Journal of Computational Mathematics, Vol.11, No.3, 1993, 225–235.

SQUARE MATRIX PADÉ APPROXIMATION AND CONVERGENCE ACCELERATION OF SEQUENCES*[1)]

Xu Guo-liang Zhuang Guo-zhong

(*Computing Center, Academic of Sincica, Beijing, China*)

Abstract

This paper provides a new method for approximating matrix-valued functions—-square Padé approximation. Some computational methods of the approximants are given. For accelerating matrix sequences, a family of nonlinear extrapolation formulas based on the square Padé approximation is given, a convergence acceleration theorem is proved and numerical examples are presented.

§1. Introduction

For a given formal power series $f(z) = \sum_{i=0}^{\infty} c_i z^i, c_i \in C$, let the polynomials $P_v(z)$ and $Q_u(z)$ be of degree v and u respectively and solve the (v, u) Padé approximation problem:

$$(fQ_u - P_v)(z) = \sum_{i \geq v+u+1} E_i z^i, \quad Q_u(0) = 1,$$

then we call $P_v(z)/Q_u(z)$ the classical Padé approximant. For a formal power series $f(z) = \sum_{i=0}^{\infty} c_i z^i$ with matrix coefficients $c_i \in C^{p \times m}$, we can directly generalize, when $p = m$, the definition of the classical Padé approximation to the matrix case (see [1], [7]). If $c_i \in C^{p \times m}$ is not a square matrix ($p \neq m$), but mu can be divided by p, then we can still define matrix Padé approximant successfully (see [8]). However, in other cases it is difficult to define matrix Padé approximant because of the trouble with matching the number of unknowns to be determined with the number of equations that determine the unknowns. In paper [9], different matrix Padé approximants have been derived. However, these definitions have a defect that all the elements of the numerator (or denominator) don't have the same degree. In this paper, we define a new kind matrix Padé approximant, which can eliminate the above defect. The basic idea is, instead of setting some coefficients of the residual to be zero, we let them satisfy some minimization conditions in Euclid (i.e.,Frobenius) norm. Hence, the approximant derived in this sense is called square matrix Padé approximant. Although other norms can be used, Euclid norm seems to be the most convenient one.

---

* Received September 6, 1991.

1) The Project Supported by National Science Foundation of China for Youth.

In section 2, we first give the definition of square Padé approximant under two normalization conditions, and then discuss its existence and uniqueness. In section 3, we give some effective computational methods of square matrix Padé approximant by utilizing fully the special structure of the related matrix. In section 4, we derive a set of accelerating formulas for matrix sequences from two different square matrix Padé approximants and obtain an accelerating convergence theorem . In the last section, numerical examples for accelerating the convergence of vector sequences are presented.

§2. Definition, Existence and Uniqueness

Let $F(z) = \sum_{i=0}^{\infty} c_i z^i, \quad c_i \in C^{p\times m}; \quad H_n^{(p,q)} = \{\sum_{i=0}^{n} a_i z^i : a_i \in C^{p\times q}\}$. If the polynomial $P_v(z) = \sum_{i=0}^{v} A_i z^i \in H_v^{(p,m)}$ and $Q_u(z) = \sum_{i=0}^{u} B_i z^i \in H_u^{(m,m)}$ satisfy

$$F(z)Q_u(z) - P_v(z) = \sum_{i=0}^{\infty} E_i z^i, \quad E_i \in C^{p\times m}$$

and

$$E_i = 0, \qquad i = 0, 1, \cdots, v, \tag{2.1}$$

$$\sum_{i=1}^{k} \|E_{v+i}\|_F = \min, \tag{2.2}$$

$$\sum_{i=1}^{u} \|B_i\|_F = \min, \tag{2.3}$$

under one of the following normalization conditions

$$B_0 = I, \tag{2.4}$$

$$\sum_{i=0}^{u} B_i = I, \tag{2.5}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, for $A = (a_{ij})_{1,1}^{p,q}$, $\|A\|_F = \Big(\sum_{i=1}^{p}\sum_{j=1}^{q}|a_{ij}|^2\Big)^{1/2}$, we call the approximant $P_v(z)(Q_u(z))^{-1}$ the square matrix Padé approximant of $F(z)$, and denote it by $[v, u, k]_F$.

If $mu = pk$, and the matrix Padé approximant $[v/u]$ (See [8]) exists, then (2.2) becomes $E_{v+i} = 0, i = 1, 2, \cdots, k$. Hence the usual matrix Padé approximant is a special case of our square matrix Padé approximant. If $k < 1$, (2.2) is neglected, (2.3) implies $B_i = 0$ (under the condition (2.4)), thus $Q_u(z) = I$, $P_v(z) = \sum_{i=0}^{v} c_i z^i$. In the following we will always assume $k \geq 1$. The condition (2.3) is proposed so that the solution is unique. From (2.1), we have

$$A_i = \sum_{j=0}^{u} c_{i-j} B_j, \quad i = 0, 1, \cdots, v \tag{2.6}$$

here we use the convention: $c_i = 0$, if $i < 0$. Hence A_i can be determined if all B_i's have been computed. From

$$E_i = \sum_{j=0}^{u} c_{i-j} B_j, \quad i = v+1, \cdots, v+k, \tag{2.7}$$

we have

$$\begin{bmatrix} c_{v+1} & c_v & \cdots & c_{v+1-u} \\ c_{v+2} & c_{v+1} & \cdots & c_{v+2-u} \\ \vdots & \vdots & & \vdots \\ c_{v+k} & c_{v+k-1} & \cdots & c_{v+k-u} \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \\ \vdots \\ B_u \end{bmatrix} = \begin{bmatrix} E_{v+1} \\ E_{v+2} \\ \vdots \\ E_{v+k} \end{bmatrix}. \tag{2.8}$$

From (2.8) and (2.2)–(2.3) we know that, under the condition (2.4), B_i's are the minimal least square solution of the following equation

$$\begin{bmatrix} c_v & \cdots & c_{v+1-u} \\ \vdots & & \vdots \\ c_{v+k-1} & \cdots & c_{v+k-u} \end{bmatrix} \begin{bmatrix} B_1 \\ \vdots \\ B_u \end{bmatrix} = - \begin{bmatrix} c_{v+1} \\ \vdots \\ c_{v+k} \end{bmatrix}. \tag{2.9}$$

Under the condition (2.5), by setting $B_i = B'_i - B'_{i+1}$, for $i = 0, 1, \cdots, u-1$; $B_u = B'_u$; or $B'_i = \sum_{j=i}^{u} B_j$, for $i = 1, \cdots, u$, we have another minimal least square problem

$$\begin{bmatrix} \Delta c_v & \cdots & \Delta c_{v+1-u} \\ \vdots & & \vdots \\ \Delta c_{v+k-1} & \cdots & \Delta c_{v+k-u} \end{bmatrix} \begin{bmatrix} B'_1 \\ \vdots \\ B'_u \end{bmatrix} = \begin{bmatrix} c_{v+1} \\ \vdots \\ c_{v+k} \end{bmatrix}, \quad \Delta c_i = c_{i+1} - c_i. \tag{2.10}$$

Both (2.9) and (2.10) always have their unique minimal least square solutions.

§3. Computation

Equations (2.9) and (2.10) have the same form, thus we need only consider computing the solution of (2.9).

3.1. Direct Computation

Lots of classical methods can be used to solve the minimal least square problem, e.g., if we denote

$$H(i,j,k) = \begin{bmatrix} c_i & \cdots & c_{i+1-j} \\ \vdots & & \vdots \\ c_{i+k-1} & \cdots & c_{i+k-j} \end{bmatrix},$$

then the minimal least square solution of (2.9) can be expressed by

$$\begin{bmatrix} B_1 \\ \vdots \\ B_u \end{bmatrix} = -H(v,u,k)^+ \begin{bmatrix} c_{v+1} \\ \vdots \\ c_{v+k} \end{bmatrix},$$

here $H(v,u,k)^+$ is the Moore–Penrose generalized inverse which can be computed, for instance, by QR factorization[2]: If

$$H(v,u,k) = QR, \quad Q \in C^{kp\times r}, \quad R \in C^{r\times mu}, \quad Q^*Q = I,$$

with $r = \operatorname{rank} H(v,u,k) = \operatorname{Rank} Q = \operatorname{rank} R$, then

$$H(v,u,k)^+ = R^*(RR^*)^{-1}Q^*.$$

However, since $H(v,u,k)$ is a block Toeplitz matrix, there are some more efficient methods for solving (2.9). At first, we discuss the method which directly solves the normal equation

$$H^*(v,u,k)H(v,u,k)\begin{bmatrix} B_1 \\ \vdots \\ B_u \end{bmatrix} = -H^*(v,u,k)\begin{bmatrix} c_{v+1} \\ \vdots \\ c_{v+k} \end{bmatrix}, \tag{3.1}$$

here we assume $H(v,u,k)$ has full rank in column. Let

$$S(v,u,k) = (s_{ij})_{i,j=1}^{u} = H^*(v,u,k)H(v,u,k) \tag{3.2}$$

then $s_{ij} = \sum_{l=1}^{k} c_{v-i+l}^* c_{v-j+l} \in C^{m\times m}$ which implies $s_{ij} = s_{ji}^*$, and

$$s_{i,j} = s_{i-1,j-1} + c_{v+1-i}^* c_{v+1-j} - c_{v+k+1-i}^* c_{v+k+1-j}. \tag{3.3}$$

So s_{ij} can be computed from $s_{i-1,j-1}$. Hence the elements of matrix $S(v,u,k)$ can be easily obtained by (3.3) when the first row (or column) has been computed.

Because $S(v,u,k)$ has low displacement rank, the method in [3] can be used to obtain $S^{-1}(v,u,k)$. In fact, for a matrix $R_N = [r_{ij}]_{i,j=1}^{N}$ $(r_{ij} \in C^{p\times p})$, define the "shift–difference" operator $\delta[\,\cdot\,]$ as

$$\delta[R_N] = \begin{bmatrix} r_{1,1} & \cdots & r_{1,N} \\ \vdots & & \vdots \\ r_{N,1} & \cdots & r_{N,N} \end{bmatrix} - \begin{bmatrix} r_{0,0} & \cdots & r_{0,N-1} \\ \vdots & & \vdots \\ r_{N-1,0} & \cdots & r_{N-1,N-1} \end{bmatrix}, \tag{3.4}$$

and write $\delta[R_N]$ as

$$\delta[R_N] = D\sum D^*, \quad D \in C^{Np\times \alpha p}, \quad \sum \in C^{\alpha p\times \alpha p} \tag{3.5}$$

with $\sum$ being a diagonal (signature) matrix consisting of ± 1's, then we can use the generalized Levinson–Szegö algorithm(for the limit of space, we do not list it here) in [3] to compute R_N^{-1}, which needs approximate $(\alpha+2)p^2N^2$ operations. This number may be small compared to $\frac{1}{3}p^3N^3$–the number of operations required for a conventional matrix inversion algorithm.

From (3.3) we have

$$\delta[S(v,u,k)] = C^*(v-1, v+k-1, u-1)\begin{bmatrix} I_p & 0 \\ 0 & -I_p \end{bmatrix} C(v-1, v+k-1, u-1), \tag{3.6}$$

where

$$C(i,j,k) = \begin{bmatrix} c_i & c_{i-1} & \cdots & c_{i+1-k} \\ c_j & c_{j-1} & \cdots & c_{j+1-k} \end{bmatrix}.$$

Since $s_{ij} \in C^{m\times m}$, (3.6) does not have the same form as (3.5) completely. In order to write it in the form of (3.5), let $p = \alpha m - \beta, \quad 0 \le \beta < m$, where both α and β are integers. Denote

$$\tilde{c}_i = \begin{bmatrix} c_i \\ 0_{\beta\times m} \end{bmatrix}, \quad \tilde{C}(i,j,k) = \begin{bmatrix} \tilde{c}_i & \tilde{c}_{i-1} & \cdots & \tilde{c}_{i+1-k} \\ \tilde{c}_j & \tilde{c}_{j-1} & \cdots & \tilde{c}_{j+1-k} \end{bmatrix},$$

where $0_{\beta\times m}$ is a $\beta \times m$ null matrix, then (3.6) can be written as

$$\begin{aligned} \delta[S(v,u,k)] = \tilde{C}^*(v-1, v+k-1, u-1)\,\mathrm{diag}\,(I_m,\cdots,I_m,-I_m,\cdots,-I_m) \\ \times \tilde{C}(v-1, v+k-1, u-1). \end{aligned} \tag{3.7}$$

Now, expressions (3.7) and (3.5) have the same form. Hence the matrix inversion method of [3] can be utilized.

3.2. Recursive Computation

a) u and k fixed, increasing v, i.e., compute $S(v+1,u,k)^{-1}$ from $S(v,u,k)^{-1}$. Because

$$S(v+1,u,k) = S(v,u,k) + C^*(v, v+k, u)\begin{bmatrix} -I_p & 0 \\ 0 & I_p \end{bmatrix} C(v, v+k, u), \tag{3.8}$$

then by the Sherman–Morrison formula

$$(A + USV^*)^{-1} = A^{-1} - A^{-1}U(V^*A^{-1}U + S^{-1})^{-1}V^*A^{-1}, \tag{3.9}$$

we have

$$S(v+1,u,k)^{-1} = S^{-1}(v,u,k) - D^*\left(DC^*(v,v+k,u) + \begin{bmatrix} -I_p & 0 \\ 0 & I_p \end{bmatrix}\right)^{-1} D,$$

where $D = C(v, v+k, u)S(v,u,k)^{-1}$, here we only need to compute the inversion of a much lower order matrix.

b) v and u fixed, increasing k, i.e., compute $S(v,u,l)^{-1}$ from $S(v,u,k)^{-1}$, $l > k$. Since

$$S(v,u,l) = S(v,u,k) + \begin{bmatrix} c^*_{v+k} & \cdots & c^*_{v+l-1} \\ \vdots & & \vdots \\ c^*_{v+k+1-u} & \cdots & c^*_{v+l-u} \end{bmatrix}\begin{bmatrix} c_{v+k} & \cdots & c_{v+k+1-u} \\ \vdots & & \vdots \\ c_{v+l-1} & \cdots & c_{v+l-u} \end{bmatrix}, \tag{3.11}$$

we can compute $S(v,u,l)^{-1}$ using (3.9). Especially, when $l = k+1$, (3.11) turns into

$$S(v,u,k+1) = S(v,u,k) + \begin{bmatrix} c^*_{v+k} \\ \vdots \\ c^*_{v+k+1-u} \end{bmatrix}\begin{bmatrix} c_{v+k} & \cdots & c_{v+k+1-u} \end{bmatrix}$$

by (3.9), we need only compute a $p \times p$ matrix inverse.

Note 3.1. If $l < k$, we can compute $S(v,u,l)^{-1}$ similarly.

c) v and k fixed, increasing u, i.e., compute $S(v,u',k)^{-1}$ from $S(v,u,k)^{-1}$ $(u' > u)$. Because we can obtain $S(v,u',k)$ from $S(v,u,k)$ by adding $(u'-u)$ columns to the right and $(u'-u)$ rows to the bottom, then the matrix bordering inversion technique can be utilized. However, this method doesn't make use of the special structure of $S(v,u,k)$. If

$S(v,u,k)^{-1}$ is computed by the generalized Levinson–Szegö algorithm, it will be more efficient to continue using this algorithm $(u'-u)$ steps. In fact, the $k-th$ step of this algorithm forms the parameters while computing R_k^{-1} of the nested matrices $\{R_m\}_{m=0}^{N}$.

§4. Application in convergence acceleration

For a given matrix sequence $\{s_0, s_1, \cdots\}$, $s_i \in C^{p\times q}$, and two integers m and l, we shall derive the following accelerating formulas from different approaches:

$$h_n^{(l,m)}(i,j) = s_{n+j+i-2} - \Delta S_{n+i-1}^{(1,m)}(\Delta^2 S_n^{(l,m)})^{+}\Delta S_{n+j-1}^{(l,1)}, \tag{4.1}$$

for $i = 1, \cdots, l;\ j = 1, \cdots, m$, where

$$S_k^{(l,m)} = \begin{bmatrix} s_k & \cdots & s_{k+m-1} \\ \vdots & & \vdots \\ s_{k+l-1} & \cdots & s_{k+m+l-2} \end{bmatrix},$$

and $\Delta S_k^{(l,m)} = S_{k+1}^{(l,m)} - S_k^{(l,m)}, \Delta^2 S_k^{(l,m)} = \Delta S_{k+1}^{(l,m)} - \Delta S_k^{(l,m)}$.

Note 4.1. If $q = 1, m = p, l = 1, i = j = 1$, formula (4.1) coincides with Henrici's transformation[4].

Let $f_i(z) = s_i + \sum_{j=1}^{\infty} \Delta s_{j+i-1} z^j, \quad i = 0, 1, \cdots$, where $\Delta s_j = s_{j+1} - s_j$, for $j \geq 0$.

Proposition 1. For two given integers m and l, let

$$F(z) = \begin{bmatrix} f_0 & f_1 & \cdots & f_{m-1} \\ \vdots & \vdots & & \vdots \\ f_{l-1} & f_l & \cdots & f_{m+l-2} \end{bmatrix}. = S_0^{(l,m)} + \sum_{j=1}^{\infty} \Delta S_{j-1}^{(l,m)} z^j,$$

and $[n+1,1,1]_F(z) = P_{n+1}(z)Q_1(z)^{-1}$ be the square Padé approximant of $F(z)$ under the condition (2.4). Then, if $Q_1(1)$ is nonsingular and $\Delta S_n^{(l,m)}$ is full rank in column, we have

$$[n+1,1,1]_F(1) = S_n^{(l,m)} - \Delta S_n^{(l,m)}(\Delta^2 S_n^{(l,m)})^{+}\Delta S_n^{(l,m)}.$$

Proof. Now we compute $[n+1,1,1]_F(1)$. From (2.6) and (2.7)

$$A_i = \Delta S_{i-1}^{(l,m)} + \Delta S_{i-2}^{(l,m)} B_1, \quad i = 0, 1, \cdots, n+1 (\Delta S_{-1}^{(l,m)} = S_0^{(l,m)}, \Delta S_{-2}^{(l,m)} = 0);$$

$$E_{n-2} = \Delta S_{n+1}^{(l,m)} + \Delta S_n^{(l,m)} B_1,$$

$$B_1 = (-\Delta S_n^{(l,m)})^{+}\Delta S_{n+1}^{(l,m)},$$

then

$$Q_1(1) = I + B_1 = (\Delta S_n^{(l,m)})^{+}\Delta S_n^{(l,m)} - (\Delta S_n^{(l,m)})^{+}\Delta S_{n+1}^{(l,m)} = -(\Delta S_n^{(l,m)})^{+}\Delta^2 S_n^{(l,m)}, \tag{4.2}$$

$$[n+1,1,1]_F(1) = P_{n+1}(1)(Q_1(1))^{-1} = \Big(\sum_{i=-1}^{n} \Delta S_i^{(l,m)} + \sum_{i=-1}^{n-1} \Delta S_i^{(l,m)} B_1\Big)(I+B_1)^+$$

$$= (S_{n+1}^{(l,m)} + S_n^{(l,m)} B_1)(I+B_1)^+ = S_n^{(l,m)} + \Delta S_n^{(l,m)}(I+B_1)^+$$

$$= S_n^{(l,m)} - \Delta S_n^{(l,m)}(\Delta^2 S_n^{(l,m)})^+ \Delta S_n^{(l,m)}.$$

Note 4.2. Obviously, $h_n^{(l,m)}(i,j)$ defined in (4.1) is the (i,j)-th block of $[n+1,1,1]_F(1)$.

An important property of (4.1) is that , if $\Delta^2 S_n^{(l,m)}$ is full rank in column, $h_n^{(l,m)}(i,j)$ doesn't depend on j, for $j = 1, \cdots, m+1$. In fact

$$h_n^{(l,m)}(i,j+1) - h_n^{(l,m)}(i,j) = \Delta s_{n+i+j-2} - \Delta S_{n+i-1}^{(1,m)}(\Delta^2 S_n^{(l,m)})^+ \Delta^2 S_{n+j-1}^{(l,1)}$$

$$= \Delta s_{n+j+i-2} - \Delta S_{n+i-1}^{(1,m)} e_j = 0,$$

here e_j denotes the j-th column of the unit matrix.

Proposition 2. Let $f(z) = \sum_{j=0}^{\infty} \Delta s_{j-1} z^j$, $\Delta S_{-1} = S_0$, and $[n+m,m,l]_f(z)$ be the square Padé approximant of $f(z)$ under the condition (2.5). Then, if $\Delta^2 S_n^{(l,m)}$ is full rank in column,

$$[n+m,m,l]_f(1) = h_n^{(l,m)}(1,j), \quad j = 1, \cdots, m+1.$$

Proof. We only need to consider the case $j = m+1$. Under the condition (2.5),

$$[n+m,m,l]_f(1) = \sum_{i=0}^{n+m} A_i = \sum_{i=0}^{m} s_{n+m-i} B_i = s_{n+m} - \sum_{i=1}^{m}(s_{n+m} - s_{n+m-i}) B_i$$

$$= s_{n+m} - \sum_{i=1}^{m}(\sum_{j=1}^{i} \Delta s_{n+m-j}) B_i = s_{n+m} - \sum_{i=1}^{m} \Delta s_{n+m-i} \sum_{j=i}^{m} B_j$$

$$= s_{n+m} - \sum_{i=1}^{m} \Delta s_{n+m-i} B_i'.$$

It follows from (2.10) that

$$[n+m,m,l]_f(1)$$

$$= s_{n+m} - [\Delta s_{n+m-1}, \cdots, \Delta s_n] \begin{bmatrix} \Delta^2 s_{n+m-1} & \cdots & \Delta^2 s_n \\ \vdots & & \vdots \\ \Delta^2 s_{n+m+l-2} & \cdots & \Delta^2 s_{n+l-1} \end{bmatrix}^+ \begin{bmatrix} \Delta s_{n+m} \\ \vdots \\ \Delta s_{n+m+l-1} \end{bmatrix}$$

$$= s_{n+m} - \Delta S_n^{(1,m)}(\Delta^2 S_n^{(l,m)})^+ \Delta S_{n+m}^{(l,1)} = h_n^{(l,m)}(1,m+1).$$

Theorem 3. *For a given matrix sequence $\{s_0, s_1, \cdots\}$, $s_i \in C^{p\times q}$ if there exist matrices $a_i \in C^{q\times q}, i = 0, \cdots, m$ and a matrix $s \in C^{p\times q}$, such that*

$$\sum_{i=0}^{m}(s_{n+i} - s) a_i = 0, \forall n \quad \text{and} \quad \sum_{i=0}^{m} a_i = I_q, \quad \det a_m \neq 0 \tag{4.3}$$

then, if $\Delta S_n^{(l,m)}$ *has full rank in column, we have*

$$h_n^{(l,m)}(i,j) = s, \quad i = 1, \cdots, l; \quad j = 1, \cdots, m.$$

Proof. From (4.3), we have

$$\sum_{i=0}^{m} \Delta s_{n+i} a_i = 0,$$

which implies $\Delta s_{n+m} = -\sum_{i=0}^{m-1} \Delta s_{n+i}(a_i a_m^{-1})$. Then

$$\Delta S_{n+1}^{(l,m)} - \Delta S_n^{(l,m)} = \Delta S_n^{(l,m)}(B - I) \quad \text{with} \quad B = \begin{bmatrix} 0 & 0 & \cdots & 0 & -a_0 a_m^{-1} \\ I_q & 0 & \cdots & 0 & -a_1 a_m^{-1} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & I_q & -a_{m-1} a_m^{-1} \end{bmatrix}.$$

From $(\Delta S_n^{(l,m)})^+ \Delta S_n^{(l,m)} = I$, we have now

$$H_n^{(l,m)} = S_n^{(l,m)} - \Delta S_n^{(l,m)}(B - I)^{-1}$$

where $H_n^{(l,m)} = (h_n^{(l,m)}(i,j))_{i=1,j=1}^{l,m}$. On the other hand

$$(B - I)^{-1} = \operatorname{diag}(-\sum_{i=1}^{m} a_i, -\sum_{i=2}^{m} a_i, \cdots, -\sum_{i=m}^{m} a_i)(\delta_{i-j})_{i,j=1}^{m}$$
$$+ \operatorname{diag}(\sum_{i=0}^{0} a_i, \sum_{i=0}^{1} a_i, \cdots, \sum_{i=0}^{m-1} a_i)(\delta_{j-i-1})_{i,j=1}^{m},$$

where

$$\delta_k = \begin{cases} I_q, & k \geq 0; \\ 0, & k < 0. \end{cases}$$

Thus we have $H_n^{(l,m)} = S_n^{(l,m)} - [S_n^{(l,1)} - S^{(l,1)}, S_{n+1}^{(l,1)} - S^{(l,1)},$

$$\cdots, S_{n+m-1}^{(l,1)} - S^{(l,1)}] = [S^{(l,1)}, \cdots, S^{(l,1)}] \text{ where } S^{(l,1)} = \left.\begin{bmatrix} s \\ \vdots \\ s \end{bmatrix}\right\} l,$$

which completes the proof.

§5. Numerical Examples

Matrix or vector sequences are often generated in numerical computation. Now, we will give three examples of vector sequences and compare, according to some principle, the effect of the given accelerating formulas in §4 with the commonly used vector ϵ−algorithm given by Wynn [6]: For a vector sequence $\{s_n\}_{n=0}^{\infty}$,

$$\varepsilon_{-1}^{(n)} = 0, \quad \varepsilon_0^{(n)} = s_n, \quad \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + (\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)})^{-1}$$

where the Samelson inverse y^{-1} of $y \in C^p$ is defined by $y^{-1} = \bar{y}/\|y\|_2^2$.

Comparing Principle: The largest subscripts of the initial vector sequence used in the computation of two compared values are the same. For instance, we compare

$h^{(l,m)}_{k-l-m}(1,1)$ with ε^{k-2m}_{2m}, $l = 1, 2, \cdots; m = 1, 2, \cdots; k = l+m, l+m+1, \cdots$; the largest subscript used is k.

The entries in Tables 1–3 denote the numbers of significant digits SD defined by

$$\text{SD} = -\log_{10}\| \text{ Exact Solution } - \text{ Approximate Solution } \|,$$

Example 1. In this example, we want to find the numerical solution $\vec{y}(1)$ of the ordinary differential equations

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_1, \\ y_3' = -y_3, \end{cases} \qquad \begin{cases} y_1(0) = -1, \\ y_2(0) = 0, \\ y_3(0) = 1 \end{cases}$$

by the well known Euler method.

The initial sequence is generated by taking step lengths 2^{-k} $(k = 0, 1, \cdots)$ successively. The true solution is $\vec{y}(1) = (-cos(1), sin(1), e^{-1})^T$.

Table 1 (Example 1)

$k(\geq 0)$	6	8	10	12	14	16
s_k	2.18	2.78	3.39	3.99	4.59	5.19
$h^{(1,1)}_{k-2}$	3.60	4.81	6.01	7.22	8.42	9.62
ε^{k-2}_2	3.91	5.11	6.31	7.52	8.72	9.93
$h^{(1,2)}_{k-3}$	4.88	6.69	8.50	10.3	12.1	13.2
ε^{k-4}_4	4.86	6.64	8.43	10.2	12.0	12.7
$h^{(1,3)}_{k-4}$	5.70	8.14	10.6	12.9	11.2	12.9
ε^{k-6}_6	5.00	7.55	10.0	12.5	11.1	13.1

Example 2[5]**.** Here we produce the initial vector sequence by the quadratic iteration $s_{n+1} = G(s_n)$, where $s_0 = (1.5, 1.6, 1.7, 1.8)^T$, $G(s) = b + As + Q(s)$, with

$$A = \begin{bmatrix} 2.25 & 0.01 & 0.05 & 0.50 \\ 0.01 & 1.75 & 0.00 & 0.05 \\ 0.05 & 0.00 & 1.75 & 0.01 \\ 0.50 & 0.05 & 0.01 & 2.25 \end{bmatrix}, \quad b = \begin{bmatrix} -0.81 \\ -0.31 \\ -0.31 \\ -0.81 \end{bmatrix}, \quad Q(x) = -0.5 \begin{bmatrix} x_1^2 + x_1 x_4 \\ x_2^2 \\ x_3^2 \\ x_1 x_4 + x_4^2 \end{bmatrix}.$$

The sequence $\{s_n\}$ converges to the vector $s = (1, 1, 1, 1)^T$.

Table 2 (Example 2)

$k(\geq 0)$	4	6	8	10	12	14
s_k	1.02	1.28	1.51	1.72	1.92	2.12
$h_{k-2}^{(1,1)}$	1.37	1.77	2.10	2.42	2.72	3.04
ε_2^{k-2}	1.39	1.84	2.23	2.59	2.92	3.25
$h_{k-3}^{(1,2)}$	1.84	2.41	2.89	3.14	3.47	3.85
ε_4^{k-4}	1.41	2.11	2.65	3.15	3.64	4.12
$h_{k-4}^{(1,3)}$	2.67	2.92	3.27	3.62	3.57	
ε_6^{k-6}	2.11	2.88	3.54	4.19	4.80	
$h_{k-5}^{(1,4)}$		2.86	2.62	2.93	3.52	
ε_8^{k-8}		2.88	3.75	4.55	5.27	

Example 3[6]. The initial vector sequence is produced by the quadratic iteration $s_{n+1} = G(s_n)$, where $s_0 = (1.6, 1.7, 2.2, 1.9)^T$, $G(s) = b + As + Q(s)$, with

$$A = \begin{bmatrix} 3.9 & -3.7 & 2.4 & -0.6 \\ 2.4 & -2.0 & 2.2 & -0.6 \\ 2.4 & -3.6 & 4.1 & -0.9 \\ 2.8 & -5.2 & 4.8 & -0.4 \end{bmatrix}, \quad b = \begin{bmatrix} -0.75 \\ -0.75 \\ -0.75 \\ -0.75 \end{bmatrix}, \quad Q(x) = -0.25 \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ x_4^2 \end{bmatrix}$$

The iteration has both $(1,1,1,1)^T$ and $(3,3,3,3)^T$ as its fixed points, for the starting point given here, the iterative sequence converges to the latter one.

Table 3 (Example 3)

$k(\geq 0)$	8	10	12	14	16	18
s_k	2.20	2.80	3.40	3.99	4.59	5.19
$h_{k-2}^{(1,1)}$	1.73	2.32	2.90	3.48	4.08	4.81
ε_2^{k-2}	1.88	2.45	3.01	3.54	4.04	4.63
$h_{k-3}^{(1,2)}$	2.60	3.86	5.12	6.30	7.41	8.48
ε_4^{k-4}	2.42	3.53	4.69	5.85	6.98	8.08
$h_{k-4}^{(1,3)}$	2.88	4.41	6.18	7.04	8.14	9.31
ε_6^{k-6}	2.32	3.56	5.10	6.97	7.78	8.87
$h_{k-5}^{(1,4)}$	2.47	5.12	6.06	7.13	8.24	9.36
ε_8^{k-8}	2.45	3.62	5.12	6.90	7.70	9.22

Comparing our method with the powerful vector ε-algorithm, we can see from these examples that the square Padé approximation, a nonlinear method, is also an efficient approach in convergence acceleration of sequences and in some cases it is even more powerful than vector $\varepsilon-$algorithm.

References

[1] D.Bessis and P.R. Graves-Morris (ed.), Topics in the theory of Padé approximants, Inst. of Phys., Bristol, 1973, 19–44.

[2] I.W. Daniel, W.B. Gragg, L.Kaufman and G.W. Stewart, Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization, *Math. Comp.*,**30** (1976), 772–795.

[3] B.Friedlander, M.Morf, T Kailath and L.Liung, New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices, *Lin. Alg. Appl.*, **27** (1979), 31–60.

[4] H. Sadok, About Henrici's transformation for accelerating vector sequences, *J. Comp. Appl. Math.*, **29** (1990), 101–110.

[5] D.A. Smith, W.F.Ford and A.Sidi, Extrapolation methods for vector sequences, *SIAM Rev.*, **29** (1987), 199–233.

[6] P. Wynn, Acceleration techniques for iterative vector and matrice problems, *Math. Comp.*, **16** (1962), 301–322.

[7] Xu Guo-liang,The existence and uniqueness of matrix Padé Approximants, *J.Comp. Math.*, **8** : 1 (1990), 65–74.

[8] Xu Guo-liang and Li Jia-kai, Generalized matrix Padé Approximants, *Approx. Theory and its Appl.*, **5** : 4 (1989), 47–60.

[9] Xu Guo-liang and A.Bultheel, Matrix Padé Approximation: Definitions and Properties, *Lin. Alg. Appl.*, 137/138(1990), 67–136.