centered finite differences. That is to say, given $V(t) = (v_0(t), v_1(t))^\top$ and $Y(t) \in \mathbb{R}^{N+1}$, a vector approaching $(y(t, x_0), \dots, y(t, x_N))^\top$, Y is solution of

$$\dot{Y} = AY + BV \quad (t > 0), \quad (10a)$$

with the initial condition

$$Y(0) = y^0 e_{N+1} \quad (10b)$$

the target state,

$$Y(T) = y^1 e_{N+1} \quad (10c)$$

the state constraint,

$$Y(t) \geq 0 \quad (t > 0 \text{ a.e.}) \quad (10d)$$

and the control constraint,

$$|V_0(t)| \leq M \quad \text{and} \quad |V_1(t)| \leq M \quad (t > 0 \text{ a.e.}), \quad (10e)$$

where we have set $e_{N+1} = (1, \dots, 1)^\top \in \mathbb{R}^{N+1}$,

$$A = N^2 \begin{pmatrix} 2 & -2 & 0 & \dots & \dots & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & 1 & -2 & 1 \\ 0 & \dots & \dots & \dots & 0 & 2 & -2 \end{pmatrix} \in M_{N+1}(\mathbb{R}) \quad \text{and} \quad B = 2N \begin{pmatrix} 1 & 0 \\ 0 & \vdots \\ \vdots & \vdots \\ \vdots & \vdots \\ \vdots & \vdots \\ \vdots & 0 \\ 0 & 1 \end{pmatrix} \in M_{N+1,2}(\mathbb{R}).$$

Now we can discretize (10a) using explicit Euler scheme and similarly to the previous example, we obtain an optimisation problem of finite dimension with constraints whose AMPL formulation is

```

1 # Parameters of the problem
2 param Nx=30; # Number of space discretisation points
3 param Nt=300; # Number of time discretisation points
4 # One have to check a posteriori that the number of time steps is large enough so that the
5 # CFL condition, T*Nx^2/Nt <=1/2, is satisfied
6 param dx=1/Nx; # Space step
7 param M =20; # Bound on the controls
8
9 # Variables of the system
10 # i stands for the time index and j for the space index
11 var y {i in 0..Nt, j in 0..Nx} >=0; # State of the control problem
12 # Neuman controls in 0 and 1. The controls are in [-M,M].
13 var v0 {i in 0..Nt} >=-M, <=M;
14 var v1 {i in 0..Nt} >=-M, <=M;
15 var T >=0; # Control time
16 var dt=T/Nt; # Time step
17
18 # Define the cost function
19 minimize cost: T;
20
21 # Define the constraints
22 # y is solution of the discretize system
23 subject to y_dyn {i in 0..Nt-1, j in 1..Nx-1}:
24 (y[i+1,j]-y[i,j])*(dx)^2=(y[i,j-1]-2*y[i,j]+y[i,j+1])*dt;
25 # Neuman boundary conditions in 0 and 1
26 subject to left_boundary {i in 1..Nt-1}: y[i,1]-y[i,0] =v0[i]*dx;
27 subject to right_boundary {i in 1..Nt-1}: y[i,Nx]-y[i,Nx-1]=v1[i]*dx;
28 # y(0)=y0 and y(T)=y1
29 subject to y_init {j in 0..Nx}: y[0,j] =5;
30 subject to y_end {j in 0..Nx}: y[Nt,j]=1;
31
32 # Solve with IpOpt
33 option solver ipopt;
34 option ipopt_options "max_iter=2000 linear_solver=mumps halt_on_ampl_error yes";
35 solve;
36
37 # Write the solution in the file out.txt
38 printf: " # T = %24.16e\n", T > out.txt;
39 printf: " # Nx = %d\n", Nx >> out.txt;

```

```

40 printf: " # Nt = %d\n", Nt    >> out.txt;
41 printf: " # Data\n"          >> out.txt;
42 printf {i in 0..Nt}: " %24.16e\n", v0[i] >> out.txt;
43 printf {i in 0..Nt}: " %24.16e\n", v1[i] >> out.txt;
44 printf {i in 0..Nt, j in 0..Nx}: " %24.16e\n", y[i,j] >> out.txt;
45
46 end;

```

Once the file `out.txt` is written, it can be for instance read by Scilab with the following code

```

1 fid= mopen('out.txt','r');           // Open out.txt
2 T = mfscanf(fid,'%s %s %s');        // Read ``# T =``
3 T = mfscanf(fid,'%f');              // Read value of T
4 Nx = mfscanf(fid,'%s %s %s');       // Read ``# Nx =``
5 Nx = mfscanf(fid,'%d');             // Read value of Nx
6 Nt = mfscanf(fid,'%s %s %s');       // Read ``# Nt =``
7 Nt = mfscanf(fid,'%d');             // Read value of Nt
8 s = mfscanf(fid,'%s %s'); s=[];     // Read ``# Data``
9 v0 = mfscanf(Nt+1,fid,'%f'); v0=v0'; // Read the Nt+1 values of v0
10 v1 = mfscanf(Nt+1,fid,'%f'); v1=v1'; // Read the Nt+1 values of v1
11 y = mfscanf((Nt+1)*(Nx+1),fid,'%f'); // Read the (Nt+1)*(Nx+1) values of y
12 mclose(fid);                       // Close out.txt
13
14 y = matrix(y,Nx+1,Nt+1); y=y';     // Reshape y as a matrix, line i is the solution at time i/Nt
15 x = 0:1/Nx:1; t = 0:T/Nt:T;       // Define the space and time discretisations
16 printf('time:\t%f\n',T);          // Display the control time
17 plot(t,[v0;v1]); sleep(2000); clf(); // plot controls and wait 2s
18 plot2d(x,y(1,:),rect=[0 0 1 10]); sleep(100); // plot the initial state and wait 0.1s
19 for i=2:1:Nt,
20     plot(x,y(i,:),rect=[0 0 1 10]); sleep(10); // plot the state at each time instants
21 end
22 plot(x,y($,:),rect=[0 0 1 10]);    // plot the final state

```

Based on these two codes, we obtain

- for $y^0 = 1$, $y^1 = 5$, $M = 20$ and the discretization parameters $N_x = 30$, $N_t = 450$ the results displayed on the video:
<https://mycore.core-cloud.net/index.php/s/FaXAYDKtdB4So6j>
(the minimal time obtained is $T \simeq 0.2093$);
- for $y^0 = 5$, $y^1 = 1$, $M = 800$ and the discretization parameters $N_x = 30$, $N_t = 450$ the results displayed on the video:
<https://mycore.core-cloud.net/index.php/s/FYG6RgN0IEovwCN>
(the minimal time obtained is $T \simeq 0.1938$).

Similarly, we can consider the time optimal control problem,

$$\begin{array}{l}
\min T \\
\left| \begin{array}{l}
T > 0, \\
u_0, u_1 \in L^\infty(0, T) \\
u_0(t) \geq 0 \quad \text{and} \quad u_1(t) \geq 0 \quad (t \in (0, T) \text{ a.e.}), \\
y \text{ solution of (11) satisfies } y(T, \cdot) = y^1,
\end{array} \right.
\end{array}$$

with (11) given by,

$$\left\{ \begin{array}{ll}
\dot{y}(t, x) = \partial_x^2 y(t, x) & (t > 0, x \in (0, 1)), \\
y(t, 0) = u_0(t) & (t > 0), \\
y(t, 1) = u_1(t) & (t > 0), \\
y(0, x) = y^0(x) & (x \in (0, 1)).
\end{array} \right. \quad (11)$$

Numerically, we obtain

- for $y^0 = 5$, $y^1 = 1$, $M = 50$ and the discretization parameters $N_x = 30$, $N_t = 450$ the results displayed on the video:
<https://mycore.core-cloud.net/index.php/s/SUZPRgM25Qg9rma>
 (the minimal time obtained is $T \simeq 0.1931$);
- for $y^0 = 1$, $y^1 = 5$, $M = 50$ and the discretization parameters $N_x = 20$, $N_t = 200$ the results displayed on the video:
<https://mycore.core-cloud.net/index.php/s/UDvMVhum5drcaRX>
 (the minimal time obtained is $T \simeq 0.0498$);
- for $y^0 = 1$, $y^1 = 5$, $M = 3000$ and the discretization parameters $N_x = 20$, $N_t = 200$ the results displayed on the video:
<https://mycore.core-cloud.net/index.php/s/mahwuADcC7ZvMCg>
 (the minimal time obtained is $T \simeq 0.0438$).

Useful links

- IpOpt project: <https://projects.coin-or.org/Ipopt>
- AMPL project: <http://ampl.com/products/ampl/>
- NEOS solvers: <https://neos-server.org/neos/solvers/>

References

- [1] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming. 2nd ed.* Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 2nd ed. edition, 2010.
- [2] R. Fourer, D. M. Gay, and B. W. Kernighan. A modeling language for mathematical programming. *Manage. Sci.*, 36(5):519–554, 1990.
- [3] J. Lohéac, E. Trélat, and E. Zuazua. Minimal controllability time for the heat equation under state constraints. In preparation.
- [4] E. Trélat. *Contrôle optimal. Théorie et applications.* Paris: Vuibert, 2005.
- [5] E. Trélat. Optimal control and applications to aerospace: some results and challenges. *J. Optim. Theory Appl.*, 154(3):713–758, 2012.
- [6] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1 (A)):25–57, 2006.