Symbolic Manipulation of Rational Matrices and Applications

by

N. P. Karampetakis and P. Tzekis

Department of Mathematics, Aristotle University of Thesaloniki, Thessaloniki 54006, Greece

tel. (fax) ++31 997951 email : karampet@ccf.auth.gr

Abstract.

Rational matrices are extensively used in the analysis, synthesis and design of control systems [2], [3], [4]. The main purpose of this work is to present useful symbolic computational tools for the study of the structure of rational matrices and furthermore for the solution of basic synthesis control problem.

1. Introduction.

Tedious, repetitive and complex calculations are involved in many problems in linear algebra. It is in general beneficial to implement the corresponding problem computationally rather than solving the problem by hand in terms of certain governing factors. Such factors may include the time, accuracy and cost of obtaining such a solution.

The advent of complete sybolic computational packages, such as Maple [1], Mathematica, MACSYMA, Reduce, MuMath and Scratchpad, have made the computational implication of such problems particularly attractive. The main advantages, in regards to linear algebra, lie in the following.

Symbolic Storage - Variables can be stored in an "exact" form (i.e. 1/3 as opposed to 0.33333...) resulting in no loss of accuracy during a calculation. The main disadvantage of uses "exact numbers" are : a) the large size of memory they use, and b) the slow speed they have, such kind of programmes. Thus we can easily see that symbolic computational programmes may be used in case where accuracy is more important than speed and memory. Additionally variables can be left "unassigned" (i.e.) without holding any numerical values) which enables polynomial operations to be defined, say, in an indeterminate s.

Inbuilt Procedures - The existence of hundreds of inbuilt procedures covering both general and certain specialized areas of mathematics. In Maple, for example, the package *linalg* consist of numerous procedures used in the solution of problems in linear algebra such as smith form and the inverse of a polynomial matrix.

Programming Language - The existence of unique high-level programming languages allowing specific procedures to be written. In Maple over 90% of these inbuilt procedures are written in its own procedural programming language and therefore any of them can be implemented in any new procedures written. They can therefore be viewed upon as *building blocks* for subsequently more specified and advanced procedures.

Perhaps understandably, however, neither one of these symbolic computer packages contain existing procedures or indeed packages for specialized work in the study of rational matrices and its applications in analysis, synthesis and design of control systems. However they are clearly flexible enough for the implementation of such work to be carried out. This is the motivation of this paper and we present a package of procedures, developed for use in Maple. More specifically in section 2 a brief description of the rational matrix procedures is given. The procedures concerns the structure of rational functions and matrices i.e. the Smith form of rational matrices over different rings [3], known realization techniques of rational matrices, the solution of matrix diophantine equations over different rings and the solution of known synthesis and design problems in control systems such as the model matching problem, the stabilizing compensator problem, the asymptotic tracking problem [2], [3], [4]. Section 3 demonstrates the uses of these rational matrix procedures with detailed examples.

2. Rational matrix routines.

A subset of the polynomial matrix procedures developed in Maple V are presented in Tables 1-5 along with an overview of their functionality.

3. Examples

In this section we interface the procedures implemented above with the symbolic computational package Maple [1] via the introduction of a rational matrix package *ratmat*

Maple enables new packages, or groups of related problems, to be formed and implemented directly. The advantages of forming packages is that they are easier to work with and groups of procedures can be read into a worksheet as a whole instead of individually. A package is read into a Maple session via the command :

```
>with(package):
```

A procedure, say *file1*, existing within the package can be subsequently implemented via

```
>file1(arg s1);
```

In the sequel we use the procedures, as presented in previous sections, via Maple; the machine used is a Pentium 100Mhz.

Rational matrices

Example 1. Consider the rational matrices :

$$T_1(s)=\begin{bmatrix}1 & s^2\\ 0 & 1\end{bmatrix}\ ;\ T_2(s)=\begin{bmatrix}s-1 & s^2\\ 1 & s+1\end{bmatrix}$$

We can easily found the Smith form at $s=\infty$ of the compound matrix $[T_1(s)\ \ T_2(s)]$, let $S^{\infty}_{[T_1(s)\ \ T_2(s)]}(s)$ using the procedure *SmithForm* as well as the left and right biproper matrices *U(s), V(s)* such that $U(s)S^{\infty}_{[T_1(s)\ \ T_2(s)]}(s)V(s)=[T_1(s)\ \ T_2(s)]$ as shown in the following Maple session :

```
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> with(ratmat):
>T1:=matrix(2,2,[1,s^2,0,1]);
```

$$T_1 := \begin{bmatrix}1 & s^2\\ 0 & 1\end{bmatrix}$$

```
> T2:=matrix(2,2,[s-1,s^2,1,s+1]);
```

$$T_2 := \begin{bmatrix}s-1 & s^2\\ 1 & s+1\end{bmatrix}$$

```
> T:=augment(T1,T2);
```

$$T(s)=\begin{bmatrix}1 & s^2 & s-1 & s^2\\ 0 & 1 & 1 & s+1\end{bmatrix}$$

```
> S:=SmithForm(T,U,V,'infinite',s);
```

$$S := \begin{bmatrix}s^2 & 0 & 0 & 0\\ 0 & s & 0 & 0\end{bmatrix}$$

```
> print(U,V,evalm(U &*S &*V));
```

$$\begin{bmatrix}1 & 0\\ \frac{1}{s^2} & 1\end{bmatrix},\begin{bmatrix}\frac{1}{s^2} & 1 & \frac{s-1}{s^2} & 1\\ -\frac{1}{s^3} & 0 & \frac{s^2-s+1}{s^3} & 1\\ 0 & 0 & 1 & 0\\ 1 & 0 & 0 & 0\end{bmatrix},$$

$$\begin{bmatrix}1 & s^2 & s-1 & s^2\\ 0 & 1 & \frac{s-1}{s^2}+\frac{s^2-s+1}{s^2} & s+1\end{bmatrix}$$

The greatest common left divisor $T_{GL}(s)$ of $T_1(s), T_2(s)$ will also has the same zero structure at $s=\infty$ as shown in [8] and can be easily checked using the procedure *arraygcd* :

```
> arraygcd(T1,T2,'left','infinite',s);
```

$$\begin{bmatrix}s^2 & 0\\ 1 & s\end{bmatrix}$$

Realization of rational matrices

Example 2. Consider the matrix $C(s)\in R(s)^{2\times 2}$:

$$C(s)=\begin{bmatrix}\frac{1-s}{(s+1)^2} & \frac{1}{(2s+1)}\\ 0 & \frac{1}{(s+1)}\end{bmatrix}$$

A left and right coprime matrix fraction description of *C(s)* in Ω_S is calculated via the procedure *mfd :*

```
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> with(ratmat):
>C:=matrix(2,2,[(1-s)/(s+1)^2,1/(2*s+1),0,1/(s+1)]);
```

$$C := \begin{bmatrix}\frac{1-s}{(s+1)^2} & \frac{1}{(2s+1)}\\ 0 & \frac{1}{(s+1)}\end{bmatrix}$$

```
> mfd(C,s,'left','Shur',A1,B1);
                        true
> print(A1,B1,evalm(inverse(A1)&*B1));
```

$$\begin{bmatrix}\frac{1}{2} & 0\\ -\frac{1}{2}\frac{2s+1}{s+1} & \frac{1}{2}\end{bmatrix},\begin{bmatrix}-\frac{1}{2}\frac{2s^2-s-1}{(2s+1)(s+1)^2} & \frac{1}{2}\frac{2s+1}{s+1}\\ \frac{(s-1)\left(s+\frac{1}{2}\right)}{(s+1)^3} & 0\end{bmatrix},$$

$$\begin{bmatrix}-\frac{2s^2-s-1}{(2s+1)(s+1)^2} & \frac{1}{2s+1}\\ -\frac{2s^2-s-1}{(s+1)^3}+2\frac{(s-1)\left(s+\frac{1}{2}\right)}{(s+1)^3} & \frac{1}{s+1}\end{bmatrix}$$

```
> mfd(C,s,'right','Shur',A2,B2);
                        true
> print(A2,B2,evalm(B2 &*inverse(A2)));
```

$$\begin{bmatrix}0 & \frac{1}{4}\\ 1 & \frac{1}{4}\frac{2s^2-s-1}{(s+1)^2}\end{bmatrix},\begin{bmatrix}\frac{1}{2s+1} & 0\\ \frac{1}{s+1} & \frac{1}{2}\frac{(s-1)\left(s+\frac{1}{2}\right)}{(s+1)^3}\end{bmatrix},$$

$$\begin{bmatrix}-\frac{2s^2-s-1}{(2s+1)(s+1)^2} & \frac{1}{2s+1}\\ -\frac{2s^2-s-1}{(s+1)^3}+2\frac{(s-1)\left(s+\frac{1}{2}\right)}{(s+1)^3} & \frac{1}{s+1}\end{bmatrix}$$

Solutions of matrix diophantine equations

Example 3. Consider the following matrix diophantine equation :

$$\underbrace{\begin{bmatrix} 1 & s^2 \\ 0 & 1 \end{bmatrix}}_{A(s)} X(s) + \underbrace{\begin{bmatrix} s-2 & s^3 \\ 0 & s-3 \end{bmatrix}}_{B(s)} Y(s) = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{C(s)}$$

Using the procedure *mdesolve* we can find the polynomial solutions of the above matrix diophantine equation.

```
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> with(ratmat):
>A:=matrix(2,2,[1,s^2,0,1]);
```

$$A := \begin{bmatrix} 1 & s^2 \\ 0 & 1 \end{bmatrix}$$

```
> B:=matrix(2,2,[s-2,s^3,0,s-3]);
```

$$B := \begin{bmatrix} s-2 & s^3 \\ 0 & s-3 \end{bmatrix}$$

```
> C:=matrix(2,2,[1,0,0,1]);
```

$$C := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

```
> mdesolve(A,B,C,X1,Y1,'polynomial',s);
```

true

```
> print(X1,Y1);
```

$$\begin{bmatrix} 1-(-s+2)Q_{1,1}-3s^2Q_{2,1} & -s^2+(-s+2)Q_{1,2}-3s^2Q_{2,2} \\ (-s+3)Q_{2,1} & 1+(-s+3)Q_{2,2} \end{bmatrix}, \quad \begin{bmatrix} Q_{1,1} & Q_{1,2} \\ Q_{2,1} & Q_{2,2} \end{bmatrix}$$

where the matrix *Y1* is an arbitrary polynomial matrix. Using now the procedure *bmdesolve* we can find the respective solutions of the bilateral matrix diophantine equation :

$$\underbrace{\begin{bmatrix} 1 & s^2 \\ 0 & 1 \end{bmatrix}}_{A(s)} X(s) + Y(s) \underbrace{\begin{bmatrix} s-2 & s^3 \\ 0 & s-3 \end{bmatrix}}_{B(s)} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{C(s)}$$

```
> bmdesolve(A,B,C,X2,Y2,'polynomial',s);
```

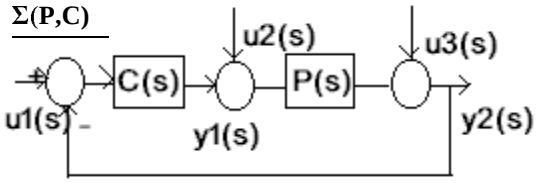
true

```
> print(X2,Y2);
```

$$\begin{bmatrix} 1-(-s+2)Q_{1,1}+(s^3-2s^2)Q_{3,1} & -s^2-s^3Q_{1,1}+(-s+3)Q_{2,1}+ \\ & s^5Q_{3,1}+(s^3-3s^2)Q_{4,1} \\ (-s+2)Q_{3,1} & 1-s^3Q_{3,1}+(-s+3)Q_{4,1} \end{bmatrix}, \quad \begin{bmatrix} Q_{1,1} & Q_{2,1} \\ Q_{3,1} & `Q_{4,1} \end{bmatrix}$$

Implementation of matrix diophantine equations in Control

Example 4. Consider the closed loop system :



where

$$P(s) = \left[\frac{1}{s(s-1)} \right]$$

Using the procedure *stabcomp* we can find all the proper and Hurwitz (Shur) stable compensators *C(s)* of the above closed loop system :

```
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> with(ratmat):
>P:=matrix(1,1,[1/(s*(s-1))]);
```

$$P := \left[\frac{1}{s(s-1)} \right]$$

```
> stabcomp(P,'Shur',s);
```

$$\left[\frac{\left(\frac{1}{4}\frac{1+26s}{2s+1} + \frac{s(s-1)w_{1,1}}{\left(s+\frac{1}{2}\right)^2} \right)(2s+1)^2}{4s^2+12s+5-4w_{1,1}} \right]$$

```
> stabcomp(P,'Hurwitz',s);
```

$$\left[\frac{\frac{1}{2}\left(\frac{9}{2}\frac{s}{2s+1} + \frac{(s-1)w_{1,1}}{s+\frac{1}{2}} \right)s(2`s+1)}{s^2+2s-w_{1,1}} \right]$$

where $w_{1,1}$ is a proper and Shur or Hurwitz stable repsectively rational function. In case where the closed loop transfer function matrix is :

$$H(s)=\left[\frac{1}{(s+1)(s+2)}\right]$$

then we can find the stabilizing compensator *C(s)* which gives rise to this closed loop transfer function matrix via the procedure *matching*.

```
> H:=matrix(1,1,[1/((s+1)*(s+2))]);
```

$$H:=\left[\frac{1}{(s+1)(s+2)}\right]$$

```
> matching(P,H,'Shur',s);
```

$$\left[4\frac{(s-1)s\left(s+\frac{1}{2}\right)^2}{4s^4+16s^3+17s^2+7s+1}\right]$$

```
> matching(P,H,'Hurwitz',s);
```

false

Let $u_2(t)=0, u_3(t)=0$ and $u_1(t)=e^t$ (its Laplace transform is $\frac{1}{s-1}$. We can find using the procedure *tracking* all the stabilizing compensators *C(s)* such that the exponentially stable system *Σ(P,C) asymptotically tracks* the reference input $u_1(t)$.

```
> PS:=matrix(1,1,[1/(s-1)]);
```

$$PS:=\left[\frac{1}{s-1}\right]$$

```
> tracking(P,PS,'Shur',s);
```

$$\left[\frac{\left(\frac{2s+5}{2s+1}+\frac{s(s-1)\left(\frac{25}{16}\frac{1+26s}{2s+1}+\frac{V_{1,1}\left(\frac{1}{2}s-1\right)}{\left(s+\frac{1}{2}\right)^2}\right)(2s+1)^3}{\left(s+\frac{1}{2}\right)^2}\right)}{27s^2-155s-6+26s^3-4V_{1,1}s+8V_{1,1}}\right]$$

```
> tracking(P,PS,'Hurwitz',s);
```

$$\left[\frac{1}{2}\left[(2s+1)^3 s\times\left(2\frac{s+2}{2s+1}+\frac{(s-1)\left(\frac{225}{4}\frac{s^2}{(2s+1)^2}+\frac{V_{11}(s-2)}{s+\frac{1}{2}}\right)}{s+\frac{1}{2}}\right)\right]\right]$$

$$9s^4+9s^3-54s^2-4V_{1,1}s^2+6V_{1,1}s+4V_{1,1}$$

4. Conclusions

In this work we have presented a number of procedures useful for the analysis of rational functions and matrices. The importance of this analysis for the solution of matrix diophantine equations over different rings and therefore for the solution of synthesis control problems has also been shown. The above procedures can be used either on the ring of polynomial, or proper, or proper and Shur stable, or proper and Hurwitz stable matrices and this is one of their advantages. The basic reference for the procedures described above is the book of Vardulakis [3] while there are a lot of books in this specific field as for example the ones of Kucera [2] and Vidyasagar [4]. Some of the procedures presented in this work can also be applied for the solution of other synthesis control problems such as the simultaneous stabilization problem, the multicompensator problem e.t.c.. An interesting Web page, where can someone make use of these procedures is http://dirac.math.auth.gr which is under construction, while the Maple worksheet with all the proposed procedures is available under request by the authors.

Acknowledgments

This work was supported by the Greek National Foundation.

Bibliography

[1] Char B.W., Geddes K.G. Gonnet G.H. and Watt S.M., 1991, *Maple V Language Reference Manual*, Springer Verlag.

[2] Kucera V., 1979, *Discrete Linear Control, The Polynomial Equation Approach.*, John Wiley & Sons, Chichester ; Kucera V., 1991, *Analysis and Design of Discrete Linear Control Systems*, Prentice Hall ; Kucera V., 1993, Diophantine equations in Control - A Survey., *Automatica*, Vol. **29**, No.6, pp.1361-1375.

[3] Vardulakis A.I.G., 1991, *Linear Multivariable Control, Algebraic Analysis and Synthesis Methods.*, John Willey & Sons, Chichester.

[4] Vidyasagar M., 1985, *Control System Synthesis, A Factorization Approach*, MIT Press, Cambridge.

Rational function procedures	
Name	**Function Description**
deg(*t*:function, *WhatIs*:type, *s*:variable)	Computes the discrete valuation of *t(s)* over different rings. *t* is a known rational function dependent on *s* while the parameter WhatIs ∈ {finite, infinite, Hurwitz, Shur}. If *WhatIs*='finite' then the procedure calculates the $\delta_C(t(s))$. If *WhatIs*='infinite' then the procedure calculates the $\delta_\infty(t(s))$. If *WhatIs*='Hurwitz' then the procedure calculate the $\delta_{\Omega_H}(t(s))$. If *WhatIs*='Shur' then the procedure calculates the $\delta_{\Omega_S}(t(s))$.
division(t_1:function, t_2:function, *WhatIs*:type, *s*:variable, *quot*:function, *remn*:function)	Checks if $t_2(s)$ divides exact $t_1(s)$. t_1 and t_2 are known rational functions dependent on *s* while the parameter *WhatIs* ∈ {finite, infinite, Hurwitz, Shur}. The procedure checks if $t_2(s)$ divides exact $t_1(s)$ and returns true or false respectively. *quot* is the quotient of the division of $t_2(s)$ with $t_1(s)$ while *remn* is its remainder.

Table 1. Rational function procedures

Rational matrix procedures	
Name	**Function Description**
SmithForm(*T*:matrix, *U*:matrix, *V*:matrix, *WhatIs*:type, *s*:variable)	Computes the Smith form of *T(s)* over different rings. *T* is a known rational matrix dependent on *s*. *U(s)* and *V(s)* are the left and right transforming matrices i.e $U(s)S^{\Omega}_{T(s)}(s)V(s)=T(s)$ while the parameter *WhatIs* ∈ {finite, infinite, Hurwitz, Shur}. The procedure produces the finite, infinite, Hurwitz and Shur Smith form depending on the value of the variable *WhatIs.*
arraygcd(T_1:matrix, T_2:matrix, *WhatIs1*:type, *WhatIs2*:type, *s*:variable)	T_1 and T_2 are known rational matrices dependent on *s*. The procedure produces the greatest common left (*WhatIs1*='left') or right (*WhatIs1*='right') divisor of T_1 and T_2 at $C, s=\infty, \Omega_H, \Omega_S$. The type of the g.c.d. depends on the value of *WhatIs2* ∈{finite, infinite, Hurwitz, Shur}.
mcmillan(*T*:matrix, *s*:variable)	*T* is a known rational matrix dependent on *s*. The procedure produces the McMillan degree of *T(s).*
ginverse(*T*:matrix)	*T* is a known rational matrix. The procedure computes the generalized inverse of *T(s).*
maxdeg(*A*:matrix, *WhatIs1*:type, *WhatIs2*:type, *s*:variable)	*A* is a known rational matrix dependent on *s*, while *WhatIs2*∈{finite, infinite, Shur, Hurwitz}. If *WhatIs1*='true' then the output of this procedure is the maximum evaluation of *all* the elements of *A(s)* over the ring defined by the variable *WhatIs2*. If *WhatIs1*='false' then the output of this procedure is the maximum evaluation of *all the nonzero* elements of *A(s)* over the ring defined by the variable *WhatIs2*.
mindeg(*A*:matrix, *WhatIs1*:type, *WhatIs2*:type, *s*:variable)	Similarly to *maxdeg* computes the least evaluation of *A(s).*
lcmden(*A*:matrix, *WhatIs1*:type, *s*:variable)	*A* is a known rational matrix dependent on *s*, while *WhatIs*∈{finite, Shur, Hurwitz}. The output of this procedure is the least common multiple of the parts of the denominators of all the elements of *A(s)* which have zeros in *C* (if *WhatIs*='finite') or in Ω_S (if WhatIs='Shur') or in Ω_H (if WhatIs='Hurwitz').
belongs(*A*:matrix, *s*:variable , *WhatIs1*:type, *WhatIs2*:type)	*A* is a known rational matrix dependent on *s*. The output of this procedure is true if *A(s)* belongs to the ring defined by *WhatIs1* ∈ {polynomial, proper, properstable, strictlyproper, strictlyproperstable, stable} and/or *WhatIs2* ∈{Shur,Hurwitz} otherwise returns false.

Table 2. Rational matrix procedures

Realization matrix procedures	
Name	**Function Description**
mfd(*T*:matrix, *s*:variable, *WhatIs1*:type, *WhatIs2*:type, A_1:matrix, B_1:matrix)	*T* is a known rational matrix dependent on *s*. *WhatIs1*∈{left, right} specifies the type of the coprime MFD (left or right) we are looking for. *WhatIs*2∈{ finite, infinite, Hurwitz, Shur} specifies the ring *S* where the matrices $A_1(s)$ and $B_1(s)$ must belong. The procedure computes the numerator matrix $B_1(s)$ and the denominator matrix $A_1(s)$ of the *S*-coprime MFD of *T(s)*.

Table 3. Realization matrix procedures

Matrix diophantine equations procedures	
Name	**Function Description**
mdesolve(*A*:matrix, *B*:matrix, *C*:matrix, *X*:matrix, *Y*:matrix, *WhatIs*:type, *s*:variable)	*A, B, C* are known rational matrices, dependent on *s*. *WhatIs*∈{general, constant, polynomial, proper, Hurwitz, Shur} specifies the kind of solutions *X(s), Y(s)* we are looking for. Thus in case where *WhatIs*='general' (constant, polynomial, proper, Hurwitz, Shur) we are looking for all the rational (constant, polynomial, proper, proper and Hurwitz stable, proper and Shur stable) solutions *X(s), Y(s)* of the matrix diophantine equation *A(s)X(s)+B(s)Y(s)=C(s)*. The output of this procedure is true if the matrix diophantine equation *A(s)X(s)+B(s)Y(s)=C(s)* has a solution over the specific ring or false otherwise. *X(s), Y(s)* are the solutions we are looking for.
spesolve(*A*:matrix, *B*:matrix, *C*:matrix, *X*:matrix, *Y*:matrix, *WhatIs*:type, *s*:variable)	Computes a special solution of the right matrix diophantine equation *A(s)X(s)+B(s)Y(s)=C(s)* over different rings.
bmdesolve(*A*:matrix, *B*:matrix, *C*:matrix, *X*:matrix, *Y*:matrix, *WhatIs*:type, *s*:variable)	Computes the solution of the bilateral matrix diophantine equation *A(s)X(s)+ Y(s) B(s)=C(s)* over different rings.

Table 4. Matrix diophantine equations procedures

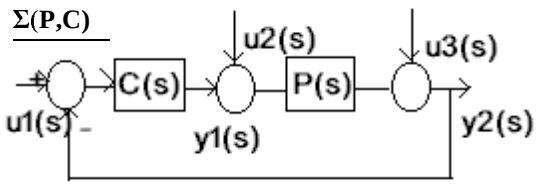
Controller synthesis procedures	
Name	**Function Description**
stabcomp(*P*:matrix, *WhatIs*:type, *s*:variable)	*P* is a known proper rational matrix dependent on *s*. If *WhatIs*='Hurwitz' then returns the family of all the proper and Hurwitz stable compensators *C(s)* of Σ(P,C) while if *WhatIs*='Shur' then returns the family of all the proper and Shur stable compensators *C(s)* of Σ(P,C). 
matching(*P*:matrix, *H*:matrix, *WhatIs*:type, *s*:variable)	*P* is a known proper rational matrix dependent on *s* and represents the transfer function of the open loop system. *H* is a known proper and *Hurwitz* (*Shur*) stable rational matrix dependent on *s* and represents the transfer function of the closed loop system. Both matrices must have full row rank. The type of the matrix *C(s)* depends on the variable *WhatIs*. If *WhatIs*='Hurwitz' then returns the family of all the proper and Hurwitz stable compensators *C(s)* of Σ(P,C) while if *WhatIs*='Shur' then returns the family of all the proper and Shur stable compensators *C(s)* of Σ(P,C).
tracking(*P*:matrix, *f*:rational function, *WhatIs*:type, *s*:variable)	Computes the family of all the stabilizing compensators *C(s)* of the closed loop system Σ(P,C) which asymptotically tracks the reference input *f(t)* (Laplace transform of *f(t)*).

Table 5. Controller synthesis procedures