RESIDUAL GENERATION FOR FAULT DIAGNOSIS OF SYSTEMS DESCRIBED BY GENERAL LINEAR DIFFERENTIAL-ALGEBRAIC EQUATIONS

Erik Frisk and Mattias Nyberg

Dept. of Electrical Engineering, Linköping University,
SE-581 83 Linköping, Sweden
Phone: +46 13285714, Email: frisk@isy.liu.se, matny@isy.liu.se

Abstract: An algorithm for designing linear residual generators is presented. A main result is that the algorithm is able to design residual generators for *any* model described by general linear differential-algebraic equations. Previous algorithms have been restricted to models on transfer function, state space, or descriptor form. The presented algorithm can handle all these types of models, but also more general model descriptions not handled by previous algorithms. This is important since more general linear differential-algebraic equations models are often the result of object-oriented equation-based modeling. Also included is an extension to the stochastic case. Since the algorithm is based on well studied algebraic manipulations of polynomial matrices, it will have good numerical performance. The algorithm is demonstrated on a linearized model of a three-link robot.

Keywords: residual generation, parity relation, analytical redundancy relation, consistency relation, fault diagnosis, differential-algebraic system, polynomial matrix

1. INTRODUCTION

Fault diagnosis consists of detecting and isolating faults acting on a process. In many methods, e.g. *structured residuals* (Gertler, 1998), the concept of residuals play a central role. Commonly, a set of residuals is used, where different subsets of residuals are sensitive to different subsets of faults and in this way isolation between several faults is achieved.

In this paper, residual generation based on linear models is considered. Previous works on linear residual generation has primarily dealt with models on transfer function form (Gertler, 1998; Viswanadham *et al.*, 1987), state space form (Frisk and Nyberg, 2001; Chow and Willsky, 1984; Chen and Patton, 1999), or descriptor form (Hou, 2000; Kratz *et al.*, 1995). In contrast, the key result of this paper is that we consider all models that are described by any set of linear differential-algebraic equations. The three previous cases become special cases of this more general model class.

An algorithm for constructing residual generators for this more general model class will be presented. For the common special cases of models described by transfer functions, state space, or descriptor form, the algorithm becomes especially straightforward. The algorithm is based on polynomial matrix algebra for which efficient computational tools are commercially available.

The ability to handle this more general class of models is also beneficial from a modeling perspective. This is because models not on state-space, descriptor, or transfer function form are often the result when using a physically based object-oriented modeling approach (Åström *et al.*, 1998). Typically, different components are modeled by separate models and then connected together via algebraic constraints. Each component is modeled by specifying physical laws that connect internal variables to each other. Such a modeling strategy will naturally produce models that contain algebraic constraints, and also derivatives of higher order than one. Thus, these kinds of models are often not in transfer function, state space, or descriptor form.

In contrast to control, fault diagnosis from a principle point of view, does not distinguish between process inputs and outputs. The only thing that should matter in fault diagnosis is whether a variable is unknown or known. This is the point of view taken in this paper. All known signals, e.g. inputs and outputs, are collected in a vector $z(t)$ and all unknown signals, e.g. internal states and disturbances, are collected in a vector $x(t)$. Also this issue is closely related to object-oriented modeling. The reason is that object-oriented modeling languages (Mattson *et al.*, 1998; Piela *et al.*, 1991) are often equation based, meaning that no notion of causality is contained in the models, e.g. it is not specified if a signal is an input or an output to a component.

The paper starts by in Section 2 more formally describe the residual-generation problem for systems described by linear differential-algebraic equations. Section 3 contains the description of the design algorithm including motivating examples and proofs. A general detectability condition is then presented in Section 4. Section 5 discusses an extension of the basic algorithm to cover also the stochastic case, where the system is also affected by stochastic signals. The specific cases transfer functions, state space, or descriptor form, are discussed in Section 6. Section 7 finally illustrates the algorithm on a linearized model of a three-link robot.

2. PROBLEM FORMULATION

This paper only deals with the time-continuous case, and we will use p to denote the differentiation operator. A similar algorithm for the time-discrete case can also be constructed by changing p to the time-shift operator q. In the presentation that follows, both the operator p and the Laplace variable s will be used and care has been devoted to use the two correctly which is the reason for repeated switching between the two.

It is assumed that the model of the system to be diagnosed is linear and on the general form

$$H(p)x(t) + L(p)z(t) + F(p)f(t) = 0 \qquad (1)$$

where $x(t)$ is a any signal vector, $z(t)$ is the observed vector, $f(t)$ the vector of fault signals, and $H(p)$, $L(p)$, and $F(p)$ arbitrary polynomial matrices in the operator p. The signal vector $x(t)$ contains all unknown signals which includes internal system states and unknown inputs such as disturbances. We assume that the goal is to detect faults included in the $f(t)$-vector, while faults that will be decoupled are included among the unknown signals in $x(t)$. The signal vector $z(t)$ contains all known signals, such as sensor and actuator values.

The general form (1) contains the more common cases transfer function, state-space system, and descriptor systems as special cases. All these special cases will be exemplified in Section 6.

2.1 *Residual Generators and Consistency Relations*

As said in the introduction, the goal is to design linear residual generators, which we define as follows:

Definition 1. (Residual Generator). The rational polynomial vector $R(p)$, of dimension $1 \times n_z$, is a *residual generator* if the filter $R(p)$ is proper and for all signals $z(t)$, and any signal $r(t)$ fulfilling $r(t) = R(p)z(t)$, it holds that

$$\exists x(t) \; . \; H(p)x(t) + L(p)z(t) = 0 \Longrightarrow \lim_{t\to\infty} r(t) = 0$$

The residual generator $R(p)$ can be written as

$$R(p) = \frac{K(p)}{c(p)}$$

where $K(p)$ is a polynomial vector and $c(p)$ is a scalar polynomial. If $c(s)$ has only stable zeros and it holds that $K(p)z(t) \rightarrow 0$ for each $z(t)$ that satisfies the model, then $R(p)$ will be a residual generator. It can also be realized that if $R(p)$ is a residual generator, it must hold that $K(p)z(t) \rightarrow 0$ for each $z(t)$ that satisfies the model. This is the motivation to suggest the following design strategy for residual generators:

(1) Find a $K(p)$ for which it holds that $K(p)z(t) \rightarrow 0$ for each $z(t)$ that satisfies the model.
(2) Find a stable $c(s)$ that together with $K(p)$ makes $R(p)$ proper.

The difficult part is of course step 1, i.e. to find $K(p)$. The rest of the paper will therefore be devoted to this problem. To find $K(p)$ is equivalent to finding so called *consistency relations* or *parity equations*. We define consistency relation as follows:

Definition 2. (Consistency Relation). The scalar expression $K(p)z(t)\overset{\infty}{=}0$, where $K(p)$ is polynomial, is a *consistency relation* if for all signals $z(t)$ it holds that

$$\exists x(t) \; . \; H(p)x(t) + L(p)z(t) = 0 \Rightarrow \lim_{t\to\infty} K(p)z(t) = 0$$

In the next section we will present a design algorithm that for any system described by the general model (1), produces consistency relations $K(p)z(t)\overset{\infty}{=}0$.

3. GENERAL DESIGN ALGORITHM FOR CONSISTENCY RELATIONS

The design algorithm below is based on polynomial matrix algebra, e.g. see (Kailath, 1980). The basic ideas are similar to the method presented in (Frisk and Nyberg, 2001), but here we consider a more general class of models. Given is a model on the general form (1). The algorithm will then find a $Q_r(p)$, and possible consistency relations can be obtained as $\gamma(p)Q_r(p)z(t)\overset{\infty}{=}0$ where $\gamma(p)$ is any polynomial vector.

Design Algorithm

(1) If $[H(p)\ L(p)]$ has not full row rank, premultiply $[H(p)\ L(p)]$ with unimodular matrices to obtain

$$\begin{bmatrix} \tilde{H}(p) & \tilde{L}(p) \\ 0 & 0 \end{bmatrix}$$

where $[\tilde{H}(p)\ \tilde{L}(p)]$ has full row rank. This can for example be performed by taking $[H(p)\ L(p)]$ to column Hermite form.
(2) Find a minimal polynomial basis for the left null-space of $\tilde{H}(p)$, denoted $N_{\tilde{H}}(p)$.
(3) Form $Q(p) = N_{\tilde{H}}(p)L(p)$.
(4) Factorize $Q(p)$ as $Q(p) = Q_{stab}(p)Q_r(p)$, where $Q_{stab}(s)$ is square and full rank, and $Q_r(s)$ has no strictly stable zeros and $Q_{stab}(s)$ has no instable zeros (i.e. no zeros s_0 where $Re\{s_0\} \geq 0$).

Theorem 1. Assume that the design algorithm above is used. Then

(a) $Q_r(p)$ has full rank
(b) For all polynomial vectors $\gamma(p)$, $\gamma(p)Q_r(p)z(t)\overset{\infty}{=}0$ will be a consistency relation.

All lemmas needed to prove Theorem 1 will be given later in Section 3.3. The reason for this is to be able to focus on principles and main ideas and return to the technicalities later.

The proof of Theorem 1 follows:

PROOF. It is easily proven that a matrix $[A(p)\ B(p)]$ has full row rank if and only if $N_A(p)B(p)$ has full row rank. Thus, $Q(p)$ will have full row rank. Since $Q(p) = Q_{stab}(p)Q_r(p)$ and $Q_{stab}(p)$ is square and full rank, it is obvious that also $Q_r(p)$ has full rank which proves the a)-part. The b)-part follows directly from Lemma 3. □

The algorithm can be easily implemented in Matlab using (*The Polynomial Toolbox 2.5*, 2001), which contains functions for each of the steps in the algorithm.

Remark 1. Although not a part of the theorem, from Lemma 4 it also follows that $Q_r(s)$ is a polynomial basis for all $f(p)$ such that $f(p)z(t)\overset{\infty}{=}0$ is a consistency relation.

Remark 2. In some cases it may be desirable only to obtain consistency relations for which it also holds that

$$\exists x(t) \; . \; H(p)x(t) + L(p)z(t) = 0 \Rightarrow K(p)z(t) = 0 \qquad (2)$$

By removing step 4 from the algorithm, only consistency relations fulfilling (2) are obtained.

3.1 *Motivation of the Design Algorithm*

The first step in the algorithm consists of removing dependent equations from the model. If this is not done, $Q(p) = N_{\tilde{H}}(p)L(p)$, and therefore also $Q_r(s)$ will not have full rank. That is, to fulfill part (a) in Theorem 1, dependent rows in $[H(p)\ L(p)]$ must be removed. As long as this is done using only unimodular matrices no solutions will be lost since unimodular matrices are polynomially invertible.

We will now present two examples motivating the 4:th step of the algorithm concerning zeros of $Q(s)$. If the original model equations are given in state-space form, the origin of any zeros of $Q(s)$ is uncontrollable modes of the original state-space realization. Therefore, for most cases $Q(s)$ has no zeros. But for the case when $Q(s)$ has zeros, stable and unstable zeros need to be handled differently. First consider the model equation:

$$(p+1)z_1(t) = (p+1)z_2(t) \tag{3}$$

for which $Q(s) = [s+1 \quad -(s+1)]$, i.e. $Q(s)$ has a stable zero at $s = -1$ which gives $Q_r(s) = [1 \quad -1]$. Then, according to Theorem 1, $z_1(t) - z_2(t) \overset{\infty}{=} 0$ is a consistency relation. It is easily proven that this is a consistency relation by solving the differential equation (3) which gives that

$$z_1(t) - z_2(t) = e^{-t}(z_1(0) - z_2(0)) \tag{4}$$

which clearly satisfies Definition 2. This example shows how factoring out a stable polynomial from $Q(s)$ is safe.

Assume now, contradictory to the algorithm, that for the model (3), $(s+1)$ is not factored out from $Q(s)$. That means that the output from the algorithm will be $Q_r(s) = [s+1 \quad -(s+1)]$. We saw in (4) that $z_1(t) - z_2(t) \overset{\infty}{=} 0$ is a consistency relation. The problem is that this consistency relation is no longer possible to obtain from the matrix $Q_r(s)$, i.e. there exists no polynomial $\gamma(p)$ such that $\gamma(p)Q_r(p) = [1 \quad -1]$. This is the motivation to why we in step 4 of the algorithm, factor out all stable parts of $Q(p)$.

To see why it is not safe to factor out an unstable polynomial, consider the model

$$(p-1)z_1(t) = (p-1)z_2(t) \tag{5}$$

For this system, $Q(s) = [s-1 \quad -(s-1)]$. Factoring out $s-1$, contrary to what is stated in step 4 of the algorithm, results in a relation $z_1(t) - z_2(t) \overset{\infty}{=} 0$. This is however not a consistency relation. This can be seen by solving the differential equation (5), which gives

$$z_1(t) - z_2(t) = e^{t}(z_1(0) - z_2(0))$$

which clearly does not satisfy Definition 2 if $z_1(0) \neq z_2(0)$.

The discussion above have motivated all the steps of the algorithm presented. To obtain a stable residual generator, we have seen that we can not factor out unstable parts from $Q(s)$. To obtain a $Q_r(s)$ such that also the most simple consistency relations can be written $\gamma(p)Q_r(p)z(t) \overset{\infty}{=} 0$, we have seen that stable parts of $Q(s)$ must be factored out.

3.2 *Simplified Algorithm*

In many cases, matrix $[H(p) \; L(p)]$ will already be full rank and irreducible. Thus step 1 is not needed since $[H(p) \; L(p)]$ is already full rank. Consider now the following result (included without proof due to space limitations):

Lemma 1. If $[H(s) \; L(s)]$ is full row-rank for all s, then $N_H(s)L(s)$ has full row-rank for all s.

This lemma implies that $Q(s)$ will have full rank for all s, i.e. $Q(s)$ has no zeros and step 4 becomes unnecessary. Thus, when $[H(p) \; L(p)]$ is irreducible, a simplified algorithm consisting of only steps 2 and 3 can be used.

For most systems, it actually holds that $[H(p) \; L(p)]$ is irreducible. Section 6 will give quite mild conditions on when the simplified design algorithm can be used together with models on transfer function, state space, or descriptor form.

3.3 *Theoretical Justifications of the Algorithm*

The following lemma is used in step 2 and 3 of the algorithm.

Lemma 2. Assume the double-sided Laplace transform of $N_A(p)B(p)z(t)$ exists. Then

$$\exists x(t) \; . \; A(p)x(t) + B(p)z(t) = 0 \tag{6}$$

if and only if

$$N_A(p)B(p)z(t) = 0 \tag{7}$$

PROOF. The only-if part follows by premultiplying (6) with $N_A(p)$. For the if-part, apply first the double-sided Laplace transform to (7). This results in $N_A(s)B(s)z(s) = 0$ which is equivalent to that $\exists x(s). \; A(s)x(s) + B(s)z(s) = 0$. Now the application of the inverse Laplace transform results in (6). □

The following lemma is used to prove part (b) in Theorem 1.

Lemma 3. All rows in $Q_r(p)z(t) \overset{\infty}{=} 0$ are consistency relations.

PROOF. From Lemma 2, we know that $Q(p)z(t) = 0$. Applying the single-sided Laplace transform results in

$$Q(s)z(s) + \overline{Q}(s)\bar{z}_0 = 0$$

where $\bar{z}_0 = [z^{(d-1)}(0), \; z^{(d-2)}(0), \; \ldots \; \dot{z}(0), \; z(0)]^T$ and $d = \deg(Q(s))$. Further on, we also know that

$$Q_{stab}(s)Q_r(s)z(s) + \overline{Q}(s)\bar{z}_0 = 0$$

Since $Q_{stab}(s)$ is square and full rank, its inverse exists and thus

$$Q_r(s)z(s) + Q_{stab}^{-1}(s)\overline{Q}(s)\bar{z}_0 = 0$$

The inverse Laplace transform of this expression is

$$Q_r(p)z(p) + \Delta(t) + g(t) = 0$$

where $\Delta(t)$ contains only Dirac functions and some orders of derivatives of Dirac functions, and $g(t) \to 0$ when $t \to \infty$. That is, $Q_r(p)z(p) \to 0$ when $t \to \infty$ and all rows in $Q_r(p)z(t) \overset{\infty}{=} 0$ are therefore consistency relations. □

The following lemma is used in Remark 1, and to prove the detectability condition and the stochastic extension in Sections 5 and 4 respectively.

Lemma 4. If $f(p)z(t) = 0$ is a consistency relation, then there exists a rational $\phi(p)$ such that $f(p) = \phi(p)Q_r(p)$.

PROOF. Define $M(s)$ to be a matrix whose columns form a polynomial basis for the right null-space to $Q_r(s)$. Let then $f(s)$ be written as

$$f(s) = \alpha(s)Q_r(s) + \beta(s)M^T(s)$$

Multiplying from the right with $M(s)$ implies

$$\begin{aligned} f(s)M(s) =& \alpha(s)Q_r(s)M(s) + \beta(s)M^T(s)M(s) = \\ =& \beta(s)M^T(s)M(s) \end{aligned} \tag{8}$$

The inverse Laplace transform of $Q_r(s)M(s) = 0$ becomes $Q_r(p)M(p)\delta(t) = 0$. By multiplying from the left with $Q_{stab}(p)$ we obtain $Q(p)M(p)\delta(t) = 0$.

Now consider the signal $y(t) = \sum_{i=0}^{\infty} M(p)\delta(t)$. It holds that $Q(p)y(t) = 0$. Since $f(p)z(t) = 0$ is a consistency relation, it must hold that $f(p)y(t) \to 0$ when $t \to \infty$. The only possibility to fulfill this is that $f(p)M(p)\delta(t) = 0$ for $t \geq 0$. The Laplace transform of this expression becomes $f(s)M(s) = 0$.

Equation (8) can now be written

$$0 = f(s)M(s) = \beta(s)M^T(s)M(s)$$

Since $M^T(s)M(s)$ has full rank, this implies that $\beta(s) = 0$, which proves the theorem. □

4. DETECTABILITY CONDITION

A fault is said to be detectable in a system if there exists any residual generator that is sensitive to that fault. With the presented theory it is possible to state a detectability condition for systems described by the general model form (1).

Theorem 2. There exists a residual generator $R(p)$ producing the residual $r(t)$ such that the transfer function $G_{rf}(s)$ from the fault $f(t)$ to $r(t)$ is non-zero if and only if

$$\operatorname{Rank}\left[H(s)\ F(s)\right] > \operatorname{Rank} H(s) \tag{9}$$

PROOF. If (9) is fulfilled, it holds that $N_H(p)L(p)z(t) = -N_H(p)F(p)f(t)$ and $N_H(p)F(p) \neq 0$. Choosing any row in $N_H(p)L(p)z(t)$, and adding stable poles, will result in a $G_{rf}(s) \neq 0$.

According to Lemma 4, all consistency relations can be written as $\phi(p)N_H(p)L(p)$ where $\phi(p)$ is rational. This also means that all possible residual generators can be written as $\phi'(p)N_H(p)L(p)$ with $\phi'(p)$ rational. If (9) is not fulfilled, it holds that $N_H(p)F(p) = 0$ and also that $G_{rf}(p) = \phi'(p)N_H(p)F(p) = 0$. Thus there cannot exist any residual generator with $G_{rf}(s) \neq 0$. □

5. STOCHASTIC CASE

To show the generality of the design methodology, we will now present a small extension to cover also residual generation for stochastic differential-algebraic models. To the model (1), add a stochastic term, i.e.

$$H(p)x(t) + L(p)z(t) + V(p)n(t) + F(p)f(t) = 0$$

where $n(t)$ is assumed to be white noise. The residual generation problem for stochastic systems is addressed in (Nikoukhah, 1994) from which the following definition is taken.

Definition 3. (Innovation filter). A finite-dimensional linear time-invariant filter $R(p)$ is called an innovation filter if it is stable with the least number of outputs such that, in the absence of failure,

(1) its output $r = R(p)z(t)$ is zero-mean, white and decoupled from $u(t)$ and $d(t)$,
(2) if $R'(s)$ is any finite-dimensional linear time-invariant system such that $r'(t) = R'(s)z(t)$ is decoupled from $u(t)$ and $d(t)$, then there exists a linear system $T(p)$ such that $R'(s) = T(s)R(s)$.

Assumption. From now on it is assumed that perfect decoupling of both the noise n and unknown signals x is not possible. Also, for sake of brevity, it is assumed that the conditions for the simplified algorithm is fulfilled, i.e. $[H(s)\ L(s)]$ is assumed to be full-rank and irreducible.

Let $R(p)$ be a residual generator. Lemma 4 then gives that there exists a rational $\varphi(s)$ such that $R(s) = \varphi(s)N_H(s)L(s)$. Straightforward calculations give that the spectrum of the residual can be written

$$\Phi_r(s) = \varphi(s)N_H(s)V(s)V^T(-s)N_H^T(-s)\varphi^T(-s) \tag{10}$$

Before the main result on innovation filter design is presented, we introduce the notation

$$W(s) = N_H(s)V(s)V^T(-s)N_H^T(-s)$$

Theorem 3. Let $R(s) = P^{-1}(s)N_H(s)L(s)$ where $P(s)$ is a spectral co-factor of $W(s)$. Then it holds that $R(p)$ is an innovation filter if $R(p)$ is stable and proper. If $R(p)$ is not an innovation filter, no innovation filter exists.

PROOF. Since it was assumed that perfect decoupling of the noise was not possible, $W(s)$ will have full rank. Then if $P(s)$ is a J-spectral co-factor of $W(s)$, $W(s) = P(s)P^T(-s)$. Now, if $R(s) = P^{-1}(s)N_H(s)L(s)$ is a residual generator, equation (10) directly gives that the spectrum of $r(t)$ is constant since $\varphi(s) = P^{-1}(s)$. Thus, $R(s)$ is an innovation filter if $R(s)$ is also stable and proper.

Assume $R(s)$ is not stable and proper and that $R'(s)$ is an innovation filter. Since $R'(s)$ is a disturbance decoupling residual generator it can be written $R'(s) = \varphi'(s)N_H(s)L(s)$. Also, since $R'(s)$ produces white residuals, Equation (10) gives that there exists an $\eta(s)$ such that $\varphi'(s) = \eta(s)P^{-1}(s)$ and $\eta(s)$ satisfies $\eta(s)\eta^T(-s)$ constant. Since this holds for all s and also in the limit when $s \to \infty$. This gives that both $\eta(s)$ and $\eta^{-1}(s)$ is proper. Also, since $R'(s)$ is stable, so is $\eta(s)$. But $R(s) = \eta^{-1}(s)R'(s)$ which is a contradiction to $R(s)$ not being an innovation filter. Thus, it is proven that if $R(s)$ is not an innovation filter, no innovation filter exists. □

For more details on innovation filter design using polynomial methods, see (Frisk, 2001).

6. SPECIAL CASES

In this section, three common special cases of model descriptions are addressed: models on state-space form, descriptor form, and transfer function form. In particular it will be shown under what conditions the simplified algorithm from Section 3.2 can be used.

6.1 *State Space Model*

First, consider models given by state-space equations on the form:

$$\dot{x}(t) = Ax(t) + B_u u(t) + B_d d(t) + B_f f(t) \tag{11a}$$
$$y(t) = Cx(t) + D_u u(t) + D_d d(t) + D_f f(t) \tag{11b}$$

To state the state-space description on the general form (1), matrices $H(p)$, $L(p)$, and $H(p)$ becomes:

$$H(p) = \begin{bmatrix} C & D_d \\ -(pI - A) & B_d \end{bmatrix},\ L(p) = \begin{bmatrix} -I & D_u \\ 0 & B_u \end{bmatrix} \tag{12a}$$
$$F(p) = \begin{bmatrix} D_f \\ B_f \end{bmatrix} \tag{12b}$$

It will now be shown that mild controllability conditions on (11) is sufficient to be able to use the simplified design algorithm. The first step in the design algorithm is immediate; a state-space description always render full row-rank $[H(s)\ L(s)]$. This can be seen directly in

$$[H(s)\ L(s)] = \begin{bmatrix} C & D_d & -I & D_u \\ -(sI - A) & B_d & 0 & B_u \end{bmatrix} \tag{13}$$

which is full row-rank.

Now, assume that the pair $\{A, [B_u\ B_d]\}$ is controllable. Then the PBH test gives that $[-(sI-A)\ B_d\ B_u]$ has full row-rank for all s which in turn gives that

$$[H(s)\ L(s)] = \begin{bmatrix} C & D_d & -I & D_u \\ -(sI-A) & B_d & 0 & B_u \end{bmatrix}$$

has full row-rank for all s. Lemma 1 then gives that $Q(s)$ computed in step 3 of the design algorithm has no zeros and no factorization in step 4 is necessary (or even possible). Thus, the controllability assumption is sufficient for the simplified design algorithm, from Section 3.2, to be valid. Further, in (Frisk and Nyberg, 2001) it is also proven that $Q(s)$ is with the controllability assumption also a *minimal polynomial basis*.

6.2 *Descriptor Form*

Another common special case is so called descriptor systems which can be modeled by equations on the form

$$E\dot{x}(t) = Ax(t) + B_u u(t) + B_d d(t) + B_f f(t) \quad (14a)$$
$$y(t) = Cx(t) + D_u u(t) + D_d d(t) + D_f f(t) \quad (14b)$$

The only difference from the state-space form is the matrix E which can be singular or even non-square. Collecting the model equations on form (1) is similar to the state-space matrices (12) with the difference that $H(p)$ is given by

$$H(p) = \begin{bmatrix} C & D_d \\ -(pE-A) & B_d \end{bmatrix}$$

For this case, similar to the state-space case, assume R-controllability (Yip and Sincovec, 1981) from u and d. Then the matrix $[sE - A\ B_u\ B_d]$ will have full row-rank for all s (Yip and Sincovec, 1981) which, by similar reasoning as for the state-space case, implies that $[H(s)\ L(s)]$ is irreducible. Therefore, also in the descriptor case a mild controllability assumptions is sufficient for the simplified design algorithm to be valid.

6.3 *Transfer Function Model*

A third common model description is transfer functions:

$$y(t) = G_u(p)u(t) + G_d(p)d(t) + G_f(p)f(t)$$

where $G_u(s)$, $G_d(s)$, and $G_f(s)$ are rational functions in s. To get a polynomial description like in (1), derive a polynomial matrix fraction description (MFD), i.e. find polynomial matrices $D(s)$, $N_u(s)$, and $N_d(s)$ such that

$$D(s)[G_u(s)\ G_d(s)] = [N_u(s)\ N_d(s)]$$

Then, the fault-free transfer-function model can be stated on the form

$$N_d(p)d(t) + [-D(p)\ N_u(p)] \begin{bmatrix} y(t) \\ u(t) \end{bmatrix} = 0$$

If the MFD is irreducible, meaning that $D(s)$ and $[N_u(s)\ N_d(s)]$ are co-prime, similar reasoning as for the other two special cases gives that the simplified design algorithm is applicable.

7. ILLUSTRATIVE EXAMPLE

The example used to illustrate the design procedure is based on, but not identical to, a descriptor model described in (Hou and Müller, 1996) and used for diagnosis in (Hou, 2000). The model is an idealized description of a three-link planar manipulator/robot. The process works by moving the end effector repeatedly, e.g. cleaning a facade. The manipulator is equipped with three actuators that can apply torques at all three joints. Three sensors is used measuring the height of the end effector, the contact force in the x direction, and a tracking signal. The fault-free model is stated on descriptor form in (Hou and Müller, 1996) where also numerical values for model parameters can be found. The model has 8 states: Cartesian coordinates of the end effector (3 states), derivatives of the Cartesian coordinates (3 states), and two Lagrangian multipliers.

In Section 1, the benefits and consequences of component and object-oriented based modeling is discussed. To illustrate these matters, an additional sensor and two fault models are added. The first state x_1 is the height of the end effector. Now, assume that an accelerometer is also attached to the end-effector, thus $\ddot{x}_1(t)$ is measured. Also, two faults are modeled, one fault acting on the first actuator and a sensor fault on the first sensor. Collecting the original model, the additional sensor, and the fault models results in the following model description:

$$pEx(t) = Ax(t) + B_u u(t) + B_1 f_1(t) \quad (15a)$$
$$y_1(t) = x_2(t) + f_2(t) \quad (15b)$$
$$y_2(t) = x_7(t) \quad (15c)$$
$$y_3(t) = x_8(t) \quad (15d)$$
$$y_4(t) = p^2 x_1(t) \quad (15e)$$

where B_1 is equal to the first column in B_u. Note that the new sensor-equation (15e) was straightforward to introduce and no parts of the model equations had to be modified. In a state-space/descriptor setting, additional states would have been necessary all depending on the original system. This illustrates the modeling principles discussed in Section 1 regarding modularity, object-oriented, and component based modeling. If the model was to be used for control, perhaps a state-space/descriptor formulation would have been preferable, but for diagnosis applications this general class of models is equally useful.

Now, the model (15) can easily be stated on form (1) with unknown variables $x(t)$, and observables $z(t) = [y^T(t)\ u^T(t)]^T$. It is obvious that the model above is not on state-space form and it can easily be verified that it can not be transferred to state-space form either. The model is not on descriptor form either. Although it is possible to transfer the model to a descriptor form, it is not a trivial operation and with the proposed design algorithm, model equations (15) can be used directly without any additional transformations.

In this example, two residual generators $R_1(p)$ and $R_2(p)$ are to be designed, used for detection and isolation of the two faults f_1 and f_2. In the residual generators fault f_1 and f_2 should be decoupled in $R_1(p)$ and $R_2(p)$ respectively, i.e. residual r_1 should only react to fault f_2 and vice versa. For both designs, by computing rank and greatest left divisor, matrix $[H(s)\ L(s)]$ is seen to have full row-rank and no zeros. This fact, together with Lemma 1 gives that the simplified algorithm from Section 3.2 is applicable. Full Matlab code for design of a residual generator is given below. The code uses two toolboxes, (*The Polynomial Toolbox 2.5, 2001)* and control toolbox. Note that all operations are standard operations in these toolboxes and no diagnosis specific algorithms need to be developed and/or written. The model matrices $H(s)$ and $L(s)$ are assumed to be defined and the resulting residual generator $R(s)$ is in state-space form (an LTI object in control toolbox).

```
1  Q = null(H.').'*L;
2  gamma = [1 1 1];
3  c = (s+2)^3;
4  [Ra,Rb,Rc,Rd] = lmf2ss(gamma*Q,c);
5  R = ss(Ra,Rb,Rc,Rd);
```

All steps are self explanatory except maybe line 4 which transforms a left MFD to a state-space description. For both designs, $Q(s)$ is a polynomial matrix of degree 2 with 3 rows, i.e. there exists exactly 3 linearly independent consistency relations where f_1 respectively f_2 are decoupled. The two design choices that exists in the design are the choice of $\gamma(s)$ (property b in Theorem 1) and the choice of denominator $c(s)$ such that the residual generator is stable and realizable on state-space form. Here these choices are done ad-hoc since no additional design specifications are given. The row vector $\gamma(s)$ is selected to use all consistency relations with $\gamma(s) = [1\ 1\ 1]$ and all poles of the residual generator, i.e. zeros of $c(s)$, are placed in $s = -2$. For the residual generator to be proper, it turns out that the denominator polynomial need to be of at least degree two, here a third order denominator polynomial is used.

A first step when evaluating the designs is to validate that the desired decoupling properties are satisfied. Computing the sizes of the transfer functions from the decoupled faults to residuals in Matlab we get $\|G_{r_1 f_1}(s)\|_\infty = 0$ and $\|G_{r_2 f_2}(s)\|_\infty = 0$. This verifies that faults f_1 and f_2 are decoupled in r_1 and r_2 respectively according to design specifications. Figure 1 shows transfer functions from faults to the residuals, and it is clear that each residual will react according to design specifications. Also, the cut-off frequency is, as specified by the denominator polynomial, approximately 2 rad/s. One important thing that has not been

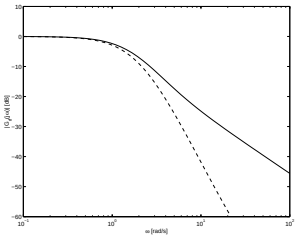

Fig. 1. Transfer function from fault f_2 to residual r_1 (solid line), and from fault f_1 to residual r_2 (dashed line).

mentioned is numerical performance of the design procedure. This is not pursued further here, interested readers is referred to previous works on both state-space and descriptor models that shows good numerical performance (compared to other approaches) on example models (Frisk and Nyberg, 2001).

8. CONCLUSIONS

An algorithm for designing linear consistency relations has been presented. It is also shown how consistency relations can be used to form residual generators. A main result is that the algorithm designs consistency relations for any model described by general linear differential-algebraic equations. Previous algorithms have been restricted to models on transfer function, state space, or descriptor form. In other words, the presented algorithm can handle all types of models handled by earlier algorithms, but also a more general class of models not handled by previous algorithms. To be able to handle general linear differential-algebraic models is important since they are often the result of object-oriented equation-based modeling.

Also included is an extension to the stochastic case, i.e. when the system is affected by noise with known distributions. Here the goal becomes to find innovation filters instead of consistency relations.

For most models, the algorithm becomes very simple; it basically consists of one Matlab-command. Since the algorithm is based on well studied algebraic manipulations of polynomial matrices, the suggested Matlab-implementation will also have good numerical performance.

9. REFERENCES

Chen, J. and R. J Patton (1999). *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Kluwer Academic Publishers.

Chow, E.Y. and A.S. Willsky (1984). Analytical redundancy and the design of robust failure detection systems. *IEEE Trans. on Automatic Control* **29**(7), 603–614.

Frisk, E. (2001). Residual generation in linear stochastic systems - a, polynomial approach. In: *Proc. IEEE Conference on Decision and Control*. Orlando, USA.

Frisk, E. and M. Nyberg (2001). A minimal polynomial basis solution to residual generation for fault diagnosis in linear systems. *Automatica* **37**, 1417–1424.

Gertler, J. (1998). *Fault Detection and Diagnosis in Engineering Systems*. Marcel Dekker.

Hou, M. (2000). *Fault detection and isolation for descriptor systems*. Chap. 5. in Patton *et al.* (2000).

Hou, M. and P.C. Müller (1996). Tracking control for a class of descriptor systems. In: *Proc. 13th IFAC World Congress*. San Fransisco, USA. pp. 109–114.

Kailath, Thomas (1980). *Linear Systems*. Prentice-Hall.

Kratz, F., S. Bousghiri and W. Nuninger (1995). A finite memory observer structure of continuous descriptor systems. In: *Proc. American Control Conference*. Seattle, USA. pp. 3900–3904.

Mattson, S.E., H. Elmqvist and M. Otter (1998). Physical system modeling with modelica. *Control Engineering Practice* **6**(4), 501–510.

Nikoukhah, R. (1994). Innovations generation in the presence of unknown inputs: Application to robust failure detection. *Automatica* **30**(12), 1851–1867.

Patton, R.J., Frank, P.M. and Clark, R.N., Eds.) (2000). *Issues of Fault Diagnosis for Dynamic Systems*. Springer.

Piela, P., P.T. Epperly, K. Westerberg and A. Westerberg (1991). ASCEND:an object-oriented computer environment for modeling and analysis: the modeling language. *Computers and Chemical Engineering* **15**(1), 53–72.

Åström, K.J., H. Elmqvist and S.E. Mattson (1998). Evolution of continuous-time modeling and simulation. In: *Proc. 12:th European Simulation Multiconference ESM'98*. Manchester, UK.

The Polynomial Toolbox 2.5 (2001). Polyx, Czech Republic. URL: http://www.polyx.com.

Viswanadham, N., J.H. Taylor and E.C. Luce (1987). A frequency-domain approach to failure detection and isolation with application to GE-21 turbine engine control systems. *Control - Theory and advanced technology* **3**(1), 45–72.

Yip, E.L. and R.F. Sincovec (1981). Solvability, controllability, and observability of continuous descriptor systems. *IEEE Trans. on Automatic Control* **26**(2), 702–707.