数式処理 *J.JSSAC (2005)*
Vol. 11, No. 3,4, pp. 153 - 164

Noda2005 論文

Strelitz test for stable polynomials and its application to design problems of control systems

Taku TAKESHIMA*

Fujitsu Laboratories, Ltd.

Abstract

Stability is a fundamental problem in control systems. The basic problem is mathematically described as "Decide whether or not a given univariate polynomial has all roots in the left half part of the complex number plane without prior computation of the roots," and was solved by Routh and Hurwitz independently. Much later, Strelitz presented an algorithm to decide a polynomial to be stable without division by any coefficients of the polynomial. This is a favorable property for an algorithm to find a necessary and sufficient condition for polynomials with symbolic parameters to be stable. In this article, applying the Strelitz test for parametric polynomials are presented, and how a typical stability problem, *i.e.*, *D*-stability, is translated into the scope of the Strelitz test.

1 Introduction

Stability is a fundamental problem in control systems. The basic problem is mathematically described as "Decide whether or not a given univariate polynomial has all roots in the left half part of the complex number plane without prior computation of the roots," and was solved by Routh and Hurwitz independently.

Among variety of works, Kimura and Hara[3][4] showed a considerably wide class of design problems in robust control systems, including stability related problems, can be formulated as sign definite condition (SDC) problems, and presented a primitive algorithm based on Euclidean remainder sequence to convert original problems to SDC.

Then later, Anai and Hara[1][2] reformulated SDC problem as a special quantifier elimination problem, and showed it is solved smartly and more efficiently through Sturm-Habicht sequence computation. Their method is well applicable for exceptional cases when the Euclidean sequence becomes abnormal by the instantiation (specialization) of symbolic parameters.

In this article, we show that typical *D*-stability problems, treated in the above papers, can be and much better be solved by an alternative formulation without SDC reformulation. They can be solved directly as stable polynomial problems based on "sum-of-roots polynomial" introduced by Strelitz[5]. In view of algebraic computation, the presented method bears superior properties over that of SDC.

*takesima@labs.fujitsu.com

2 Sum-of-roots polynomial and the Strelitz test for a stable polynomial

An univariate polynomial with real coefficients is called a stable polynomial if all of its roots lie in the left half part of the complex plane (left half plane, in short.) Routh-Hurwitz test is a well known decision procedure for a polynomial to be stable.

The Strelitz test is an alternative one, which has good properties in view of algebraic computation, subsequently gives better performance to polynomials with symbolic coefficients.

Remark 1
Routh-Hurwitz criterion is computed through a process equivalent to counting the number of roots (a modified polynomial remainder sequence or subresultant sequence), while the Strelitz criterion does not attempt to count roots in its computing process. Routh-Hurwitz criterion is considered over-quality when only stability is in question.

Now we introduce the key object *sum-of-roots polynomial* for the Strelitz test. Let $f \in \mathbb{R}[z]$ be a *monic* polynomial with a positive degree n, and $\{\alpha_i\}_{i=1,\dots,n}$ be the set [1] of all roots of f. Define a new polynomial $g \in \mathbb{R}[z]$, monic with degree $\frac{n(n-1)}{2}$, by

$$g = \prod_{1 \le i < j \le n} (z - (\alpha_i + \alpha_j)).$$

Then, g is called the *sum-of-roots polynomial* of f.

Note that, every coefficient of g is real since every root $\alpha_i + \alpha_j$ has its conjugate among $\{\alpha_i + \alpha_j\}_{1 \le i < j \le n}$.

To compute g efficiently, Strelitz shows an algorithm based on Newton-Girard formulae, which will be described in a later section.

We may consider a *bigger* sum-of-roots polynomial G of f defined by

$$G = \prod_{1 \le i, j \le n} (z - (\alpha_i + \alpha_j)) = 2^n f(\frac{z}{2}) \times g(z)^2.$$

Then, by the definition of resultant, it is easy to see that

$$G = \mathrm{res}_t(f(t), f(z-t)).$$

One might use this resultant to obtain g, but, mainly due to space problem, it is applicable only for a small f, *i.e.*, with very small n and having small number of parametric (symbolic) coefficients which necessarily appear in f for design problems.

We prepare two propositions before stating the main property of the sum-of-roots polynomial.

Proposition 2
If f is stable, then g is also stable.

Proof Since every root of g is the sum of two roots of f, both lying in the left half plane, it consequently lies in the left half plane. ∎

Proposition 3
Let $f \in \mathbb{R}[z]$ be monic. Then, the following two statements cannot hold at the same time.

[1] Strictly speaking, we have to use "multi-set of roots," although we use "set of roots" according to the custom.

1. All the coefficients of f are positive.
2. f has a non-negative real root.

Proof Assume both of the statements hold. Then, the second statement says there is a non-negative real number, say γ, satisfying $f(\gamma) = 0$. But, by the first statement, $f(\gamma)$, being the sum of products of positive coefficients and non-negative powers of γ, never becomes 0. This is a contradiction. ∎

The main property of the sum-of-roots polynomial is now stated as follows.

Theorem 4 (Strelitz)
A monic polynomial f with real coefficients is stable if and only if the coefficients of both f and its sum-of-roots polynomial g are all positive.

Proof (only if part): Assume f is stable. Then, if f has a real root, say γ, it is negative. Thus, f has a linear factor $(z - \gamma)$, the coefficients of which, 1 and $-\gamma$, are clearly positive. And if f has a pair of conjugate roots, say α and $\overline{\alpha}$, f has a quadratic factor $(z-\alpha)\times(z-\overline{\alpha}) = z^2 - (\alpha+\overline{\alpha})z + \alpha\overline{\alpha}$, the coefficients of which are again obviously positive since the real part of α, and that of $\overline{\alpha}$ too, are negative for stable f. Because f is nothing but the product of all these linear factors and quadratic ones with multiplicity counted, all of its coefficients must be positive. Since g is also stable by Proposition 2, all of its coefficients are positive by the same argument.

(if part): Assume all the coefficients of both f and g are positive. Then, if f has a real root, say γ, it must be negative by Proposition 3. If f has a pair of conjugate roots, say α and $\overline{\alpha}$, g has a root $\alpha + \overline{\alpha}$, which is a real number. Here, Proposition 3 applied for g again tells that $\alpha + \overline{\alpha}$ must be negative, which in turn means the real part of α is negative. Thus, every root of f lies in the left half plane. Therefore, f is stable, and hence g is also stable by Proposition 2. ∎

In this paper, we call the condition for a polynomial to be stable described in Theorem 4 the Strelitz condition or the Strelitz criterion, and the decision procedure based on it the Strelitz test.

Remark 5
If $f(z)$ monic has imaginary coefficients, then we will apply the theorem to $f(z)\overline{f(\overline{z})} \in \mathbb{R}[z]$ for testing all the roots of f lie in the left half plane.

3 Computing the sum-of-roots polynomial

The coefficients of the sum-of-roots polynomial g defined in section 2 are all symmetric with respect to the roots of f, and therefore integral polynomials in the coefficients of f.

Strelitz presented an algorithm to compute g efficiently through power-sums. The algorithm computes the coefficients of g from the coefficients of f, by using Newton-Girard formulae for power-sums of polynomial roots, together with his recurrence formula which relates power-sums of roots of f and those of g.

Put $f = a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0$ and $g = b_m z^m + a_{m-1}z^{m-1} + \cdots + b_1 z + b_0$ for positive n and $m = \frac{n(n-1)}{2}$ with $a_n = 1$ and $b_m = 1$.

Let $\{\alpha_i\}_{i=1,\ldots,n}$ be the set of all n roots of f, and $\{\beta_i\}_{i=1,\ldots,m} = \{\alpha_i + \alpha_j\}_{1\le i<j\le n}$ be the set of all $m = \frac{n(n-1)}{2}$ roots of g.

Moreover, we introduce the power-sums of f and g, by putting $\sigma_j = \sum_{i=1}^{n} \alpha_i^j$, $(j = 0, 1, ..., m)$ and $s_j = \sum_{i=1}^{m} \beta_i^j$, $(j = 0, 1, ..., m)$ respectively.

Then, the computation of g from f is sketched by the algorithm (Algorithm 6) below which consists of three steps.

Algorithm 6 (SORP: sum-of-roots polynomial)
Input: coefficients $\{a_i\}_{i=0,...,n}$ of monic polynomial $f \in \mathbb{R}[z]$;
Output: coefficients $\{b_l\}_{l=0,...,m}$ of the sum-of-roots polynomial of f;

Step 1: Represent σ_j by a_i, $(i = 0, 1, ..., n)$ for $j = 0, 1, ..., m$.

Step 2: Represent s_k by a_i, $(i = 0, 1, ..., n)$, through σ_j, $(j = 0, 1, ..., m)$ which are computed in Step 1, for $k = 0, 1, ..., m$.

Step 3: Represent b_l by a_i, $(i = 0, 1, ..., n)$, through s_k, $(k = 0, 1, ..., m)$ which are computed in Step 2, for $l = m, m-1, ..., 0$.

In the following subsections, recurrence formulae to the above Algorithm 6 will be given.

3.1 Power-sums and coefficients of a polynomial

In Step 1 and Step 3 of Algorithm 6, well-known Newton-Girard formulae are used.

Newton-Girard formulae for a_i and σ_j are described as follows.

$$\begin{array}{rcl} \sigma_0 & = & n, \\ \sigma_1 + a_{n-1} & = & 0, \\ \sigma_2 + \sigma_1 a_{n-1} + 2a_{n-2} & = & 0, \\ & \dots & \\ \sigma_n + \sigma_{n-1}a_{n-1} + 2\sigma_{n-2}a_{n-2} + \cdots + na_0 & = & 0. \end{array} \tag{1}$$

And for $j > n$,

$$\sigma_j + \sigma_{j-1}a_{n-1} + \cdots + \sigma_{j-n}a_0 = 0.$$

By using these formulae as recurrence relations, we can compute σ_j, a polynomial in a_i's, successively upwards for $j = 0, 1, 2, ..., m$.

Note: We need no divisions but only ring operations on $\mathbb{Z}[a_n, ..., a_0]$ *in the process.*

Newton-Girard formulae for s_k and b_l are obtained by replacing symbols m, b_l, s_k for n, a_i, σ_j, respectively, in equations (1). And by using the formulae as recurrence relations, we can compute b_l from s_k successively downwards for $l = m, m-1, ..., 1, 0$.

Note: In this process, we need exact division by integers besides ring operations on $\mathbb{Z}[s_m, .., s_0]$*, and subsequently, on* $\mathbb{Z}[a_n, ..., a_0]$*, since* s_k*'s are known to be integral in* $a_n, ..., a_0$ *by the formulae shown in the next subsection.*

3.2 Power-sum relationships

Strelitz presented recurrence formulae that relate power-sums ($\{s_j\}_{j=0,1,...,m}$) for g to those ($\{\sigma_i\}_{i=0,1,...,m}$) of f, *viz.*,

$$2s_j = \sum_{p=0}^{j} \binom{j}{p} \sigma_p \sigma_{j-p} - 2^j \sigma_j, \ (j = 0, 1, ..., m). \tag{2}$$

Equation (2) is derived from the following identity.

$$\left(\sum_{k=1}^{n} e^{\alpha_k \cdot t}\right)^2 = \sum_{k=1}^{n} e^{2\alpha_k \cdot t} + \sum_{p,q=1,p\neq q}^{n} e^{(\alpha_p+\alpha_q)t}.$$

Expanding exponential functions in both sides of the above equation in Taylor series, and collecting terms of the same powers of t, we get

$$\left(\sum_{j=0}^{\infty} \frac{1}{j!}\sigma_j t^j\right)^2 = \sum_{j=0}^{\infty} \frac{2^j}{j!}\sigma_j t^j + \sum_{j=0}^{\infty} \frac{2}{j!} s_j t^j.$$

Equating the coefficients of the same degree in t on the both sides, we get

$$\sum_{p=0}^{j} \frac{1}{p!(j-p)!}\sigma_p \sigma_{j-p} = \frac{2^j}{j!}\sigma_j + \frac{2}{j!} s_j \; (j = 0, 1, ...).$$

Multiplying both sides by $j!$ and expressing $2s_j$ in the other two terms give the result.

Note: Equation (2) tells that to represent s_j in σ_i, we need division by 2 besides ring operations on $\mathbb{Z}[\sigma_m, ..., \sigma_0]$, and further consideration tells that the division is an exact one in operating in $\mathbb{Z}[a_n, a_{n-1}, ..., a_0]$.

Taking the above three notes into account, we need only operating in $\mathbb{Z}[a_n, a_{n-1}, ..., a_0]$ in the whole process to obtain the sum-of-roots polynomial. This is the reason, the sum-of-roots can be used to get stability conditions for polynomials with symbolic parameters.

4 Strelitz test as a special QE

Decision procedure for a polynomial being stable can be considered as a special quantifier elimination (QE) procedure that eliminates the main variable from the given polynomial yielding "true" or "false" as a result. Further, if we take the coefficients from a polynomial ring $\mathbb{R}[\mathbf{p}]$ where $\mathbf{p} = (p_1, ..., p_k)$ are real valued parameters, the same process naturally gives a condition for the question(proposition) to be true. The resulting condition is expressed as a first order logic formula which is composed of atomic formulae with logical connectives, *e.g.*, disjunction, conjunction and negation, where an atomic formulae is either an equality or inequality of polynomials (sometimes rational functions) in $\mathbf{p}$.

Polynomials with such an extended coefficient domain usually appear in design problems. And as other procedures originally designed for constant coefficients met with instantiation (specialization) problems, so does the Strelitz test meet with leading coefficient problems described in the next section.

5 Leading coefficient problem

The sum-of-roots polynomial defined in section 3 is computed only for monic polynomials with real coefficients. As far as polynomials that have only number coefficients is concerned, this is of no problem, theoretically at least. We can always normalize the polynomial in question to a monic polynomial, preserving roots unchanged, by multiplying the reciprocal of its leading coefficient a number.

For design problems, however, we often have to deal with polynomials with several unknown parameters in their coefficients. Of course, sum-of-roots polynomial is in principle applicable to such cases to some extent, if we dare to operate in rational function domain.

Three problems arise, however. One is the complexity increase when we deal with rational function coefficients. The second, more essential, is homomorphism anomaly that inevitably arise from instantiation of parameters to number values. Plainly and simply speaking we have to take care of the cases where the leading coefficient of f vanishes by instantiation. The last problem is rather practical one. We have always take care of sign constraints of the leading coefficients of the polynomials in the process.

Remark 7
Sturm-Habicht sequence is an amendment to Sturm sequence and the subresultant sequence for avoiding rational function coefficients and the homomorphism anomaly as well as sign constraint problems.

In the following, two methods are proposed to solve those problems when applying the Strelitz criterion to polynomials with a non-number leading coefficient.

5.1 Formal reciprocal method

We first consider the case where the leading coefficients do not vanish, and next the case they do vanish.

non-vanishing leading coefficient case: If the leading coefficient of f is a polynomial in several parameter variables, we shall modify computation of g as follows.

Let

$$f = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0$$

such that $a_n \in \mathbb{R}[\mathbf{p}] \backslash \mathbb{R}$.

Introduce a new indeterminate, say t, and multiply it to f. Then, replace only the leading coefficient ta_n of tf by 1 so that we obtain

$$f_t = tf = z^n + ta_{n-1} z^{n-1} + \cdots + ta_1 z + ta_0. \tag{3}$$

At the same time, we shall keep a new equality that means non-vanishing condition for the leading coefficient a_n,

$$ta_n = 1 \tag{4}$$

as a constraint condition for later recovery of g. By this constraint, a_n cannot become 0, and t stands for a rational function $\frac{1}{a_n}$, a formal reciprocal to a_n. Note that this modified monic polynomial $f_t \in \mathbb{R}[t, \mathbf{p}][z]$ has the same roots as f does for the same instantiation of $\mathbf{p}$ (and for induced instantiation of t) except for the homomorphism anomaly case, which we explain later.

Then, for this modified monic polynomial $f_t \in \mathbb{R}[t, \mathbf{p}][z]$, we can compute its sum-of-roots polynomial g_t by the procedure sketched in Algorithm 6 in section 3 without operating in the rational function domain. Then, g, the sum-of-roots polynomial of f, can be recovered from g_t by replacing $\frac{1}{a_n}$ for t in g_t by using the constraint (4).

Thus, in this case the coefficients of g may usually have denominators of powers of a_n. In subsequent applications, our interest is concerned with only the positiveness of the coefficients of g. Therefore, in the actual computation, we can cancel the denominators of g by multiplying an appropriate powers of a_n. In such a case, we must be careful of the sign change of constraints on coefficients. For uniform treatment the multiplier may well be taken from even powers of a_n.

The condition for f being stable varies according to the two possibilities of constraint for a_n, *i.e.*, $a_n > 0$ or $a_n < 0$.

In case of $a_n > 0$, it is stated that the coefficients of f and g are all positive. And, in case of $a_n < 0$, it is stated that the coefficients of f are all *negative*, and the coefficients of g are all positive.

vanishing leading coefficient case: If the leading coefficient of f is to vanish, *i.e.*, for the case where we set up a condition

$$a_n = 0, \tag{5}$$

we simply recurse the problem to that of stable polynomial problem of f_R with a formal degree $n-1$ defined by

$$f_R = a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + \cdots + a_1 z + a_0$$

with the constraint condition (5). This recursive reduction of the problem terminates when the formal degree of f_R becomes 0.

Remark 8
It may possible we have a disjunction of stable conditions for n recursively generated polynomials. We can stop recursion as soon as the leading coefficient becomes a number.

Below, we show a basic algorithm for a special QE based on the formal reciprocal method described above, where the formal reciprocals and their constraints remain in the resulting formula..

We define and use several auxiliary functions in the algorithm: LC(f) for the leading coefficient of f, SORP(f) for the sum-of-roots polynomial of monic f, and REST(f) for the polynomial $f - \mathrm{Lt}(f)$, where Lt(f) stands for the leading term of f.

For $f \in \mathbb{R}[\mathbf{p}][z]$ with $\deg_z(f) \geq 0$, function StableCond1(f) defined by Algorithm 9 returns a first order logic formula, in $\mathbf{p}$ and also in dynamically generated indeterminate t's, which gives an equivalent condition for f to be stable.

Algorithm 9 (StableCond1)
function StableCond1(f)
  begin
    if $\deg_z(f) = 0$ **then return** (**true**);
    if $\mathrm{LC}(f) \in \mathbb{R}$ **then**
      begin
        $f' := f/\mathrm{LC}(f)$;
        let $f' =: z^n + a'_{n-1}z^{n-1} + \cdots + a'_1 z + a'_0$;
        $g := \mathrm{SORP}(f')$;
        let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        return $((a'_{n-1} > 0) \wedge \cdots \wedge (a'_1 > 0) \wedge (a'_0 > 0)$
            $\wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0))$
      end
    else
      begin
        Choose a new indeterminate t;
        Construct f_t from f defined by equation (3);
        $g := \mathrm{SORP}(f_t)$;
        let $f =: a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0$;

let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
PositiveCase := $(ta_n = 1) \wedge (t > 0) \wedge (a_n > 0) \wedge (a_{n-1} > 0) \wedge \cdots \wedge (a_1 > 0)$
$\wedge (a_0 > 0) \wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0)$;
NegativeCase := $(ta_n = 1) \wedge (t < 0) \wedge (a_n < 0) \wedge (a_{n-1} < 0) \wedge \cdots \wedge (a_1 < 0)$
$\wedge (a_0 < 0) \wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0)$;
NullCase := $(a_n = 0) \wedge$ StableCond1(REST(f));
return (PositiveCase $\vee$ NegativeCase $\vee$ NullCase)
end
end

Remark 10
There can be several variations of the algorithm depending on the way how we treat t, the formal reciprocal. Indeterminate t's differ from each other at each recursive invocation of the function StableCond1(f). Thus for Algorithm 9, at most n different t's can appear in the resulting first order logic formula. We can take another way where we replace $\frac{1}{a_n}$ for t at each assignment statement for the program variables, PositiveCase and NegativeCase, at every recursive invocation. It is a matter of simplification of the resulting logic formula, and we leave it for another study.

Remark 11
In actual implementation, we have to be careful not to attempt to violate the operation ordering of the first order logic formulae in the computing process. In the algorithm shown above, we preferred simplicity than formality. This remark also applies to Algorithm 12 shown in the next subsection.

5.2 Scalar linear conversion of coordinate

For the second alternative, we make use of a well known convention to convert a non-monic polynomial into a monic one for factoring univariate integral polynomials.

For the case where the leading coefficient a_n to vanish, we shall take the same treatment as in the previous paragraph 5.1. So, we describe here only the case where $a_n \neq 0$.

A monic polynomial $f_M \in \mathbb{R}[\mathbf{p}][w]$ is obtained by

$$f_M = a_n^{n-1} f(\frac{w}{a_n}). \tag{6}$$

Because, if an instantiation of a_n is negative, this scalar linear transformation $z \mapsto w$ causes transposition with respect to the origin, and subsequently yields swapping of the left and right half planes, we have to take care of such a case.

For the case where the leading coefficient a_n is constrained to be positive, *i.e.*, $a_n > 0$, the positiveness of the coefficients of the sum-of-roots polynomial g_M of f_M gives, together with the positiveness of the coefficients of f, the desired condition for f being stable.

On the other hand, for the case of the leading coefficient a_n being negative, *i.e.*, $a_n < 0$, we must use

$$\widehat{f_M}(w) = (-1)^n f_M(-w) = (-1)^n a_n^{n-1} f(\frac{-w}{a_n}) \tag{7}$$

instead of f_M defined by (6). In this case, the *positiveness* of the coefficients of the sum-of-roots polynomial $\widehat{g_M}$ of $\widehat{f_M}$ gives, together with the *negativeness* of the coefficients of f, the desired condition for f to be stable.

Below, we show a basic algorithm based on the scalar linear conversion described above. We assume the same setting for f and use the same auxiliary functions as in Algorithm 9.

Algorithm 12 (StableCond2)

function StableCond2(f)
  begin
    if $\deg_z(f) = 0$ **then return** (**true**);
    if $\mathrm{LC}(f) \in \mathbb{R}$ **then**
      begin
        $f' := f/\mathrm{LC}(f)$;
        let $f' =: z^n + a'_{n-1}z^{n-1} + \cdots + a'_1 z + a'_0$;
        $g := \mathrm{SORP}(f')$;
        let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        return $((a'_{n-1} > 0) \wedge \cdots \wedge (a'_1 > 0) \wedge (a'_0 > 0)$
            $\wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0))$
      end
    else
      begin
        Construct f_M from f defined by equation (6);
        Construct $\widehat{f_M}$ from f defined by equation (7);
        $g_M := \mathrm{SORP}(f_M)$;
        $\widehat{g_M} := \mathrm{SORP}(\widehat{f_M})$;
        let $f =: a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0$;
        let $g_M =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        let $\widehat{g_M} =: z^m + b'_{m-1}z^{m-1} + \cdots + b'_1 z + b'_0$;
        PositiveCase := $(a_n > 0) \wedge (a_{n-1} > 0) \wedge \cdots \wedge (a_1 > 0) \wedge (a_0 > 0)$
            $\wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0)$;
        NegativeCase := $(a_n < 0) \wedge (a_{n-1} < 0) \wedge \cdots \wedge (a_1 < 0) \wedge (a_0 < 0)$
            $\wedge (b'_{m-1} > 0) \wedge \cdots \wedge (b'_1 > 0) \wedge (b'_0 > 0)$;
        NullCase := $(a_n = 0) \wedge$ StableCond2(REST(f));
        return (PositiveCase $\vee$ NegativeCase $\vee$ NullCase)
      end
  end

6 Problem formulation in the Strelitz test

In this section, we show how a typical problem in robust control system design is translated into the scope of the Strelitz test. The example problems are taken from [3].

6.1 *D*-stability

D-stability, viewed from our stand point, is a problem to obtain an equivalent condition for a polynomial with real parametric coefficients to have all roots in a specified area D in the complex number plane. Although it is trivial that arbitrary area cannot be dealt with, we do not go into this problem and only exemplify a basic set of possible D's.

Because any circle in the complex sphere can be mapped to each other by a linear fractional transformation of the complex sphere, we can transform the interior of any circle, including any half plane divided by any straight line on the complex plane, into the left half plane, so that our purpose can well be achieved by the Strelitz test.

Note that, if we can deal with areas D_1, D_2, then we can do with $D_1 \cap D_2$ as well, because the answer of the problem for $D_1 \cap D_2$ is given by a conjunction of the answers for D_1 and D_2.

We show two basic examples that can be dealt with by the Strelitz test.

Circled area: The problem here is to obtain the condition for a given polynomial $f \in \mathbb{R}[\mathbf{p}][z]$, $\deg_z f = n > 0$ to have all roots in D, the interior of a circle described by the following inequality for $z \in D$:

$$(z-c)\overline{(z-c)} < r^2,$$

where $r > 0$ and $c \in \mathbb{C}$. (The boundary is a circle centered at c with radius r.) In practical settings, c will be a negative real number and D will have no common point with the right half plane (and consequently with the imaginary axis, too).

Area D, the interior of the given circle, is transformed into the left half plane by a linear fractional transformation $z \mapsto w$ such that

$$w = \frac{(z-c)+r}{(z-c)-r}.$$

The inverse transformation is given by

$$z = r\frac{w+1}{w-1} + c.$$

Therefore, if we can assume c a real number as a practical assumption, the problem is to find an equivalent condition for a polynomial $\widetilde{f} \in \mathbb{R}[\mathbf{p}][w]$ to have all roots in the left half plane, where $\widetilde{f}$ with $\deg_w(\widetilde{f}) = n$, is defined by

$$\widetilde{f}(w) = (w-1)^n f(r\frac{w+1}{w-1} + c).$$

Thus, the problem is nothing but to find an equivalent condition for $\widetilde{f}$ to be stable, and it is of no doubt that the Strelitz test readily carry it out.

If we do wish to deal with a case for an imaginary c, then construct

$$F(w) = \widetilde{f}(w)\overline{\widetilde{f}(\overline{w})} \in \mathbb{R}[\mathbf{p}][w],$$

and put it for the Strelitz test with paying an extra cost for the doubled degree.

Remark 13
A practical example for an imaginary c may appear in the situation where $D = D_1 \cap D_2$ such that D_1 and D_2 are circled areas centered at c and $\overline{c}$ respectively, and share a non-empty interval on the real axis. For such a situation, the condition for $F(w)$ to be stable gives a D-stability of $f(z)$.

Wedge-shaped area: Another example for D is a wedge-shaped area given by a set intersection of two half plane D_1 and D_2, *i.e.*, $D = D_1 \cap D_2$. Our mission is to find an equivalent condition that all the roots lie in D.

Because we are dealing with $f \in \mathbb{R}[\mathbf{p}][z]$ such that $\mathbf{p}$ are real parameters, and hence the roots are located symmetrically with respect to the real axis, the wedge-shaped area D can be assumed, without loss of generality, to be symmetric with respect to the real axis. By this, we are allowed to test only any one of the half plane components, say D_1 of D, since if a root α is located in D_1 but not in D_2, the conjugate $\overline{\alpha}$ is not located in D_1, and this violation of the questioned condition shall be detected as soon as it is tested for D_1.

Let D_1 be specified such that its boundary line passes through $z_1 \in \mathbb{C}$ and cross the real axis at $c \in \mathbb{R}$, and be located on the left side of the vector $z_1 - c$ on the boundary line.

Then, D_1 is described by the following inequality for $z \in D_1$.

$$i\,\overline{(z_1 - c)}(z - c) - i(z_1 - c)\overline{(z - c)} < 0.$$

D_1 is transformed into the left half plane by

$$w = i\,\overline{(z_1 - c)}(z - c).$$

This is also a kind of linear fractional transformation with only shifting, rotation and stretching or shrinking. Its inverse map is

$$z = \frac{1}{i\,\overline{(z_1 - c)}} w + c.$$

Therefore, D-stability of f is transformed into the usual stability of $\widetilde{f}(w)$ defined by

$$\widetilde{f}(w) = f(\frac{1}{i\,\overline{(z_1 - c)}} w + c).$$

Unfortunately $\widetilde{f}(w)$ may usually contain imaginary coefficients. The Strelitz test is applied to the following F.

$$F(w) = \widetilde{f}(w)\overline{\widetilde{f}(\overline{w})} \in \mathbb{R}[\mathbf{p}][w].$$

7 Discussions

In papers [3] and [4], it is claimed that major problems in control system design can be effectively transformed into SDC. And in papers [1] and [2], SDC with parameters can be solved efficiently by Sturm-Habicht sequence computation.

There are, however, several examples that are suited for another reformulation in the point of computational complexity and simplicity of the algorithm. Some of the reformulation to the SDC problems presented in those papers takes rather long detour arriving at the usual stability problems, *i.e.*, finding the conditions for a polynomial to have all roots on the left half plane, although at a glance its appearance is different. Typical one is the D-stability problem reformulation in [3].

Here, we explain their reformulation process in our terminology for comparison. In the following, we use $< f_1, ..., f_s >$ for a polynomial ideal generated by $f_1, ..., f_s$, and $\mathrm{Zero}_{\mathbb{C}}(I)$ for the set of complex zeros of ideal I, $\mathrm{Zero}_{\mathbb{R}}(I)$ for the set of real zeros of ideal I.

1. The problem is to decide or to find the condition for a polynomial $f(z)$ to have all the roots in specified area $D \subset \mathbb{C}$.

2. The complex plane $\mathbb{C}$ is identified with the Euclidean plane $\mathbb{R}^2$, so that $D \subset \mathbb{R}^2$. Then, the problem is translated into a real bi-variate problem to decide or to find the equivalent condition for the statement $\mathrm{Zero}_{\mathbb{R}}(< f_r(x, y), f_i(x, y) >) \subset D \subset \mathbb{R}^2$, where $f(x + iy) =: f_r(x, y) + i f_i(x, y)$.

3. D is mapped into the left half of the Euclidean plane $\mathbb{R}_{<0} \times \mathbb{R} \subset \mathbb{R}^2$. And the statement $\mathrm{Zero}_{\mathbb{R}}(< f_r(x, y), f_i(x, y) >) \subset D$ is translated into $\mathrm{Zero}_{\mathbb{R}}(< g_r(t, \omega), g_i(t, \omega) >) \subset \mathbb{R}_{<0} \times \mathbb{R}$, where $g(t + i\omega) = g_r(t, \omega) + i g_i(t, \omega)$ holds for some fortunate and convenient $g(z) \in \mathbb{R}[z]$.

4. Let $m(t)$ be a multiple of the minimal polynomial of t with respect to the ideal $< g_r(t,\omega), g_i(t,\omega) >$. Then, the problem in $\mathbb{R}^2$ is translated into SDC of $m(t)$, *i.e.*, to decide or to find the condition for $m(t)$ to have no roots in $[0,+\infty)$, and also into a supplemental work, to decide or to find the condition for no real ω to satisfy $g_r(\tau,\omega) = g_i(\tau,\omega) = 0$ when $m(t)$ happens to have a root $\tau \in [0,+\infty)$ [2)].

5. SDC is solved by Sturm-Habicht sequence computation.

The 3rd problem is nothing but an usual stability problem if the translation of D into $\mathbb{R}_{<0} \times \mathbb{R}$ is obtained by a linear fractional transformation. The two examples, in [3] are just such ones, although it was described in terms of the coordinates on the Euclidean plane. As a conclusion, the 4th step is extraneous; 2nd and 3rd steps together can better be replaced by a linear fractional transformation; then subsequently, the last step can be replaced by the Routh-Hurwitz test, or better be done by the Strelitz test.

We mention that "gain margin" problem is solved by the Strelitz condition as well. Research into more problems will find such problems which can better be dealt with by the Strelitz test.

8 Conclusion

Applying the Strelitz test for parametric polynomials and formulating a typical stability problem in the Strelitz condition are presented. Although its applicability is limited to problems which are translated into stability problems, the simple structure of the algorithm is preferable for symbolic computation.

Finally, the author would like to mention that we can efficiently compute the (second) largest sum-of-roots polynomial defined by

$$G_L(z) = \prod_{i_1}(z-\alpha_{i_1}) \cdot \prod_{i_1<i_2}(z-(\alpha_{i_1}+\alpha_{i_2}))\cdots \prod_{i_1<i_2<\cdots<i_n}(z-(\alpha_{i_1}+\alpha_{i_2}+\cdots\alpha_{i_n})),$$

if we apply a similar method to compute the (small) sum-of-roots polynomial for each factor of G_L separately.

References

[1] Anai, H. and Hara, S.: A Robust Control System Design by a Special Quantifier Elimination Methods using a Sturm-Habicht sequence, *Proc. IMACS-ACA'99*, El Escorial(Spain), 1999.

[2] Anai, H. and Hara, S.: Fixed-structure robust controller synthesis based on sign definite condition by a special quantifier elimination, *Proc. ACC2000*, Chicago (USA), 2000, 1312-1316.

[3] Kimura, T. and Hara, S.: A robust control system design by a parameter space approach based on sign definite condition, *Proc. KACC'91*, Seoul(Korea), 1991, 1533-1538.

[4] Kimura, T. and Hara, S.: Robust Control Analysis Considering Real Parametric Perturbations Based on Sign Definite Conditions, *Proc. IFAC'93, Vol. 1*, Sydney(Australia), 1993, 37-40.

[5] Strelitz, SH.: On the Routh-Hurwitz Problem, *American Mathematical Monthly*, **84**(7), 1977, 542-544.

[2)] This supplemental work is not an easy task especially if $f(z)$ has symbolic parameters.