Numerical algorithms for one and two target optimal controls

Sung-Sik Kwon

Department of Mathematics and Computer Science, North Carolina Central University
1801 Fayetteville St. Durham, NC 27707
Email: skwon@nccu.edu

Jon W. Tolle

Department of Statistics and Operations Research
University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3180
Email: tolle@email.unc.edu

Abstract

In this paper one and two target optimal controls of a linear diffusion equation are considered. These problems can be formulated as nonlinear, PDE-constrained optimization problems in infinite dimension. We consider two formulations, one having an inequality constraint mandating the primary target to be satisfied within a given tolerance ϵ, and the other having weighted norms in the cost function to fulfill the same requirement. In this paper we develop iterative numerical algorithms that estimate the weights, α, in terms of ϵ. This result allows us to estimate the solution of the inequality constrained optimization, which is generally difficult and computationally expensive, by the solution of the optimization problem having the weighted norms.

Keywords. PDE constrained optimization, conjugate duality theory, numerical algorithm.

AMS Classification. 65K10, 49M29, 49M05.

1 Introduction

One and two-target optimal control problems arising from a system governed by a PDE can be formulated as infinite dimensional, nonlinear optimization problems, typically containing inequality constraints. However, handling of these inequality constraints are generally tricky and computationally expensive. In this paper we investigate alternative formulations and propose numerical algorithms which can avoid solving this difficult problem by using its dual problem and other formulations.

Throughout this paper we consider the following control system.

$$\begin{aligned} \frac{\partial y}{\partial t} - \frac{\partial^2 y}{\partial x^2} &= v(x,t) \qquad (x,t) \in (0,L) \times (0,T) \\ y(0,t) = y(L,t) &= 0 \qquad t \in (0,T) \\ y(x,0) &= 0 \qquad x \in (0,L). \end{aligned} \tag{1}$$

$v(x,t)$ is the control function that is used to control the state vector $y(x,t)$. As a first step toward studying various formulations [**?**], [**?**], we consider the simplest, one-target case with only equality constraints. The object of this case is to find $v(x,t)$ (at minimum cost) so

that the state of the system at time T, $y(x,T)$, is close to a given $L^2(0,L)$ target y_T. The problem can be formulated as

$$(P1) \qquad \min \quad \frac{1}{2}\|v(x,t)\|^2_{L^2((0,L)\times(0,T))} + \frac{\alpha}{2}\|y(x,T) - y_T(x)\|^2_{L^2(0,L)}$$

subject to (1) is satisfied.

The output least squares method was first applied to solve $(P1)$, but we observed that the algorithm ran prohibitively slow even with very coarse mesh. This was somewhat expected since the method requires the forward solution of (1) to be solved frequently (for each choice of v), and computing a pde is generally expensive computationally . Thus we need to look for a faster way that avoids the direct calculation of forward solutions of (1); we will do this by treating both v and y as independent variables.

2 Approximating the PDE constraint

Let us begin by creating mesh points on $[0,L] \times [0,T]$: Divide $[0,L]$ into m subintervals of the equal length and let $h = \frac{L}{m}$. Define $x_i = ih, i = 0,1,...,m$. Similarly, divide $[0,T]$ into n subintervals, let $k = \frac{T}{n}$, and define $t_j = jk, j = 0,1,...,n$. Then we look for U_i^j that approximate the true solution $\hat{y}$ at the mesh points, i.e., $U_i^j \approx \hat{y}(x_i,t_j)$. We have tested three different methods for solving this problem: forward difference, backward difference, and Crank-Nicolson methods. The Crank-Nicolson method [**?**] which has the best error estimate and stability among these, indeed, outperformed the other two as expected. Thus we use Crank-Nicolson formulation here. Letting $\lambda = \frac{k}{h^2}$, we need to solve

$$-\frac{\lambda}{2}U_{i-1}^{j+1} + (1+\lambda)U_i^{j+1} - \frac{\lambda}{2}U_{i+1}^{j+1} = \frac{\lambda}{2}U_{i-1}^{j} + (1-\lambda)U_i^{j} + \frac{\lambda}{2}U_{i+1}^{j} + v(x_i,t_j) \qquad (2)$$

Typically, an evolution equation such as (2) is solved iteratively at each time horizon using the known information from the previous time step. This is memory-efficient method since we only need to store a matrix of size $O(m^2)$ for computing an ode for a fixed time. However, our primary interest is in representing (1) (for the entire space-time domain) in a discretized form as a matrix equation. To do this, we first need to rewrite U_i^j as a long, one-dimensional vector y of size $n(m-1)$. Note that in this vector we do not include the known values, i.e., initial values $\{U_i^0\}$ and the boundary values $\{U_0^j\}, \{U_m^j\}$. Using the enumeration

$$y_{(j-1)(m-1)+i} = U_i^j, \qquad i = 1,2,...,m-1,\ j = 1,2,...,n-1$$

and similarly

$$v_{(j-1)(m-1)+i} = v(x_i,t_j), \qquad i = 1,2,...,m-1,\ j = 1,2,...,n-1$$

system (2) can be represented in a matrix equation

$$Ay = v \qquad (3)$$

where $y = (y_1, y_2, ..., y_{(n-1)(m-1)})^t$, $v = (v_1, v_2, ..., v_{(n-1)(m-1)})^t$, and A is the square matrix resulting from the equation (2). We refer the details of constructing the matrix A to the MATLAB source file, **CNFullDiscret3.m**.

Example 1 We test the above discretization scheme, i.e.,(3) to find an approximate solution of (1) with an input $v(x,t) = 2$. Fix $L = T = 1$ and $m = 10, n = 20$. The analytic solution can be found in this case and is given by

$$\hat{y}(x,t) = x(1-x) + \sum_{n=1}^{\infty} b_n e^{-n^2\pi^2 t} \sin n\pi x$$

where

$$b_n = 2\int_0^1 x(x-1)\sin n\pi x\, dx$$

We compare the solution obtained from (3) and the analytic solution evaluated at $m+1$ mesh points at time t = 0.95. **(Source file: CNFullDiscret3.m)**

```
>> CNFullDiscret3

Result at t = 0.950000
           x-value        approx sol          true sol
                 0                 0                 0
   0.10000000000000   0.08998205658483   0.08998893530471
   0.20000000000000   0.15998988515105   0.15997895369889
   0.30000000000000   0.20997008088413   0.20997103225166
   0.40000000000000   0.23996933775797   0.23996594636946
   0.50000000000000   0.24996928220374   0.24996419389389
   0.60000000000000   0.23996933775797   0.23996594636946
   0.70000000000000   0.20997008088413   0.20997103225166
   0.80000000000000   0.15998988515105   0.15997895369889
   0.90000000000000   0.08998205658483   0.08998893530471
   1.00000000000000                 0  -0.00000000000000

norm of the difference between the approx sol and true sol is
0.009170
```

3 One-target problem

In this section, we discuss the numerical solutions of one target problem, which can be stated as follows: Given a target function $y_T(x) \in L^2(0,L)$ and a tolerance $\epsilon > 0$, find a control function $v(x,t) \in L^2((0,L)\times(0,T))$ that forces the state solution $y(v)$ at time T to be within $\epsilon-$neighborhood of the target, i.e., $\|y(x,T;v) - y_T\| \le \epsilon$. Since there may be infinitely many such vs, we look for the one that has the minimum norm. This problem has been considered in three different formulations including (P1), and the connection among these formulations has been addressed theoretically [**?**]. A goal of this section is to verify some of these theoretical developments by using a numerical example. Throughout this section, we assume $L = T = 1$ and $y_T = x(1-x)$. From Example 1 we know at least $v(x,t) \equiv 2$ drives the system to satisfy the target requirement (assuming $\epsilon > 0.01$), that $y(x,T)$ is close to y_T. What we do not know is whether $v \equiv 2$ is the minimum norm solution

that we are seeking. It is very unlikely that $v(x,t) \equiv 2$ (or something similar to this) would yield the minimum due to the following intuitive reason. Since the target only concerns at the final time of y, the control function v can stay pretty close to zero except for the last few times that are close to the final time. At those last few times v can give *stronger* input to drive y to y_T. Corresponding y will also stay flat near zero until the last few times and suddenly jump to the desired state at the final time. If such v exists, its L^2 norm will be small. This phenomenon is revealed in the numerical examples considered below. However, all this may change if we add a secondary target that prevents a sudden jump (such as H_0^1 norm) in y. We plan to study this case in the future.

3.1 Formulation (P1), equality constraint with penalty term

We begin our discussion with the numerical solution of (P1) by treating both v and y as independent variables. If we represent the approximation of $\{v(x_i,t_j)\}$ as a long, one-dimensional vector v of size $sz := (n-1)(m-1)$ (using the same enumeration scheme for the interior mesh points as we have done for vector y) we have the following discretized version of (P1).

$$\min_{(v,y)\in R^{sz}\times R^{sz}} \quad \frac{1}{2}\|v\|^2 + \frac{\alpha}{2}\|y_F - y_T\|^2$$

subject to

$$[-I \;\; A]\begin{pmatrix} v \\ y \end{pmatrix} = 0$$

where y_T is the given target vector in R^{m-1} and A the same matrix that we saw in section 2. The vector y_F should a vector in R^{m-1} that approximates the true solution $\hat{y}$ at the final time $t = t_n$. Since $y \in R^{sz}$ approximates $\hat{y}$ only up to time $t = t_{n-1}$, y_F is not a part of y and thus needs to be calculated separately. One way to compute y_F is by solving (1) at $t = t_n$ using the approximate solution y at time $t = t_{n-1}$, which is the last $m-1$ components of the vector y. If we denote these last $m-1$ components of y by $\tilde{y}$, then by using the forward difference scheme we have

$$y_F = M_1\tilde{y}.$$

Here M_1 is an $(m-1)\times(m-1)$ matrix whose main diagonal entries are $1-2\lambda$ and super- and sub-diagonal entries are λ, resulting from the forward difference scheme. (Recall $\lambda = \frac{k}{h^2}$.) Furthermore, if we denote by 0_{m-1} the zero matrix of size of $(m-1)\times(m-1)$ and by I_{m-1} the identity matrix of size $m-1$, and let C be $(m-1)$ by $(n-1)(m-1)$ matrix consisting of $n-1$ blocks of $(m-1)\times(m-1)$ matrices such that

$$C = [0_{m-1}\, 0_{m-1}\, 0_{m-1},\, \ldots, I_{m-1}]$$

then

$$\tilde{y} = Cy$$

Finally, putting $M = M_1C$ we can rewrite the finite dimensional approximation of (P1) as

$$(P1_F) \qquad \min_{(v,y)\in R^{sz}\times R^{sz}} \quad \frac{1}{2}\|v\|^2 + \frac{\alpha}{2}\|My - y_T\|^2$$

subject to

$$[-I \;\; A] \begin{pmatrix} v \\ y \end{pmatrix} = 0$$

3.1.1 Solution of $(P1_F)$

We have tested three different routines for solving $(P1_F)$: a general-purpose MATLAB routine **fmincon**, quadratic programming routine **quadprog**, and my routine that simply computes the necessary condition. Among these fmincon was the slowest, and the other two routines ran much faster and gave very similar results. For example, when we use $m = 10, n = 20$, and $\alpha = 1000$, both routines took less than 0.2 seconds and the norm of the difference between their approximate solutions at $t = T$ was 2.549209629838923e-013. Deviations from target were about 0.002158 in both cases. Thus either quadprog routine or solving for the first order necessary condition would be suitable for handling $(P1_F)$.

3.2 Using the inequality constraint model

A direct translation of one target problem uses an inequality constraint:

$$(INEQ) \qquad \min \quad \frac{1}{2}\|v(x,t)\|^2_{L^2((0,L)\times(0,T))}$$

subject to

$$\begin{aligned} y_t - y_{xx} &= v(x,t) \\ y(0,t) = y(L,t) &= 0 \qquad t \in (0,T) \\ y(x,0) &= 0 \qquad x \in (0,L) \\ \|(y(x,T) - y_T(x))\|^2 &\leq \epsilon^2 \end{aligned}$$

A finite dimensional approximation of (INEQ) is then

$$(INEQ_F) \qquad \min_{(v,y)\in R^{sz}\times R^{sz}} \quad \frac{1}{2}v^t v$$

subject to

$$[-I \;\; A] \begin{pmatrix} v \\ y \end{pmatrix} = 0$$

$$(My - y_T)^t(My - y_T) \leq \epsilon^2$$

where M and y_T are defined the same way as in the previous section. We note that this formulation is computationally more involving than (P1) as it contains an inequality constraint. For the same size problem ($m = 10, n = 20$) that we used for $(P1_F)$, solving $(INEQ_F)$ by **fmincon** normally takes more than 6 minutes on the same machine, and the results are not very accurate. For example, the deviation from the target may slightly exceed the required tolerance ϵ.

3.3 Formulation of dual problems

Although (INEQ) is the one that best describes the one-target problem, handling its inequality constraint is difficult. In this section, we look at the dual problems for (INEQ) and (P1) and find a way to approximate the solution of (INEQ) by the solutions of (P1) [**?**],[**?**]. We summarize the results here.

The adjoint equation of (1) is defined by

$$\begin{aligned} \frac{\partial p}{\partial t} + \frac{\partial^2 p}{\partial x^2} &= 0 \qquad (x,t) \in (0,L) \times (0,T) \\ y(0,t) = y(L,t) &= 0 \qquad t \in (0,T) \\ y(x,T) &= w(x) \qquad x \in (0,L) \end{aligned} \tag{4}$$

for $w \in L^2(0,L)$. We define a linear mapping $\mathcal{L}:\ L^2((0,L)\times(0,T)) \to L^2(0,L)$ by

$$\mathcal{L}(v) = y(x,T;v)$$

where $y(x,T;v)$ is the solution of the equation (1) for the given v. Then its adjoint operator $\mathcal{L}^*$ is defined by

$$\mathcal{L}^*(w) = p(x,t;w)$$

where $p(x,t;w)$ is the solution of the system (4) for the given w. We let $\Lambda = \mathcal{L} \circ \mathcal{L}^*$ and state the dual problems.

Dual problem for (INEQ): Solving problem (INEQ) is equivalent to solving the problem: find $w^* \in L^2(0,L)$ such that for every $w \in L^2(0,L)$

$$< \Lambda(w^*), w - w^* >_{L^2} + \beta(\|w\|_{L^2} - \|w^*\|_{L^2}) \geq - < w - w^*, y_T >_{L^2},$$

in that if w^* satisfies these inequalities then the corresponding $p^*(x,t;w^*)$ given by (4) yields the optimal control for (INEQ) and the resulting $y^*(x,t;p^*)$ given by (1) is the optimal state variable.

Dual problem for (P1): Solving problem (P1) is equivalent to solving the linear system

$$(\Lambda + \frac{1}{\alpha})w = y_T \tag{5}$$

in that if w^* satisfies this equation then the corresponding $p^*(x,t;w^*)$ given by (4) yields the optimal control for (P1) and the resulting $y^*(x,t;p^*)$ given by (1) is the optimal state variable. From these dual problems, we derive a key relationship between α and ϵ:

$$\alpha = \frac{\|w^*\|_{L^2}}{\epsilon} \tag{6}$$

This suggests the following algorithm for finding an approximate solution to the problem (INEQ) using the solution to (P1).

Dual Algorithm

1. Given $\alpha^0 > 0$, set $k = 0$.

2. Solve the dual problem (5) with $\alpha = \alpha^k$ to obtain a solution w^k

3. If $\|\Lambda(w^k) - y_T\|_{L^2} > \epsilon$

   (a) Set $\alpha^{k+1} = \frac{1}{\epsilon}\| w^k\|_{L^2}$.

   (b) Set $k = k + 1$ and return to step 2.

4. Stop with approximate solution w^k to problem (INEQ).

In order to implement this algorithm, we discretize the operator Λ, and write a finite dimensional approximation of (5).

3.3.1 Discretization of Λ

Recall that the linear mapping $\mathcal{L} : L^2((0,L)\times(0,T)) \to L^2(0,L)$ is defined by

$$\mathcal{L}(v) = y(x,T;v)$$

where $y(x,T;v)$ is the solution of the equation (1) for the given v.

$$\begin{aligned} \mathcal{L}(v) &= y(x,T;v) \quad \text{in} \quad L^2(0,L) \\ &\approx My \quad \text{where} \quad Ay = v \quad \text{in} \quad R^{sz} \\ &= MA^{-1}v \end{aligned}$$

So, $\mathcal{L}$ can be approximated by MA^{-1}. By using the adjoint property, we can find an approximation of $\mathcal{L}^*$.

$$\begin{aligned} < \mathcal{L}(v), w > &\approx < MA^{-1}v, w > \\ &= v^t(A^t)^{-1}M^t w \\ &= < v, (A^t)^{-1}M^t w > \\ &\approx < v, \mathcal{L}^*(v) > \end{aligned}$$

So, $\mathcal{L}^*$ can be approximated by $(A^t)^{-1}M^t$ which is the transpose of MA^{-1}. Thus $\Lambda = \mathcal{L}\circ\mathcal{L}^*$ is approximated by

$$\Lambda \approx MA^{-1}(A^t)^{-1}M^t = M(A^tA)^{-1}M^t$$

Now we implement algorithm DUAL in which equation (5) is solved by

$$\left(M(A^tA)^{-1}M^t + \frac{1}{\alpha}I\right) w = y_T.$$

Using $m = 10, n = 20, \epsilon = 0.0001$, and $\alpha^0 = 1$, we have the following output.

```
>> p1_dual2
Number of iterations on alpha: 5

The values of alpha chosen by the algorithm
```

```
  1.0e+004 *

   0.00010000000000
   0.45569062263522
   2.16464894851623
   2.16605689188045
   2.16605713606691
   2.16605713610923

CPU time taken: 0.191000 seconds
           x-value   approx y at t=T        Target y_T
                 0                 0                 0
   0.10000000000000   0.08998585353174   0.09000000000000
   0.20000000000000   0.15997336530581   0.16000000000000
   0.30000000000000   0.20996372681881   0.21000000000000
   0.40000000000000   0.23995768578262   0.24000000000000
   0.50000000000000   0.24995563258323   0.25000000000000
   0.60000000000000   0.23995768578262   0.24000000000000
   0.70000000000000   0.20996372681881   0.21000000000000
   0.80000000000000   0.15997336530581   0.16000000000000
   0.90000000000000   0.08998585353174   0.09000000000000
   1.00000000000000                 0                 0

Deviation from the target is: targetDeviation =

    1.000000000000153e-004

Norm of the control is 1.117921.
```

As we run $(P1_F)$ by quadprog with $\alpha = 1.0e + 004 * 2.16605713610923$ found above, we obtain the following output.

```
>> p1_quad3

alp =

    2.166057136109230e+004

Optimization terminated successfully:
 Relative (projected) residual of PCG iteration <= OPTIONS.TolPCG
CPU time taken: 0.281000 seconds
           x-value   approx y at t=T        Target y_T
   0.10000000000000   0.08998585353175   0.09000000000000
   0.20000000000000   0.15997336530570   0.16000000000000
   0.30000000000000   0.20996372681893   0.21000000000000
   0.40000000000000   0.23995768578270   0.24000000000000
```

```
0.50000000000000   0.24995563258291   0.25000000000000
0.60000000000000   0.23995768578314   0.24000000000000
0.70000000000000   0.20996372681815   0.21000000000000
0.80000000000000   0.15997336530648   0.16000000000000
0.90000000000000   0.08998585353131   0.09000000000000

Deviation from the target is 0.000100

Norm of the control is 1.117921.
```

4 Two target problem

We now consider the two target case. As before, our primary objective is to find the control v of the system (1) that forces the state solution at the final time $t = T$ to be within $\epsilon-$neighborhood of a given L^2-target, i.e., $\|y(x,T;v) - y_T\| \leq \epsilon$. In addition, we have a secondary objective that the solution y is as close as possible to a given $y_1 \in L^2((\Omega) \times (0,T))$. (Note that we may take $y_1 \in L^2(0,T;H_0^1(\Omega))$.) Throughout this section, we use $y_1 \equiv 0$. Again, this problem can be formulated in more than one way, and we would like to discuss connections among different formulations. A key result is the relationship between the formulation using an inequality constraint (INEQ2) and the formulation using multiobjective function (P2). This relationship, which is described similar to (6), enable us to compute (INEQ2) via (P2).

4.1 Formulation using inequality constraint

The two-target problem is most precisely described by using the inequality constraint:

$$(INEQ2) \qquad \min \quad \frac{1}{2}\|v(x,t)\|^2_{L^2((0,L)\times(0,T))} + \frac{\alpha_2}{2}\|y(x,t)\|^2_{L^2((0,L)\times(0,T))}$$

subject to

$$\begin{aligned} y_t - y_{xx} &= v(x,t) \\ y(0,t) = y(L,t) &= 0 \qquad t \in (0,T) \\ y(x,0) &= 0 \qquad x \in (0,L) \\ \|(y(x,T) - y_T(x))\|^2 &\leq \epsilon^2 \end{aligned}$$

As we saw in the one-target case, a main drawback of this formulation is the difficulty associated with the inequality constraint. Thus we look for an alternate way of solving this problem, and in this pursuit, we study the dual problem of (INEQ2).

The conjugate duality theory of Rockafellar, et al, [**?**], [**?**] will be applied as follows: Let V and W be two Hilbert spaces and let $\mathcal{L}$ be a bounded linear transformation from V to W. If $f_1 : V \to R^e$ and $f_2 : W \to R^e$ are proper covex functions (R^e denotes the extended reals) then the duality theory gives the equivalence of two optimization problems:

$$\inf_{v\in V}\{f_1(v) + f_2(\mathcal{L}(v))\} = -\inf_{w\in W}\{f_1^*(\mathcal{L}^*(w)) + f_2^*(-w)\}$$

where f_j^* are conjugate functions of f_j, $j = 1, , 2$. That is,

$$f_1^*(v) = \sup_{v^* \in V} \{< v, v^* >_V - f_1(v^*)\}$$

and

$$f_2^*(w) = \sup_{w^* \in W} \{< w, w^* >_W - f_2(w^*)\}$$

We begin by defining several functions that we will use later. We will use the notation:

$$V = L^2((0, L) \times (0, T))$$

and

$$W = L^2(0, L).$$

Then we define a linear map $F : V \to V$ such that

$$F(v) = y$$

where y is the solution of (1) and a linear map $E : L^2(0, T; H_0^1(0, L)) \to W$ such that

$$E(y) = y(x, T).$$

We note that in practice, this map will be applied to a solution of (1), which belongs to the domain of this map. We also set $\mathcal{L} = E \circ F$ and $\mathcal{L}^*$ as the adjoint of $\mathcal{L}$.

We now define $f_1 : V \to R^e$ and $f_2 : W \to R^e$ by

$$f_1(v) = \frac{1}{2}\|v\|_V^2 + \frac{\alpha_2}{2}\|F(v)\|_V^2$$

$$f_2(w) = \begin{cases} 0 & \text{if} \quad \|w - y_T\| \leq \beta \\ +\infty & \text{otherwise} \end{cases}$$

The conjugate functions of f_1 and f_2 are

$$\begin{aligned} f_1^*(v) &= \sup_{v^* \in V} \{< v, v^* >_V - f_1(v^*)\} \\ &= \sup_{v^* \in V} \left\{< v, v^* >_V - \frac{1}{2}\|v^*\|_V^2 - \frac{\alpha_2}{2}\|F(v^*)\|_V^2 \right\} \\ &= < v, G^{-1}(v) >_V - \frac{1}{2}\|G^{-1}(v)\|_V^2 - \frac{\alpha_2}{2}\|FG^{-1}(v)\|_V^2 \end{aligned}$$

where $G = I + \alpha_2 F^* F$ and, as in the one-target case,

$$f_2^*(w) = \beta\|w\| + < w, y_T >_W$$

Before we state the dual problem, we compute

$$\begin{aligned} f_1^*(\mathcal{L}^* w) &= < \mathcal{L}^* w, G^{-1}\mathcal{L}^* w >_V - \frac{1}{2}\|G^{-1}\mathcal{L}^* w\|_V^2 - \frac{\alpha_2}{2}\|FG^{-1}\mathcal{L}^* w\|_V^2 \\ &= < w, \mathcal{L}G^{-1}\mathcal{L}^* w >_W - \frac{1}{2} < w, \mathcal{L}G^{-2}\mathcal{L}^* w >_W - \frac{\alpha_2}{2} < w, \mathcal{L}G^{-1}F^* F G^{-1}\mathcal{L}^* w >_W \\ &= < \Lambda_2(w), w >_W \end{aligned}$$

where

$$\Lambda_2 = \mathcal{L}G^{-1}\mathcal{L}^* - \frac{1}{2}\mathcal{L}G^{-2}\mathcal{L}^* - \frac{\alpha_2}{2}\mathcal{L}G^{-1}F^*FG^{-1}\mathcal{L}^*$$

We note that

$$G^{-1} = (I + \alpha_2 F^*F)^{-1} = \frac{1}{\alpha_2}\left(\frac{1}{\alpha_2}I + F^*F\right)^{-1}$$

So, for α_2 large enough, $G^{-1} \approx \frac{1}{\alpha_2}(F^*F)^{-1}$ and thus $\Lambda_2 > 0$. Now we state the dual problem:

$$-\inf_{w\in W}\{< \Lambda_2(w), w >_W + \beta\|w\| - < w, y_T >_W\} \tag{7}$$

Applying variational inequality, solving (7) is equivalent to: find $w^* \in W$ such that for every $w \in W$

$$< 2\Lambda_2(w^*), w - w^* >_W + \beta\|w\| - \beta\|w^*\| \geq < w - w^*, y_T >_W \tag{8}$$

Using $w = 0$ and $w = 2w^*$ in (8) we can show the inequality becomes the equation

$$< 2\Lambda_2(w^*), w^* >_W + \beta\|w^*\| = < w^*, y_T >_W \tag{9}$$

4.2 Multiobjective formulation

Another way of describing the two target problem is by using the multiobjective function:

$$(P2) \qquad \min \quad \frac{1}{2}\|v\|_V^2 + \frac{\alpha_1}{2}\|y(x,T) - y_T\|_W^2 + \frac{\alpha_2}{2}\|y\|_V^2$$

subject to

$$\begin{aligned} y_t - y_{xx} &= v(x,t) \\ y(0,t) = y(L,t) &= 0 \qquad t \in (0,T) \\ y(x,0) &= 0 \qquad x \in (0,L) \end{aligned}$$

This formulation is much easier to compute, but it is not clear how to choose α_1 and α_2 so that (P2) indeed solves the original two target problem narrative. Thus it crucial to investigate this issue. We begin our investigation by considering the dual problem associated with (P2). We define $f_1 : V \to R^e$ and $f_2 : W \to R^e$ such that

$$f_1(v) = \frac{1}{2}\|v\|_V^2 + \frac{\alpha_2}{2}\|F(v)\|_V^2$$

$$f_2(w) = \frac{\alpha_1}{2}\|w - y_T\|_W^2$$

Now compute the conjugate functions. We have already calculated the conjugate of f_1 in the previous section:

$$f_1^*(\mathcal{L}^*w) = < \Lambda_2(w), w >_W$$

The conjugate of f_2 is given by

$$f_2^*(w) = \frac{1}{2\alpha_1}\|w\|^2 - < w, y_T >_W .$$

Thus the dual problem for (P2) is

$$-\inf_{w\in W}\left\{<\Lambda_2(w),w>_W+\frac{1}{2\alpha_1}\|w\|^2-<w,y_T>_W\right\} \tag{10}$$

It can be shown that solving (10) is equivalent to solving

$$(2\Lambda_2+\frac{1}{\alpha_2}I)w=y_T \tag{11}$$

If w^* is a solution for (11) then after multiplying it by w^* and integrating, we get

$$<2\Lambda_2(w^*),w^*>_W+\frac{1}{\alpha_1}\|w^*\|^2=<w^*,y_T>_W \tag{12}$$

Comparing (9) and (12) we can derive

$$\alpha_1=\frac{\|w^*\|^2}{\epsilon} \tag{13}$$

where w^* is a solution of (11).

4.3 Numerical test

Unlike in the one target case, it is not clear how to obtain the solutions of the original problems from the solutions of the dual problems. However, (11) gives a clue to approximate (INEQ2) by (P2). We propose the following algorithm:

Algorithm DUAL_PRIMAL

1. Given $\alpha^0>0$, Set relchange = 1, tolerance, MAXCNT, and $k=0$.
2. Solve the dual problem (11) with $\alpha=\alpha^k$ to obtain a solution w^k
3. while relchange > tolerance & $k\leq$ MAXCNT
   (a) Set $\alpha^{k+1}=\frac{1}{\epsilon}\|\ w^k\|_{L^2}$.
   (b) Set $w^{k+1}=$ solution of (11) with $\alpha=\alpha^{k+1}$
   (c) Set relchange $=\frac{\|w^{k+1}-w^k\|}{\|w^{k+1}\|}$.
   (d) Set $k=k+1$
4. Use the α obtained from step 3 for α_1 in (P2). Fix a large enough α_2 and solve (P2).

Here is the discretized version of (P2)

$$(P2_F)\qquad \min_{(v,y)\in R^{sz}\times R^{sz}}\quad \frac{1}{2}\|v\|^2+\frac{\alpha_1}{2}\|By-y_T\|^2+\frac{\alpha_2}{2}\|y\|^2$$

subject to

$$[-I\ \ A]\begin{pmatrix} v\\ y\end{pmatrix}=0$$

Note that B is the same as M in $(P1_F)$. As in the one-target case, $\mathcal{L} = BA^{-1}$ and $\mathcal{L}^* = (A^t)^{-1}B^t$. Furthermore,

$$F = A^{-1}$$

$$G = I_{sz} + \alpha_2(AA^t)^{-1}$$

$$\Lambda_2 = \mathcal{L}G^{-1}\mathcal{L}^t - \frac{1}{2}\mathcal{L}G^{-2}\mathcal{L}^t - \frac{\alpha_2}{2}\mathcal{L}G^{-1}(AA^t)^{-1}G^{-1}\mathcal{L}^t$$

Below is the two different runs of DUAL_PRIMAL (**Source code: TwoTargets.m**). We used $m = 10, n = 20, \epsilon = 0.0001$, and $\alpha^0 = 1$.

Run 1: We used $\alpha_2 = 10$ for (P2)

```
>> TwoTargets

normofy =

   1.13322417765573


tolerance =

    1.000000000000000e-009

Number of iterations on alpha: 4

The values of alpha chosen by the algorithm
  1.0e+005 *

   0.00001000000000
   0.05642720935889
   2.52010987654221
   2.53932776528601
   2.53933112186030

CPU time taken: 0.291000 seconds
           x-value   approx y at t=T        Target y_T
                 0                 0                 0
   0.10000000000000   0.08998611784380   0.09000000000000
   0.20000000000000   0.15997364516693   0.16000000000000
   0.30000000000000   0.20996380040241   0.21000000000000
   0.40000000000000   0.23995750988554   0.24000000000000
   0.50000000000000   0.24995534828507   0.25000000000000
   0.60000000000000   0.23995750988554   0.24000000000000
   0.70000000000000   0.20996380040242   0.21000000000000
   0.80000000000000   0.15997364516693   0.16000000000000
   0.90000000000000   0.08998611784380   0.09000000000000
   1.00000000000000                 0                 0
```

```
Deviation from the target is: targetDeviation =

    1.000000000214037e-004

Norm of the control is 1.343874.
```

Run 2: We used $\alpha_2 = 100$ for (P2)

```
>> TwoTargets

normofy =

   1.12981545595572


tolerance =

    1.000000000000000e-009

Number of iterations on alpha: 5

The values of alpha chosen by the algorithm
  1.0e+006 *

   0.00000100000000
   0.00575771092576
   2.10482194472038
   2.24680373026757
   2.24683000142191
   2.24683000597579

CPU time taken: 0.300000 seconds
           x-value   approx y at t=T        Target y_T
                 0                 0                 0
   0.10000000000000   0.08998614027937   0.09000000000000
   0.20000000000000   0.15997366849576   0.16000000000000
   0.30000000000000   0.20996380635576   0.21000000000000
   0.40000000000000   0.23995749532282   0.24000000000000
   0.50000000000000   0.24995532489366   0.25000000000000
   0.60000000000000   0.23995749532282   0.24000000000000
   0.70000000000000   0.20996380635576   0.21000000000000
   0.80000000000000   0.15997366849576   0.16000000000000
   0.90000000000000   0.08998614027937   0.09000000000000
   1.00000000000000                 0                 0
```

```
Deviation from the target is: targetDeviation =

    1.000000000064129e-004

Norm of the control is 1.398574.
```

The above runs demonstrate that the algorithmic choice of α_1 is very accurate so that the inequality constraint of (INEQ2) is satisfied at the boundary. Furthermore, they also demonstrate that choice of α_2 correctly influence the norm of the state solution y. (Compare "normofy" values.)