Fast algorithms for the factorization of Hankel and Toeplitz matrices and the Padé approximation problem.

by A. BULTHEEL

Report TW 42, October 1978

Applied Mathematics and Programming Division
Katholieke Universiteit Leuven (Belgium)

Fast algorithms for the factorization of Hankel and Toeplitz matrices and the Padé approximation problem.

Abstract

The denominators of Padé approximants are the solutions of linear systems of equations with Hankel or Toeplitz matrix. If the denominators are computed for a whole path of the Padé table then here exist recursions between successive elements of these paths.
These correspond to fast algorithms for the factorization of the corresponding matrices or their inverse. These algorithms are usuallywell known in the normal case. Their generalization for the non-normal case will also be given.

1. Introduction

Given a formal power series

$$f(z) = \sum_{i=0}^{\infty} f_i z^i \tag{1}$$

We want a rational approximation (Padé approximant)

$$[m/n] = P^{[m/n]}(z) \,/\, Q^{[m/n]}(z) \tag{2}$$

being an order (m + n + 1) - fit of the given series. I.e.

$$\frac{P^{[m/n]}(z)}{Q^{[m/n]}(z)} = \frac{\sum_{i=0}^{m} p_i z^i}{\sum_{i=0}^{n} q_i z^i} = \sum_{i=0}^{m+n} f_i z^i + O(z^{m+n+1}) \quad \cdots \tag{3}$$

or

$$Z^{[m/n]} = Q^{[m/n]}(z) f(z) - P^{[m/n]}(z) = O(z^{m+n+1}) \tag{4}$$

This corresponds to the solution of the following system of linear equations.

$$\begin{bmatrix} p_0 \\ \vdots \\ p_m \\ \hline 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} f_0 & f_{-1} & \cdots & f_{-n} \\ f_1 & f_0 & & \vdots \\ \vdots & & \ddots & \\ f_m & & & \\ \hline f_{m+1} & & & \vdots \\ & \ddots & & \\ & & & f_{m-1} \\ f_{m+n} & & f_{m+1} & f_m \end{bmatrix} \begin{bmatrix} q_0 \\ \vdots \\ q_n \end{bmatrix} = \begin{bmatrix} F_1 \\ \hline F_2 \end{bmatrix} \begin{bmatrix} q_0 \\ \vdots \\ q_n \end{bmatrix} \tag{5}$$

Where it is supposed, as we will do in the following that non existing f_i are zero ($f_{-1} = f_{-2} = \ldots = 0$).

In the normal case, P and Q are uniquely defined up to a constant factor. This one can be fixed by some normalizing condition e.g. we could impose $q_n = 1$ or $q_0 = 1$. We will refer to this as a monic, resp. comonic normalization. The numerators can always be obtained by simple evaluation of

$$\begin{bmatrix} p_0 \\ \cdot \\ \cdot \\ \cdot \\ p_m \end{bmatrix} = F_1 \begin{bmatrix} q_0 \\ \cdot \\ \cdot \\ \cdot \\ q_n \end{bmatrix}$$

We thus only have to solve

$$F_2 \begin{bmatrix} q_0 \\ \cdot \\ \cdot \\ \cdot \\ q_n \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

or if we want to make this a square system

$$\begin{bmatrix} f_{m+1} & f_m & \cdots & f_{m-n+1} \\ \cdot & \searrow & \searrow & \vdots \\ \cdot & & \searrow & \\ \cdot & & & \\ & & & \\ f_{m+n+1} & & & f_{m+1} \end{bmatrix} \begin{bmatrix} q_0 \\ \cdot \\ \cdot \\ \cdot \\ \\ q_n \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \\ r_{m,n} \end{bmatrix} \qquad (6)$$

We will in the following indicate for the monic normalization $\overline{P} / \overline{Q}$ and for the comonic $\tilde{P} / \tilde{Q}$ while P / Q will mean either of the two.

The Padé table and its structure in the non normal case are well known (see e.g. [20]).

We will further introduce the following notation :

Two index - pairs (i,j) and (k,ℓ) are linked if (k,ℓ) is one of the eight positions surrounding (i,j) in the two dimensional array of these pairs.

An (ordered) set of index - pairs $\{(k_i,\ell_i),\ i = 0, 1, \ldots\}$ is called a chain if (k_i,ℓ_i) and (k_{i+1},ℓ_{i+1}) are linked $\forall i = 0, 1, 2, \ldots$

If C is a chain then for some matrix M we set $P(M, C) = \{M_{i,j} | (i, j) \in C$ and equally ordered$\}$ a path of M with chain C.

Let F be the double infinite Toeplitz matrix with elements $F_{i,j} = f_{i-j}$ which is lower triangular and T be the matrix which has as its elements $\mathsf{T}_{i,j}$, the square Toeplitz matrix that is a submatrix of F and has as upper corners $F_{i+1,0}$ and $F_{i+1,j}$

P will indicate the Padé table of some formal series.

Consider a chain C with i - th element (k_i, ℓ_i), then the denomination of the i - th element of $P(\mathsf{P}, C)$ is the solution of a system like (6) with as a matrix the i - th element of $P(\mathsf{T}, C)$.

When it is clear from the context what C is, then we will indicate the i - th element of $P(\mathsf{P}, C)$ by $P_i(z) / Q_i(z)$ and if we drop the argument z we usually mean the corresponding column vectors e.g.

$$P_i = [P_{0,i} \ \cdots \ P_{k_i,i}]^T \ ; \ Q_i = [Q_{0,i} \ \cdots \ Q_{\ell_i,i}]^T$$

and the i - th element of $P(\mathsf{T}, C)$ will be T_i

We thus have

$$T_i Q_i = e_{\ell_i} r_i$$

(e_k is the k - th unit vector)
The ratio factor r_i is the leading coefficient in the residual Z_i of the approximant $P_i(z) / Q_i(z)$.

J will be the permutation matrix $\begin{bmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{bmatrix}$

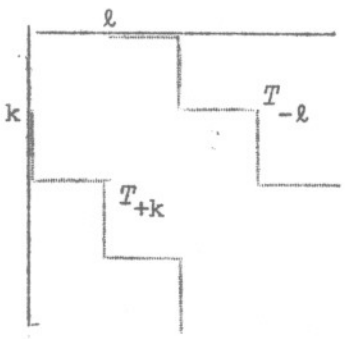
A matrix M is persymmetric if $JM^TJ = M$
(= symmetry with respect to the antidiagonal)

For some vector x we set $\tilde{x} = Jx$.

$[\emptyset]$ will denote a matrix with zero rows and/ or zero columns.

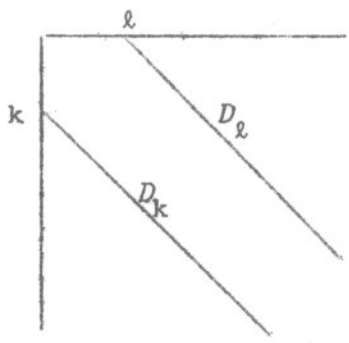
Some paths usually considered are staircases, rows, diagonals, columns, sawteeth. We will use the notation of [6] and represent graphically the notation used ($k, \ell \geqslant 0$ everywhere)

descending staircases

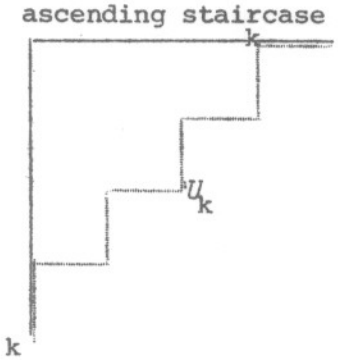


(note $T_{+0} \neq T_{-0}$)

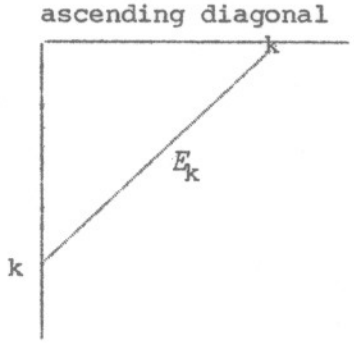
descending diagonals



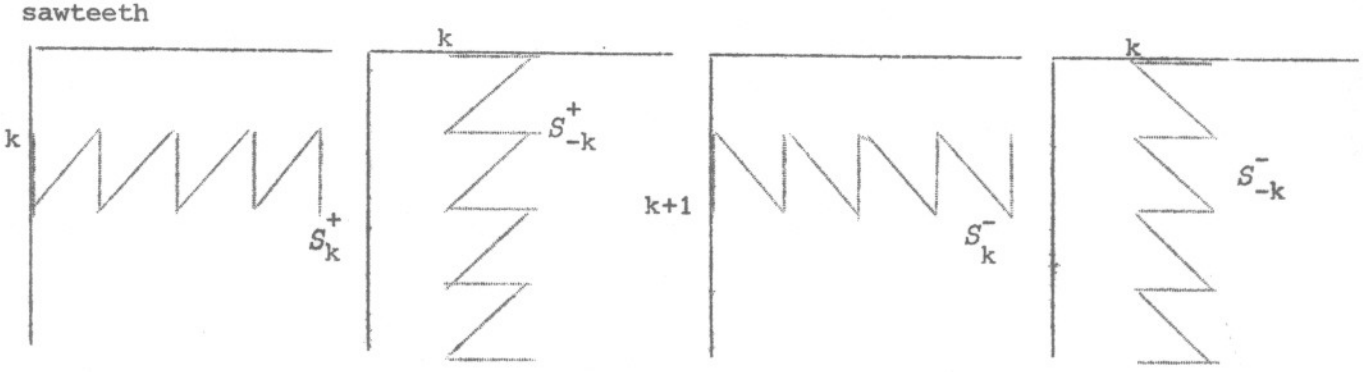
ascending staircase



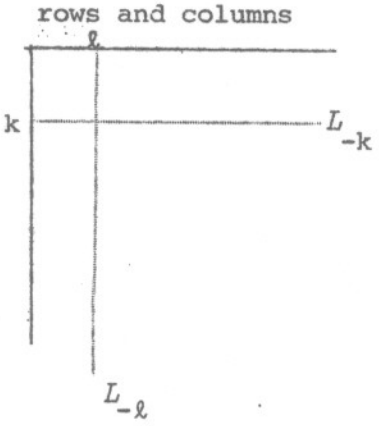
ascending diagonal



sawteeth



rows and columns



The system (6) has a Toeplitz structure. There are some well known fast algorithms (Trench [32] e.g.) requiring $O(n^2)$ operations to solve it, instead of $O(n^3)$ for classical system solvers.

If we want to compute all the Padé approximants that are on some path, then the systems that have to be successively solved differ from each other only by the introduction of one or two new f_i. The updating of the solution of these systems require only of $O(n)$ operations. We thus do not only find the solution of (6) in $O(n^2)$ operations but also we have as intermediate results the denominators of all the Padé approximants that are on some path P, going trough the required approximant, , that precede this approximant on this path.

The matrices in $P(\top, C)$ may eventually be nested such that their triangular factorizations are nested too. In that case we have as a final result also the complete triangular factorization of a Toeplitz matrix or its inverse.

The special updating of the solution of (6) was used explicitely by Brezinski in his bordering algorithm for a descending diagonal, he thus refinds the recursion of van Rossum [35] but the recursion for other paths may be looked at in a similar way.

2. Toeplitz matrices, Padé rows and - Sawteeth; normal case

Suppose we want to compute a row of the Padé table with underlying chain $L_k \quad (k \geqslant +0)$

We have to solve

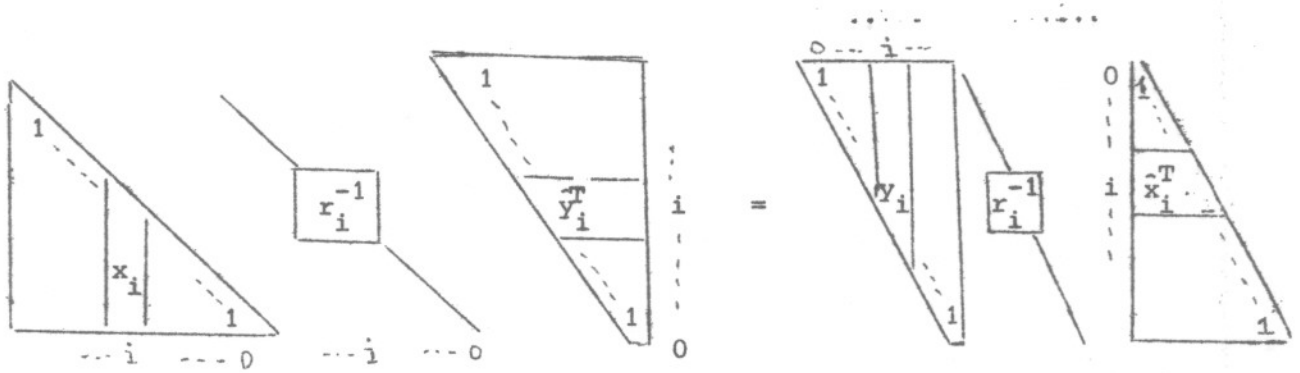
$$T_i \, Q_i = e_{i+1} \, r_i \tag{7}$$

where

$$T_{i+1} = \left\{ \begin{array}{c|c} t_0 & w_i^T \\ \hline v_i & T_i \end{array} \right\} = \left\{ \begin{array}{c|c} T_i^{\bullet} & \hat{w}_i \\ \hline \hat{v}_i^T & t_0 \end{array} \right\} \tag{8}$$

where we have set $t_j = f_{k+1+j}$

Because of the persymmetry of a Toeplitz matrix we can obtain the factorizations.

$T_n^{-1} =$ 

$= \tilde{L}^{-1} \quad \tilde{D}^{-1} \quad \tilde{U}^{-1} = \quad U^{-1} \quad D^{-1} \quad L^{-1}$ (9)

with

$$T_n[x_n|y_n] = [e_1|e_{n+1}]r_n \; ; \; x_{n0} = y_{nn} = 1$$

and from (7) $\quad \bar{Q}_n = y_n$

The highest degree coefficient of $\bar{P}_i$ is

$$\delta_i = w_i^T y_i$$

as may be seen from (5).

The recursion for a row are mentioned by Bussonnais [7] e.g. and can be restated for the monic case as :

$$y_{n+1} = \begin{bmatrix} 0 \\ y_n \end{bmatrix} + \frac{\delta_n \, r_n^{-1}}{\delta_{n-1} \, r_{n-1}^{-1}} \begin{bmatrix} y_n \\ 0 \end{bmatrix} - \frac{\delta_n}{\delta_{n-1}} \begin{bmatrix} 0 \\ y_{n-1} \\ 0 \end{bmatrix} \tag{10}$$

where

$$\begin{bmatrix} \delta_n \\ \hline r_n \end{bmatrix} = \left[\begin{array}{c} w_n^T \\ \hline \tilde{v}_{n-1}^T \; t_0 \end{array} \right] y_n$$

and with initial conditions

$$y_{-1} = [\emptyset], \; \delta_{-1} = 1, \; r_{-1} = -1$$

$$y_0 = 1, \; \delta_0 = t_{-1}, \; r_0 = t_0$$

Closely related to this are the sawteeth recursion. They compute two successive rows. S_k^+ e.g. is an interleaving of L_k and L_{k+1}.

If we indicate the denominators of the k - th row with y_i and the denominators of the (k+1) - th row with y_i^* both with a monic normalization say; thus

$$y_i = \overline{Q}_{2i} \quad ; \quad y_i^* = \overline{Q}_{2i-1}$$

where $P(P,\ S_k^+) = \{\overline{P}_i/\overline{Q}_i \ ;\ i = 0,\ 1,\ 2,\ \ldots\}$ and

if we correspondingly call the Toeplitz matrices of $P(T,\ S_k^+)$ T_i and T_i^* then we have

$$T_i\ y_i = e_{i+1}\ r_i \ ;\ T_i^*\ y_i^* = e_{i+1}\ r_i^* \tag{11}$$

with the recursions :

$$y_{i+1} = \begin{bmatrix} 0 \\ y_i \end{bmatrix} - \begin{bmatrix} y_i^* \\ 0 \end{bmatrix} \frac{\delta_i}{\delta_i^*} \tag{12}$$

$$y_{i+1}^* = y_{i+1} - \begin{bmatrix} y_i^* \\ \\ 0 \end{bmatrix} \frac{r_{i+1}}{r_i^*}$$

where

$$\begin{bmatrix} \delta_i \\ \hline r_i \end{bmatrix} = \left[\begin{array}{ccc} t_{-1} & \cdots & t_{-(i+1)} \\ \hline t_i & \cdots & t_0 \end{array}\right] y_i = \left[\begin{array}{c|c} w_i^T & \\ \hline \hat{v}_{i-1}^T & t_0 \end{array}\right] y_i \tag{13}$$

and

$$\begin{bmatrix} \delta_i^* \\ \hline r_i^* \end{bmatrix} = \left[\begin{array}{ccc} t_0 & \cdots & t_{-i} \\ \hline t_{i+1} & \cdots & t_1 \end{array}\right] y_i^* = \left[\begin{array}{c|c} t_0 & w_{i-1}^T \\ \hline & \hat{v}_i^T \end{array}\right] y_i^*$$

Instead of the definition of y_i^* from (11) we could also have introduced x_i satisfying

$$T_i \, x_i = e_1 \, s_i$$

then

$$T_i^{\star} \, x_i = [0 \ \ldots \ 0 \ f_i]^T \quad ; \quad f_i = [\hat{v}_{i-1}^T \,|\, t_0] \, x_i$$

so that x_i is proportional to $y_i^{\star}$:

$$x_i \sim y_i^{\star} \tag{14}$$

We will reserve the notation x_i for a comonic normalization and $y_i^{\star}$ for a monic normalization $(x_{oi} = y_{ii}^{\star} = 1)$.

The other sawtooth (S_k^-) would give in a monic notation :

$$y_{i+1} = \left(\begin{bmatrix} 0 \\ \\ y_i \end{bmatrix} - \begin{bmatrix} y_{i+1}^{\star} \\ \\ \end{bmatrix} R \right) / (1 - R) \ ; \ R = \frac{\delta_i}{\delta_i^{\star}} \tag{15}$$

$$y_{i+1}^{\star} = \begin{bmatrix} 0 \\ \\ y_i \end{bmatrix} - \begin{bmatrix} y_i^{\star} \\ \\ 0 \end{bmatrix} \frac{r_i}{r_i^{\star}}$$

r_i, $r_i^{\star}$, δ_i, $\delta_i^{\star}$ defined as before

Still another possibility is to take a combination of (12) and (15) to give :

$$y_{i+1} = \begin{bmatrix} 0 \\ \\ y_i \end{bmatrix} - \begin{bmatrix} y_i^{\star} \\ \\ 0 \end{bmatrix} \frac{\delta_i}{\delta_i^{\star}} \tag{16}$$

$$y_{i+1}^{\star} = \begin{bmatrix} 0 \\ \\ y_i \end{bmatrix} - \begin{bmatrix} y_i^{\star} \\ \\ 0 \end{bmatrix} \frac{r_i}{r_i^{\star}}$$

I.e. we use the S_k^+ - recursion for the y_{i+1} and the S_k^- - recursion for $y_{i+1}^{\star}$.

Two of the inner products in (13) may be replaced by recurrences. (16) is exactly the Trench - Zohar [32, 39, 38] algorithm (see also [1, 12, 13, 24, 37]) with a monic normalization. I.e. with $y^{\star}_{ii} = 1$ unstead of the usual $x_{oi} = 1$. With the latter normalization it may be seen from (9) that $r^{\star}_{i}$ and r_i are actually the same and the usual formulation of the Trench - Zohar algorithm then is :

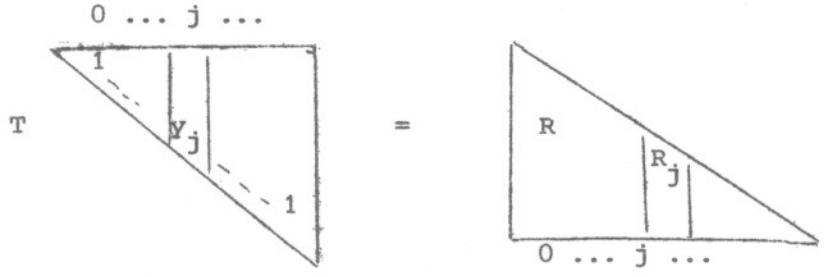
$$y_{n+1} = \begin{pmatrix} 0 \\ \\ y_n \end{pmatrix} - \begin{pmatrix} x_n \\ \\ 0 \end{pmatrix} \delta_n \, r_n^{-1} \; ; \qquad (17)$$

$$x_{n+1} = \begin{pmatrix} x_n \\ \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ \\ y_n \end{pmatrix} \sigma_n \, r_n^{-1}$$

$$\delta_n = w_n^T y_n \quad ; \quad \sigma_n = \hat{v}_n^T x_n$$

$$r_{n+1} = r_n - \delta_n \, r_n^{-1} \, \sigma_n$$

All these algorithms are thus factorization algorithms for T^{-1} (see 9) and they may be derived as such. i.e. as fast elimination methods to triangularize matrices of Toeplitz structure.
Now, we could triangularize T itself, which is closely related to Rissanen's derivation [29] , by considering



(18)

$$R_{i,\,j} \overset{\Delta}{=} [\, t_i \; \ldots \; t_{i-j} \,]\, y_j \quad , \quad i = 0, 1, \ldots \; ; \quad j = 0, 1, \ldots, i$$

Since we now have already recursions for the y_j, we may use them to find recursions for R_j.

E.g. recursions (17) from Trench's algorithm would give :

$$R_{i,\,j+1} = [\,t_{i-1} \;\cdots\; t_{i-j-1}]\,y_j - [\,t_i \;\cdots\; t_{i-j}]\,x_j\;\delta_j\;r_j^{-1}$$

$$= R_{i-1,\,j} - R^{\star}_{i-1,\,j}\;\delta_j\;r_j^{-1}$$

and

$$R^{\star}_{i-1,\,j+1} = [\,t_i \;\cdots\; t_{i-j-1}]\,x_{j+1} \qquad (19)$$

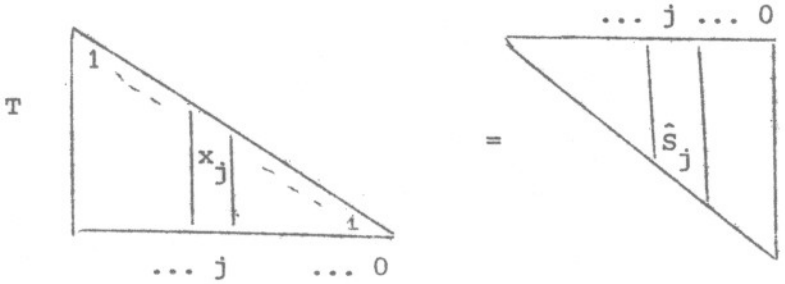
$$= [\,t_i \;\cdots\; t_{i-j}]\,x_j - [\,t_{i-1} \;\cdots\; t_{i-j-1}]\,y_j\;\sigma_j\;r_j^{-1}$$

$$= R^{\star}_{i-1,\,j} - R_{i-1,\,j}\;\sigma_j\;r_j^{-1}$$

$$R_{i,\,o} = t_i \quad ; \quad R_{i,\,i} = r_i$$

$$R^{\star}_{i,\,o} = t_{i+1} \quad ; \quad R^{\star}_{i,\,i} = \sigma_i$$

The scheme (19) makes it possible to compute R recursively, column by column. Only the value of δ_i does not follow from this scheme, so we still have to compute y_i from scheme (17). But if we would, sumultaneously with R also compute the other triangular factor of T by



(20)

and introducing the notation (z - transform)

$$R_j(z) = \sum_{i=j}^{\infty} R_{ij}\,z^{i-j} \quad \text{etc ...}$$

then we get the following scheme :

$$R_0(z) = \sum_{i=0}^{\infty} t_i z^i \qquad S_0(z) = \sum_{i=0}^{\infty} t_{-i} z^i$$

$$R_0^{\star}(z) = \sum_{i=0}^{\infty} t_{i+1} z^i \qquad S_0^{\star}(z) = \sum_{i=0}^{\infty} t_{-i-1} z^i$$

$$\text{for } i = 0, 1, 2, \ldots \tag{21}$$

$$r_i = S_i(0) = R_i(0)$$

$$\sigma_i = R_i^{\star}(0) \qquad \delta_i = S_i^{\star}(0)$$

$$\beta_i = \sigma_i r_i^{-1} \qquad \alpha_i = \delta_i r_i^{-1}$$

$$R_{i+1}(z) = R_i(z) - R_i^{\star}(z)\,\alpha_i \qquad S_{i+1}(z) = S_i(z) - S_i^{\star}(z)\,\beta_i$$

$$R_{i+1}^{\star}(z) = \frac{1}{z}\left(R_i^{\star}(z) - R_i(z)\,\beta_i\right) \qquad S_{i+1}^{\star}(z) = \frac{1}{z}\left(S_i^{\star}(z) - S_i(z)\,\alpha_i\right)$$

For the Padé approximation problem, S_i and $S_i^{\star}$ are finite series of constant degree. E.g. for the zeroth row S_i is of first degree and $S_i^{\star}$ of degree zero, but the scheme (21) is equaly valid for infinite series S_i, $S_i^{\star}$.

From (21), it is clear that we do not need the evaluation of y_i any more but only the coefficients used in their recursion. This corresponds to the continued fraction algorithm - approach of [6]. As a matter of fact (21) is about the same algorithm as RTOE

If we set

$$T_i^{\infty} = \begin{vmatrix} t_{-1} & & \\ t_0 & t_{-1} & \\ t_1 & & t_{-1} \\ & & t_0 \\ & & t_1 \end{vmatrix} \tag{22}$$

a matrix of dimension $\infty \times (i+1)$, then

$$T_i^{\infty}\,\overline{Q}_i = [\,\ldots\ 0\ \overbrace{\overline{P}_i^T}^{k+1}\ \vdots\ 0\ \ldots\ R_{0i}\ \vdots\ R_{1i}\ \ldots\,]^T \tag{23}$$

where the subdivision of the RMS of (23) corresponds to the subdivision of (22).

$\overline{P}_i(z)\ /\ \overline{Q}_i(z)$ is the, monically normalized, Padé - approximant, the i - th of row k and

$$z^{k+i+1}\ R_i(z) = \overline{Z}_i(z) \tag{24}$$

the corresponding residual.

We know that $\overline{Q}_i$, $\overline{P}_i$, $\overline{Z}_i$ have the same recurrence relation. (24), (21) and (17) then will make it possible to prove by induction that $S_i^{\star}$ is exactly $\hat{\overline{P}}_i$

From (14) we find the interpretation of the other two quantities involved in (21) : $R_i^{\star}$ and S_i. They correspond to the residuals and the numerators for row (k+1) with a comonic normalization :

$$x_i = \tilde{Q}_i\ ;\ R_i^{\star} = \tilde{Z}_i \quad \text{and} \quad \hat{S}_i = \tilde{P}_i$$

where $\tilde{Q}_i$, $\tilde{P}_i$, $\tilde{Z}_i$ are quantities corresponding to the (k+1) - th row.

Of course, the other schemes : (10), (12), and (15) will also give recursions for the columns R_i of (18). The recursion (10) is readily transcribed in a recursion for the residual R_i, but we need beside of this the recursion for the numerators or for the denominators to obtain the necessary recurrence coefficients. Note that the computation of the numerators is equivalent to the computation of the factor $S^{\star}$ obtained from a factorization of the form (20) for $T^{\star}$. For schemes (12) and (15) we may, as for the Trench - Zohar algorithm (16) compute the factorizations of T_i and $T_i^{\star}$ where T_i is the i - th matrix of $P(\mathbb{T}, L_k)$

$$T_i = \left\{ \left[\begin{array}{cccc} t_0 & t_{-1} & \cdots & t_{-i} \\ \hline t_1 & \ddots & \ddots & t_{-1} \\ t_i & \ddots & \ddots & t_0 \\ \hline t_{i+1} & \cdots & \ddots & t_1 \end{array}\right] \right\} = T_i^{\star}$$

and $T_i^{\star}$ is the i - th matrix of $P(\mathbb{T}, L_{k+1})$.

The S_k^+ - algorithm handles the sequence

$$\ldots\ T_i,\ T_i^{\star},\ T_{i+1},\ T_{i+1}^{\star},\ \ldots$$

and thus the two new elements t_{i+1} and $t_{-(i+1)}$, added to the information when passing from T_i to T_{i+1} are introduced one at a time : first t_{i+1} (left bottom) and then $t_{-(i+1)}$ (right top) while the other sawtooth S_k^- - algorithm handles

$$\ldots,\ T_i^{\star},\ T_i,\ T_{i+1}^{\star},\ T_{i+1},\ \ldots$$

and seen as a factorization of $T_i^{\star}$ introduces first the new t_{-i} (right top) and then t_{i+2} (left bottom).
Roughly speaking this corresponds to first add the new row and then the new column or reversely first the new column and then the new row.

We thus may conclude that continued fraction - algorithm or division algorithms [27, 6] for a row are working with residuals and numerators of the Padé approximant, i.e. the factorizations of the matrices T_i belonging to $P(\top, L_k)$, and eventually of the $T_i^{\star} \in P(\top, L_{k+1})$, while the explicit computation of the denominator - coefficients such as the Trench - Zohar algorithm correspond to a factorization of the inverses of the same matrices T_i and $T_i^{\star}$. And we could also have a mix of the two for the factorizations like (18) and (20) which would correspond to a Rissanen - like approach [29].

3. Hankel matrices, Diagonals and staircases; normal case

We have already mentioned the algorithm of Brezinski computing recursively a descending diagonal D_k. If

$$P(\top, D_k) = \{T_0, T_1, \ldots\}$$

then we have to solve the systems

$$T_i\ Q_i = e_{i+1}\ r_i \qquad (25)$$

The matrices T_i are growing in the "south west" direction. Since it is more habitual to have a growth in "south east" direction, we reverse the column order to obtain Hankel matrices

$$H_i = T_i\ J \qquad (26)$$

So let us solve for y_i :

$$H_i \, y_i = e_{i+1} \, r_i \quad ; \quad y_{ii} = 1 \tag{26}$$

with

$$H_{i+1} = \left(\begin{array}{c|c} H_i & v_i \\ \hline v_i^T & h_{2i+2} \end{array} \right) \quad ; \quad H_0 = h_0 \quad ; \quad v_i^T = [h_{i+1} \ \ldots \ h_{2i+1}] \tag{27}$$

and for simplicity we assumed $k \geqslant 0$ and the elements of T_i are renumbered such that the left top element of H_i is h_0 :

$$h_j = f_{m+1+j} \tag{28}$$

then clearly :

$$\tilde{Q}_i = \hat{y}_i$$

One possible recursive solution is well known [3, 21, 29, 33, 35] and even dates back to Chebyshev. It may be resumed as follows

$$y_{i+1} = \begin{pmatrix} 0 \\ \\ y_i \end{pmatrix} - \begin{pmatrix} y_{i-1} \\ 0 \\ 0 \end{pmatrix} r_i \, r_{i-1}^{-1} - \begin{pmatrix} y_i \\ \\ 0 \end{pmatrix} \left(\frac{\lambda_i}{r_i} - \frac{\lambda_{i-1}}{r_{i-1}} \right) \tag{29}$$

$$\begin{pmatrix} \lambda_i \\ \hline r_i \end{pmatrix} = \left(\begin{array}{c} v_i^T \\ \hline v_{i-1}^T \mid h_{2i} \end{array} \right) y_i$$

with initial conditions

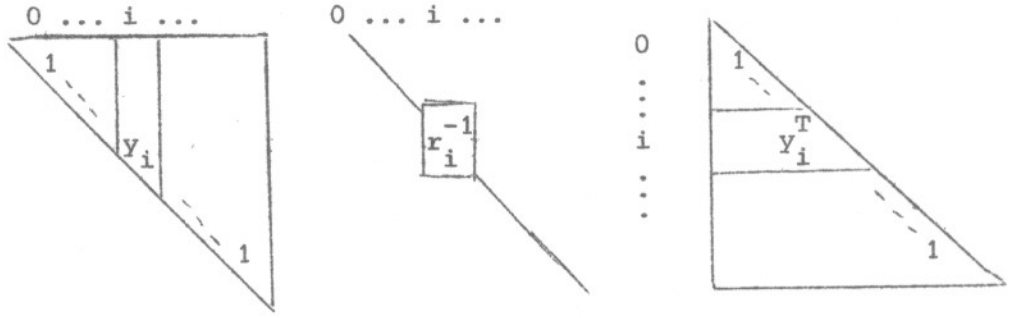
$$y_0 = 1 \quad ; \quad y_{-1} = [\emptyset]$$

$$r_0 = h_0 \quad ; \quad r_{-1} = 1$$

$$\lambda_0 = h_1 \quad ; \quad \lambda_{-1} = 0$$

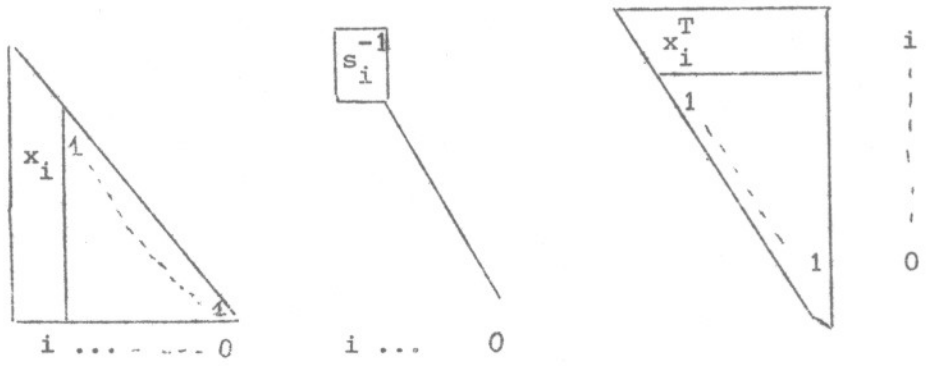
Remark that h_{2i+2} is only needed to have some normalization, i.e. if we compute r_{i+1}.

Again (29) computes the factorization of H^{-1}

$H^{-1} =$ 

(30)

(29) is the Hankel - equivalent of the Toeplitz recursion (10).
The other factorization

$H_i^{-1} =$ 

(31)

$= \quad L_i \quad D_i \quad L_i^T$

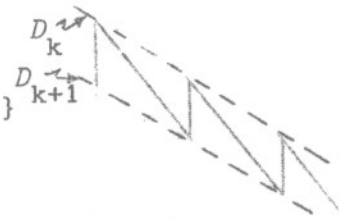
is not nested. I.e. the j - th column of L_i $j < i$ is <u>not</u> x_j.

Although Hankel and Toeplitz matrice may have much in common, there are some essential differences that prevent us from deriving completely dual algorithms. Let us briefly summarize those differences that are responsible for this lack of duality.

1) For the Toeplitz case, there is a symmetry in the growth directions "south - east" and "north - west" (see (8))while this is not so for Hankel matrices (27). Therefore we have that both factorizations (9) of the Toeplitz matrix are nested, while for the Hankel matrix only (30) is and (31) is not.

2) A Hankel matrix is symmetric while a Toeplitz matrix is persymmetric. A consequence of this is that for the Toeplitz case, there are two sawteeth (S_k^+ and S_k^-) giving the recursions (12) and (15), and their combination (16). For the Hankel matrix this may be translated as follows : The two rows of the sawteeth become the two descending diagonals D_k and D_{k+1} (for simplicity suppose $(k \geqslant +0)$ of the staircase T_k and the staircase T_k replaces S_k^+. The S_k^- would correspond to a chain

$$\{(k+1,0),\ (k,0),\ (k+2,1),\ (k+1,1),\ \ldots\} \tag{32}$$



but this chain allows no recursion.
There is no dual of T_k because whether we introduce first a column and then a row or conversely, with a Hankel matrix this makes no difference because of its symmetry. We must always introduce h_{2i+1} first and only then h_{2i+2}. There will thus not exist an algorithm for Hankel matrices that is a perfect copy of the Trench - Zohar recursion.

Let us look thus at the staircase algorithm.
We will indicate the quantities refering to the diagonal D_{k+1} with an asterix e.g. $(k \geqslant +0)$

$$P(\overline{\top},\ D_k) = \{H_i J\ ;\ i = 0,\ 1,\ 2,\ \ldots\}$$

and

$$P(\overline{\top},\ D_{k+1}) = \{H_i^{\star} J\ ;\ i = 0,\ 1,\ 2,\ \ldots\}$$

$$P(\overline{\top},\ T_k) = \{H_i J,\ H_i^{\star} J\ ;\ i = 0,\ 1,\ 2,\ \ldots\}$$

$$H_i = \left\{ \begin{vmatrix} h_0 & h_1 & \cdots & h_i \\ h_1 & & & \\ \vdots & & \iddots & \\ h_i & & & h_{2i} \\ h_{i+1} & \cdots & & h_{2i+1} \end{vmatrix} \right\} = H_i^{\star} \tag{33}$$

now introduce y_i and $y_i^{\star}$ as

$$H_i\, y_i = e_{i+1}\, r_i \qquad ; \qquad y_{ii} = 1 \tag{34}$$

$$H_i^{\star}\, y_i^{\star} = e_{i+1}\, r_i^{\star} \qquad ; \qquad y_{ii}^{\star} = 1$$

then the recursion is :

$$y_{i+1} = \begin{Bmatrix} 0 \\ y_i^{\star} \end{Bmatrix} - \begin{Bmatrix} y_i \\ 0 \end{Bmatrix} \frac{r_i^{\star}}{r_i} \tag{35}$$

$$y_{i+1}^{\star} = y_{i+1} - \begin{Bmatrix} y_i^{\star} \\ 0 \end{Bmatrix} \frac{r_{i+1}}{r_i^{\star}}$$

$$r_i = [h_i \ \ldots \ h_{2i}]\, y_i \qquad ; \qquad r_i^{\star} = [h_{i+1} \ \ldots \ h_{2i+1}]\, y_i^{\star}$$

It is not difficult to see that, if we take a comonic normalization for $y^{\star}$, i.e. introduce x_i as

$$H_i\, x_i = e_1\, U_i \qquad ; \qquad x_{0i} = 1 \tag{36}$$

then

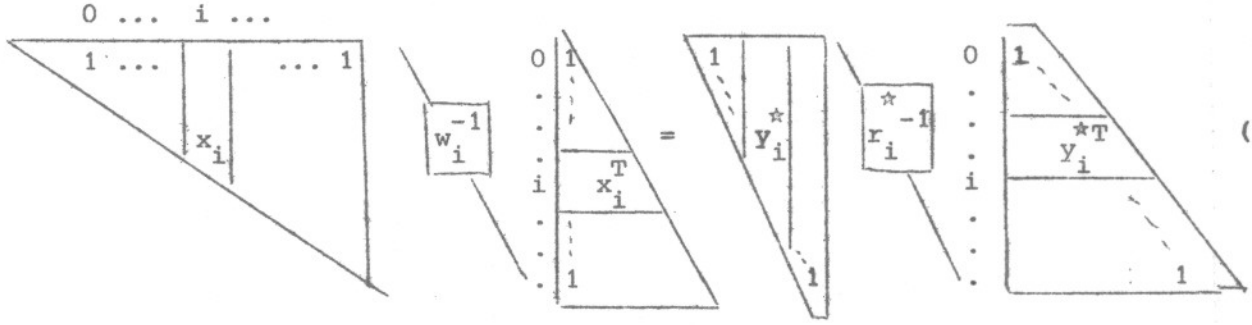
$$H_i^{\star}\, x_i = [0 \ \ldots \ 0 \ v_i]^T$$

and thus

$$x_i \sim y_i^{\star}$$

This introduces the remarkable result that the new columns x_i introduced in the non - nested factorization (31) are actually the columns of the nested factorization (37).

H^{*-1} =

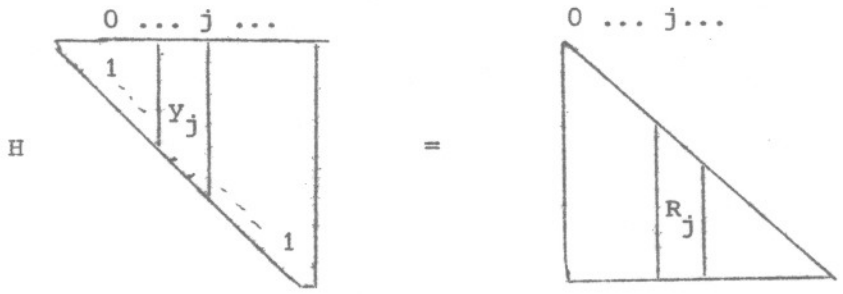


(37)

where $w_i = x_{ii}$ $v_i = r_i^* x_{ii} = r_i^* y_{0i}^{*-2}$

The two recursive schemes (29) and (35) may now be used to derive corresponding algorithms for the factorization of H (and eventually H^*) itself.

Let



(38)

then the recursions for R_{ji} as obtained from (29) may be summarized in the notation

$$R_j(z) = \sum_{i=j}^{\infty} R_{ij}\, z^{i-j}$$

as :

$$R_{-1}(z) = \frac{1}{z} \quad ; \quad R_0(z) = \sum_{i=0}^{\infty} h_i\, z^i$$

$$[r_{-1},\ \lambda_{-1}] = [-1,\ 0]$$

for $j = 0, 1, 2, \ldots$

$$[r_j,\ \lambda_j] = [R_{jj},\ R_{j+1,\,j}] = [R_j(0),\ R_j'(0)] \tag{39}$$

$$R_{j+1}(z) = R_j(z) - \frac{1}{z}\left(R_{j-1}(z)\,\frac{r_j}{r_{j-1}} + R_j(z)\,\left(\frac{\lambda_j}{r_j} - \frac{\lambda_{j-1}}{r_{j-1}}\right)\right)$$

which is a direct transcription of (29).

The two coefficients needed in the recursion appear to be the first two non - zero coefficients in the residual. This is in contrast with (19) in the Toeplitz case where we needed the two different factorizations of T to find the necessary recursion coefficients.
Recursions (39) are the essence of the RTQA - algorithm of [6].

The scheme (35) can be used to derive factorizations of the form (38) for two adjacent diagonals i.e. the matrices H_i and $H_i^{\star}$ say :

$$R_{i,\,j+1} = R^{\star}_{ij} - R_{ij}\,\frac{r^{\star}_j}{r_j} \qquad ; \qquad r_j = R_{jj} \tag{40}$$

$$R^{\star}_{i,\,j} = R_{i+1,\,j} - R^{\star}_{i,\,j-1}\,\frac{r_j}{r^{\star}_{j-1}} \qquad ; \qquad r^{\star}_j = R^{\star}_{jj}$$

A short remark for the case $k \leqslant -0$ is in place here. We could start the recursions with $y_{-1}(z) = z^{k-1}$ for (29) and $y_{-1}(z) = z^{k}$ for (35) and for y_0 : the first $k+1$ coefficients of $\frac{1}{f(z)}$.
This corresponds to the solution of the systems

$$\begin{bmatrix} 0 & 0 & f_0 & f_1 \\ 0 & & & f_2 \\ & & & \\ f_0 & & & \\ & & & \\ f_1 & f_2 & \cdots & f_{k+2} \end{bmatrix} y_0 = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ r_0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & 0 & f_0 \\ 0 & & f_1 \\ & & \\ f_0 & f_1 & \cdots \quad f_{\ell} \end{bmatrix} y_{-1} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ r_{-1} \end{bmatrix} \tag{41}$$

$$y_{0,\,k} = 1 \quad ; \quad y_{-1,\,\ell} = 1 \qquad\qquad \ell = k \text{ for (29) and}$$
$$\ell = k + 1 \text{ for (35)}$$

The solution of which is trivial.
How the factorizations work, will be handled in the non - normal case.
Another possibility to part with the annoying case $k \leqslant -0$ is to obtain $[m/n]_f$ as $[n/m]\,1/f$. This would require the inversion of the f(z) series first.

The algorithms for ascending staircases and diagonals [6, 7, 8] solve systems with matrices taken from $P(\Gamma, E_k)$ say and these are Toeplitz matrices growing in "north - east" direction.
Converting this in a growth direction that is "south - east", we again obtain Hankel matrices. The systems are :

$$H_i \, x_i = e_1 \, r_i \quad ; \quad x_{0i} = 1 \tag{42}$$

with

$$H_i = J \, T_i \quad ; \quad x_i = \tilde{Q}_i$$

As i grows, at a certain point zero's will be introduced in H (because $f_{-1} = f_{-2} = \ldots = 0$) and the algorithm will end at the point where the system

$$\begin{pmatrix} f_{k+1} & \cdots & f_0 & & \\ \vdots & & & f_1 & \\ & & & & 0 \\ f_0 & & & & \\ f_1 & 0 & \cdots & & 0 \end{pmatrix} \quad x_\ell \quad = \quad \begin{pmatrix} r_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

is reached, which is of course the same as the one to start with in (41).
In the case of a Laurent - Padé table, this is not true.

The recursions for x_i are not difficult to find and are similar to the previous Hankel - recursions. E.g. for the diagonal : putting $h_j = f_{k+1-j} \quad j = 0, 1, 2, \ldots$

We have

$$x_{i+1} = \Gamma_i \left\{ \begin{bmatrix} x_{i-1} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \\ x_i \end{bmatrix} \frac{\nu_{i-1}}{\nu_i} - \begin{bmatrix} x_i \\ \\ 0 \end{bmatrix} \Lambda_i \right\} \qquad (43)$$

$$\rho_i = [h_{i+1} \ \cdots \ h_{2i+1}]\, x_i \quad ; \quad \nu_i = [h_{i+2} \ \cdots \ h_{2i+2}]\, x_i$$

$$\Lambda_i = \frac{\rho_i \ \nu_i - \rho_{i-1} \ \nu_{i-1}}{\nu_i^2}$$

$$\Gamma_i = (1 - \Lambda_i)^{-1} \text{ for normalization of } x_{i+1}$$

We now have that the role of residual and numerator are interchanged : i.e. The R - series in (38) are now finite and will represent numerators while the non - nested factorization of H into upper - x lower-triangular will contain residual coefficients in the last column added. ν_i and ρ_i are thus the two coefficients of the highest degree - terms of the numerators i.e.

$$[\nu_i, \rho_i] = [\tilde{P}_{ii}, \tilde{P}_{i-1,i}] \qquad (44)$$

As a conclusion of this section we give some references related to the cited algorithms :
The algorithms that use the explicit evaluation of the Padé denominators (x_i or y_i) factorize the inverse of the Hankel matrix. The recursion (29) can be found in several places : [3, 28, 29, 33,] and because of the known relation between orthogonal polynomials and Padé - denominators [3, 21 e.g.] may even be attributed to Lancsoz and Chebyshev [2, 21, 27] . The other one : (35) is in fact the Watson algorithm [7, 8] . The recursive factorizations of H correspond to continued fraction algorithms. See [6, 27, 28, 29] for (39) and [7, 8, 20] for (40). The coefficients appearing in the latter are also those evaluated by the QD algorithm of Rutishauser, but the approach is different.
For ascending staircases / diagonals, (43) e.g. is a relation that can be found in [7], and the staircase algorithm would correspond to Baker's [8] . The same computations are also done by the Euclid algorithm [9, 26] .

Remarks

1. We do not treat vertical paths like columns or vertical sawteeth because there is no nesting of the matrices involved and thus there will be no factorization algorithm hidden behind the Padé recursions.

2. There is nowhere in the algorithms made explicit use of the fact that $f_i = 0$ for $i < 0$ so that all the algorithms also extend to the Laurent - Padé table [22].

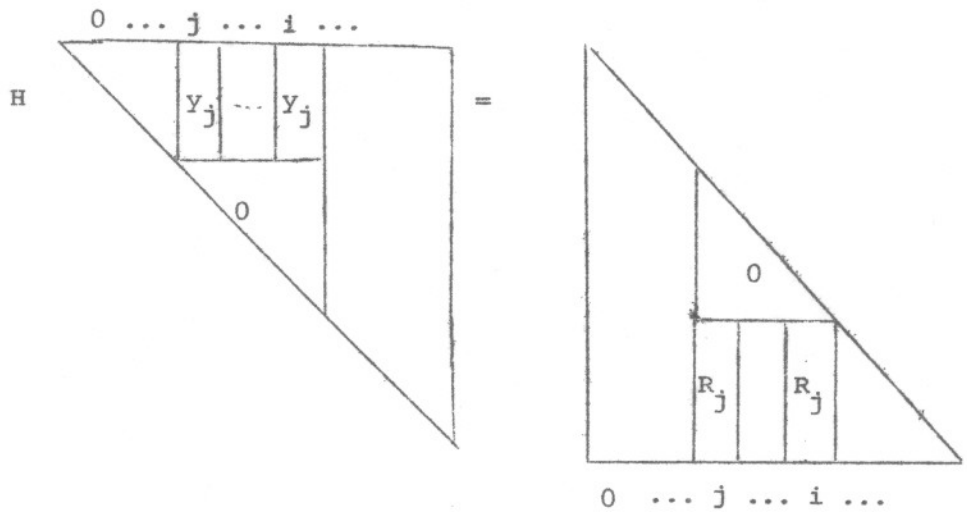
3. There is a well known relationship between orthogonal polynomials and Padé approximation [4, 21, 35] e.g. and so the algorithms can even so be interpreted as orthogonalization procedures. In this context also least squares solutions of rectangular systems with Toeplitz - (Hankel-) like structure see [12, 13] e.g.

4. It must well be remembered that all the recursions mentioned so far will break down whenever the Padé table is not normal i.e. when one or more of the principal minors of the matrices treated are zero. There exist however also algorithms for singular Padé tables. We will see in the next sections what factorizations they represent.

4. Hankel matrix factorization, non normal case

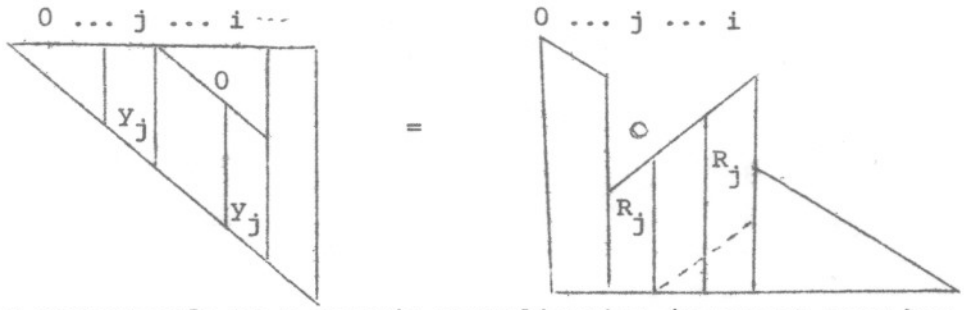
We treat the singular Hankel matrix case first because this has been the most extensively studied in literature.

The factorization algorithms for Hankels will break down if some $r_i = R_i(0) = 0$, we may then have factorizations like



(45)

or

H 

(46)

The second one corresponds to a comonic normalization (constant term has to be zero) and the first to a monic normalization.

(46) is also a triangular factorization of a row - permutated

H : viz. diag $(J_1\ J_2\ \ldots\ )H$ (47)

where J_i is an order reversing J - type matrix, with size equal to the number of elements that lie on the diagonal and in the (singular) block of y_j. There is always some freedom for the factorization part lying in a singular block and (45), (46) are only two of the many possibilities.

Several algorithms are known [5, 6, 11, 30, 25, 27] for descending diagonals and staircases, all will obtain one of the mentioned factorizations.

The algorithm most directly formulated as a factorization algorithm is the one by Rissanen [30]. The main idea is to take

$\begin{pmatrix} 0 \\ y_i \end{pmatrix}$ as a candidate for y_{i+1} if the corresponding R_{i+1} is not in the linear span of R_j, $j \leq i$

then $y_{i+1} = \begin{pmatrix} 0 \\ y_i \end{pmatrix}$

Thus the tap depicted in (46) is generated. If R_{i+1} is in the linear span of R_j ; $j \leq i$ then we use $\begin{pmatrix} y_{j-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, $\begin{pmatrix} y_i \\ 0 \end{pmatrix}$, $\begin{pmatrix} y_{i-1} \\ 0 \\ 0 \end{pmatrix}$, ..., $\begin{pmatrix} y_j \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ to eliminate the undesired coefficients of R_{i+1} above the diagonal. In the normal case i = j and we obtain (29) again.

Rissanen uses the explicit evaluation of y_i and R_i in the recursions. A statement of the algorithm can be found in [30]. The formulations in [5, 11, 25] use the explicit computation of the y_i while only the necessary coefficients of R_i are computed as the products of H - rows with y_i. They may thus be seen as algorithms for the computation of the triangular factors of H^{-1}. [6, 27] use only the residuals R_i during the computations and they thus compute the factor of H itself. [5, 6, 27] also contain algorithms for the staircases. The algorithms are somewhat lengthy to formulate, therefore we will not copy them merely to adapt them to the notation used in this text. We will content ourselves with the above survey. Remark that the systems in (41) may now also be treated by these general algorithms.

The algorithms for ascending diagonals and staircases do also work with Hankel matrices as mentioned in section 3 and they thus can be treated by the some factorization algorithms. The singular Padé table was treated by Cordellier [10] and Mc Eliece - Shearer [26]. Both relie upon computations with the numerators, using the finite-ness of the series representing these numerators. They are thus not directly applicable to Laurent - Padé tables, but a reformulation as singular Hankel algorithms like before will make them more generaly applicable. The factorizations are like (45) and (46) but the R_j will now represent the numerators in reversed order and y_i the denominators. While in the descending algorithms these are residuals and reversed order denominators respectively.

5. Toeplitz matrix factorization, non - normal case

For a Toeplitz matrix, we can reduce the problem to a Hankel matrix problem as Rissanen did in [30]. This solution however does not give as intermediate results the approximants on the same row / sawteeth and preceding the one involved.

The RTOGE algorithms in [6] solves this problem, but it makes use of the fact that $f_{-1} = f_{-2} = \ldots = 0$.

A more general formulation not dependent upon this fact has not been given in the literature as far as we know and we will derive this below. This formulation makes it applicable to Laurent - Padé tables also.

Suppose T_i^{∞} is defined as in (22).

The relation (23) will now be

$$T_i^\infty y_i = [\ldots \underset{\substack{\neq \\ 0}}{s_{0i}} \underbrace{0 \ldots 0}_{\lambda_i} \mid \underbrace{0 \ldots 0}_{i} \underbrace{0 \mid 0 \ldots 0}_{\gamma_i} \underset{\substack{\neq \\ 0}}{r_{0i}} \ldots]^T \tag{48}$$

If y_i is the denominator of the Padé element in the first column of its (singular) block, then that block has size $\lambda_i + \gamma_i + 1$. The last element in that block on row k is then $y_n (n = i + \lambda_i + \gamma_i)$ satisfying :

$$y_n = [\underbrace{0 \ldots 0}_{\lambda_i} \underbrace{y_i^T}_{i+1} \underbrace{0 \ldots 0}_{\gamma_i}]^T \text{ and } T_n^\infty y_n = [\ldots \underset{\substack{\neq \\ 0}}{s_{0i}} \mid \underbrace{0 \ldots 0}_{n} \underset{\substack{\neq \\ 0}}{r_{0i}} \mid \ldots]^T \tag{49}$$

And more general for $p = 0, 1, \ldots \ell$ $(\ell = \lambda_i + \gamma_i + 1 = n - i + 1)$

$$T_{n+1}^\infty [\underbrace{0 \ldots 0}_{p} \underbrace{y_i^T}_{i+1} \underbrace{0 \ldots 0}_{\ell-p}]^T =$$

$$= [\ldots \mid s_{p-\lambda_i-1,\, i} \ldots \underset{\substack{\neq \\ 0}}{s_{0i}} \underbrace{0 \ldots 0}_{n+\lambda_i-p+1} \underbrace{0 \mid 0 \ldots 0}_{p-\lambda_i-1} \underset{\substack{\neq \\ 0}}{r_{0i}} \ldots]^T$$

$$\text{for } p = \lambda_i + 1, \ldots, \ell \tag{50}$$

$$= [\ldots \mid \underbrace{0 \ldots 0}_{n-\lambda_i+p} \underset{\substack{\neq \\ 0}}{r_{0i}} \ldots r_{\lambda_i+1-p,\, i} \mid \ldots]^T$$

$$\text{for } p = 0, 1, \ldots, \lambda_i$$

while

$$T_{n+1}^\infty [\underbrace{0 \ldots 0}_{\gamma_i+1} \underbrace{y_{i-1}^T}_{i} \underbrace{0 \ldots 0}_{\lambda_i+1}]^T =$$

$$[\ldots \mid s_{\gamma_i,\, i-1} \ldots \underset{\substack{\neq \\ 0}}{s_{0,\, i-1}} \underbrace{0 \ldots 0}_{i-1} \underset{\substack{\neq \\ 0}}{r_{0,\, i-1}} \ldots r_{\lambda_i+1,\, i-1} \mid \ldots] \tag{51}$$

The s - coefficients are numerator coefficients while the r's are residuals coefficients.

It will then be clear from (50) and (51) that the $s_{0,\,i-1} \cdots s_{\gamma_i,\,i-1}$ and $r_{0,\,i-1}, \cdots r_{\lambda_i,\,i-1}$ can be eliminated from (51) by suitable multiples of (50).

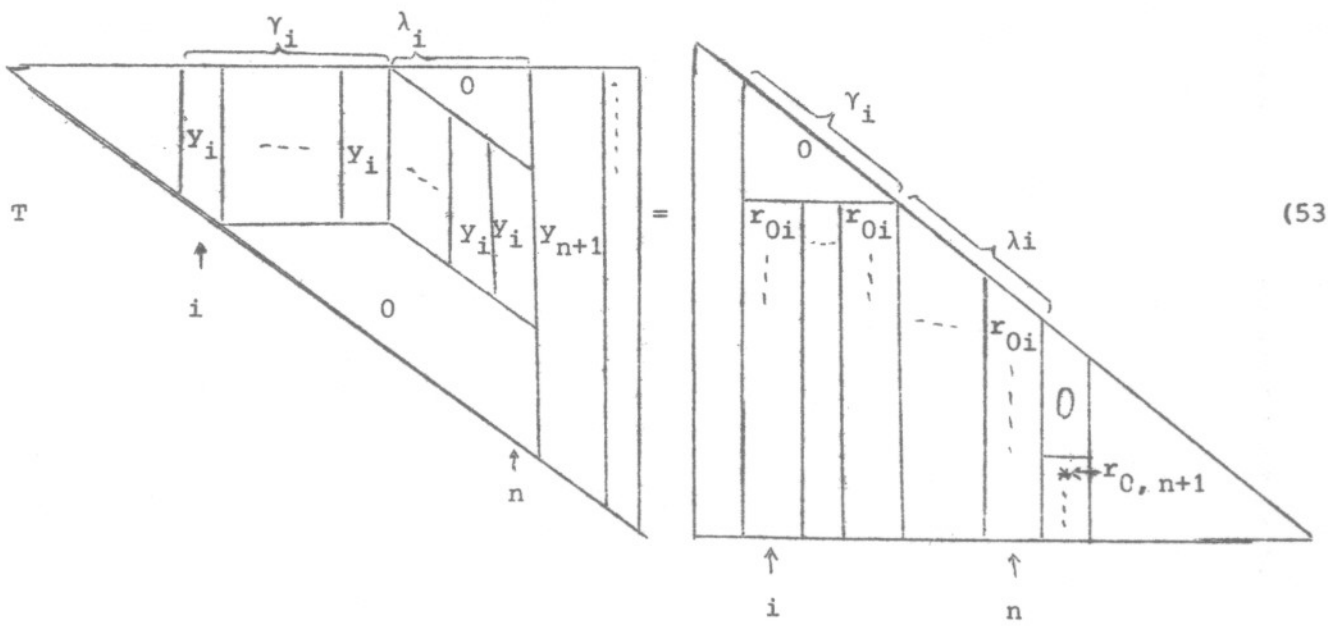
Thus there exist $\mu_i : i = 0, 1, \ldots, \ell$ such that

$$y_{n+1} \sim y_{i-1} - \sum_{p=0}^{\ell} \mu_p \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y_i \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \} \, p \\ \\ \} \, i+1 \\ \\ \} \, \ell - 1 \end{matrix} \qquad (52)$$

Let us call this elimination procedure UPDAT 1.

It is a variant of the RTOGE algorithm of [6]. To make the link, note that the elimination of the $s_{\cdot,\,i-1}$ corresponds to the division of the numerators and the elimination of the $r_{\cdot,\,i-1}$ to the division of the residuals.

The factorization hidden behind this is the following :



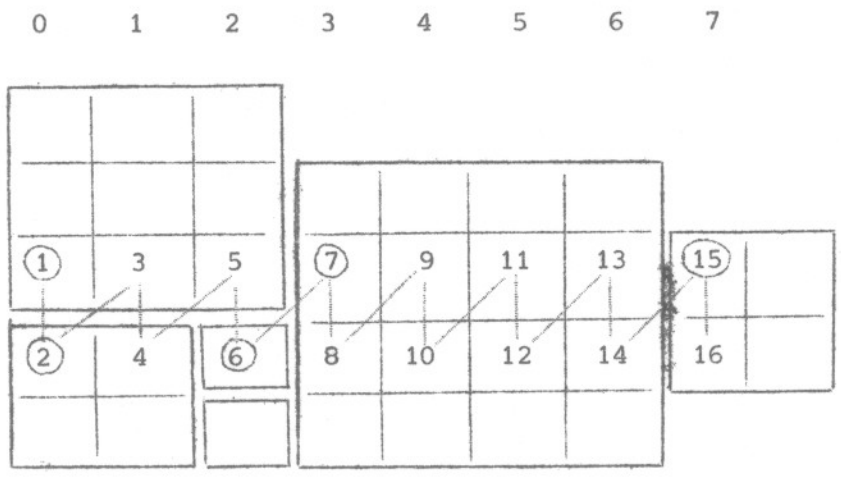
(53)

We can compute the y_i explicitely and the r-and s - coefficients that we need by multiplication of a T - row with some y. This would correspond to an algorithm computing the triangular factor of T^{-1}. This is the most direct generalization of (10). If we would only use the r - and s - coefficients, then we work with the triangular factors of T itself. i.e. we work with residuals and numerators. See in this context the discussion p 12, 13. We could also use a mix of both factorizations. e.g. compute the y_i and R_i explicitely and the s - coefficients by scalar products (rows of T with y).

The sawteeth algorithms may also be generalized to the non - normal case. The RTOV3 algorithm doing this has been chiefly outlined in [6], but again the formulation made use of the fact that $f_{-1} = f_{-2} = \ldots = 0$.

In a sawtooth algorithm we compute some element (denominator of Padé approximant) as a combination of the last element preceding it and the last one preceding this which was not a shifted version of the latter i.e. does not differ merely by an insertion of zeros in front or at the end.

To make things clear let us look at an example :



only the encircled elements are relevant. For the computation of 7 e.g. we use the elements 6 and 5, the latter being a shifted version of 1. For 15 we use 14 and 6, where 14 is again a shifted version of 7.
Thus a new element will be computed from two previous elements P and PP say : P is the last one computed thus far and PP is the first one met along the sawtooth walking along this path from P in the reverse direction and that is different from P in a "relevant way". PP will always be in the last column of its (singular) block.

In the normal case, the Zohar - Trench algorithm uses a combination of the two sawtooth algorithms. I.e. it uses S_k^+ to compute the recursion for an element of row k and S_k^- for an element of row k + 1. Since in the normal case all elements contain relevant information it computes the elements from the (i + 1) st column as a combination of the elements from the i - th column. In the non - normal case this may result in a considerable loss of efficiency in the computational effort and in storage requirement. In the example we would compute 15 from 14 and 6 while 16 is computed separately from 5 and 13, while actually 15 and 16 are the some elements. We will therefore not go further into this and formulate only a generalized S_k^+ algorithm ($k \geqslant + 0$).

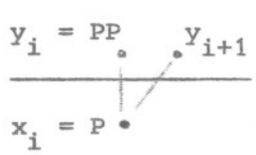
As before introduce the notation :

The denominator of the i - th element in the k - th row of the Padé table will be indicated by y_i, while the denominator of the same column in the (k + 1) st - row is named x_i, then y_i satisfies (48) while x_i satisfies

$$T_i^{\infty} x_i = [\ldots \underset{\substack{\neq \\ 0}}{u_{0,i}} \underbrace{0 \ldots 0 \mid 0}_{\mu_i} \underbrace{0 \ldots 0}_{i} \mid \underbrace{0 \ldots 0}_{\psi_i} \underset{\substack{\neq \\ 0}}{v_{0,i}} \ldots]^T \tag{54}$$

At the upper or lower border of a singular block we have the following situations;

$PP = y_i$; $P = x_i$ and $y_{i+1} = ?$



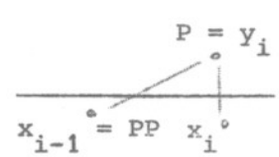
then we have the procedure refered to as A :

$$A : \underline{\text{if}}\ \lambda_i > 0 \quad \underline{\text{then}} \quad y_{i+1} \sim \begin{pmatrix} 0 \\ y_i \end{pmatrix} \tag{55}$$

$$\underline{\text{else}} \quad Y_{i+1} \sim \begin{pmatrix} 0 \\ y_i \end{pmatrix} - \begin{pmatrix} x_i \\ 0 \end{pmatrix} \frac{s_{0,i}}{u_{0,i}}$$

and at the lower border :

$PP = x_{i-1}$; $P = y_i$; $x_i = ?$



$$\text{B : } \underline{\text{if}} \quad \psi_i > 0 \quad \underline{\text{then}} \quad x_i \sim \begin{bmatrix} x_{i-1} \\ \\ 0 \end{bmatrix} \tag{56}$$

$$\underline{\text{else}} \quad x_i \sim \begin{bmatrix} x_{i-1} \\ \\ 0 \end{bmatrix} - y_i \quad \frac{v_{0,\,i-1}}{r_{0i}}$$

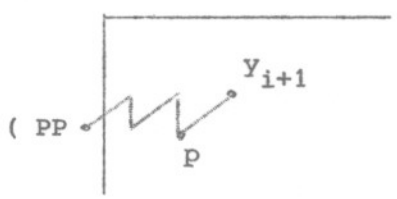
We do not bother for the normalization and therefore we write only $\sim$.

The algorithm for S_k^+ may now be formulated as :

1) $\boxed{P = x_i}$ find y_{i+1}

if $\mu_i > 0$ (this is $[t_0 \ \ldots \ t_{-i}]\, x_i = 0$)

then $y_{i+1} \sim [x_i^T \ 0]^T$; $P \leftarrow y_{i+1}$



else ($\mu_i = 0$)

begin

if $\psi_i > 0$ (i.e. $[t_{i+1} \ \ldots \ t_1]\, x_i = 0$

then (x_i is in the first row but not in the last column of its singular block

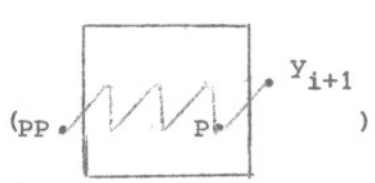
thus $PP = y_i$)

execute A

else (x_i is in the last column of its singular block)

if $PP = y_i$ then execute A

else



execute UPDAT 2 ;

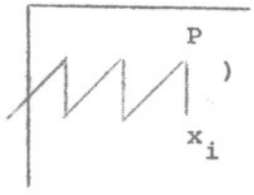
$PP \leftarrow P \leftarrow y_{i+1}$;

end ;

2) $\boxed{P = y_i}$ find x_i

if $\gamma_i > 0$ (i.e. if $[t_{i+1} \ldots t_1]y_i = 0$

then $x_i = y_i$; $P \leftarrow x_i$



else ($\gamma_i = 0$)

if $\lambda_i > 0$ (i.e. if $[t_0 \ldots t_{-i}]y_i = 0$)

then (y_i is in the last row but not in the last column of its singular block : thus PP = x_{i-1})

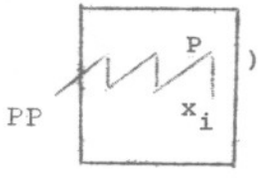
execute B;

PP $\leftarrow P \leftarrow x_i$;

else (y_i is in the last column of its singular block)

if PP = x_{i-1} then execute B

else (



)

$$\begin{bmatrix} x_i \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ y_i \end{bmatrix} \quad ; \quad P \leftarrow x_i \quad ;$$

The UPDAT 2 procedure may be briefly outlined as follows : If the size of the block to which PP belongs is greater than 1, then PP is actualy equal to some y_i, $i < n$ and thus UPDAT 1 may be used, if its size is 1, then we have the following situation : let PP = x_{i-1} say

then

$$T_{n+1}^{\infty} [\underbrace{0 \ldots 0}_{p} \underbrace{y_i^T}_{i+1} \underbrace{0 \ldots 0}_{\ell-p}]^T \qquad (57)$$

for $p = 0, 1, \ldots, \ell$, and $\ell = \lambda_i + \gamma_i + 1$, where ℓ is the size of the singular block of y_i and $n = \lambda_i + \gamma_i + i$

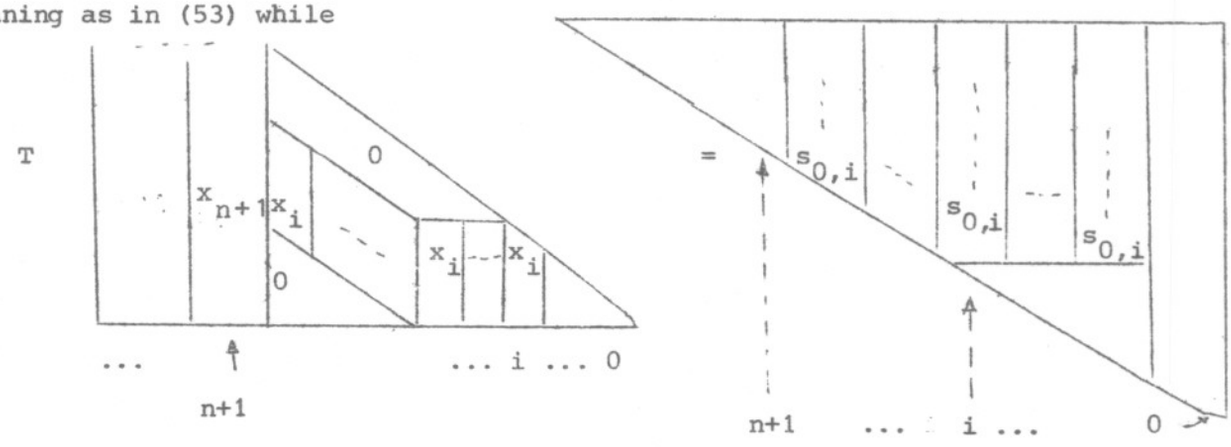
$n = \lambda_i + \gamma_i + i$

Further we have :

$$T^{\infty}_{n+1} [\underbrace{0 \ldots 0}_{\gamma_i} \underbrace{x^T_{i-1}}_{i} \underbrace{0 \ldots 0}_{\lambda_i+2}]^T = [\ldots | u_{\gamma_i, i-1} \cdots \underset{\substack{\neq \\ 0}}{u_{0,i-1}} \underbrace{0 \ldots 0}_{i-1} \underset{\substack{\neq \\ 0}}{v_{0,i-1}} \cdots v_{\lambda_i+1, i-1} |] \tag{58}$$

and the numbers between verticals can be eliminated with multiples of (57) just like in the UPDAT 1 algorithm.

The interpretation of the quantities involved in terms of the factorization of the Toeplitz matrix is not difficult : The y_i and r - quantities have the same meaning as in (53) while



We can again have different versions : computing the r_{ji} and s_{ji}, thus making the x_i and y_i superfluous would again correspond to working with residuals for the k - th row and numerators for the (k + 1) st row. Another possibility is to work with denominators y_i and x_i of k - th and (k + 1)- st row only, i.e. with the factors of T^{-1} and T^{*-1}, or one could take a mix of both : etc ...

6. Relation with other Fields

We have given to Padé table recursions the interpretation of fast algorithms for the triangularization of Hankel and Toeplitz matrices. Fast refers to the fact that they only need $O(n^2)$ arithmetic operations instead of the usual $O(n^3)$. The formulations given are also applicable for the Laurent - Padé table and a special case of this (symmetric Toeplitz matrix, first row of table) is important in digital filtering, where orthogonal polynomials [4, 22, 19] represent optimal linear predictors. The so called cascade realization or ladder forms correspond to a continued fraction approach. The recursion coefficients are called reflection coefficients there and the algorithms actually factorize Toeplitz matrices and / or their inverses. Dewilde etal.[14] give also the relation to network synthesis.
The usual Padé approximation is known to be identical with the partial minimal realisation problem of systems theory [11, 16].
The algorithm may also be generalized to block Hankel and Toeplitz matrices. Examples may be found in [1, 2, 11, 16, 29, 36, 37].
There exist also fast algorithms for matrices that do not differ very much from Hankel or Toeplitz matrices. See [18].

If there is still more structure in the matrices then the algorithms may even be made faster e.g. a band structure is exploited in [23, 15, 37], see also [17].

Acknowledgment.

We are endebded to Prof. P. Dewilde who introduced the Levinson - algorithm to us as a fast Choleski - factorization.

References

1. H. Akaike
Block Toeplitz matrix inversion.
SIAM J. Appl. Math. Vol. 24 (2) p. 234 - 241, 1973.

2. E. Bareiss
Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices.
Numer. Math. Vol. 13, p. 404 - 424, 1969.

3. C. Brezinski
Computation of Padé approximations and continued fractions.
Journ. of Comp. and Appl. Math. Vol. 2 (2), p. 113 - 123, 1976.

4. C. Brezinski
Padé approximants and Orthogonal Polynomials in [31], p. 3 - 14.

5. A. Bultheel
Recursive algorithms for non normal Padé tables.
K.U.Leuven, Belgium, Report TW 40, July 1978.

6. A. Bultheel
Division algorithms for continued fractions and the Padé table.
K.U.Leuven, Belgium, Report TW 41, August 1978.

7. D. Bussonnais
"Tous" les algorithmes de calcul par recurrence des approximants de Padé d'une série. Construction de fractions continues correspondantes.
Séminaire d'Analyse Numerique, n° 293, Grenoble 1978.

8. G. Claessens
A new look at the Padé table and the different methods for computing its elements.
Journ. of Comp. and Appl. Math. Vol. 1 (3), p. 141 - 152, 1975.

9. G. Claessens
Some aspects of the rational Hermite interpolation table and its applications.
Ph D Thesis, Universitaire Instelling Antwerpen, Wilrijk, 1976.

10. B. Cordellier
Deux algorithmes de calcul recursif des elements d'une table de Padé non normale.
Presented at the conference on Padé approximation, Lille, 1978.

11. L. S. De Jong
Numerical aspects of realization algortihms in linear systems theory.
Thesis, T. H. Eindhoven, The Netherlands, 1975.

12. R. De Meersman
A method for least squares solution of systems with a cyclic rectangular coefficient matrix.
Journ. Comp. and Appl. Math. Vol. 1 (1), p. 51 - 54, 1975.

13. R. De Meersman
A derivation of the Levinson algorithm for solving linear systems with symmetric positive definite Toeplitz matrix.
Journ. Comp. and Appl. Math. Vol. 2 (1), p. 63 - 65, 1976.

14. P. Dewilde - A. Vieira - Th. Kailath
On a generalized Szego - Levinson realization algorithm for optimal linear predictors based on a network systhesis approach.
Subm. IEEE Trans. CAS, 1978.

15. B. W. Dickinson
Efficient solution of linear equations with banded Toeplitz matrices.
Subm. IEEE Trans. ASSP, 1978.

16. B. W. Dickinson - M. Morf - T. Kailath
A minimal realization algorithm for matrix sequences.
IEEE Trans. on Automatic Control. AC - 10 (1), p. 31 - 38, 1974.

17. D. C. Farden
The solution of a special set of hermitian Toeplitz linear equation.
Trans. on Math. Software (TOMS) Vol. 3 (2), p. 159 - 163, 1977.

18. B. Friedlander - M. Morf - T. Kailath - L. Ljung
New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices.
Sub. SIAM J. Appl. Math.

19. L. Y. Geronimus
Orthogonal polynomials.
Consultants bureau, New York, 1961.

20. W. B. Gragg
The padé table and its relation to certain algorithms of numerical analysis.
SIAM Review, Vol. 14, p. 1 - 62, 1972.

21. W. B. Gragg
Matrix interpretations and applications of the continued fraction algorithm. Rocky Mountain J. of Math. Vol. 4 (2) p. 213 - 225, 1974.

22. W. B. Gragg
Laurent, Fourier and Chebyshev.
Padé tables in [31], p. 61 - 72, 1977.

23. A. K. Jain
Fast inversion of banded Toeplitz matrices by circular decomposition.
IEEE Trans. an Acoustics, speech, signal process., ASSP - 26 (2), p. 121 - 126, 1978.

24. N. Levinson
The Wiener RMS (root mean square) error criterion in filter design and prediction.
J. Math. and Phys. Vol. 25, p. 261 - 278, 1947.

25. J. L. Massey
Shift - Register synthesis and BCH decoding.
IEEE Transactions on Information Theory IT - 15 (1), p. 122 - 127, 1967.

26. R. J. Mc Eliece - J. B. Shearer
A property of Euclid's algorithm and an application to Padé approximation.
SIAM J. Appl. Math. Vol. 34 (4), p. 611 - 616, 1978.

27. J. A. Murphy - M. R. O'Donohoe
A class of algorithms for obtaining rational approximants to functions which are definied by power series.
Journ. of Appl. Math. and Physics (ZAMP) Vol. 28, p. 1121 - 1131, 1977.

28. J. L. Phillips
The triangular decomposition of Hankel matrices.
Math. of Comp. Vol. 25 (115), p. 599 - 602, 1971.

29. J. Rissanen
Algorithms for triangular decomposition of block Hankel and Toeplitz matrices with application to factoring positive matrix polynomials.
Math. of Comp. Vol. 27 (121) p. 147 - 154, 1973.

30. J. Rissanen
Solution of linear equations with Hankel and Toeplitz matrices.
Numer. Math. Vol. 22, p. 361 - 366, 1974.

31. E. B. Saff - R. S. Varga (eds.)
Padé and rational approximation, Theory and applications.
Academic Press, New York, 1977.

32. W. F. Trench
An algorithm for the inversion of finite Toeplitz matrices.
SIAM J. Appl. Math. Vol. 12, p. 515 - 512, 1964.

33. W. F. Trench
An algorithm for the inversion of finite Hankel matrices.
SIAM J. Appl. Math. Vol. 13, p. 1102 - 1107, 1965.

34. W. F. Trench
Inversion of Toeplitz band matrices.
Math. Comp. Vol. 28 (128), p. 1089 - 1095, 1974.

35. H. Van Rossum
Systems of orthogonal and quasi orthogonal polynomials connected with the Padé table I - II - III.
Indigationes Mathematicae XVII, p. 517 - 534, 675 - 682, 1955.

36. H. Van Rossum
Padé approximants and Indefinite inner product spaces in [31], p. 111 - 120.

37. R. A. Wiggins - E. A. Robinson
Recursive solution to the multichannel filtering problem.
J. Geophys. Res. Vol. 70, p. 1885 - 1891, 1965.

38. S. Zohar
Toeplitz matrix inversion the algorithm of W. F. Trench.
J. ACM Vol. 16, p. 592 - 601, 1969.

39. S. Zohar
The solution of a Toeplitz set of linear equations.
J. ACM Vol. 21, p. 272 - 276, 1974.