

## Chapter 2

# THE STV APPROACH TO FINANCIAL OPTIMAL CONTROL MODELS

### 1. Introduction

In this chapter, a particular case of the optimal financial control problems, which has one state function and one control function that is approximated by a step-function, is discussed. Before the problem is defined, it is necessary to cover some concepts and transformations in Section 2.2 and Section 2.3, to explain the problems and algorithms that will be introduced in later sections and chapters. As part of the computer software package SCOM (that was constructed by Craven and Islam), “nqq” is used as a DE solver in the algorithms in this research. This program is described in Section 2.4. Then a simplified control problem is introduced in Section 2.5. The computational algorithms, which are used to solve this simplified control problem, are indicated in Section 2.6. Some problems with different fitting functions will be discussed later in Chapter 5. Lastly, graphs and tables in Section 2.7 represent all the computing results of this problem. The analysis of the results is discussed in Section 2.8

### 2. Piecewise-linear Transformation

The idea of piecewise-linear transformation of the time variable was first introduced by Teo [40, 1991], but the time intervals were mapped into  $(j, j + 1)$  instead of  $(jh, (j + 1)h)$ ,  $h = 1/n$ , where  $n$  is the number of time intervals. In Lee, Teo, Rehbock and Jennings [51, 1997], the time transformation is described by  $dt/ds = v(s)$ , where  $v$  is a piece-wise constant transformation. In this book, a similar idea is used, but the implementation is a little simpler, not requiring another differential equation. A non-linear transformation of the time scale given by Craven [15, 1998], is introduced in Section 2.3. The transfor-

mations in this chapter are the essential part of the algorithms in the research, and will be directly applied in Section 2.6.

Consider a financial optimal control model, to minimize an integral:

$$\text{MIN } J = \int_0^1 [x(t) - \phi(t)]^2 dt \quad (2.1)$$

subject to:

$$\dot{x}(t) = u(t), x(0) = x_0 \quad (2.2)$$

$$0 \leq u(t) \leq 1, 0 \leq t \leq 1 \quad (2.3)$$

The time interval  $[0,1]$  is divided as follows:

$$0 = t_0 < t_1 < t_2 < \dots < t_j < t_{j+1} < \dots < t_{r-1} < t_r = 1;$$

so there are  $r$  subintervals:

$$[t_0, t_1], [t_1, t_2], \dots, [t_{j-1}, t_j], \dots, [t_{r-1}, t_r].$$

A scaled time  $\tau$  is constructed here to replace the real time  $t$  in computation;  $\tau$  will be used later in the “nqj” package (differential equation solver).

The scaled time  $\tau$  takes values:

$$\tau = 0, h, 2h, \dots, jh, (j+1)h, \dots, (r-1)h, rh = 1$$

where  $h = 1/r$ , thus the time intervals  $[0, 1]$  is divided into  $r$  equal intervals.

The relationship between  $\tau$  and  $t$  can be expressed by the following formula:

$$t = \varphi(\tau) = h^{-1} * (t_{j+1} - t_j) * (\tau - h * (j - 1)) + t_j \quad (2.4)$$

where  $\tau \in [0, 1]$ .

Thus the subdivision point  $t = t_j$  maps to point  $\tau = jh$ .

Now the control  $u(t)$  takes values:

$$u(t) = u_0, u_1, u_2, \dots, u_j, \dots, u_{r-1}$$

in successive subintervals:

$$[0, h], [h, 2h], [2h, 3h], \dots, [jh, (j+1)h], \dots, [(r-1)h, 1]$$

Define  $x(t) := x(\varphi(\tau))$ .

Then the dynamic equation  $x(0) = x_0, \dot{x}(t) = u(t)$  transforms to:

$$\begin{aligned} \dot{x}(t) &= \dot{x}(\varphi(\tau))(d/d\tau)\varphi(\tau) \\ &= u(t) := h^{-1} * (t_{j+1} - t_j) * u_j \\ &\quad (\text{for } jh < \tau < (j+1)h) \end{aligned} \quad (2.5)$$

The objective function transforms to the sum of integrals:

$$J = \sum_{j=0}^{r-1} J_j = \sum_{j=0}^{r-1} \int_{jh}^{(j+1)h} [x(\varphi(\tau)) - \phi(\varphi(\tau))]^2 h^{-1} * (t_{j+1} - t_j) d\tau \quad (2.6)$$

### 3. Non-linear Time Scale Transformation

In some time-optimal control problems or optimal control problems that are over a long time interval, the large number of subintervals of the time scale will cause computational difficulty. The optimal control functions lead to stable values monotonically when time  $t$  becomes large. So it would be useful if a suitable non-linear transformation of time scale is practical, (see reference [15, 1998]). With a suitable non-linear time scale, a few subintervals may be enough to get the same accuracy of large subintervals.

Goh and Teo [33, 1987] introduced a change of time scale for a time-optimal control problem while the terminal time  $T$  is a variable. Let  $t = T\tau$  where  $\tau$  is the new time variable mapping in  $[0,1]$ .  $t$  is again written for  $\tau$ , then problem becomes a fixed-time optimal control problem with time interval  $[0,1]$  and parameter  $T$ . This allows  $T$  to be computed accurately without being interpolated between subdivision points.

This makes the original financial optimization problem as the following:

$$\text{MIN}_{x(\cdot), u(\cdot)} \int_0^T f(x(t), u(t), t) dt + \Phi(x(T))$$

subject to:

$$\begin{aligned} x(0) &= x_0, \dot{x}(t) = m(x(t), u(t), t) \quad (0 \leq t \leq T) \\ g(x(t), u(t)) &\leq b(t) \quad (0 \leq t \leq T) \\ q(x(T)) &= 0 \end{aligned}$$

Transform into:

$$\text{MIN}_{x(\cdot), u(\cdot)} \int_0^1 f(x(t), u(t), t) T dt + \Phi(x(1))$$

subject to:

$$\begin{aligned} x(0) &= x_0, \dot{x}(t) = m(x(t), u(t), t) T \quad (0 \leq t \leq 1) \\ g(x(t), u(t)) &\leq b(t) \quad (0 \leq t \leq 1) \\ q(x(1)) &= 0 \end{aligned}$$

This transformation will be used later in the computational methods 4.1 - 4.5 for the financial model in Chapter 4.

Now consider, an objective function with a discount factor  $e^{-\alpha t}$  where  $\alpha$  is positive, thus:

$$\int_0^T e^{-\alpha t} f(x(t), u(t)) dt$$

Define time  $t$  as:

$$t = -\alpha^{-1} \log(1 - \beta\tau), \text{ where } \beta = 1 - e^{-\alpha T}, \tau \in [0, 1]$$

The objective function becomes:

$$(\beta/\alpha) \int_0^1 f(x(t), u(t)) dt$$

and the differential equation changes from  $\dot{x}(t) = m(x(t), u(t), t)$  to:

$$\dot{x}(t) = \frac{(\beta/\alpha)m(x(t), u(t))}{(1 - \beta t)} \quad (0 \leq t \leq 1), \quad x(0) = x_0$$

For  $T$  is fixed and large, the denominator satisfies  $1 - \beta\tau \geq 1 - \beta > 0$ . If  $T$  is a variable to be optimized over, then  $1 - \beta$  replaces the variable  $T$ , in an interval  $0 < \epsilon \leq 1 - \beta \leq \frac{1}{2}$ .

For example, in a fixed-time optimal control problem, if there is some “distinguished time”  $\hat{T} \in [0, T]$ , then an additional constraint  $r(x(\hat{T})) = 0$  must be satisfied. A transformation of  $t \in [0, T]$  to  $\tau \in [0, \frac{1}{2}]$  maps  $\hat{T}$  to  $\frac{1}{2}$ , thus to a subdivision point. In particular:

$$\tau = \frac{1}{2}t/\hat{T} \quad (0 \leq t \leq \hat{T}), \quad t = 2\hat{T}\tau \quad (0 \leq \tau \leq \frac{1}{2})$$

$$\tau = \frac{1}{2} + \frac{1}{2} \frac{(t - \hat{T})}{(T - \hat{T})} \quad (\hat{T} \leq t \leq T), \quad t = \hat{T} + 2(T - \hat{T})(\tau - \frac{1}{2}) \quad (\frac{1}{2} \leq \tau \leq 1)$$

At  $\tau = \frac{1}{2}$ , the piecewise-linear transformation is not differentiable, and the co-state may be discontinuous.

#### 4. A Computer Software Package Used in this Study

The computer package SCOM ([18, 2001]) is a tool to solve step-function optimal control problems on MATLAB 5.2, on a Macintosh computer. It is noted that a program written and compiled in C language will run faster than a MATLAB program for the same computation. However MATLAB is a matrix computation language, so it requires much less programming work in calculating matrix operations on MATLAB than other computer languages, such as C language and Fortran, etc. The program “constr” is a constrained optimization package in MATLAB’s Toolbox, based on a SQP (sequential quadratic programming) method. “constr” will use gradients if supplied; otherwise it will estimate gradients by finite difference. In non-linear programming methods, the SQP method is very successful. The method closely mimics Newton’s method for constrained optimization, just as it is done for unconstrained optimization. Using a quasi-Newton updating method, an approximation is made of the Hessian of the Lagrangian function at each major iteration. It is then used to generate a Quadratic Programming sub-problem whose solution is used to form a search direction for a line search procedure. In the research, we build another subroutine to be called as the objective function in “constr”.

The state functions in this research were solved by the differential equation solver “nqq” in the SCOM package (see details in Appendix C). When results of the state come out on the grid-points, the objective function  $J(u)$  becomes a function  $\tilde{J}(u_1, u_2, \dots, u_N)$  of  $N$  variables that can also be calculated by “nqq”.

When the subroutines for the state functions and objective function are constructed, the MATLAB program “constr” is then used to obtain the optimal solution of the problem with respect to optimal switching times. In the program, the control  $u$  is approximated by a step-function.

## 5. An Optimal Control Problem When the Control can only Take the Value 0 or 1

From the point of view of control theory, the bang-bang optimal control happens when the systems of the optimal control problems are linear in control. The Nerlove-Arrow model [63, 1962] is an example of a bang-bang control following a singular arc control. Now, introduce a typical bang-bang optimal control problem.

We consider an optimal control problem:

$$\text{MIN} J(u) = \int_0^T [x(t) - \phi(t)]^2 dt \quad (2.7)$$

subject to:

$$x(0) = 0 \quad (2.8)$$

$$\dot{x}(t) = u(t) \quad (2.9)$$

$$u(t) = 0 \text{ or } 1 \quad (0 \leq t \leq 1) \quad (2.10)$$

In this case, the statefunction  $x(\cdot)$  is assumed piece-wise smooth, and the control  $u(t)$  jumps many times between 0 and 1 during the time interval  $[0, T]$ . Here, simplify the problem to  $T = 1$ . Now use a step-function:

$$v(t) = \frac{1}{2} + (-1)^k [v(0) - \frac{1}{2}] \text{ for } k = 0, 1, 2, \dots, n$$

$$v(0) = 0 \text{ or } 1$$

to approximate the control  $u(t)$ . Here the target function is:  $\phi(t)$ . Let  $\phi(t) = \frac{1}{2}t$ .

Observe that if having instead  $u(t) \in [0, 1]$ , then  $u(t) = \frac{1}{2}$  ( $0 \leq t \leq 1$ ) would be optimal, with  $J(u) = 0$ . But if  $u(t)$  must be either 0 or 1, then

$u(t)$  will jump many times between 0 and 1. Suppose a further restriction is made that, for some given integer  $n$ , time interval  $[0, 1]$  is divided into  $n$  equal subintervals  $[0, 1/n], [1/n, 2/n], \dots, [j/n, (j+1)/n], \dots, [(n-1)/n, 1]$ ; ( $u(t)$  takes a constant value (1 or 0) on each subinterval  $[j/n, (j+1)/n]$ , ( $j = 0, 1, \dots, n-1$ ). So if  $u(t)$  takes the values in this pattern 1, 0, 1, 0, ..., 1, 0 on the successive subinterval, then each subintervals contributes  $1/(12n^3)$  to  $J(u)$ ,  $J(u) = 1/(6n^2) \rightarrow 0$  as  $n \rightarrow \infty$ . But the optimal value  $J = 0$  is not reached, unless an infinite number of jumps are allowed.

Now a term  $K * n$  is added to the objective function. (2.7) is modified as follows:

$$\text{MIN } F = J(u) + kn = \int_0^T [x(t) - \phi(t)]^2 dt + Kn$$

Here,  $K$  is the cost of changing control and  $n$  is the number of time intervals. The objective function of the optimal control problem becomes a cost function. The algorithms in the next section are used for solving these kind of optimal control problems.

## 6. Approaches to Bang-Bang Optimal Control with a Cost of Changing Control

In this section, the computational methods for solving bang-bang optimal control problems with a cost of switching control are introduced. Simply modifying certain parts of these methods can satisfy a class of similar problems.

Suppose the number  $N$  of switching times is fixed by  $N = nm$ , say switching times  $t_1, t_2, \dots, t_{nm}$ . Now consider (2.7) as a function of the switching times, say  $J(u) = Q(t_1, t_2, \dots, t_{nm})$ . Then the minimum of this function is computed, starting from a small value of  $N$ . Here the components of the vector  $um = (um_1, um_2, \dots, um_{nm})$  are the lengths between each switching time. The vector  $um$  must satisfy the upper and lower bounds 0 and 1, thus  $0 \leq t_{j-1} \leq t_j \leq 1$ , and a constraint  $\sum_{i=1}^N um_i = 1$  must be satisfied. The vector  $xm = (xm_1, xm_2, \dots, xm_{nm})$  represents the values of the state function takes at each switching time. The algorithms are as follows:

Algorithm 2.1 Main Program (see project1\_1.m in Appendix A.1)

**Step 1.** Initialization. Set the maximum number of function evaluations,  $par$ , which is the system parameter of the MATLAB “constr” function, and also another system parameter  $par(13) = 1$  (1 represents the number of the equation constraint in the minimization problem), and a vector of the parameters which are used in the whole subroutines,  $par = [\text{number of the state components, number of control components, } nm = \text{number of total subintervals}], \text{ arbitrary starting lengths of switching time intervals}$

$um0 = (um0_1, um0_2, \dots, um0_{nn})$ , and vector  $ul$  = lower bound of  $um$ , vector  $uu$  = upper bound of  $um$ , thus  $uu \leq um \leq ul$ . Initialize the value of the state function  $xinit$ .

**Step 2.** Call the MATLAB “constr” function. In turn, “constr” calls the “Minimizing Program” to calculate the minimization of the calling program with respect to the optimal vector  $um$ .

**Step 3.** Input the optimal result  $um$  into “Minimizing Program” again to obtain the results of the objective function  $J_{nn}$  (the last value of the calculation) and the state vector  $xm$ , corresponding to the optimal  $um$ .

**Step 4.** Attach a cost  $K$  to  $nn$ . Add  $K * nn$  to the objective function (2.7) and calculate  $F = J_{nn} + K * nn$ .

**Step 5.** Set a bigger  $nn$ ; then go back to Step 2; EXIT when the result of cost function  $J$  stops decreasing.

Algorithm 2.2: Minimizing Program (see project1\_2.m in Appendix A.1)

**Step 1.** Initialization. Input vectors  $um$ ,  $par$  and initial state  $xinit$ . Set the initial state  $xm(1) = xinit$ , and  $nx = par(1)$ , the number of the state components,  $nu = par(2)$ , the number of the control components; initialize scaled time  $t = 0$ , subinterval counter  $it = 1$ ,  $hs = 1/nn$ (length of each equal subinterval). Choose the “Input function for dynamic equation” as the right side of equation (2.9) to calculate the differential equation, input  $um$ .

**Step 2.** Call SCOM package function “nqq” with the stated “Input function for dynamic equation” to solve the differential equation (2.9) of the state function  $x(t)$ . Tabulate the solutions as the components of the vector  $xm = [x(1), x(2), \dots, x(nn)]$ .

**Step 3.** Set the initial state  $zz = 0$ . Set initial scaled time  $t = 0$ , subinterval counter  $it = 1$  again, choose the “input function for integration calculation” for SCOM function “nqq” for solving the integration of the objective function (2.7), input vector  $xm$  and  $um$ .

**Step 4.** Call SCOM function “nqq” with the stated “Input function for integration calculation” to calculate:

$$J(u(t)) = \sum_{j=0}^{nn-1} J_i$$

where:

$$J_i = \int_{jh}^{(j+1)h} [x(\varphi(\tau)) - \phi(\varphi(\tau))]^2 h^{-1} (t_{j+1} - t_j) d\tau$$



$$h = 1/n$$

when:

$$jh < \tau < (j+1)h$$

by solving the differential equation:

$$w(0) = 0, \dot{w}(t) = [x\varphi(\tau) - \phi(\varphi(\tau))]^2 h^{-1}(t_{j+1} - t_j)$$

when:

$$jh < \tau < (j+1)h$$

Tabulate the results in  $w(\cdot)$  as the components of vector

$$jm = [j(1), \dots, j(nn)].$$

**Step 5.** Take the last result of the vector  $jm$  as the value of the objective function, and calculate the constraint function of “Minimizing program”, which is:  $g(um) = \sum_{i=1}^N um_i - 1$ .

Algorithm 2.3: Input function for dynamic equation (see project1\_3.m in Appendix A.1)

**Step 1.** Initialization. Input scaled time  $t$ , subinterval counter  $it$ , the length of subintervals  $hs$ , and vector  $um$ , and set the number of subintervals  $nn = 1/hs$ .

**Step 2.** Set the control policy as vector  $u = [u(1), u(2), \dots, u(nn)]$  with alternating values 1 and 0 (as in 2.10) in successive subintervals.

**Step 3.** Construct the right side of the differential equation for the state function using the piecewise-linear transformation in (2.5).

Algorithm 2.4: Input function for integration calculation (see project1\_4.m in Appendix A.1)

**Step 1.** Initialization. Input scaled time  $t$ , subinterval counter  $it$ , the length of subintervals  $hs$ , vector  $xm$  representing the values of the state function at each switching time, and also a new initial state  $z$  for integral, and the vector  $um$ . Set the number of subintervals  $nn = 1/hs$ .

**Step 2.** Use the linear interpolation to get an estimate “ $xmt$ ” of the state, in a time  $t$  between grid-points  $0, h, 2h, \dots, nn * h$ , where  $h = 1/nn$ .

**Step 3.** Add up components of the lengths of the switching time intervals in  $um$  to obtain the switching times “ $t_j$ ” in (2.4).

**Step 4.** Construct the right side of the equation (2.4) to obtain the time variable  $t$ .

**Step 5.** Calculate the integral in (2.6) at scaled time  $t$ .

When the number of switching time  $N$  increases, the cost function decreases because of the better approximation. While calculating a minimization problem  $KN + J_N$ , the term  $KN$  increases with  $N$  increasing, and  $J_N$  decreases with bigger  $N$ . It can be found that the cost of changing control is very critical in the cost function. A proper chosen cost  $K$  will efficiently lead the cost function to the minimum. The analysis of the cost will be discussed in later sections.

## 7. An Investment Planning Model and Results

In this section, the computational results reported by a set of graphs will be introduced to verify the algorithms developed in Section 2.6. First we will introduce the fitting function. In this example, the fitting function is set to be  $0.4 * t$ . The state function  $x(t)$  is used to approximate this fitting function. The formulae for this financial optimal control model is shown as follows:

$$\text{MIN } J = \int_0^1 |x(t) - 0.4 * t^2| dt \quad (2.11)$$

subject to:

$$\dot{x}(t) = u(t) \quad (2.12)$$

$$x_0 = 0 \quad (2.13)$$

$$u(t) = 1, 0 \quad (2.14)$$

$$0 \leq t \leq 1 \quad (2.15)$$

where  $x(t)$  = stock price,  $u(t)$  = the proportion of total investment in stocks compared to other forms of financial investment.

Although the above model is an illustrative model, it can, however, represent an interesting financial decision making problem. The state equation represents the dynamics of the price of a stock. It is assumed that the change in the price of a stock is determined by the proportion of allocation of total funds for purchasing a stock. The objective of this control problem is to determine the value of  $u(t)$  (which only takes the value of 1 or 0) which can optimize the objective function so as to minimize the deviation of the state variable from its target value specified. Therefore this model (2.11 to 2.15) is an investment planning model with some sub-utilization criterion included in the model.

The target function is  $0.4 * t^2$ . First set  $n = 2$  ( $n$  is the number of time intervals), and control takes as 1,0 in time intervals  $[0, t_1]$ ,  $[t_1, 1]$ . The model 2.11 to 2.15 was solved with these parameter values and the results are shown

in Appendix B.1 and are represented by Figure 2.1. In Figure 2.1, it is shown that during the planning horizon  $[0,1]$ , the control only jumps once at time  $t = 0.127$ . The state vector  $xm$  takes the value  $[0.127, 0.127]$  at the grid-points of two subintervals  $[0, 0.127]$  and  $[0.127, 1]$ . The approximation between the state function  $xm$  and the fitting function  $0.4 * t^2$  is not very good when the number of the switching times is quite small ( $n$  is only 2). “\*\_” represents the state function  $x(t)$  and “.” represents the fitting function  $0.4 * t^2$  in the following graph.

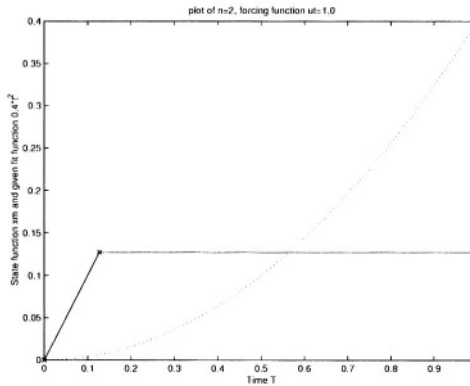


Figure 2.1. Plot of  $n=2$ , forcing function  $u_t=1,0$

Then increase  $n$  to 4. During the whole time  $[0,1]$ , the control jumps three times. A better approximation is shown in Figure 2.2 with more jumps of the control.

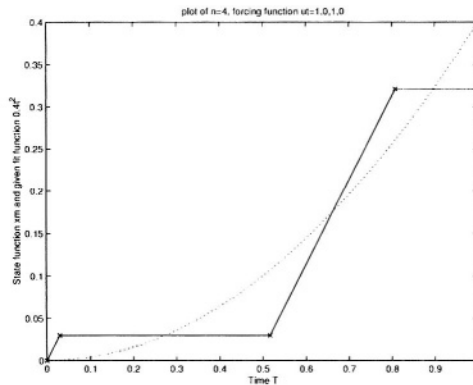


Figure 2.2. Plot of  $n=4$ , forcing function  $u_t=1,0,1,0$

Set  $n = 6$ , and run the program. A much better approximation than  $n = 4$  is shown in Figure 2.3 because of the increased switching times.

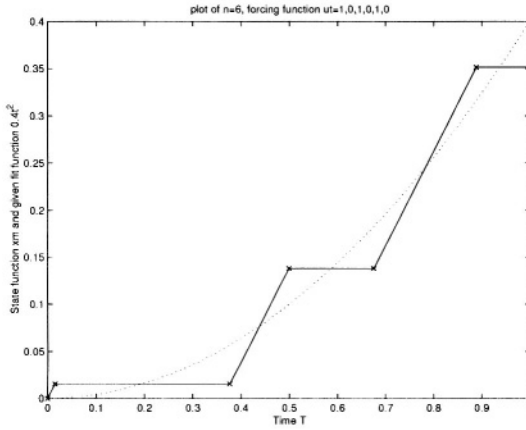


Figure 2.3. Plot of  $n=6$ , forcing function  $u_t=1,0,1,0,1,0$

When  $n = 8$ , a very close fit between state  $x_m$  and the given fitting function  $\phi(t)$  is shown in Figure 2.4. The result proves a very good convergence of the algorithms.

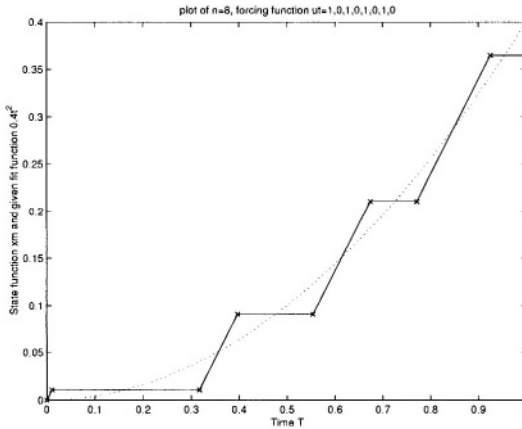


Figure 2.4. Plot of  $n=8$ , forcing function  $u_t=1,0,1,0,1,0,1,0$

In Figure 2.5, although the approximation between  $x_m$  and  $\phi(t)$  is still getting better when  $n = 10$ , the difference between the result of  $n = 8$  and  $n = 10$  is not as big as between  $n = 2$  and  $n = 4$ . The decreasing of the results of the objective function slows down when  $n$  is very large.

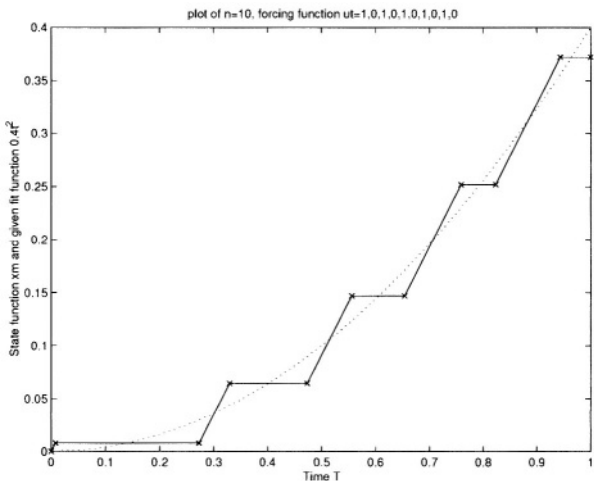


Figure 2.5. Plot of  $n=10$ , forcing function  $u_t=1,0,1,0,1,0,1,0,1,0$

The results of the objective function according to different numbers of the time intervals  $n$  are put in Table 2.1. The decreasing of the results of the objective function follows the increases number of the time intervals  $n$ . When  $n$  is small, the result of the objective function will decrease very fast, however this decrease will slow down when  $n$  becomes big. A good illustration is shown in Figure 2.6.

Table 2.1. Objective functions with the number of the switching times

$n$	$J$
2	0.062696
4	0.029085
6	0.018364
8	0.013369
10	0.010463

In Figure 2.6, the results of the objective function against the different number of the time intervals  $n$  are shown. From the graph, we will find out that the descent of the results of the objective function slows down when the number of time intervals  $n$  increases. The conclusion can be made that more jumps of the control in the time period  $[0, 1]$  give a better association between the state  $x(t)$  and fitting function  $0.4t^2$  (i.e. makes the financial system more stable along

its desired path) and leads the financial system to reach the best fit when  $n$  becomes infinity.

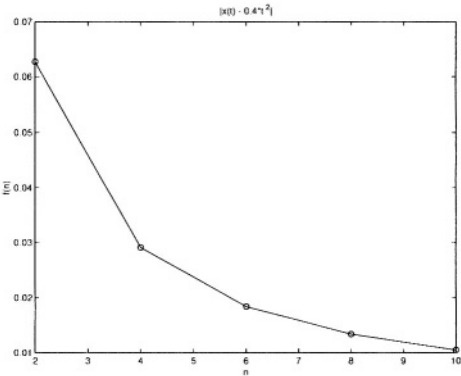


Figure 2.6. Plot of the values of the objective function to the number of the switching times

The optimal control problem in (2.11)-(2.15) is computed and the results are shown in the above graphs. The number of time intervals  $n = 2, 4, 6, 8, 10$  is set consequently. A better fit comes into being Figures 2.1, 2.2, 2.3, 2.4 and 2.5. In Figure 2.6, it is shown that the result of the objective function  $J$  decreases while the number of time intervals  $n$  increases. Same results are also shown in Table 2.1. Now a proper cost  $K$  is set and attached with the number of the time intervals  $n$  to the objective function  $J$ . The objective function (2.11) becomes:

$$\text{MIN } F = J(n) + kn = \int_0^1 |x(t) - 0.4 * t^2| dt + Kn \tag{2.16}$$

Use the algorithms to solve this new optimal control problem with the same constraints as in (2.12)-(2.15). Three different values of cost  $K$  are chosen for  $F(n) = J(n) + K * n$ . The results of adding  $K * n$  to the objective function are shown in Table 2.2, corresponding to  $n = 2, 4, 6, 8, 12$ .

Table 2.2. Costs of the switching control attached to the objective function

$K$	$n = 2$	$n = 4$	$n = 6$	$n = 8$	$n = 10$
$K=0.02$	0.102692	0.109085	0.138364	0.173369	0.210463
$K=0.002$	0.066696	0.037085	0.030364	0.029369	0.030463
$K=0.0002$	0.063096	0.029885	0.019564	0.014969	0.012463

From the results in Table 2.2, a conclusion can be made that only when cost  $K$  takes a certain value, it will lead the cost function to a minimum infinite switching times. When  $K = 0.02$ , the results of the cost function are increasing from  $n = 2$  to  $n = 10$ , and the increases become faster. When  $K = 0.0002$ , the results are decreasing, but this decrease begins to slow down when  $n$  increases. It is conjectured that when the number of the switching times  $n$  is getting very big, this decreases will stop at a certain value of  $n$ . Meaningful results from  $K = 0.002$  are given, and a minimum is obtained at  $n = 8$ . The following Figure 2.7 shows the results of the cost function against the number of the time intervals  $n$  while the cost  $K = 0.002$  is attached to the objective function. The bottom of the line in the figure is the minimum point  $n = 8$ .

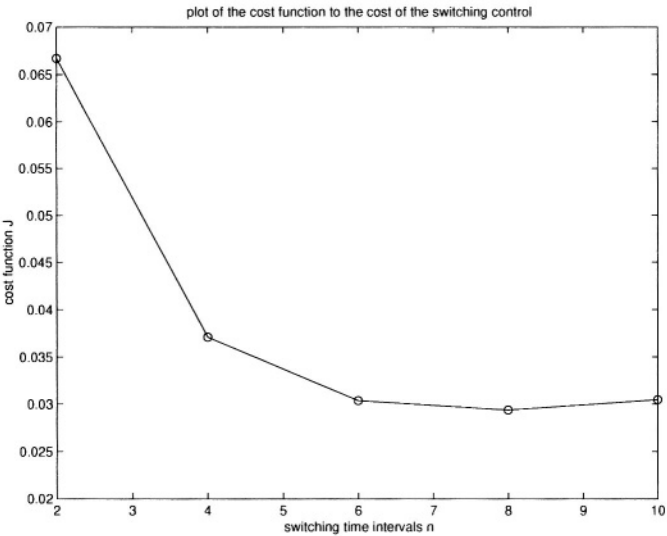


Figure 2.7. Plot of the cost function to the cost of switching control

This is an effective example of computational algorithms. All the results confirm the accuracy of computational algorithms in Section 2.6. A hand calculation of the part of the differential equation to exam the algorithms also gives a proof of computational accuracy. Some other experiments with different fitting functions will be discussed later in Chapter 5.

## 8. Financial Implications and Conclusion

The crucial aspect of the computational experiments undertaken here is that switching times and the cost of switching times have significant implications for optimal investment planning. As the graphical results showed in Section 2.7, different numbers of switching times lead to different results for the objective function of the financial model. While the number of switching times increases, the value of the objective function decreases. That means a better fit is obtained. When a cost of changing control at each switching time is added to the original objective, the perspective changes. In the present case, when the number of switching times increases, the value of the term, which includes cost and the number of the switching, also increases. This increase slows down the decreasing of the original objective function. When the number of switching times increase to a certain value, the cost function stops decreasing and instead increases. The intermediate point between the decreasing and increasing is the optimal point.

The value of the cost of switching significantly influence the values of the switching times and optimal control. Comparing with the result of the objective function, if the cost is too big, the cost functional will increase all the time. It can be explained that it costs too much to change a control at each time interval. The financial system can never reach an optimal solution. But if the cost is too small, it could not affect the cost function at all. Then the control will jump infinitely in the time period to search an optimal solution, which is difficult to realize in real computation. Therefore, the value of the cost of switching time affects the optimal number of switching time. In terms of the value of the optimal control, it is found that several jumps in the strategy of investment in the stock market, shown by 1,0 values of investment in the stock market are optimal. A higher cost for switching reduces the optimal value of switching times compared to a situation when there is no cost for control switching.

An optimal investment strategy, therefore, should always be made on the consideration of the cost of control switching to determine how often the investment strategy can be changed between 1 and 0.

In the next chapter, a financial oscillator model for optimal aggregative investment planning will be presented. Since the oscillator problem has a state function, which is a second-order differential equation, the computational algorithms require transforming the second-order differential equation into two equivalent first-order differential equations. A new time scaled transformation for better integral calculation will be introduced in Section 3.3. The corresponding transformations of the state functions and the objective function will also be contained in Section 3.3. The computational algorithms for the financial oscillator problem will be described in Section 3.4. The computer software packages for these algorithms will be introduced in Appendix A.2. Different patterns of the control might lead the computation to different minimum



points. The control with different initial starts was put into the program. The final result of the comparison will be the optimum of the control problem. The computing results of a particular oscillator problem are represented in graphs and tables in Section 3.6.



<http://www.springer.com/978-0-387-23569-1>

Optimal Control Models in Finance

A New Computational Approach

Chen, P.; Islam, S.M.N.

2005, XVIII, 201 p., Hardcover

ISBN: 978-0-387-23569-1