Computing Integrals Involving the Matrix Exponential

Charles Van Loan

TR 77-298
December 1976

*Department of Computer Science
Cornell University
Ithaca, New York 14853

*Partially supported by NSF grant MCS76-08686

COMPUTING INTEGRALS INVOLVING THE MATRIX EXPONENTIAL

Charles F. Van Loan +
Department of Computer Science
Cornell University
Ithaca, New York
14853

Abstract

A new algorithm for computing integrals involving the matrix exponential is given. The method employs diagonal Padé approximation with scaling and squaring. Rigorous truncation error bounds are given and incorporated in a FORTRAN subroutine. The computational aspects of this program are discussed and compared with existing techniques.

+Partially supported by NSF Grant MCS76-08686

1. Introduction

Let A, B, and Q_c be real matrices of dimensions nxn, nxp, and nxn respectively. Assume that Q_c is symmetric ($Q_c^T = Q_c$) and positive semidefinite ($x^T Q_c x \geq 0$). In this paper we present a new method for computing the integrals

(1.1) $$H(\Delta) = \int_0^\Delta e^{As} B \, ds$$

(1.2) $$Q(\Delta) = \int_0^\Delta e^{A^T s} Q_c e^{As} \, ds$$

(1.3) $$M(\Delta) = \int_0^\Delta e^{A^T s} Q_c H(s) \, ds$$

(1.4) $$W(\Delta) = \int_0^\Delta H(s)^T Q_c H(s) \, ds$$

The need for computing these integrals arises in several applications, notably, the optimal sampled-data regulator problem [1], [3].

The method we shall propose involves (a) computing the exponential of a certain block triangular matrix and (b) combining various submatrices of the result to obtain (1.1)-(1.4). To illustrate the basic idea, if

$$\exp\left(\begin{bmatrix} -A^T & Q_c \\ 0 & A \end{bmatrix} \Delta\right) = \begin{bmatrix} F_2(\Delta) & G_2(\Delta) \\ 0 & F_3(\Delta) \end{bmatrix}$$

and

$$K_1(t) = \int_0^t e^{A_1(t-s)} D_1 \, e^{A_4 s} \, ds \; +$$

$$+ \int_0^t \int_0^s e^{A_1(t-s)} [C_1 \, e^{A_3(s-r)} B_3 + B_1 \, e^{A_2(s-r)} C_2] e^{A_4 r} \, dr \, ds$$

$$+ \int_0^t \int_0^s \int_0^r e^{A_1(t-s)} B_1 e^{A_2(s-r)} B_2 e^{A_3(r-w)} B_3 e^{A_4 w} \, dw \, dr \, ds$$

<u>Proof</u>

Since all powers of C have the same block triangular structure, it is clear that e^{Ct} has the form indicated. By equating submatrices in the equation

$$\frac{d}{dt} [\, e^{Ct} \,] = C \, e^{Ct} \qquad I = e^{Ct} \Big|_{t=0}$$

we are led to the following differential equations:

$$\dot{F}_j(t) = A_j F_j(t) \qquad F_j(0) = I \,, \quad j=1,2,3,4$$

$$\dot{G}_j(t) = A_j G_j(t) + B_j F_{j+1}(t) \qquad G_j(0) = 0 \,, \quad j=1,2,3$$

$$\dot{H}_j(t) = A_j H_j(t) + B_j G_{j+1}(t) + C_j F_{j+2}(t) \qquad H_j(0) = 0, \; j=1,2$$

$$\dot{K}_1(t) = A_1 K_1(t) + B_1 H_2(t) + C_1 G_3(t) + D_1 F_4(t) \qquad K_1(0) = 0$$

The Theorem follows by solving these equations respectively for $F_j(t)$, $G_j(t)$, $H_j(t)$, and $K_1(t)$. ∎

If we apply this theorem to the $(3n+p)\times(3n+p)$ matrix

$$\hat{C} = \begin{bmatrix} -A^T & I & 0 & 0 \\ 0 & -A^T & Q_c & 0 \\ 0 & 0 & A & B \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

we find

$$e^{\hat{C}t} = \begin{bmatrix} \hat{F}_1(t) & \hat{G}_1(t) & \hat{H}_1(t) & \hat{K}_1(t) \\ 0 & \hat{F}_2(t) & \hat{G}_2(t) & \hat{H}_2(t) \\ 0 & 0 & \hat{F}_3(t) & \hat{G}_3(t) \\ 0 & 0 & 0 & \hat{F}_4(t) \end{bmatrix}$$

where

$$\hat{F}_3(t) = e^{At}$$

$$\hat{G}_2(t) = e^{-A^T t} \int_0^t e^{A^T s} Q_c e^{As} ds$$

$$\hat{G}_3(t) = \int_0^t e^{A(t-s)} B ds$$

$$\hat{H}_2(t) = e^{-A^T t} \int_0^t \int_0^s e^{A^T s} Q_c e^{Ar} B dr ds$$

and

$$\hat{K}_1(t) = e^{-A^T t} \int_0^t \int_0^s \int_0^r e^{A^T r} Q_c e^{Aw} B dw dr ds$$

It turns out that the integrals (1.1)-(1.4) can be expressed in terms of these submatrices of $e^{\hat{C}t}$ when we set $t = \Delta$:

This procedure is extremely easy to implement. All that is involved is a single call to any Padé matrix exponential subroutine followed by some elementary matrix computations. Ward's algorithm with its complete error analysis is particularly well suited [9]. For problems of small dimension, this is certainly a justifiable approach. However, in the interest of efficiency, the algorithm we shall detail does not repeatedly square the matrix $R_{qq}(\frac{\hat{C}\Delta}{2^j})$ as suggested by (3.2). Instead, setting $t_0 = \Delta/2^j$ we shall estimate $H(t_0)$, $Q(t_o)$, $M(t_0)$, and $W(t_0)$ using submatrices of $R_{qq}(\hat{C}t_0)$ and then repeatedly exploit the doubling formulae:

$$W(2t) = 2W(t) + H(t)^T M(t) + M(t)^T H(t) + H(t)^T Q(t) H(t) \tag{3.3}$$

$$M(2t) = M(t) + e^{A^T t} [\, Q(t)H(t) + M(t)\,] \tag{3.4}$$

$$Q(2t) = Q(t) + e^{A^T t} Q(t)\, e^{At} \tag{3.5}$$

$$H(2t) = H(t) + e^{At} H(t) \tag{3.6}$$

$$e^{2At} = e^{At}\, e^{At} \tag{3.7}$$

These formulae follow from the definitions (1.1)-(1.4). (See [1] for details.) Summarizing, our algorithm is as follows:

Algorithm

1. Set $\hat{C} = \begin{bmatrix} -A^T & I & 0 & 0 \\ 0 & -A^T & Q_c & 0 \\ 0 & 0 & A & B \\ 0 & 0 & 0 & 0 \end{bmatrix}$ and let j be the smallest

non-negative integer such that $\frac{\|\hat{C}\Delta\|}{2^j} \leq \frac{1}{2}$. Set $t_0 = \Delta/2^j$.

2. For some $q \geq 1$ compute

$$Y_0 = R_{qq}\left(\frac{\hat{C}\Delta}{2^j}\right) \equiv \begin{bmatrix} F_1(t_0) & G_1(t_0) & H_1(t_0) & K_1(t_0) \\ 0 & F_2(t_0) & G_2(t_0) & H_2(t_0) \\ 0 & 0 & F_3(t_0) & G_3(t_0) \\ 0 & 0 & 0 & F_4(t_0) \end{bmatrix}$$

and set

$$\begin{aligned} F_0 &= F_3(t_0) \\ H_0 &= G_3(t_0) \\ Q_0 &= F_3(t_0)^T\, G_2(t_0) \\ M_0 &= F_3(t_0)^T\, H_2(t_0) \\ W_0 &= [B^T F_3(t_0)^T K_1(t_0)] + [B^T F_3(t_0)^T K_1(t_0)]^T \end{aligned}$$

3. For $k = 0,\ldots,j-1$

$$\begin{aligned} W_{k+1} &= 2\, W_k + H_k^T M_k + M_k^T H_k + H_k^T\, Q_k\, H_k \\ M_{k+1} &= M_k + F_k^T [\, Q_k H_k + M_k\,] \\ Q_{k+1} &= Q_k + F_k^T\, Q_k\, F_k \\ H_{k+1} &= H_k + F_k H_k \\ F_{k+1} &= F_k^2 \end{aligned}$$

4. $F \equiv F_j$, $H \equiv H_j$, $Q \equiv Q_j$, $M \equiv M_j$ and $W \equiv W_j$ are then approximates to $e^{A\Delta}$, $H(\Delta)$, $Q(\Delta)$, $M(\Delta)$, and $W(\Delta)$ respectively.

inequality and recalling that $\|\hat{C}t_0\| \leq \frac{1}{2}$ we have

$$(4.7) \qquad \|\hat{E}\| \leq \varepsilon$$

From this it is easy to establish (4.5) because the Frobenius norm of any submatrix of $\hat{E}$ is less than the Frobenius norm of $\hat{E}$. It's also clear from [7] that $\hat{E}$ has the same block structure as $\hat{C}$:

$$\hat{E} = \begin{bmatrix} E_9 & E_2 & E_5 & E_7 \\ 0 & E_8 & E_3 & E_6 \\ 0 & 0 & E_1 & E_4 \\ 0 & 0 & 0 & E_{10} \end{bmatrix}$$

Now by scrutinizing Steps 1 and 2 of the algorithm, it is clear tha that

$$F_1(t_0) = F_2(t_0) = R_{qq}(-A^T t_0) = [[R_{qq}(At_0)]^{-1}]^T = [F_3(t_0)^{-1}]^T$$

and

$$F_4(t_0) = R_{qq}(0) = I \quad .$$

On the other hand, the equation $Y_0 = e^{(\hat{C}+\hat{E})t_0}$ coupled with Theorem 1 tells us that

$$\begin{aligned} F_1(t_0) &= e^{(-A^T+E_9)t_0} \\ F_2(t_0) &= e^{(-A^T+E_8)t_0} \\ F_3(t_0) &= e^{(A+E_1)t_0} \\ F_4(t_0) &= e^{(0+E_{10})t_0} \end{aligned}$$

and therefore, $E_8 = E_9 = -E_1^T$ and $E_{10} = 0$. Thus, $\hat{E}$ has the structure defined by (4.2).

By equating the (3,3) and (1,2) blocks of (4.6), we see that (4.3) and $A^TE_2 = E_2A^T$ hold. This latter equality implies that E_2 commutes with E_1^T since E_1 is a function of A [7]. Thus (4.4) is verified completing the proof of the lemma. ▌

Lemma 2

If $F_3(t_0)$, $G_2(t_0)$, $G_3(t_0)$, $H_2(t_0)$ and $K_1(t_0)$ are defined by Step 2 of the algorithm, then

$$F_3(t_0) = e^{(A+E_1)t_0}$$

$$G_2(t_0) = e^{-(A+E_1)t_0} \int_0^{t_0} e^{(A+E_1)^T s} (Q_c+E_3)\, e^{(A+E_1)s}\, ds$$

$$G_3(t_0) = \int_0^{t_0} e^{(A+E_1)s} (B + E_4)\, ds$$

$$H_2(t_0) = \int_0^{t_0} e^{-(A+E_1)^T(t_0-s)} E_6\, ds \; +$$

$$+ \int_0^{t_0}\int_0^{s} e^{-(A+E_1)^T(t_0-s)} (Q_c+E_3)\, e^{(A+E_1)(s-r)} (B+E_4)d$$

$$K_1(t_0) = \int_0^{t_0} e^{-(A+E_1)^T(t_0-s)} E_7\, ds \; +$$

$$+ \int_0^{t_0}\int_0^{s} e^{-(A+E_1)^T(t_0-s)} E_5\, e^{(A+E_1)(s-r)} (B+E_4)\, dr\, ds$$

$$+ \int_0^{t_0}\int_0^{s} (I+E_2) e^{-(A+E_1)^T(t_0-r)} E_6\, dr\, ds$$

$$+ \int_0^{t_0}\int_0^{s}\int_0^{r} (I+E_2) e^{-(A+E_1)^T(t_0-r)} (Q_c+E_3) e^{(A+E_1)(r-w)} \ldots$$

$$\ldots (B+E_4)\, dw\, dr\, ds$$

Theorem 2

If F is defined by the Algorithm, then

$$\| F - e^{A\Delta} \| \leq \varepsilon\Delta\, \theta(\Delta) e^{\varepsilon\Delta}$$

Proof

This follows from (4.9) with $u = \Delta$. ∎

Theorem 3

If H is defined by the Algorithm, then

$$\| H - H(\Delta) \| \leq \varepsilon\, \Delta\, \theta(\Delta)\, e^{\varepsilon\Delta} \left[1 + \frac{\alpha\ \Delta}{2}\right]$$

Proof

According to Lemma 3 (k=j) and the definition of $H(\Delta)$,

$$H - H(\Delta) = \int_0^{\Delta} [e^{(A+E_1)s} - e^{As}]B\, ds + \int_0^{\Delta} e^{(A+E_1)s} E_4\, ds$$

The Theorem follows by taking norms and applying (4.5),(4.8) and (4.9). ∎

Theorem 4

If Q is defined by the Algorithm, then

$$\| Q - Q(\Delta) \| \leq \varepsilon\, \Delta\, \theta(\Delta)^2 e^{2\varepsilon\Delta} [1 + \alpha\Delta]$$

Proof

From Lemma 3 (k=j) and the definition of $Q(\Delta)$,

$$Q - Q(\Delta) = \int_0^{\Delta} [e^{(A+E_1)^T s}(Q_c+E_3)e^{(A+E_1)s} - e^{A^T s}Q_c\, e^{As}]\, ds$$

$$= \int_0^{\Delta} [e^{(A+E_1)s} - e^{As}]^T Q_c e^{(A+E_1)s} + e^{A^T s} Q_c [e^{(A+E_1)s} - e^{As}]\, ds$$

$$+ \int_0^{\Delta} e^{(A+E_1)^T s} E_3\, e^{(A+E_1)s}\, ds$$

The theorem follows by taking norms and using (4.5), (4.8), and (4.9). ∎

<u>Theorem 5</u>

If M is defined by the algorithm, then

$$\|M - M(\Delta)\| \leq \varepsilon\,\Delta\,\Theta(\Delta)^2\, e^{2\varepsilon\Delta}\,[1 + \varepsilon + \Delta\alpha\ ]^2$$

<u>Proof</u>

From Lemma 3 (k=j) and the definition of $M(\Delta)$ we have

$$M - M(\Delta) = \int_0^\Delta \int_0^s e^{(A+E_1)^T s}(Q_c+E_3)e^{(A+E_1)(s-r)}(B+E_4)\,dr\,ds$$

$$+ \int_0^\Delta e^{(A+E_1)^T s}\,E_6\,ds - \int_0^\Delta \int_0^s e^{A^T s}\,Q_c\,e^{A(s-r)}\,B\,dr\,ds$$

$$= \int_0^\Delta \int_0^s [e^{(A+E_1)s} - e^{As}]^T Q_c\, e^{(A+E_1)(s-r)}\,B\,dr\,ds$$

$$+ \int_0^\Delta \int_0^s e^{A^T s}\,Q_c\,[e^{(A+E_1)(s-r)} - e^{A(s-r)}]\,B\,dr\,ds$$

$$+ \int_0^\Delta \int_0^s e^{(A+E_1)^T s}\,[Q_c + E_3]\,e^{(A+E_1)(s-r)}\,E_4\,dr\,ds$$

$$+ \int_0^\Delta \int_0^s e^{(A+E_1)^T s}\,E_3\,e^{(A+E_1)(s-r)}\,B\,dr\,ds$$

$$+ \int_0^\Delta e^{(A+E_1)^T s}\,E_6\,ds$$

The Theorem follows by taking norms and invoking (4.5), (4.8), and (4.9).

The lemma follows from this result, (4.11)-(4.14), and the fact that

$$\|F_3(t_0)^T K_1(t_0) - \hat{F}_3(t_0)^T \hat{K}_1(t_0)\| \leq \|\mathcal{E}_1\| + \|\mathcal{E}_2\| + \|\mathcal{E}_3\| + \|\mathcal{E}_4 - \hat{F}_3(t_0)^T \hat{K}_1(t_0)\|$$

Theorem 6

If W is defined by the algorithm, then

$$\|W - W(\Delta)\| \leq \varepsilon\, \theta(\Delta)^2 e^{2\varepsilon\Delta}\, 4[1 + 1.5(\alpha+\varepsilon)\Delta]^3$$

Proof

Subtracting the doubling formula

$$W(t_{k+1}) = 2W(t_k) + H(t_k)^T M(t_k) + M(t_k)^T H(t_k) + H(t_k)^T Q(t_k) H(t_k)$$

from

$$W_{k+1} = 2\, W_k + H_k^T M_k + M_k^T H_k + H_k^T Q_k H_k$$

and taking norms gives

$$(4.15)\quad \|W_{k+1} - W(t_{k+1})\| \leq 2\,\|W_k - W(t_k)\| + 2\,\|H_k^T M_k - H(t_k)^T M(t_k)\| + \|H_k^T Q_k H_k - H(t_k)^T Q(t_k) H(t_k)\|$$

By applying Lemma 1, Lemma 3, (4.8), and (4.9) the following bounds can be derived:

$$\|H_k^T M_k - H(t_k)^T M(t_k)\| \leq \varepsilon\, \theta(t_{k+1})^2\, e^{2\varepsilon\Delta} t_k^2\, (\alpha+\varepsilon)\, [\tfrac{3}{2}\, (\alpha+\varepsilon) t_k + 1]^2$$

and

$$\|H_k^T Q_k H_k - H(t_k)^T Q(t_k) H(t_k)\| \leq \varepsilon\, \theta(t_{k+1})^2\, e^{2\varepsilon\Delta} t_k^3\, (\alpha+\varepsilon)^2\, [3 + t_k(\alpha+\varepsilon)]$$

It is thus clear from (4.15) that

$$(4.16) \qquad \| W_{k+1} - W(t_{k+1}) \| \leq 2 \| W_k - W(t_k) \| + \delta_k$$

where

$$\delta_k = \varepsilon \, \theta(t_{k+1})^2 \, e^{2\varepsilon\Delta} \, t_k^2 \, (\alpha+\varepsilon) \, [3(\alpha+\varepsilon)t_k + 2]^2$$

A simple induction argument involving (4.16) shows

$$(4.17) \qquad \| W - W(\Delta) \| = \| W_j - W(t_j) \| \leq 2^j \| W_0 - W(t_0) \| + \sum_{k=0}^{j-1} 2^{j-k-1} \delta_k$$

By using elementary properties of geometric series and the fact that $t_k = 2^{k-j}\Delta$, it is easy to verify that

$$(4.18) \qquad \sum_{k=0}^{j-1} 2^{j-k-1} \delta_k \leq \frac{1}{2} \varepsilon \, \theta(\Delta)^2 \, e^{2\varepsilon\Delta} \Delta^2 \, (\alpha+\varepsilon) [3(\alpha+\varepsilon)\Delta + 2]^2$$

Now $\| C \, t_0 \| \leq .5$ implies $\alpha t_0 \leq .5$ and since $\theta(t_0) \leq \theta(\Delta)$ we have from Lemma 4,

$$2^j \| W_0 - W(t_0) \| \leq 2 \, \varepsilon \, \Delta \, \theta(\Delta)^2 \, (1 + \alpha)^2$$

The theorem follows by substituting this result together with (4.18) into (4.17). ∎

$$D_{33}\,F_3(t_0) = N_{33}$$
$$D_{33}\,G_3(t_0) = N_{34} - D_{34}$$
$$N_{33}^T\,G_2(t_0) = N_{23} - D_{23}\,F_3(t_0)$$
$$N_{33}^T\,H_2(t_0) = N_{24} - D_{23}\,G_3(t_0) - D_{24}$$
$$N_{33}^T\,K_1(t_0) = N_{14} - D_{12}\,H_2(t_0) - D_{13}G_3(t_0) - D_{14}$$

These linear systems can be solved using Gaussian elimination. Since $\left\| \frac{\hat{C}A}{2^J} \right\| < \frac{1}{2}$, it is easy to show that both D_{33} and N_{33}^T are diagonally dominant and therefore, that no pivoting is necessary [2,p.152]. Arrays are needed to form the submatrices $D_{33}, D_{34}, D_{23}, D_{24}, D_{12}, D_{13}, D_{14}, N_{33}, N_{34}, N_{23}, N_{24}$, and N_{14}. After that, they can be overwritten as the above linear systems are solved.

The implementation of the doubling formulae is straight forward and does not require any special commentary. Suffice it to say that no additional storage is necessary to execute that portion of the program.

Regarding storage and efficiency, it may be that not all of the matrices F,H,Q,M, and W are desired. For example, suppose that W is not wanted. We can effectively compute F,H,Q, and M by working with

$$\begin{bmatrix} -A^T & Q_c & 0 \\ 0 & A & B \\ 0 & 0 & 0 \end{bmatrix}$$

instead of $\hat{C}$. We merely ignore all the computations which are specific to the construction of W. The truncation error bounds (3.8)-(3.11) still hold. Similar techniques exist if only Q,H, or F are wanted.

Through an integer variable CODE, the user can specify certain subsets of the matrices F,H,Q,M, and W which are to be computed. This allows for a saving in both storage and execution time as the following table indicates:

CODE	Matrices Computed	Approximate Storage Required	Magnitude of Work (multiplicative operations)
1	F	$4n^2$	$[q+j+\frac{1}{3}]n^3$
2	F,H	$4n^2 + 4np$	$[q+j+\frac{1}{3}]n^3 + [q+j]n^2p$
3	F,Q	$8n^2$	$[3q+\frac{5}{2}j+\frac{2}{3}]n^3$
4	F,H,Q,M	$8n^2 + 7np$	$[3q+\frac{5}{2}j+\frac{2}{3}]n^3 + [3q+3j]n^2p$
5	F,H,Q,M,W	$11n^2 + 10np$	$[4q+\frac{5}{2}j-\frac{5}{6}]n^3 + [4q+\frac{11}{2}j+2]n^2p$

The volume of computation is seen to depend upon the scale parameter j and the degree q of the Padé approximant which is used. The selection of j was described in Section 3. The integer q is chosen in accordance with a user specified tolerance TOL. From Theorems 2-6 we know that if TOL > 0 then q can be picked so

$$(5.1)\quad \|F - F(\Delta)\| \leq \varepsilon\Delta e^{\varepsilon\Delta}\, \theta(\Delta) \leq \mathrm{TOL}\ \theta(\Delta)$$

$$(5.2)\quad \|H - H(\Delta)\| \leq \varepsilon\Delta e^{\varepsilon\Delta}\, [1 + \frac{\alpha\Delta}{2}]\, \theta(\Delta) \leq \mathrm{TOL}\ \theta(\Delta)$$

$$(5.3)\quad \|Q - Q(\Delta)\| \leq \varepsilon\Delta e^{2\varepsilon\Delta}[1 + \alpha\Delta]\, \theta(\Delta)^2 \leq \mathrm{TOL}\ \theta(\Delta)^2$$

are to 10 significant digits:

$$F(\Delta) = \begin{bmatrix} .477528143 & -.522155363 & -.351058933 \\ .855482148 & -.994523657 & -.702117866 \\ -.855482148 & 1.012839296 & .720433505 \end{bmatrix}$$

$$H(\Delta) = \begin{bmatrix} 1.999431436 & -3.394449325 \\ 1.148224072 & -6.155423359 \\ -0.166539711 & 7.627949901 \end{bmatrix}$$

$$Q(\Delta) = \begin{bmatrix} 9.934877720 & -11.08568953 & -9.123023900 \\ -11.08568953 & 13.66870748 & 11.50451512 \\ -9.123023900 & 11.50451512 & 10.29179555 \end{bmatrix}$$

$$M(\Delta) = \begin{bmatrix} 3.515982340 & -24.87596341 \\ -2.516164470 & 30.94693518 \\ -1.194242580 & 24.29316617 \end{bmatrix}$$

$$W(\Delta) = \begin{bmatrix} 12.29648659 & -5.373425530 \\ -5.373425530 & 105.9996704 \end{bmatrix}$$

With TOL = 10^{-3} , the computed versions of these matrices were found to be correct through the sixth decimal place. (In this example, q = 4, j = 7, and THETA = 4.2.)

We do not expect the accuracy of our computed matrices to undercut the value of TOL by such amounts in all problems. Indeed, one must be wary of rounding errors which have not been accounted for in our analysis. However, as experience with Ward's Padé scaling and squaring algorithm for matrix exponentials suggests, we can be fairly confident of our error bounds so long as TOL is not in the immediate neighborhood of the machine precision.

6. Conclusions

We conclude by contrasting our algorithm with some of the other techniques which have been suggested for computing the various integrals (1.1)-(1.4).

Johnson and Phillips [4] have proposed the computation of $H(\Delta)$ through the formula

$$H(\Delta) = \left[\sum_{k=0}^{m-1} (e^{At})^k \right] H(t) \qquad mt = \Delta$$

the idea being that for small t, e^{At} and $H(t)$ can be accurately computed. (Their discussion assumes B=I but it is easy to extend their results for general B.) However, if $m = 2^j$, their algorithm requires about $2^j[n^3 + n^2p]$ operations to compute $H(\Delta)$ from $H(t)$ in contrast to our algorithm where the corresponding figure is only $j[n^3 + n^2p]$.

In search for an efficient squaring algorithm, Kallstrom [5] has proposed repeated application of

$$H(2t) = H(t) \; [\; 2I + AH(t) \;] \quad .$$

Unfortunately, this formula only holds if B is the identity and therefore one has to compute $H(\Delta)$ by applying Kallstrom's formula to the problem

$$\tilde{H}(\Delta) = \int_0^{\Delta} e^{As}\, ds$$

and then forming $H(\Delta) = \tilde{H}(\Delta)B$. This increases the volume of com-

BIBLIOGRAPHY

[1] E.S.Armstrong and A.K.Caglayan, An Algorithm for the Weighting Matrices in the Sampled-Data Optimal Linear Regulator Problem, NASA Technical Note NASA TN D-8372, NASA Langley Research Center, Hampton Va., 1976.

[2] G.Dahlquist and A.Bjork, Numerical Methods, Prentice Hall, Englewood Cliffs, NJ, 1974.

[3] P.Dorato and A.Levis, Optimal Linear Regulators:The Discrete Time Case, IEEE Trans.Auto.Contr. AC-16, Dec. 1971, p.613-620.

[4] J.C.Johnson and C.L.Phillips, An Algorithm for the Computation of the Integral of the State Transition Matrix, IEEE Trans.Auto.Contr. AC-16 , 1971, p.204-205.

[5] C.Kallstrom, Computing Exp(A) and $\int$Exp(As)ds , Report 7309, Division of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1973.

[6] A.H.Levis, Some Computational Aspects of the Matrix Exponential, IEEE Trans.Auto.Contr.,AC-14, 1969, p.410-411.

[7] C.B.Moler and C.Van Loan, Nineteen Dubious Ways to Compute the Exponential of a Matrix, (to appear, SIAM Review).

[8] C.Van Loan, The Sensitivity of the Matrix Exponential, (to appear, SIAM J.Numer.Anal.)

[9] R.C.Ward, Numerical Computation of the Matrix Exponential with Accuracy Estimate, (to appear, SIAM J.Numer.Anal.)