Symbolic Computations on Rings of Rational Functions and Applications in Control Engineering

N.P. Karampetakis[1], E.N. Antoniou[2], A.I.G. Vardulakis[1], and S. Vologiannidis[3]

[1] Aristotle University of Thessaloniki, Department of Mathematics,
54124 - Thessaloniki, Greece
{karampet,avardula}@math.auth.gr
[2] Technological Educational Institute of Thessaloniki, Department of Sciences,
54101 - Thessaloniki, Greece
eantonio@gen.teithe.gr
[3] Technological Educational Institute of Serres, Department of Information and Communication Sciences,
62100 - Serres, Greece
svol@teiser.gr

Abstract. A collection of algorithms implemented in Mathematica 7.0, freely available over the internet, and capable to manipulate rational functions and solve related control problems using polynomial analysis and design methods is presented. The package provides all the necessary functionality and tools in order to use the theory of $\Omega-$stable functions, and is expected to provide the necessary framework for the development of several other algorithms that solve specific control problems.

1 Introduction

Polynomial methods are modern design techniques for complex multi-variable systems, signals and processes based on manipulations of polynomials, polynomial matrices, and other similar objects. The theoretical background of polynomial design techniques for control systems can be traced back to the late fifties. However, their frontal attack to control theory started in the seventies when the first really important results were achieved. One of the most important results is without doubt the parameterization of all controllers that stabilize a given plant, now referred to as Youla-Kucera parameterization. In the eighties, the polynomial methods were used to solve robust control problems and employed also in the field of signal processing. The Youla–Kucera (YK) parametrization of all stabilizing controllers [1], [4] is particularly useful for controller design because all the closed-loop system transfer functions depend affinely in the same parameter that can be optimized over the set of proper stable rational functions [2]. The significance of polynomial methods as a theoretical tool for engineers and applied mathematicians has been proven through the years.

An important research area for control engineering is the development of robust and efficient algorithms solving control problems. The software packages

R. Moreno-Díaz et al. (Eds.): EUROCAST 2009, LNCS 5717, pp. 587–594, 2009.

currently available fall into two categories. The first one includes packages that use numerical methods, having the advantages of high speed and low memory requirements and as a trade-off the loss of numerical accuracy. Well known such packages are "Control Systems Toolbox" [7], "Polynomial toolbox" for MATLAB [5] and "The Control and Systems Library" SLICOT, (see [8], [9]). The second category includes packages using symbolic calculations, that can obtain exact solutions even to parametrical control problems, with higher computational requirements, such as "Control System Professional" for MATHEMATICA [6]. The package presented in the present paper uses the symbolic computation approach.

Many control problems require the design of a compensator satisfying specific requirements, such as the pole placement to a certain subregion Ω of the complex plane $\mathbb{C}$. These kind of problems have already been solved theoretically [10], by the use of $\Omega-$generalized polynomials. Although there exist some toolboxes to manipulate polynomial matrices, such as the "Polynomial toolbox" for MATLAB, there is no software package dealing with the theory and applications of $\Omega-$stable functions/matrices. In this paper we present a collection of algorithms implemented in Mathematica 7.0, freely available over the internet, able to manipulate rational functions and solve related control problems using polynomial analysis and design methods. The package provides all the necessary functionality and tools in order to use the theory of $\Omega-$stable functions and matrices. The existing functions are capable of solving control problems concerning both SISO and MIMO systems, but for brevity reasons the latter is not presented in the following sections.

2 Ω-Stable Rational Functions and the Rings Package

Let Ω be a subset of the extended complex plane $\bar{\mathbb{C}} = \mathbb{C} \cup \infty$, symmetric with respect to the real axis which excludes at least either one point on the real axis $a \in \mathbb{R}$ or ∞. In the following the set Ω will play the role of the forbidden poles region in $\bar{\mathbb{C}}$. Let Ω^C denote the complement of Ω with respect to $\bar{\mathbb{C}}$, i.e. $\Omega \cup \Omega^C = \bar{\mathbb{C}}$. Given a rational function $t(s) \in \mathbb{R}(s)$ we can always factorize it as

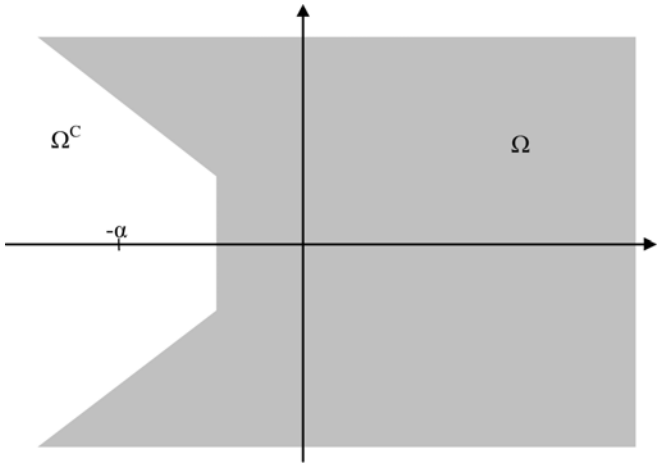
$$t(s) = t_\Omega(s)\hat{t}(s),$$

where $t_\Omega(s) = n_\Omega(s)/d_\Omega(s)$, $\hat{t}(s) = \hat{n}(s)/\hat{d}(s)$, with $n_\Omega(s), d_\Omega(s)$ polynomials having zeros in Ω and $\hat{n}(s), \hat{d}(s)$ polynomials with zeros in Ω^C.

A rational function $t(s)$ will be called *Ω-stable* iff it has all its poles in Ω^C. Then $t(s)$ can be written as

$$t(s) = n_\Omega(s)\frac{\hat{n}(s)}{\hat{d}(s)}, \tag{1}$$

where the polynomials $n_\Omega(s), \hat{n}(s), \hat{d}(s)$ have zeros in the regions described above. If we denote by $S_\Omega = \{t(s) : t(s) \text{ is } \Omega\text{-stable}\}$ the set of all Ω-stable rational functions, it can be easily seen that the set S_Ω endowed with the operations of



addition and multiplication is a commutative ring, with unity element the real number 1 and no zero divisors (i.e. it is an integral domain). The units of S_Ω are the elements of S_Ω whose inverse also belongs to S_Ω.

In order to be able to manipulate rational functions and solve related control problems, we have implemented the *Rings* package, using Mathematica 7.0. The package provides all the necessary functionality and tools for the experimentation with the theory of $\Omega-$stable functions. In a Mathematica notebook in order to load the *Rings* package, we give

```
<<Rings‘
```

The next line provides the global setting for the working ring of proper and Hurwitz stable rational functions:

```
$ForbiddenPolesArea = RightComplexPlane;
```

The `RightComplexPlane` is simply a membership function that gives `True` for values in the extended closed right-half complex plane and false elsewhere. The actual definition of the function is the following:

```
RightComplexPlane[s_]:= (Re[s]>=0) || (Abs[s]==Infinity);
```

Depending on the choice of Ω one may end up with a variety of rings. We give some examples:

- $\Omega_h = \{s : \mathrm{Re}(s) \geq 0\} \cup \{\infty\}$, the ring of proper, Hurwitz stable rational functions.
- $\Omega_s = \{s : |s| \geq 1\} \cup \{\infty\}$, the ring of proper, Schur stable rational functions.
- $\Omega_{pol} = \mathbb{C}$, the ring of polynomials $\mathbb{R}[s]$.
- $\Omega_{pr} = \{\infty\}$, the ring of proper rational functions $\mathbb{R}_p(s)$.

The *Rings* package provides a set of membership functions, to define to most commonly used Ω's, resulting in the above rings of rational functions. The predefined membership functions that can be used as values of `$ForbiddenPolesArea` are

`RightComplexPlane`	$\longrightarrow$ Corresponding to Ω_h
`UnitDiscComplement`	$\longrightarrow$ Corresponding to Ω_s
`FiniteComplexPlane`	$\longrightarrow$ Corresponding to Ω_{pol}
`InfinityPoint`	$\longrightarrow$ Corresponding to Ω_{pr}

In order to check whether a given rational function $t(s)$ belongs to the ring S_Ω defined by the `$ForbiddenPolesArea` setting, we can use

```
RingQ[ts,s]
```

where the argument s indicates the function's indeterminate variable. As in most of the functions of the package one can override the default ring setting provided by `$ForbiddenPolesArea`, using an extra option of the form

```
RingQ[ts,s,ForbiddenPolesArea->UnitDiscComplement]
```

In the latter example the rational function $t(s)$ will be checked against the ring of Schur stable functions. A very important result (see [3]) following from the above discussion, is that the field of rational functions can be considered as a quotient field of S_Ω. Any rational function $t(s) \in \mathbb{R}(s)$ can be written (non - uniquely) as a ratio of two Ω-stable rational functions, i.e.

$$t(s) = \frac{n(s)}{d(s)},$$

where $n(s), d(s) \in S_\Omega$. Given a rational functions $t(s)$ one can obtain the above fractional representation using the function

```
{ns,ds} = RingFraction[ts,s]
```

which returns the numerator $n(s)$ and the denominator $d(s)$ respectively.

Furthermore, one may define the mapping $\delta_\Omega : S_\Omega \to \mathbb{Z}$ via

$$\delta_\Omega(t(s)) = \begin{cases} \#\text{ of zeros of } t(s) \text{ in } \Omega, & \text{if } t(s) \neq 0 \\ -\infty, & \text{if } t(s) = 0 \end{cases}. \tag{2}$$

The mapping $\delta_\Omega(.)$ satisfies

$$\delta_\Omega(t_1(s)t_2(s)) = \delta_\Omega(t_1(s)) + \delta_\Omega(t_2(s)),$$

$$\delta_\Omega(t_1(s) + t_2(s)) \geq \min\{\delta_\Omega(t_1(s)), \delta_\Omega(t_2(s))\},$$

and thus it serves as a *discrete evaluation* or *degree* for the ring S_Ω. For the above mentioned choices of Ω, where $t(s)$ is written as in (1), the discrete evaluation $\delta_\Omega(t(s))$ can be calculated as follows

- For the ring of proper Hurwitz or Schur stable rational functions, $\delta_\Omega(t(s)) = \deg \hat{d}(s) - \deg \hat{n}(s)$.
- For the ring of polynomials $\mathbb{R}[s]$, $\delta_\Omega(t(s)) = \deg n_\Omega(s)$.
- For the ring of proper rational functions $\mathbb{R}_p(s)$, $\delta_\Omega(t(s)) = \deg d(s) - \deg n(s)$.

In the *Rings* package the calculation of the degree of a given Ω-stable rational function $t(s)$, can be done by

```
RingDegree[ts,s]
```

It can be easily seen that the units of S_Ω are exactly the elements $u(s) \in S_\Omega$, satisfying $\delta_\Omega(u(s)) = 0$ i.e. rational functions having no poles and zeros in Ω.

Going one step further, as shown in [3], one may define the euclidean division between two elements of S_Ω. Given $a(s) \neq 0, b(s) \in S_\Omega$ there exist $q(s), r(s) \in S_\Omega$ (called the quotient and remainder respectively), such that

$$b(s) = q(s)a(s) + r(s), \tag{3}$$

satisfying

$$\delta_\Omega(r(s)) < \delta_\Omega(a(s)) \text{ or } r(s) = 0 \tag{4}$$

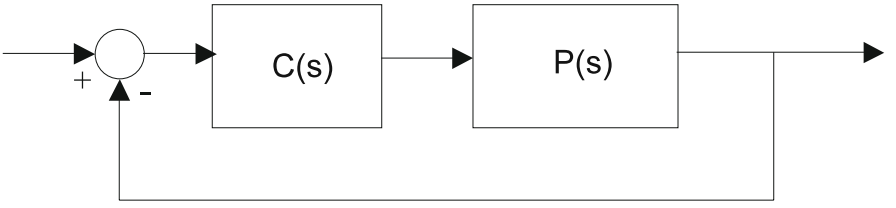
The integral domain S_Ω equipped with the discrete evaluation $\delta_\Omega(.)$ and the above defined division is thus a euclidean ring. The *Rings* package provides the following division related functionality

`RingDivision[ns,ds,s]`	⟶	Returns the pair {quotient,remainder} of the division of ns by ds
`RingQuotient[ns,ds,s]`	⟶	Returns the quotient of the division of ns by ds
`RingRemainder[ns,ds,s]`	⟶	Returns the remainder of the division of ns by ds
`RingGCD[as,bs,s]`	⟶	Returns the greatest common divisor of as and bs
`RingCoprime[as,bs,s]`	⟶	Checks whether as,bs are coprime rational functions

2.1 Feedback Stabilization of SISO Plants

One of the fundamental problems in control theory is the problem of stabilization of a possibly unstable plant via feedback. Given a linear plant described by its transfer function $P(s)$, the goal of stabilization via feedback, amounts in finding a controller $C(s)$ such that the closed loop system in figure 2.1 has all its poles in a prescribed region of the complex plane.

The stability region may differ depending on the nature of the problem or certain performance requirements imposed by the designer. For instance in continuous time it is desired to move all poles in the open left half complex plane



($\mathrm{Re}\, s < 0$) or even better in some subregion of the left half complex plane symmetrically located with respect to the real axis. Accordingly, in the discrete time case stability is achieved by moving the poles of the closed loop system inside the unit disc or in some subregion of the unit disc.

Let $P(s)$ be a proper rational transfer function describing the plant that we wish to stabilize using the feedback scheme of figure 2.1. We shall restrict our interest to the continuous time case thus it is desired to obtain Hurwitz stability for the closed system. According to the previous section if we set $\Omega = \{s : \mathrm{Re}(s) \geq 0\} \cup \{\infty\}$, we can write $P(s)$ as a ratio of two Ω-stable rational functions, i.e.

$$P(s) = \frac{n(s)}{d(s)},$$

with $n(s), d(s) \in S_\Omega$. We seek to find a proper rational controller $C(s)$, which we assume to be of the form

$$C(s) = \frac{y(s)}{x(s)},$$

where $x(s), y(s) \in S_\Omega$. The closed loop transfer function is given by

$$G(s) = \frac{n(s)y(s)}{d(s)x(s) + n(s)y(s)}. \tag{5}$$

For the stabilization of the closed loop system it is required to determine $x(s)$, $y(s) \in S_\Omega$ such that $G(s)$ is itself Ω-stable. This can be done if the expression $d(s)x(s) + n(s)y(s)$ is a unit of S_Ω or simply 1, i.e.

$$d(s)x(s) + n(s)y(s) = 1. \tag{6}$$

The above equation is *diophantine equation* over the ring of Ω-stable rational functions and can be solved using the euclidean algorithm. If a particular solution of (6) $x_0(s), y_0(s) \in S_\Omega$ is determined, then every other solution can be obtained from the parameterization

$$\begin{aligned} x(s) &= x_0(s) + n(s)t(s) \\ y(s) &= y_0(s) - d(s)t(s) \end{aligned} \tag{7}$$

where $t(s) \in S_\Omega$ is an arbitrary Ω-stable rational function. We illustrate this methodology via the following

Example 1. In order to meet certain performance criteria, the package allows the user to define custom stability areas and hence custom rings of $\Omega-$stable rational functions. Such an area of the complex plane that can guarantee approximately 5% overshoot and 1.5sec settling time of the closed loop system is defined by the function

```
MyArea[s_]:=(Re[s]>=-2.7)||((-2.4<=Arg[s])&&(Arg[s]<=2.4))
```

We also need to provide an arbitrary constant $a \in \mathbb{R}$, such that $-a \in \Omega^C$. Such a choice could be

```
$MyAreaAutomaticAlpha = 3;
```

We can either set `$ForbiddenPolesArea` to `MyArea`, to change the working ring globally, or use the individual option setting for temporary use. We prefer to switch to the custom ring globally, so

```
$ForbiddenPolesArea = MyArea;
```

For the ring corresponding to `MyArea` the transfer function $P(s)$ of a given the plant can be factorized using the `RingFraction` function, which gives

```
Ps=(s-3)/(s^2 +2 s-8);
{Ns, Ds} = RingFraction[Ps, s]
```

$$\left\{\frac{s-3}{s^2+7s+12}, \frac{s-2}{s+3}\right\}$$

In order to compute a stabilizing controller for the given plant, he have to solve the diophantine equation $d(s)x(s) + n(s)y(s) = 1$, where $x(s), y(s)$ are proper and Hurwitz stable rational functions. This can be accomplished by

```
{xo, yo} = RingDiophantineSolve[Ds, Ns, 1, s]
```

$$\left\{\frac{s+33}{s+3}, -\frac{25(s+4)}{s+3}\right\}$$

so a stabilizing controller is given by

```
Cs = yo/xo
```

$$-\frac{25(s+4)}{s+33}$$

while the family of stabilizing controllers is parametrized by the forumla

$$C(s) = \frac{y(s)}{x(s)} = \frac{y_0(s) - d(s)t(s)}{x_0(s) + n(s)t(s)}$$

where $t(s) \in S_\Omega$ is an arbitrary Ω-stable rational function.

This can be computed using

```
Cs = (yo-Ds*t[s])/(xo+Ns*t[s])
```

$$\frac{-\frac{(s-2)t(s)}{s+3} - \frac{25(s+4)}{s+3}}{\frac{(s-3)t(s)}{s^2+7s+12} + \frac{s+33}{s+3}}$$

3 Conclusions

In this paper we have presented a collection of algorithms implemented in Mathematica 7.0, available freely over the internet, able to manipulate rational functions and solve control related problems using polynomial analysis and design methods. The package provides all the necessary functionality and tools in order to use the theory of $\Omega-$stable functions. The user can choose one of the common rings of $\Omega-$stable rational functions, such as the ring of Hurwitz stable, Schur stable, proper rational functions or the ring of polynomials. It is also allowed to introduce user defined rings by describing the set Ω using a simple procedure.

Some commonly used algorithms for the analysis of control systems have been implemented. The package also contains algorithms for the solution of matrix Diophantine equations over a variety of rings, solving several synthesis and design control problems such as stabilization, pole placement, dead-beat control, model matching, disturbance rejection, minimum variance control, LQG or H_2 optimal control, H_∞ optimization, tracking problems etc.The *Rings* package is expected to provide the necessary framework for the implementation of many of the algorithms for the solution of control synthesis and design problems that exist in the literature.

References

1. Kucera, V.: Stability of discrete linear feedback systems. In: Proc. IFAC World Congr., Boston, MA, vol. 1, paper 44.1 (1975)
2. Kucera, V.: Diophantine equations in control - A survey. Automatica 29(6), 1361–1375 (1993)
3. Vardulakis, A.I.G.: Linear Multivariable Control - Algebraic Analysis and Synthesis Methods. John Willey & Sons Ltd., New York (1991)
4. Youla, D.C., Jabr, H.A., Bongiorno, J.J.: Modern Wiener–Hopf design of optimal controllers. IEEE Trans. Autom. Control AC-21(3), 319–338 (1976)
5. Polynomial Toolbox for Matlab, Polyx Ltd., `http://www.polyx.cz/`
6. Control System Professional Version 2, Wolfram Research Inc., `http://www.wolfram.com/products/applications/control/`
7. Control System Toolbox, The MathWorks Inc., `http://www.mathworks.com/products/control/`
8. Sima, V.: SLICOT-based advanced automatic control computations. In: Advances in Automatic Control, pp. 337–350. Kluwer Academic Publishers, Dordrecht (2003)
9. The Control and Systems Library SLICOT, NICONET, `http://www.win.tue.nl/niconet/`
10. Pernebo, L.: An algebraic theory for the design of controllers for linear multivariable systems. I. Structure matrices and feedforward design. IEEE Trans. Automat. Control 26(1), 171–182 (1981)