Petr Hušek; Michael Šebek; Jan Štecha

Numerical operations among rational matrices: standard techniques and interpolation

Kybernetika, Vol. 35 (1999), No. 5, [587]--598

Persistent URL: http://dml.cz/dmlcz/135309

KYBERNETIKA — VOLUME 35 (1999), NUMBER 5, PAGES 587 - 598

NUMERICAL OPERATIONS AMONG RATIONAL MATRICES: STANDARD TECHNIQUES AND INTERPOLATION[1,2]

PETR HUŠEK, MICHAEL ŠEBEK AND JAN ŠTECHA

Numerical operations on and among rational matrices are traditionally handled by direct manipulation with their scalar entries. A new numerically attractive alternative is proposed here that is based on rational matrix interpolation. The procedure begins with evaluation of rational matrices in several complex points. Then all the required operations are performed consecutively on constant matrices corresponding to each particular point. Finally, the resulting rational matrix is recovered from the particular constant solutions via interpolation. It may be computed either in polynomial matrix fraction form or as matrix of rational functions. The operations considered include addition, multiplication and computation of polynomial matrix fraction form. The standard and interpolation methods are compared by experiments.

1. INTRODUCTION

Rational matrices (such as transfer matrices of linear multivariable systems) are often expressed in the form of *polynomial matrix fractions*, i. e.

$$H(s) = W_L^{-1}(s)\, V_L(s) = V_R(s)\, W_R^{-1}(s)$$

where $H(s)$ is a rational matrix and $W_L(s)$, $W_R(s)$, $V_L(s)$, $V_R(s)$ are polynomial matrices.

The polynomial matrix fractions are natural generalization of scalar polynomial fractions describing single-input single-output (SISO) systems. They make it possible to use all algebraic methods developed originally for SISO systems. However, they also have several disadvantages:

— Individual transfer functions between particular inputs and outputs are not directly visible.

[1]This work was supported by the Ministry of Education of the Czech Republic under Project VS97/034.

[2]A version of this paper was presented at the 5th Mediterranean Conference on Control and Systems held in Paphos (Cyprus) on June 21–23, 1997.

— Even very basic operations, such as addition or multiplication of rational matrices, appear to be rather complicated when expressed via polynomial matrix fractions.

— The degrees resulting in the polynomial matrix fraction are usually much higher than degrees encountered in the original rational matrix. For this reason numerical problems may often arise.

To illustrate the last item, consider a 3×3 transfer matrix having all its fractions coprime with mutually coprime denominators of degree 2. Such a transfer matrix belongs to a system of order $3 \times 3 \times 2 = 18$. The resulting polynomial matrix fraction consists generically of polynomial matrices with degrees 6.

To avoid this degree blow up during numerical computation with rational matrices, it may sometimes be convenient to deal with the original description using the *matrix of rational fractions*, when the rational matrix

$$H(s) = \left[\frac{n_{ij}(s)}{d_{ij}(s)}\right]$$

is stored via other two polynomial matrices the matrix of denominators $D(s) = [d_{ij}(s)]$ and the matrix of numerators $N(s) = [n_{ij}(s)]$. When employing interpolation technique *each* of the two representations above can be achieved.

2. INTERPOLATION

Let us briefly summarize the idea of polynomial and rational matrix interpolation pioneered by Ansaklis and Gao in [1]. Let us recall, that the interpolation of scalars runs as follows: we chose the sufficient number of interpolation points s_j, then we evaluate the interpolated object in these points and finally recover it from both series of values. To generalize this procedure for matrices, we moreover postmultiply the evaluated matrices by some vectors a_j. This simplifies the interpolation of polynomial and rational matrices.

Let us denote $S(s) :=$ blockdiag $[1, s, \ldots, s^{d_i}]^T$ where d_i, $i = 1, \ldots, m$, are non-negative integers and let $a_j \neq 0$ and b_j be $(m \times 1)$ and $(p \times 1)$ complex vectors, respectively, and let s_j be complex scalars.

Theorem 1. (Polynomial matrix interpolation, see [1].) Given interpolation (points) triplets (s_j, a_j, b_j), $j = 1, \ldots, l$, and non-negative integers d_i with $l = \sum d_i + m$ such that the $(\sum d_i + m) \times l$ matrix

$$S_l := [S(s_1)\, a_1, \ldots, S(s_l)\, a_l]$$

has full rank, there exists a unique $(p \times m)$ polynomial matrix $Q(s)$, with ith column degree equal to d_i, $i = 1, \ldots, m$ for which

$$Q(s_j)\, a_j = b_j, \; j = 1, \ldots, l$$

$Q(s)$ can be written as

$$Q(s) = QS(s)$$

where Q $(p \times (\sum d_i + m))$ contains the coefficients of the polynomial entries. Q must satisfy

$$QS_l = U_l$$

where $U_l := [b_1, \ldots, b_l]$. Since S_l is non-singular, Q and therefore $Q(s)$ are uniquely determined.

For $p = m = 1$, S_l is called *Vandermonde matrix*. In the multidimensional case, we shall call it *block Vandermonde matrix*.

Quite naturally, rational matrix interpolation can be handled as a special case of polynomial matrix interpolation.

Theorem 2. (Rational matrix interpolation, see [1].) Assume that interpolation triplets (s_j, a_j, b_j), $j = 1, \ldots, l$, where s_j are complex scalars and $a_j \neq 0, b_j$ complex $(m \times 1), (p \times 1)$ vectors respectively, and non-negative integers d_i, $i = 1, \ldots, p+m$, with $l = \sum d_i + m$ are given such that the $(\sum d_i + (p+m)) \times l$ matrix

$$S_l := [S(s_1)c_1, \ldots, S(s_l)c_l]$$

where $c_j = [a_j^T, b_j^T]^T$, has full column rank. There exists a unique $(p \times m)$ rational matrix $H(s)$ of the form $H(s) = W_L^{-1}(s)\, V_L(s)$ where $W_L(s), V_L(s)$ are $(p \times p)$ and $(p \times m)$ polynomial matrices respectively and the column degrees of the polynomial matrix $[V_L(s), -W_L(s)]$ are d_i $i = 1, \ldots, p+m$, with the leading coefficient matrix of $D(s)$ being I_p (non-singular), which satisfies

$$H(s_j)\, a_j = b_j \quad j = 1, \ldots, l$$

The solution can be determined by solving the equation

$$[V_L, -W_L][S_l, P] = [0, R]$$

where $[V_L(s), -W_L(s)] = [V_L, -W_L]S(s)$ with $S(s) :=$ blockdiag $[1, s, \ldots, s^{d_i}]^T$ $i = 1, \ldots, p+m$. Equations $[V_L, -W_L]P = R$ expresses the additional constraints on the coefficients.

3. COMPUTING POLYNOMIAL MATRIX FRACTIONS

The first task to be discussed is the calculation of left[1] polynomial matrix fraction

$$W_L^{-1}(s)\, V_L(s) = H(s)$$

to a given rational matrix $H(s)$.

Traditional procedure [3] consists in putting in each row all the fractions to their least common denominator $d_i = \text{lcm}(d_{ij})$ that, in fact, generically equals to their

[1] The dual case of right fraction can be handled similarly.

product. Then the matrix $W_L(s) = \text{diag}\{d_i\}$ together with $V_L(s)$ computed accordingly form a left polynomial matrix fraction description of $H(s)$. Moreover, the fraction is generically coprime.

Alternatively, the interpolation can be employed as in Theorem 2. From the computational point of view, the critical step in this algorithm appears to be the inversion of the block Vandermonde matrix (S_l in Theorem 2). This matrix is badly conditioned and its dimension grows very quickly. Its condition number depends heavily on the choice of the interpolation points.

As experiments reveal, the choice of real numbers with magnitudes depending on the coefficients of the $H(s)$ that are balanced around zero with even distances improves the condition number considerably. To be more specific, a rational matrix interpolated at unbalanced points has typically the same condition number as a rational matrix with twice higher degrees interpolated at balanced points. Another improvement can be achieved by placement of interpolation points on the unit circle at the complex plain.

This procedure for computing polynomial matrix fraction works well for a rational matrix of dimension (2×2) with element numerators and denominators of degree 4 for which the resulting polynomial matrix denominator has degree 8. The corresponding block Vandermonde matrix is (34×34) and exhibits condition number about 10^{10}. The computation takes[2] 3 seconds and its relative accuracy of the result is about 10^{-8}.

When Theorem 2 is applied to interpolate scalar rational functions, then degrees up to 20 can be efficiently processed.

In the generic case of all fractions in $H(s)$ coprime and all their denominators of the same degree and mutually coprime, the resulting fraction is coprime. If this is not the case, none of the two procedures mentioned above guarantees directly that resulting polynomial matrix fraction is coprime. In the standard procedure, coprimeness can only be accomplished by additional operations of computing and then extracting the greatest common left divisor of polynomial matrices $V_L(s)$ and $W_L(s)$. This step does not exhibit good numerical properties[3]. When using interpolation, however, coprimeness depends on our ability to estimate the lowest existing degree of the denominator polynomial matrix $W_L(s)$, for which of course, the fraction is coprime. The following procedure is recommended to obtain the coprime polynomial matrix fraction:

1. Determine the highest possible degree of the resulting matrix fraction and interpolate it;

2. Then decrease the estimated degree and compute the new matrix fraction.

3. Check (e. g. by evaluating the rational matrix and the resulting matrix fraction), if the result is correct. If it is, continue by step 2; if it isn't, take the last correct matrix fraction.

[2] All computing mentioned throughout the paper was made on a relatively slow PC with Pentium 65 MHz 16 MB RAM and MATLAB 4.1.

[3] The well-known of "almost common factors".

Numerical experience with the traditional method based on the least common multiples of denominators in each row are as follows: If the result need not be coprime matrix fraction, this procedure has no numerical limitation but the resulting degrees become out of control. When a coprime matrix fraction is desired, some limitations arise by computing the greatest common divisors. By its nature, the procedure is relatively slow. For the example considered above, this simply method transfers that rational matrix in about 6 seconds with higher relative accuracy (about 10^{-13}).

The comparison of the two algorithms for computing can be summarized as follows: For relatively small rational matrices (up to (2×2) with elements of degree 4) the interpolation method is quicker and sufficiently accurate. For larger matrices and/or higher degrees, interpolation brings no particular advantage when compared to the traditional method.

Let us now illustrate the use interpolation on a simple MATLAB session[4].

Example 1.

```
% The 2 x 2 rational matrix H with degrees 2 is
% expressed by the matrix of its numerators

N =

    11     0    -7     0     2   -15     2
     6     3    16    17     8    -8     0
     0     0     0     0     0     0   NaN

% and the matrix of its denominators

D =

    12     5    -2   -13    -1    -8     2
    -7    -4   -14     9    -8     8     0
     0     0     0     0     0     0   NaN

% To get the left polynomial matrix description
```

[4] Here the Polynomial Toolbox [2] is employed. It uses the following polynomial matrix format: A polynomial matrix $N(s) = N_0 + N_1 s + \cdots + N_n s^n$ is stored by the block row of the matrix coefficient and an extra row of and column zeros having the degree n in the upper right corner and NaN in a single MATLAB element N given by

$$N = \left[\begin{array}{|c|c|c|c|c|} \hline N_0 & N_1 & \cdots & N_n & \begin{matrix} n \\ 0 \\ \vdots \\ \vdots \\ 0 \end{matrix} \\ \hline 0\cdots 0 & 0\cdots 0 & 0\cdots 0 & 0\cdots 0 & NaN \\ \hline \end{array}\right]$$

```
% WL^{-1}*VL, we run the macro 'rinter'

>> [WL,VL]=rinter(N,D)

WL =

  Columns 1 through 4

    7.5000    0.0000  -20.7500    0.0000
    0.0000   -0.4375    0.0000    0.1094
         0         0         0         0

  Columns 5 through 8

   -9.3750    0.0000    3.6250    0.0000
    0.0000    2.3437    0.0000    2.8750
         0         0         0         0

  Columns 9 through 11

    1.0000         0    4.0000
         0    1.0000         0
         0         0       NaN


VL =

  Columns 1 through 4

    6.8750    0.0000  -22.2500    0.0000
    0.3750    0.3281    0.1562    2.5156
         0         0         0         0

  Columns 5 through 8

    1.6250  -22.5000    3.7500    3.7500
   -2.5000    3.2187   -3.1250    0.3750
         0         0         0         0

  Columns 9 through 11

   -2.0000    1.8750    4.0000
   -1.0000   -1.0000         0
         0         0       NaN
```

4. SUM OF RATIONAL MATRICES

For sum of two (or more) rational matrices, three algorithms are considered: standard element-wise procedure and two modifications of the interpolation (the whole matrix interpolation or elementwise one).

In the first method, the sum

$$F(s) = G(s) + H(s)$$

is achieved by adding the particular scalar elements

$$\frac{n_{G,ij}}{d_{G,ij}} + \frac{n_{H,ij}}{d_{H,ij}} = \frac{n_{G,ij}d_{H,ij} + n_{H,ij}d_{G,ij}}{d_{G,ij}d_{H,ij}}.$$

The second method, the whole interpolation, runs as follows: At first,

$$n_p = m + \sum_{i=1}^{m+p} d_i$$

of interpolation points is chosen where p and m is the number of rows and columns in $F(s)$, respectively, while d_i is the estimated degree of ith column in the composite polynomial matrix $[V_L\ W_L]$ describing $F(s)$. Then, for $j = 1, \ldots, n_p$, the points are substituted and the resulting constant matrices are added

$$F(s_j) = G(s_j) + H(s_j), \quad j = 1, \ldots, n_p.$$

Finally, the desired sum $F(s)$ is recovered by interpolation as in Theorem 2. The rational matrix resulting from the interpolation can, in fact, be obtained in two different forms: either in the form of polynomial matrix fraction (Theorem 2) or as a standard rational matrix. However, the former can only be used for very small matrices with low degrees while the latter handles well quite large degrees and, by nature of the matrix addition, is completely independent of the matrix dimensions.

The reason can be easily explained. The maximum size of the block Vandermonde matrix resulting from addition[5] one can 'invert'[6] is about 20×20. This size corresponds to a 2×2 rational matrix with degrees 2, that is, the sum of two 2×2 rational matrices with degrees 1! Although the conditioning can slightly be improved by using Tchebychev polynomial bases, this does not qualify the methods for larger matrices.

This is why we prefer the third method: Interpolation element-by-element consists of a series of scalar rational interpolations where the corresponding elements of $G(s)$ and $H(s)$ are separately evaluated, added

$$f_{ij}(s_k) = g_{ij}(s_k) + h_{ij}(s_k), \quad k = 1, \ldots, n_p;$$

and finally interpolated. Here the resulting degree can be estimated separately for each particular element.

[5] It is usually worse conditioned than for a randomly generated rational matrix.

[6] Despites the balancing mentioned above.

By this procedure, one can interpolate a rational matrix with degrees 18. This corresponds to the sum of two rational matrices of arbitrary dimension with degrees 9. The resulted rational matrix is in the form of the matrix of numerators and the matrix of denominators. It can be shown, that the computation takes the same time for both interpolation methods.

To compare relative accuracy and time consumed by computation for the last two described methods (elementwise interpolation and definition based method), let us sum two (2×2) rational matrices with elements of the degree 9. The interpolation method computes the result in 25 seconds with relative accuracy 10^{-7}, the on definition based method computes the result in 45 seconds with relative accuracy about 10^{-13}.

Let us summarize advantages and disadvantages of the proposed method for the sum of rational matrices. We have studied only the case, when the rational matrices are originally given element by element. The interpolation giving the resulted rational matrix in the form of polynomial matrix fraction is useful only for relatively small matrices (up to (2×2) with elements of degree 4). To add larger matrices and/or higher degrees, we recommend to employ the interpolation giving the sum element by element or the traditional method. The former is quicker, but its relative accuracy is a little lower and degree is limited. The latter has limitation neither on dimension nor on degrees. If reduced form of rational matrix is required, the interpolation is namely convenient if one is able to estimate the degree of each element correctly.

Let us now illustrate the interpolation resulting in polynomial matrix fraction on a simple MATLAB session.

Example 2.

```
% The 2 x 2 rational matrix G with degrees 1 is
% expressed by the matrix of its numerators

NG =

    11    11     6    -2     1
   -11    -8     8     5     0
     0     0     0     0   NaN

% and the matrix of its denominators

DG =

    -2   -11    14     2     1
     5    -8    -7   -16     0
     0     0     0     0   NaN

% The 2 x 2 rational matrix H with degrees 1 is
% expressed by the matrix of its numerators
```

```
NH =

   -11    12     4     3     1
    -5     8   -13     3     0
     0     0     0     0   NaN

% and the matrix of its denominators

DH =

    -5    -3     9     1     1
    -8     9    11     4     0
     0     0     0     0   NaN

% To get the left polynomial matrix description
% WL^{-1}*VL of the sum F=G+H, we run the macro
% 'raddint' (using the interpolation)

>> [WL,VL]=raddint(NG,DG,NH,DH)

WL =

  Columns 1 through 4

   -1.1187    0.0000    9.9791    0.0000
    0.0000    0.5844    0.0000   -0.1932
         0         0         0         0

  Columns 5 through 8

  -15.2021    0.0000    1.0000    0.0000
    0.0000   -2.3198         0    1.3084
         0         0         0         0

  Columns 9 through 11

    1.0000         0    4.0000
         0    1.0000         0
         0         0       NaN


VL =

  Columns 1 through 7
```

```
    3.6916    5.5933    9.9588  -47.2852
   -0.9205    1.1039    0.8912   -1.9351
         0         0         0         0

Columns 5 through 8

  -13.8209   53.5959    0.7446   23.0000
    4.2451   -0.7786   -3.6006    1.5412
         0         0         0         0

Columns 9 through 11

    0.8730    2.0000    4.0000
   -2.3247    0.4375         0
         0         0       NaN
```

5. PRODUCT OF RATIONAL MATRICES

For multiplication of two (or more) rational matrices are considered the same type of algorithms as in the case of their addition, i.e. standard (on definition based) procedure, interpolation of the whole matrix and element-wise interpolation. We assume again that rational matrices to be multiplied are originally given element by element.

For computing the product

$$F(s) = G(s) * H(s)$$

the first method follows the definition so that

$$f_{ij} = \sum_k g_{ik} * h_{kj}$$

where the sums and products are computed in common way.

Alternatively, interpolation method can be used in two different ways as we have seen in the previous chapter. The first procedure, the whole matrix interpolation, returns the resulting matrix in the form of polynomial matrix fraction

$$F(s) = G(s) * H(s) = W_L^{-1} V_L$$

is achieved in the following steps. At first,

$$n_p = m + \sum_{i=1}^{m+p} d_i$$

of interpolation points is chosen where p and m is the number of rows and columns in $F(s)$ respectively while d_i is the estimated degree of ith column in the composite polynomial matrix $[V_L\ W_L]$. Then, for $j = 1, \ldots, n_p$, the points are substituted and the resulting constant matrices are multiplied to get

$$F(s_j) = G(s_j) * H(s_j), \quad j = 1, \ldots, n_p.$$

Finally, the desired polynomial matrices W_L and V_L are recovered by interpolation as in Theorem 2. Practically, the procedure can only be used to multiply scalar rational functions (say both with degrees up to 8). Multiplication of two 2×2 rational matrices with degrees 1 corresponds to polynomial matrices W_L and V_L with degrees 8. For their interpolation, the corresponding block Vandermonde matrix is 34×34 and its condition number 10^{19} does not guarantee a correct result.

The second interpolation procedure returns the resulting rational matrix element by element: At first,

$$n_p = 1 + \max_{i,j}(deg(n_{F,ij}) + deg(d_{F,ij}))$$

of interpolation points is chosen. Then, for $j = 1, \ldots, n_p$, the points are substituted and the resulting constant matrices are multiplied

$$F(s_j) = G(s_j) * H(s_j), \quad j = 1, \ldots, n_p.$$

Finally, each element f_{ij} of the product $F(s)$ is recovered separately by scalar rational function interpolation following Theorem 2. Here the resulting degrees can be estimated separately for each particular element. This algorithm allows to multiply two 2×2 rational matrices with elements of the degree 4.

To compare multiplication by definition with elementwise interpolation, consider the product of two 2×2 rational matrices with degrees 4. The method based on definition gives the result with relative accuracy about 10^{-14} and the computation takes 15 seconds. The interpolation method performed element by element achieves the relative accuracy about 10^{-8} and its computation takes 8 seconds. If rational matrix with coprime fractions is required, interpolation is efficient if one is able to estimate the degree of each element correctly. Otherwise, both the methods will require computation and extraction of the greatest common divisor in each element.

Let us now summarize our experience with rational matrix multiplication. We have considered the case of rational matrices originally given element by element. and exercised three different algorithms to get their product. The whole matrix interpolation appears unsuitable for rational matrices. The procedure based on definition can be used for arbitrary dimensions and degrees of rational matrices to be multiplied, but it is very slow. The interpolation performed element by element is fast and useful for small matrices and/or low degrees.

6. MORE COMPLEX OPERATIONS WITH RATIONAL MATRICES

More complex operations on among several rational matrices are currently being tested such as inverse $(F^{-1}(s))$, closed-loop transfer matrix $(G(s)\,H(s)(I -$

$G(s)\,H(s))^{-1}$), sensitivity and complementary sensitivity functions and alike. It is expected, that the more operations can be performed within constant matrices, the relatively less important become possible difficulties arising during interpolation of their final result.

(Received April 8, 1998.)

REFERENCES

[1] P. J. Antsaklis and Z. Gao: Polynomial and Rational Matrix Interpolation: Theory and Control Applications. Internat. J. Control *58* (1993), 2, 349–404.

[2] M. Šebek and R. C. Strijbos: Polynomial control toolbox. In: Proceedings of the 4th IEEE Mediterranean Symposium on New Directions in Control & Automation, IEEE–CSS, Chania 1996, pp. 488–491.

[3] V. Kučera: Discrete Linear Control: The Polynomial Equation Approach. Academia, Praha 1979.

Ing. Petr Hušek, Trnka Laboratory for Automatic Control, Department of Control Engineering, Faculty of Electrical Engineering, Czech Technical University, Technická 2, 166 27 Praha 6. Czech Republic.
e-mail: husek@control.felk.cvut.cz

Ing. Michael Šebek, DrSc., Institute of Information Theory and Automation, Pod vodárenskou věží 4, 182 08 Praha 8. Czech Republic.
e-mail: msebek@utia.cas.cz

Prof. Ing. Jan Štecha, CSc., Trnka Laboratory for Automatic Control, Department of Control Engineering, Faculty of Electrical Engineering, Czech Technical University, Technická 2, 166 27 Praha 6. Czech Republic.
e-mail: stecha@control.felk.cvut.cz