Journal of the Nigerian Association of Mathematical Physics
Volume 14 (May, 2009), pp 341 – 348

Convergence profile of a discretized scheme for constrained problem via the penalty-multiplier method

[1]O. Olotu and [2]S. A. Olorunsola,
[1]*Department of Mathematical Sciences,*
The Federal University of Technology,
P. M. B. 704, *Akure, Ondo State, Nigeria*
[2]*Department of Mathematical Sciences,*
The University of Ado-Ekiti,
P. M. B. 5373, *Ado-Ekiti, Ekiti State, Nigeria*

Abstract

An extended discretized scheme is proposed to examine the convergence profile of a quadratic control problem constrained by evolution equation with real coefficients. With an unconstrained formulation of the problem via the penalty-multiplier method, the discretization of the time interval and differential constraint is carried out. An operator, to circumvent the cumbersome calculation inherent in some earlier schemes, such as the function space algorithm, is established and proved. An example is considered to test the effectiveness and superiority of this scheme as it compares to other schemes in terms of convergence profile.

Keywords
Convergence, evolution, penalty-multiplier, operator.

1.0 Introduction

Many algorithms [2, 3, 4, 5, 6] have been developed to solve constrained quadratic continuous optimal control problems. Here, we are considering a new method, Discretized Continuous Algorithm (DCA) which is computationally more efficient. To numerically achieve this, the time interval and the differential constraint are discretized via the Euler's formula and finite difference method respectively. An unconstrained formulation of the constrained problem was obtained using the combination of the penalty and Multiplier methods [4, 5]. With this formulation, a bilinear form expression was obtained which formed a strong footing for the construction of an operator based on [5] reviewed method on function minimization by Fletcher and Reeves [2] and others [6,10]. The construction of this operator circumvented the unusual cumbersomeness inherent in the function space algorithm (FSA) [6] marred with many integral evaluations over a given interval. To this end, we are considering a generalized class of quadratic control problems for our developed scheme.

1.1 Materials and method

1.1.1 Generalized Problem 1

$$MinJ(x,u,t)_K = \int_0^T \left(px^2(t) + qu^2(t) \right) dt \qquad (1.1)$$

Subject to

[1]Corresponding author:
[1]e-mail address: segolotu@yahoo.ca

$$\dot{x}(t) = ax(t) + bx(t-r) + cu(t), \quad x(0) = x_0, \quad 0 \le t \le T,$$
$$x(t) = h(t)\ , t \in [-r, 0], \tag{1.2}$$

were, *p, q, r* are greater than zero, *x*(*t*), *u*(*t*) are in *R*(the set of real numbers),*a*,*b* and *c* are constants not necessarily positive, and $K = H[0,T]\ x\ L_2{}^q[-r,0]\ x\ L_2{}^q[0,T]$, is the product of the Sobolev space H[0, T] of absolutely continuous functions $x(\bullet)$ such that both $x(\bullet)$ and $\dot{x}(\bullet)$ are square integrable over the interval [0,*T*] and the Hilbert space $L_2{}^q[0,T]$ of equivalence classes of real-valued functions on [0,*T*] with norm defined by $\|x(t)\|_{L_2{}^q[0,T]} = (\int_0^T |x(t)|^2\, dt)^{\frac{1}{2}}, x(t) \in L_2{}^q[0,T]$

2.0 Discretization

By discretizing (1.1), (1.2), subdivide [0,*T*] into n equal intervals [t_k,t_{k+1}] at meshpoints $x_0 < x_1 < x_2 < ... < x_{n-1}$ where *n* - 1 is the number of partition points chosen arbitrarily, thus having (*n* + 1) partition points,with *xj* = *j***Δj*, *j* = 0,1,2,…,*n*, and Δ*j* = Δ*k* is the fixed length of each subinterval for *j* = *k*. By *j***Δj*, it means *j* multiplied by Δ_j. By Euler's scheme or finite difference method

$$\dot{x}(k) = \left(x(k+1) - x(k)\right) / \Delta_k, \ \ k = 0,1,2,3,...,n-1$$
$$\dot{x}_k(t_k) = ax_k(t_k) + bx_k(t_k - r_k) + cu_k(t_k) \tag{2.1}$$

We then have the generalized problem in discretized form as

$$\sum_{k=0}^{n} \Delta_k (px_k{}^2(t_k) + qu_k{}^2(t_k))$$
$$\textit{such that}$$
$$\dot{x}_k(t_k) = (x_{k+1}(t_{k+1}) - x_k(t_k))\Delta_k = ax_k(t_k) + bx_k(t_k - r_k) + cu_k(t_k) \tag{2.2}$$

3.0 Application of the penalty-multiplier method

Applying the penalty-multiplier method [2, 3, 5, 13] to (2.2), we have

$$\begin{aligned} Min(x,u,\mu,\lambda) = \sum_{j=0}^{n} (\Delta_k (px^2(t_k) + pu_k{}^2(t_k)) + \mu[x_{k+1}(t_{k+1}) - x_k(t_k) \\ - \Delta_k ax_k(t_k) - \Delta_k bx_k(t_k - r_k) - \Delta_k cu_k(t_k)]^2 \\ + \langle \lambda_k(t_k),\ x_{k+1}(t_{k+1}) - x_k(t_k) - \Delta_k ax_k(t_k) - \Delta_k bx_k(t_k) - \Delta_k cu_k(t_k) \end{aligned} \tag{3.1}$$

$$\begin{aligned} &= \sum_{k=0}^{n} \{x_k{}^2(t_k)\alpha_k + u_k{}^2(t_k)\beta_k + y_k{}^2(t_k)\mu + x_k(t_k)u_k(t_k)\delta_k \\ &+ x_k(t_k)y_k(t_k)v_k + x_k(t_k - r_k)y_k(t_k)m_k + y_k(t_k)u_k(t_k)n_k + x_k(t_k)x_k(t_k - r_k)p_k \\ &+ x_k{}^2(t_k - r_k)c_k + x_k(t_k - r_k)u_k(t_k)q_k + \lambda_k(t_k)x_k(t_k) \\ &- \lambda_k(t_k)x_k(t_k) - \lambda_k(t_k)x_k(t_k)\Delta_k a - \lambda_k(t_k)x_k(t_k - r_k)b - \lambda_k(t_k)u_k(t_k)\Delta_k c\} \end{aligned} \tag{3.2}$$

where

$$y_k(t_k) = x_{k+1}(t_{k+1}), \quad \alpha_k = \mu + 2\mu\Delta_k a + \Delta_k{}^2 a^2 \mu + p\Delta_k, \quad \beta_k = q\Delta_k + \Delta_k{}^2 c^2 \mu,$$

$$n_k = -2\mu\Delta_k c, \quad m_k = -2\mu\Delta_k b, \quad p_k = -2\mu\Delta_k b + 2\Delta_k{}^2 ab, .$$
$$\delta_k = 2\mu\Delta_k c + 2\Delta_k{}^2 ac\mu, \quad v_k = -2\mu - 2\Delta_k a\mu, \quad c_k = \mu\Delta_k{}^2 b^2, \quad q_k = 2\mu\Delta_k{}^2 bc$$

4.0 Construction of operator *V*

We now formulate the bilinear form expression as in [1,8] for equation (3.2)

$$\begin{aligned}
\langle z_{k1}(t_k), Vz_{k2}(t_k)\rangle = \sum_{k=0}^{n} \{ & x_{k1}(t_k)x_{k2}(t_k)\alpha_k + u_{k1}(t_k)u_{k2}(t_k)\beta_k + y_{k1}(t_k)y_{k2}(t_k)\mu \\
& + x_{k1}(t_k)u_{k2}(t_k)\delta_k + x_{k2}(t_k)u_{k1}(t_k)\delta_k + y_{k1}(t_k)x_{k2}(t_k)v_k + y_{k2}(t_k)x_{k1}(t_k)v_k \\
& + y_{k1}(t_k)x_{k2}(t_k - r_k)m_k + y_{k2}(t_k)x_{k1}(t_k - r_k)m_k + y_{k1}(t_k)u_{k2}(t_k)n_k + y_{k2}(t_k)u_{k1}(t_k)n_k \\
& + x_{k1}(t_k)x_{k2}(t_k - r_k)p_k + x_{k2}(t_k)x_{k1}(t_k - r_k)p_k + x_{k1}(t_k - r_k)x_{k2}(t_k - r_k)c_k + \\
& + x_{k1}(t_k - r_k)u_{k2}(t_k)q_k + x_{k2}(t_k - r_k)u_{k1}(t_k)q_k + \lambda_{k1}(t_k)y_{k2}(t_k) + \lambda_{k2}(t_k)y_{k1}(t_k) \\
& - \lambda_{k1}(t_k)x_{k2}(t_k) - \lambda_{k2}(t_k)x_{k1}(t_k) - \lambda_{k1}(t_k)x_{k2}(t_k)a\Delta_k - \lambda_{k2}(t_k)x_{k1}(t_k)a\Delta_k \\
& - x_{k2}(t_k - r_k)\lambda_{k1}(t_k)\Delta_k b - x_{k1}(t_k - r_k)\lambda_{k2}(t_k)\Delta_k b - \lambda_{k2}(t_k)u_{k1}(t_k)\Delta_k c \\
& - \lambda_{k1}(t_k)u_{k2}(t_k)\Delta_k c
\end{aligned} \tag{4.1}$$

$$Vz_{K2}(t_k) = \begin{pmatrix} V_{11} & V_{12} & V_{13} & V_{14} \\ V_{21} & V_{22} & V_{23} & V_{24} \\ V_{31} & V_{32} & V_{33} & V_{34} \\ V_{41} & V_{42} & V_{43} & V_{44} \end{pmatrix} \begin{pmatrix} x_{k2}(t_k) \\ u_{k2}(t_k) \\ h_{k2}(t_k) \\ \lambda_{k2}(t_k) \end{pmatrix}$$

$$= \begin{pmatrix} V_{11}x_{k2}(t_k) + V_{12}u_{k2}(t_k) + V_{13}h_{k2}(t_k) + V_{14}\lambda_{k2}(t_k) \\ V_{21}x_{k2}(t_k) + V_{22}u_{k2}(t_k) + V_{23}h_{k2}(t_k) + V_{24}\lambda_{k2}(t_k) \\ V_{31}x_{k2}(t_k) + V_{32}u_{k2}(t_k) + V_{33}h_{k2}(t_k) + V_{34}\lambda_{k2}(t_k) \\ V_{41}x_{k2}(t_k) + V_{42}u_{k2}(t_k) + V_{43}h_{k2}(t_k) + V_{44}\lambda_{k2}(t_k) \end{pmatrix} \tag{4.2}$$

where $z_k(t_k) = (x_k(t_k), u_k(t_k), h_k(t_k), \lambda_k(t_k))$

Setting $\dot{x}_k(t_k) = (x_{k+1}(t_{k+1}) - x_k(t_k)) / \Delta_k$ *in* (4.1.)

We have,

$$\begin{aligned}
\sum_{k=0}^{n} & x_{k1}(t_k)x_{k2}(t_k)\alpha_k + u_{k1}(t_k)u_{k2}(t_k)\beta_k + \mu\dot{x}_{k1}(t_k)\dot{x}_{k2}(t_k)\Delta_k{}^2 \\
& + \mu\dot{x}_{k1}(t_k)x_{k2}(t_k)\Delta_k + \mu x_{k1}(t_k)\dot{x}_{k2}(t_k)\Delta_k + \mu x_{k1}(t_k)x_{k2}(t_k) \\
& + x_{k1}(t_k)u_{k2}(t_k)\delta_k + x_{k2}(t_k)u_{k1}(t_k)\delta_k + \dot{x}_{k1}(t_k)x_{k2}(t_k)\Delta_k v_k \\
& + \dot{x}_{k1}(t_k)\Delta_k x_{k2}(t_k - r_k)m_k + x_{k1}(t_k)x_{k2}(t_k - r_k)m_k \\
& + x_{k2}(t_k)\dot{x}\Delta_k x_{k1}(t_k - r_k)m_k + x_{k2}(t_k)x_{k1}(t_k - r_k)m_k \\
& + \dot{x}_{k1}(t_k)\Delta_k u_{k2}(t_k)n_k + x_{k1}(t_k)u_{k2}(t_k)n_k + \dot{x}_{k2}(t_k)\Delta_k u_{k1}(t_k)n_k \\
& + x_{k2}(t_k)u_{k1}(t_k)n_k + x_{k1}(t_k)x_{k2}(t_k - r_k)p_k + x_{k2}(t_k)x_{k1}(t_k - r_k)p_k \\
& + x_{k2}(t_k - r_k)x_{k1}(t_k - r_k)c_k
\end{aligned} \tag{4.3}$$

Now, we shall state the theorem establishing the operator V

***Theorem* 4.1**

Let the initial guess of the solution by conjugate gradient method be

$$z_0^T(t_0) = (x_0(t_0), u_0(t_0), h_0(t_0), \lambda_0(t_0))$$

Then the control operator V associated with $Vz_{k2}(t_k)$ is given by

$$V = \begin{pmatrix} V_{11} & V_{12} & V_{13} & V_{14} \\ V_{21} & V_{22} & V_{23} & V_{24} \\ V_{31} & V_{32} & V_{33} & V_{34} \\ V_{41} & V_{42} & V_{43} & V_{44} \end{pmatrix}, .$$

where the entries will be supplied in the proof.

***Proof of Theorem* 4.1**

$Solve\ \ for\ \ x_{k2}(t_k)\ \ by\ setting\ u_{K2}(t_k) = h_{k2}(t_k) = \lambda_{k2}(t_k) = 0, \quad in$ (4.3) and using remark (i) and (ii) below,

$$(i)\ x_k(t-r) = x_k(s) = \begin{cases} h(s) = s \in [-r, 0] \\ x_k(s), \quad s \in [0, T-r] \end{cases}$$

$$(ii) \quad when\ h_{k2}(t_k) = 0, \quad then \quad x_{k2}(t_k - r_k) = x_{k2}(t_k)$$

So (4.3) becomes

$$\begin{aligned}
&\sum_{k=0}^{n} (x_{k1}(t_k)x_{k2}(t_k)\alpha_k + \mu\dot{x}_{k1}(t_k)\dot{x}_{k2}(t_k)\Delta_k{}^2 + \mu\dot{x}_{k1}(t_k)x_{k2}(t_k)\Delta_k \\
&+\mu x_{k1}(t_k)\dot{x}_{k2}(t_k)\Delta_k + \mu x_{k1}(t_k)x_{k2}(t_k) + x_{k2}(t_k)u_{k1}(t_k)\delta_k \\
&+\dot{x}_{k1}(t_k)x_{k2}(t_k)\Delta_k v_k + x_{k1}(t_k)x_{k2}(t_k)\Delta_k v_k + x_{k1}(t_k)\dot{x}_{k2}(t_k)\Delta_k v_k \\
&+x_{k1}(t_k)x_{k2}(t_k)\Delta_k v_k + \dot{x}_{k1}(t_k)x_{k2}(t_k - r_k)\Delta_k m_k + x_{k1}(t_k)x_{k2}(t_k - r_k)m_k \\
&+\dot{x}_{k2}(t_k)x_{k1}(t_k - r_k)\Delta_k m_k + x_{k2}(t_k)x_{k1}(t_k - r_k)m_k \\
&+\dot{x}_{k2}(t_k)u_{k1}(t_k)\Delta_k n_k + x_{k2}(t_k)u_{k1}(t_k)n_k + x_{k1}(t_k)x_{k2}(t_k - r_k)p_k \\
&+x_{k2}(t_k)x_{k1}(t_k - r_k)p_k + c_k x_{k2}(t_k)x_{k1}(t_k - r_k) + \lambda_{k1}\dot{x}_{k2}(t_k)\Delta_k \\
&-\Delta_k a x_{k2}(t_k)\lambda_{k1} - \lambda_{k1} b x_{k2}(t_k))
\end{aligned} \tag{4.4}$$

Having used remarks (i) and (ii) above, collect like-terms to obtain

$$\begin{aligned}
&\sum_{k=0}^{n} (x_{k1}(t_k)(x_{k2}(t_k)[\alpha_k + \mu + 2v_k + 2m_k + 2p_k + .c_k] + \dot{x}_{k2}(t_k)[\Delta_k v_k + \Delta_k \mu + \Delta_k m_k \\
&+\dot{x}_{k1}(t_k)[x_{k2}(t_k)(\Delta_k v_k + \Delta_k \mu + \Delta_k m_k) + \dot{x}_{k2}(t_k)\Delta_k{}^2\mu] + u_{k1}(t_k)[\dot{x}_{k2}\Delta_k n_k \\
&+x_{k2}(\delta_k + n_k + q_k)] + h_{k1}(t_k)[m_k\dot{x}_{k2}(t_k + r_k)\Delta_k + x_{k2}(t_k + r_k)(m_k + p_k + c_k)] \\
&+\lambda_{k1}(t_k)[x_{k2}(t_k)(-a\Delta_k - b\Delta_k) + \dot{x}_{k2}(t_k - r_k)\Delta_k])
\end{aligned} \tag{4.5}$$

Now,

rewrite (4.5) as,

$$\sum_{k=0}^{n} x_{k1}(t_k)V_{11} + \dot{x}_{k1}(t_k)\dot{V}_{11} + u_{k1}(t_k)V_{21} + h_{k1}(t_k)V_{31} + \lambda_{k1}(t_k)V_{41} \tag{4.6}$$

$$V_{41}(t_k) = x_{k2}(t_k)(-a\Delta_k - b\Delta_k) + \dot{x}_{k2}(t_k - r_k)\Delta_k \tag{4.7}$$

where,

$$V_{31}(t_k) = m_k \dot{x}_{k2}(t_k + r_k)\Delta_k + x_{k2}(t_k + r_k)(m_k + p_k + c_k) \quad (4.8a)$$

To

$$V_{21}(t_k) = \dot{x}_{k2}\Delta_k n_k + x_{k2}(\delta_k + n_k + q_k) \quad (4.8b)$$

determine $V_{11}(t_k)$, set

$$\Omega_1(t_k) = (x_{k2}(t_k)[\alpha_k + \mu + 2v_k + 2m_k + 2p_k + c_k] + \dot{x}_{k2}(t_k)[\Delta_k(v_k + \mu + m_k)] \quad (4.9)$$

$$and \qquad f_1(t_k) = x_{k2}(t_k)[\Delta_k(m_k + v_k + \mu) + \dot{x}_{k2}(t_k)\mu\Delta_k{}^2 \quad (4.10)$$

Now, $\Omega_1(t_k)$ and $f_1(t_k)$ are continuous functions on [0.T], $V_{11}(t_k)$ is continuous and at least twice differentiable on [0,T]. Hence $\Omega_1(t_k) - V_{11}(t_k)$ and $f_1(t_k) - \dot{V}_{11}(t_k)$ are continuous on [0,T]

$x(.) \in D_1[o,T]$ *such that* $x(0) = x(T) = 0$ *and by* [10]

$$\int_0^T \{x_{k1}(t_k)[\Omega_1(t_k) - V_{11}(t_k)] + \dot{x}_{k1}(t_k)[f_1(t_k) - \dot{V}_{11}(t_k)]\}dt_k = 0 \quad (4.11)$$

$$Hence, \frac{d}{dt_k}\left(f_1(t_k) - \dot{V}_{11}(t_k)\right) = \Omega_1(t_k) - V_{11}(t_k) \quad (4.12)$$

This is a

$$So \ \dot{f}_1(t_k) - \ddot{V}_{11}(t_k) = \Omega_1(t_k) - V_{11}(t_k), \ 0 \le t_k \le T.$$

second order

$$Let \ \ddot{V}_{11}(t_k) - V_{11}(t_k) = \dot{f}_1(t_k) - \Omega_1(t_k) = q(t_k) \quad (4.13)$$

differen tial equation that needs to be solved. So we impose the following initial conditions

$$V_{11}(0) = p_1 \ and \ \dot{V}_{11}(0) = r_1, \quad (4.14)$$

where p_1 and r_1 are to be determined.

Let $Q(s)$ = $L(q(t_k))$ and $V_{11}(s)$ = $L(V_{11}(t_k))$ denote the Laplace Transform of $q(t_k)$ and $V_{11}(t_k)$ respectively. Taking the Laplace transform of (4.13), we have

$$s^2V_{11}(s) - p_1 s - r_1 - V_{11}(s) = Q(s) \quad (4.15)$$

$$V_{11}(s) = \frac{Q(s)}{s^2 - 1} + \frac{p_1 s}{s^2 - 1} + \frac{r_1}{s^2 - 1} \quad (4.16)$$

We take the inverse Laplace Transform of (4.16) and using convolution theorem for the first term to obtain

$$V_{11}(t_k) = \int_0^T q(s_k)\sinh(t_k - s_k)dt_k + p_1\cosh(t_k) + r_1\sinh(t_k) \quad (4.17)$$

$$But \ \Omega_1(T) - V_{11}(T) = 0, \ \Omega_1(0) - V_{11}(0) = 0 \ and \ \Omega_1(0) = p_1 \quad (4.18)$$

$$So \ V_{11}(0) = p_1$$

From (4.17) and (4.18), we have

$$V_{11}(T) = \int_0^T q(s_k)\sinh(T - s_k) + p_1\cosh(T) + r_1\sinh(T) \quad (1.24)$$

From (4.19)

, we have

$$r_1 = \frac{1}{\sinh(T)}\{-\int_0^T q(s_k)\sinh(T - s_k)ds_k - p_1\cosh(T) + \Omega_1(T)\} \quad (4.20)$$

where $Q_1(T)$ = $V_{11}(T)$. But, $q(s_k) = \dot{f}_1(s_k) - \Omega_1(s_k)$ in (4.13). Substituting (4.13) into (4.20) and integrating, we obtain

$$V_{11}(t_k) = -\sinh(T)\, f_1(0) + \int_0^T f_1(s_k)\cosh(t_k - s_k)\,dt_k$$

$$-\int_0^T \Omega_1(s_k)\sinh(t_k - s_k)\,dt_k + p_1\cosh(t_k) + r_1\sinh(t_k) \qquad (4.21)$$

Similarly following the same logic as from (4.2) to (4.21), we can solve for $u_{k2}(t_k)$ by setting $x_{k2}(t_k)$ = $h_{k2}(t_k) = \lambda_{k2}(t_k) = 0$, in (4.2) to obtain

$$V_{22}(t_k) = u_{k2}(t_k)\beta_k, \qquad (4.22)$$

$$V_{32}(t_k) = u_{k2}(t_k)q_k, \qquad (4.23)$$

$$V_{42}(t_k) = -u_{k2}(t_k)\Delta_k c, \qquad (4.24)$$

$$f_2(t_k) = u_{k2}(t_k)\Delta_k n_k \ \textit{and}\ \Omega_2(t_k) = u_{k2}(t_k)(\partial_k + n_k + q_k) \qquad (4.25)$$

$V_{12}(t_k) = V_{11}(t_k)$ and $r_2 = r_1$ except that $f_2(t_k)$ replaces $f_1(t_k)$ and $\Omega_2(t_k)$ replaces $\Omega_1\ (t_k)$ in equation (4.21). Again solve for $h_{k2}(t_k)$ by setting $x_{k2}(t_k) = u_{k2}(t_k) = \lambda_{k2}(t_k) = 0$ in (4.2), implying that □$_{k2}(t_k)$ = 0.and collecting like-terms, after some simplification, we have,

$$V_{23}(t_k) = h_{k2}(t_k)(t_k - r_k) \qquad (4.26)$$

$$V_{33}(t_k) = h_{k2}(t_k)c_k \qquad (4.27)$$

$$V_{43}(t_k) = -b\lambda_{k2}(t_k)c_k \qquad (4.28)$$

$$f_3(t_k) = h_{k2}(t_k - r_k)m_k\Delta_k \qquad (4.29)$$

$$\Omega_3(t_k) = h_{k2}(t_k - r_k)(p_k + m_k) + x_{k2}(t_k)c_k \qquad (4.30)$$

$$V_{13}(t_k) = V_{11}(t_k),\ r_3 = r_1$$

Except that $f_3(t_k)$ replaces $f_1(t_k)$ and $\Omega_3(t_k)$ replaces $\Omega_1(t_k)$ in equation (4.21). Finally, we solve for $\lambda_{k2}(t_k)$ by setting $x_{k2}(t_k) = u_{k2}(t_k) = h_{k2}(t_k) = 0$ in (4.2) implying that $\dot{x}_{k2}(t_k) = 0$

Following the same the logic as in equations (4.2) to (4.21), we have

$$V_{24}(t_k) = -c\lambda_{k2}(t_k)\Delta_k \qquad 4.31$$

$$V_{34}(t_k) = -b\lambda_{k2}(t_k)\Delta_k \qquad 4.32$$

$$V_{44}(t_k) = 0 \qquad 4.33$$

$$f_4(t_k) = \lambda_{k2}(t_k)\Delta_k \qquad 4.34$$

$$\Omega_4(t_k) = \lambda_{k2}(t_k)(-\Delta_k(a+b)) \qquad 4.35$$

$V_{14}(t_k) = V_{11}(t_k),\ r_4 = r_1$ except that $f_4(t_k)$ replaces $f_1(t_k)$ and $\Omega_4(t_k)$ replaces $\Omega_1(t_k)$ in (4.21). This completes the proof of theorem 4.1

A program is written using the conjugate gradient method (CGM) to execute and see how it compares favourably with other schemes such as the function space algorithm(FSA) , extended conjugate gradient method (ECGM), imbedding extended conjugate gradient method (MECGM), used to solve the same control problem. The result is shown in the following table with the penalty parameter fixed per cycle and the multiplier parameter varied for every iteration within the cycle

5.0 Data and analysis

***Example* 5.1**

$$Min\int_0^1 (x^2(t)+u^2(t))dt$$

such that

Her

$\Delta_k = .1$ is the stepsize, μ is the penalty constant , λ is the multiplier, and r is the delay term.

Table 5.1 shows the numerical solutions of other algorithms compared to DCA

Table 5.1: Numerical solutions of other algorithms

Penalty/multiplier parameters	**Algorithm**	**Number of iterations**	**Objective funct.**	**Constrained Satisfaction**	**Augmented Lagrangian**
1	2	3	4	5	6
μ=.275, λ=-10.15	DCA	2	1.1164	32..1949	11.9109
μ=20, λ=.1455	FSA ECGM MECGM	87 4 4	1.09513 1.10993 1.11309	$8.39*10^{-3}$ $4.8034*10^{-4}$ $8.6749*10^{-4}$	1.2629 1.119540 1.09035
μ =.30, λ= -.3683 μ =40, λ= .2914	DCA FSA ECGM MECGM	2 63 4 3	1.1153 1.09973 1.11227 1.29763	19.9758 $5.6424*10^{-3}$ $3.0816*10^{-4}$ $6.8655*10^{-3}$	15.8079 1.3254 1.1246 0.75898
μ =1, λ=-.295 μ =60, λ=.4507	DCA FSA ECGM MECGM	2 63 4 3	1.1127 1.0997 1.1130 1.1134	5.8934 $5.6421*10^{-4}$ $2.6634*10^{-4}$ $1.3200*10^{-2}$	39.2521 1.3254 1.1290 1.7528

6.0 Conclusion

For each cycle in the table, the performance of each algorithm is seen as it relates to convergence profile of the given hypothetical problem with the same test data. In Row 1, for instance, DCA compares much more favourably with the Imbedding Extended Conjugate Gradient Method(MECGM) than to either Function Space Algorithm(FSA) or Extended Conjugate Gradient Method(ECGM) judging from its higher objective value in the second and third cycles, but trails behind Imbedding Extended Conjugate Gradient Method(MECGM) for every cycle except for the first cycle.

It is interesting to note that one needs as many as twenty (20) times the number of iterations in the Function Space Algorithm (FSA)(87) or (63) to obtain approximately the same numerical values as those obtained via the Discretized Continuous Algorithm(DCA) or Imbedding Extended Conjugate Gradient Method(MECGGM)

Finally, it can be seen that DCA is second to MECGM in terms of convergence profile, though they compare much more favourably in terms of number of iterations, such as 2 and 3 for DCA and MECGM respectively. So, it can be concluded that DCA has revealed its superiority over either FSA or ECGM inherently and computationally cumbersome. Its low iteration number, if explored will mean less computational time, small memory utilization and less costly. Therefore, DCA is a new numerical algorithm via the imbedding of both penalty and multiplier methods for obtaining approximate solutions to quadratic optimization problems.

References

[1] A. V. Balakrishnan, An operator theoretical formulation of a class of control problems and a steepest descent method of solution, SIAM, Journal of Control SER, A, Vol. 1, No. 2., (1963) pp. 109-127.

[2] R. Fletcher, and C. M. Reeves, Function minimization by conjugate gradients, (CJ, Vol.7), (1974) pp. 149-154

[3] S. T. Glad, A Combination of the penalty function and multiplier methods for solving optimal control problems, JOTA Vol. 28, (1979) pp. 303-329.
[4] M. R. Hestenes, Survey; Multiplier and Gradient Methods, JOTA, VOL.4, No.5, (1969) pp.303-320.
[5] M. A. Ibiejugba, and P. Onumanyi, A control operator and some of its applications. JMAA, Vol. 103 No.1, (1984) pp.31-47
[6] I. M.Gelfand, and S. V. Formin, Calculus of variation, Patience-Hall, Inc. N.J. 1963.
[7] G. Di Pillo. L. Grippo, F. Lampariello, The multiplier method for optimal control problem, CSEI, Rice University Optimization Problems in Engineering and Economics, Naples, December, 1974) pp.1-11.
[8] E. Polar, Computational methods in optimization-A unified approach, Academic press, London, 1971.
[9] R. T. Rockafellar, The multiplier method of Hestenes and Powell applied to convex programming, Journal of theory and applications, Vol. 12, (1973) pp. 555-562.
[7] V. F. Hu, and C. Strong, “Efficient generalized conjugate algorithms, Part 1: Theory” Journal of optimization Theory and Applications, Vol. 69, (1991) pp. 129-137.
[8] H. D. Mittelman, Sufficient optimality for discretized parabolic and elliptic control problems in fast solution of discretzed optimization problems, (K.H Hoffman, R. H .W. Hoppe and V. Schuil (Eds), ISNM, 138, Birkhauser-Verlag Basel, 2001.
[10] S.S. Rao, Optimization Theory and Application, Wiley Eastern Ltd. New Delhi, 1979