A Fault Detection Toolbox for MATLAB

A. Varga

***Abstract*— The recently developed FAULT DETECTION Toolbox for MATLAB is described. The new toolbox provides a comprehensive set of high level *m*-functions to support the design of residual generation filters using reliable numerical algorithms recently developed by the author. The basic computational layer is formed by the DESCRIPTOR SYSTEMS Toolbox which contains all necessary tools to solve the underlying numerical problems. The *m*-functions based user interfaces ensure user-friendliness in operating with the functions of this toolbox via an object oriented approach.**

I. INTRODUCTION

The design of *fault detection* (FD) filters able to detect discrepancies between normal and erroneous plant operations has been an active area of research since decades (see for example the recent monographs [1], [2], [3] and hundreds of references cited therein). The need to address properly the numerical issues encountered in designing fault detection filters has been already recognized by Chen and Patton [2, p.219]. Among methods traditionally used to design fault detection filters we mention techniques based on unknown-input observers, parity equations, eigenstructure assignment, polynomial approaches, $\mathcal{H}_2$- or $\mathcal{H}_\infty$-optimal filter synthesis techniques. However, in spite of many computational approaches proposed in the literature, until recently there were no general and numerically reliable algorithms available able to address the underlying computational aspects for large dimensional systems. The existing lack of dedicated robust software tools to design FD-filters is a consequence of the existing delicate situation in algorithmic field.

A systematic research in developing numerically reliable algorithms for the design of linear FD-filters has been conducted by the author in the recent years leading to a fairly complete collection of methods able to solve various FD-problems in the most general setting [4], [5], [6]. The proposed methods have several common appealing properties: (1) they are applicable to both standard and generalized (descriptor) systems; (2) the design of appropriate filter dynamics specification is part of the problem; (3) least-order filters design problems can be addressed; (4) no technical conditions present (i.e., if a problem has a solution, then this solution can be computed). The newly developed FD-filter design methods rely on sophisticated numerical linear algebra algorithms for which robust software implementations are available in the DESCRIPTOR SYSTEMS Toolbox developed over several years by the author [7] (see [8] for a description of the last version).

A. Varga is with the German Aerospace Center, DLR - Oberpfaffenhofen, Institute of Robotics and Mechatronics, D-82234 Wessling, Germany; Email: andras.varga@dlr.de

In this paper we present the FAULT DETECTION Toolbox for MATLAB, a dedicated CACSD software which implements the new generation of numerically reliable algorithms [4], [5], [6] using the unique linear algebra tools available in the DESCRIPTOR SYSTEMS Toolbox. The new FAULT DETECTION Toolbox is primarily intended to provide the basic functionality for designing various linear fault detection filters, by covering the existing main deterministic approaches [1], [2], [3]. This toolbox allows to solve challenging fault detection problems with large state dimensions in a numerically reliable way using flexible and user-friendly interfaces.

II. OBJECT-ORIENTED APPROACH

The toolbox is based on an object oriented approach to design FD-filters (also called residual generation filters). The primary system descriptions used by the toolbox are the generalized state-space systems (called also *descriptor systems*), which represent an extension of the state-space object used in the standard CONTROL TOOLBOX of MATLAB, and the Laplace- or Z-transformed rational *transfer-function matrices* (TFMs). In what follows we describe special features of the employed system objects and data structures.

A. System representations

The toolbox can handle continuous- or discrete-time descriptor systems of the form

$$\begin{array}{rcl} E\lambda x(t) & = & Ax(t)+B_u u(t)+B_f f(t)+B_d d(t) \\ y(t) & = & Cx(t)+D_u u(t)+D_f f(t)+D_d d(t) \end{array} \quad (1)$$

where $x(t) \in \mathbb{R}^n$ is the system state vector, $u(t) \in \mathbb{R}^{m_u}$ is the system control vector, $f(t) \in \mathbb{R}^{m_f}$ is the system fault vector, $d(t) \in \mathbb{R}^{m_d}$ is the system disturbance vector, $y(t) \in \mathbb{R}^p$ is the system output vector, and where $\lambda x(t) = \dot{x}(t)$ for a continuous-time system and $\lambda x(t) = x(t+T)$ for a discrete-time system with sampling period T. It is assumed that the linear pencil $A - \lambda E$ is regular.

Alternatively, the toolbox can handle input-output representations of the form

$$\mathbf{y}(\lambda) = G_u(\lambda)\mathbf{u}(\lambda) + G_f(\lambda)\mathbf{f}(\lambda) + G_d(\lambda)\mathbf{d}(\lambda), \quad (2)$$

where $\mathbf{y}(\lambda)$, $\mathbf{u}(\lambda)$, $\mathbf{f}(\lambda)$, and $\mathbf{d}(\lambda)$ are Laplace- or Z-transformed vectors of the corresponding vectors $y(t)$, $u(t)$, $f(t)$, and $d(t)$, respectively, and where $G_u(\lambda)$, $G_f(\lambda)$, and $G_d(\lambda)$ are rational TFMs from the corresponding plant inputs to outputs. According to the system type, $\lambda = s$ in the case of a continuous-time system or $\lambda = z$ in the case of a discrete-time system.

Note. To simplify the theoretical presentation, we will exclusively consider the input-output type representations via TFMs. However, all computational issues involving TFMs are addressed via equivalent state space representations.

B. Residual generator

A linear residual generator (or fault detection filter) processes the measurable system outputs $y(t)$ and control inputs $u(t)$ and generates the residual signals $r(t)$ which serve for decision making on the presence or absence of faults. The input-output form of this filter (also called *computational form* [1]) is

$$\mathbf{r}(\lambda) = Q(\lambda) \left[\begin{array}{c} \mathbf{y}(\lambda) \\ \mathbf{u}(\lambda) \end{array} \right] \tag{3}$$

where $Q(\lambda)$ is the TFM of the filter. For a physically realizable FD filter, $Q(\lambda)$ must be proper (i.e., only finite poles) and stable (i.e., only poles with negative real parts for a continuous-time system or with magnitudes less than one for a discrete-time system). The *McMillan degree* (or dynamic *order*) of $Q(\lambda)$ is the dimension of the state vector of a minimal state-space realization of $Q(\lambda)$. The dimension q of the residual vector $r(t)$ depends on the fault detection problem to be solved. For example, for the detection of faults a single residual could be sufficient, but for isolating a fault among several possible faults a set of residuals grouped into a vector is needed.

C. Residual generation system

The residual signal $r(t)$ in (3) generally depends via the system outputs $y(t)$ of all system inputs $u(t)$, $d(t)$ and $f(t)$. The residual generation system (or *internal form* of residual generator [1]) is obtained by replacing in (3) $\mathbf{y}(\lambda)$ by its expression from (2)

$$r(\lambda) = R_f(\lambda)\mathbf{f}(\lambda) + R_d(\lambda)\mathbf{d}(\lambda) + R_u(\lambda)\mathbf{u}(\lambda) \tag{4}$$

where

$$[\, R_f(\lambda) | R_d(\lambda) | R_u(\lambda) \,] := Q(\lambda) \left[\begin{array}{c|c|c} G_f(\lambda) & G_d(\lambda) & G_u(\lambda) \\ O & O & I_{m_u} \end{array} \right]$$

For a successfully designed filter $Q(\lambda)$, the corresponding residual generation system is proper and stable and achieves specific FD requirements (e.g., decoupling of disturbances from the residuals). Thus, the residual generation system is very useful for simulation purposes allowing to visualize the overall fault detection performance.

D. Filter specification

For fault detection and isolation problems, the determination of an appropriate fault-to-residual dynamics is part of the problem solution. The ideal dynamics results in the input-output form

$$r(\lambda) = M(\lambda)\mathbf{f}(\lambda) \tag{5}$$

where $M(\lambda)$ is proper and stable (and often also diagonal), representing an *ideal* FD filter specification (satisfying thus $R_f(\lambda) = M(\lambda)$, $R_d(\lambda) = 0$, $R_u(\lambda) = 0$). The fault dynamics specification $M(\lambda)$ can be also an input parameter, in which case, $M(\lambda)$ can be an arbitrary stable and proper TFM, which still can be updated to guarantee its admissibility.

E. Fault influence matrix

The dependence of the residuals from faults is characterized by the TFM $R_f(\lambda)$ in (4). We say that the fault j is *weakly detected* in the residual component i if the (i,j)-th element of $R_f(\lambda)$ is non-zero, *strongly detected* if its DC-gain is non-zero, and *not detected* if this entry is null. A *fault influence matrix* S can be defined as a constant $q \times m_f$ structured matrix (of the same size as $R_f(\lambda)$) with the entry S_{ij} set to -1, 1 or 0 corresponding to a weakly, strongly or not detected fault j from residual i, respectively.

F. Fault detectability information

Fault detectability refers to the question of the existence of FD filters which are sensitive to certain faults. Among many existing definitions (see [9] for an ample review), we chose the following ones which serve simultaneously as easy to check criteria for fault detectability. Let $N_d(\lambda)$ be a TFM whose rows form a left nullspace basis of $G_d(\lambda)$.

The j-th fault is *weakly detectable* (or *detectable*) if

$$N_d(\lambda) g_{f,j}(\lambda) \neq 0, \tag{6}$$

where $g_{f,j}(\lambda)$ is the j-th column of $G_f(\lambda)$.

The notion of strong detectability is related to the ability to detect constant faults via nonzero constant residuals. The j-th fault is *strongly detectable* if the TFM $N_d(\lambda)g_{f,j}(\lambda)$ has no zero in $\lambda = 0$ for a continuous-time system or $\lambda = 1$ for a discrete-time system.

For a system of the form (2), the *fault detectability information* can be packed in a structured m_f-dimensional row vector, whose j-th element is set to -1, 1 or 0 according to the weak, strong or lack of detectability of fault j.

III. Design of residual generation filters

The toolbox allows to solve various fault detection problems by providing suitable design functions of residual generator filters. All design functions have the generic form

```
[Q,INFO] = fd-design(SYS,inputs,options)
```

where:

- `SYS` is a system object with input $[u^T(t), f^T(t), d^T(t)]^T$ to specify the system model in one of the forms (1) or (2);
- `inputs` is a cell array with 3 components, where `inputs{1}` specifies an index list of control inputs, `inputs{2}` specifies an index list of fault inputs, and `inputs{3}` specifies an index list of disturbance inputs. *Note:* The components of `inputs` can alternatively be specified by names associated to different input groups.
- `options` is an option structure which can be set/accessed by special routines; frequently used options are: `options.StabDeg` – desired stability degree for the poles of detector; `options.Normalize` – normalization option; `options.Tol` – tolerance for rank computations.
- `Q` is a system object containing the resulting residual generator filter in (3) in the same representation as `SYS`.
- `INFO` is a structure containing additional information, as for example `INFO.rsys` – the resulting residual generation system (4), `INFO.rf` – the resulting ideal filter specification (5), `INFO.fdstruct` – achieved fault influence structure, `INFO.fdinfo` – the fault detectability information (weak, strong, none), etc.

The user options can be specified via a dedicated option structure. The function `fdoptionset` can be used to set all user options and to generate the appropriate option structure, while a complementary function `fdoptionget` is used to extract the options specific to each filter design function. For example, the tolerance for rank computations and the stability degree can be set using

```
options = fdoptionset('Tol',0.01,'StabDeg',-2)
```

In what follows we present shortly the functionality of the available design functions.

A. Design of fault detection filters

The function **`fd`** is available to solve the following *fault detection* problem: determine a physically realizable (i.e., proper and stable) linear residual generator filter of least order having the general form (3) such that:

(i) $r(t) = 0$ when $f(t) = 0$; and

(ii) $r(t) \neq 0$ when any $f_j(t) \neq 0$, for $j = 1, \ldots, m_f$.

The FD Problem is solvable provided condition (6) is fulfilled for $j = 1, \ldots, m_f$ [9].

The condition (i) requires the full decoupling of control and disturbance inputs from the residuals and can be expressed as a left nullspace condition

$$Q(\lambda)G_e(\lambda) = 0 \tag{7}$$

where

$$G_e(\lambda) = \left[\begin{array}{cc} G_u(\lambda) & G_d(\lambda) \\ I_{m_u} & 0 \end{array} \right]$$

The condition (ii) requires the sensitivity of residual to all faults and can be equivalently expressed as

$$Q(\lambda) \left[\begin{array}{c} g_{f,j}(\lambda) \\ 0 \end{array} \right] \neq 0, \quad j = 1, \ldots, m_f \tag{8}$$

where $g_{f,j}(\lambda)$ is the j-th column of $G_f(\lambda)$.

The underlying computational method of function **`fd`**, proposed in [4], [10], relies on computing a rational left nullspace basis $N_L(\lambda)$ of the rational matrix $G_e(\lambda)$. The standard solution computed by **`fd`** is $Q(\lambda) = V(\lambda)N_L(\lambda)$, where for an ℓ-dimensional nullspace basis $N_L(\lambda)$, $V(\lambda)$ is a $q \times \ell$ TFM ensuring a desired dynamics. If $N_L(\lambda)$ is a stable and proper rational basis, then V can be even chosen a constant matrix. The number of detector outputs q can be specified via `options.FDMaxNoRes`. The detector dynamics can be specified via a given stability degree in `options.StabDeg` or a set of desired poles in `options.FDPoles`. The method to compute a rational nullspace basis $N_L(\lambda)$ via a rational basis (direct) method or via a polynomial basis (indirect) method, can be specified by setting `options.NullspaceMethod` to `'rational'` or `'polynomial'`, respectively.

If `options.FDMethod` is set to `'minimal'`, a least order detector is computed. The basis vectors in $N_L(\lambda)$ are separated in two groups $N_L(\lambda) = \left[\begin{array}{c} N_{L,1}(\lambda) \\ N_{L,2}(\lambda) \end{array} \right]$, where the number of rows of $N_{L,1}(\lambda)$ is at least q (specified via `options.FDMaxNoRes`). The detector is computed as

$$\widetilde{Q}(\lambda) := V(\lambda)(N_{L,1}(\lambda) + X(\lambda)N_{L,2}(\lambda))$$

where the rational matrices $V(\lambda)$ and $X(\lambda)$ are determined such that the resulting filter has the least possible McMillan degree and has the desired dynamics. The computation of $X(\lambda)$ for a given separation of $N_L(\lambda)$ leading to a least order of $N_{L,1}(\lambda) + X(\lambda)N_{L,2}(\lambda)$ relies on minimal dynamic covers techniques and is described in [11]. A similar technique is employed for choosing appropriate $V(\lambda)$.

For more flexibility, the function **`fd`** is implemented to allow the design of FD filters for a given fault influence structure vector S, which can be specified via the option `options.FDSpec`. Only those faults which correspond to nonzero elements of S are detected, while those corresponding to zero elements of S (called also *nuisance* faults) are decoupled from the residual signal. The default setting is $S = [\,1\ \ldots\ 1\,]$ (i.e., all faults are detected).

The normalization of the residual signals with respect to maximum or minimum nonzero magnitudes of DC-gains can be performed by setting `options.Normalize` to `'max'` or `'min'`, respectively. The corresponding tolerance for nonzero DC-gain magnitudes can be specified via `options.FDGainTol`. For assessing the weak fault detectability conditions for the achieved $R_f(\lambda)$ in (4), a tolerance for the nonzero $\mathcal{H}_\infty$-norms of single elements can be set via `options.FDTol`.

B. Design of fault detection and isolation filters

The function **`fdi`** is available to solve the following *fault detection and isolation* (FDI) problem: determine a physically realizable linear residual generator filter of the general form (3) having $q = m_f$ outputs and least dynamical order such that:

(i) $r_j(t) = 0$ when $f_j(t) = 0$; for $j = 1, \ldots, m_f$

(ii) $r_j(t) \neq 0$ when $f_j(t) \neq 0$, for $j = 1, \ldots, m_f$.

The FDI Problem is solvable provided the condition

$$\operatorname{rank}\,[\,G_f(\lambda)\ G_d(\lambda)\,] = m_f + \operatorname{rank}\,G_d(\lambda) \tag{9}$$

is fulfilled [12].

The design approach underlying **`fdi`** is based on solving the following model matching problem [5]: choose a suitable $M(\lambda)$ (i.e., stable, proper, diagonal and invertible) and find a least McMillan degree solution $Q(\lambda)$ of the linear equation with rational matrices

$$Q(\lambda) \left[\begin{array}{ccc} G_f(\lambda) & G_d(\lambda) & G_u(\lambda) \\ O & O & I_{m_u} \end{array} \right] = M(\lambda) \left[\begin{array}{ccc} I_{m_f} & O & O \end{array} \right] \tag{10}$$

which is stable and proper. This equation arises by imposing for the internal form of the filter (4) the specification (5), thus achieving an *exact* decoupling of faults from the disturbance and system inputs as required by conditions (i) and (ii). A desired filter specification $M(\lambda)$ for the fault-to-residual dynamics can be done via `options.FDIResFilter`.

If `options.FDIMethod` is set to `'minimal'`, a least order detector is computed. For a given $M(\lambda)$ the general solution of (10) can be expressed as

$$\widehat{Q}(\lambda) = Q_0(\lambda) + Y(\lambda)N_L(\lambda),$$

where $Q_0(\lambda)$ is a particular solution of (10) and $N_L(\lambda)$ is a rational basis matrix for the left nullspace of

$$G_e(\lambda) = \left[\begin{array}{ccc} G_f(\lambda) & G_d(\lambda) & G_u(\lambda) \\ O & O & I_{m_u} \end{array} \right] \qquad (11)$$

The procedure to solve (10) proposed in [5] computes first $Q_0(\lambda)$ for $M(\lambda) = I$ and $N_L(\lambda)$, then determines a suitable $Y(\lambda)$ to obtain a solution $\widehat{Q}(\lambda)$ of least McMillan degree, and finally chooses an appropriate $M(\lambda)$ ensuring the stability and properness of $Q(\lambda) = M(\lambda)\widehat{Q}(\lambda)$. The resulting filter specification $M(\lambda)$ in (5) is provided in the output parameter `INFO.rf`. The normalization of $M(\lambda)$ with respect to maximum nonzero magnitudes of DC-gains can be performed by setting `options.Normalize` to `'max'`.

C. Design of a bank of fault detection filters

The function **`fdbank`** represents an extension of the function **`fd`** to solve the following more general design problem: given a $q_f \times m_f$ desired fault influence matrix S (i.e., with all entries 0's or 1's), determine a bank of q_f single-output FD filters

$$\mathbf{r}_i(\lambda) = Q_i(\lambda) \left[\begin{array}{c} \mathbf{y}(\lambda) \\ \mathbf{u}(\lambda) \end{array} \right], \quad i = 1, \ldots, q_f \qquad (12)$$

where the fault influence matrix achieved by $Q_i(\lambda)$ is equal to the i-th row of S (modulo sign of elements). The influence matrix can be specified by setting `options.FDBankSpec` equal to S.

The selection of fault influence matrices is thoroughly discussed in [1]. For example, the FDI Problem can be solved by using this function with $S = I_{m_f}$, while a standard FD problem corresponds to $S = [\,1 \ldots 1\,]$. This function is typically employed to solve for an arbitrary number of faults the weak FDI problem in the case when no two faults can simultaneously occur [1, Ch. 7].

D. Design of model detection filters

An alternative approach to detect single faults at a time is to use a bank of so-called model detection filters. Assume we have q_m models describing the normal and faulty systems, specified in the input-output form

$$\mathbf{y}^{(i)}(\lambda) = G_u^{(i)}(\lambda)\mathbf{u}(\lambda) + G_d^{(i)}(\lambda)\mathbf{d}(\lambda), \quad i = 1, \ldots, q_m \qquad (13)$$

where $y^{(i)}(t)$ is the output of the i-th system with control input $u(t)$ and disturbance input $d(t)$, respectively, and where $G_u^{(i)}(\lambda)$ and $G_d^{(i)}(\lambda)$ are the TFMs from the corresponding plant inputs to outputs.

The function **`md`** can be used to solve the following design problem: determine a bank of q_m FD filters

$$\mathbf{r}^{(i)}(\lambda) = Q^{(i)}(\lambda) \left[\begin{array}{c} \mathbf{y}^{(j)}(\lambda) \\ \mathbf{u}(\lambda) \end{array} \right], \quad i = 1, \ldots, q_m \qquad (14)$$

such that for $j = 1, \ldots, q_m$:

(i) $r^{(i)}(t) = 0$ when $i = j$; and

(ii) $r^{(i)}(t) \neq 0$ when $i \neq j$.

This function is useful when used in conjunction with fault tolerant control schemes where reconfiguration actions take place after detecting a specific faulty situation described by a corresponding fault model. Similarly, model detection can be seen as an alternative to existing Kalman-filter based approaches used for switching between different controllers in multi-model adaptive control. The main potential advantages of using FD-filters are their ability to decouple the influence of non-stochastic disturbances from the residuals and their lower dynamical orders.

E. $\mathcal{H}_2$-design of fault detection and isolation filters

The existence condition (9) for the solution to the "exact" FDI problem is seldom fulfilled in practical applications and therefore the best achievable result can be obtained by solving the rational equation (10) approximately (e.g., in a *least-squares* sense). The function **`fdih2`** is available to solve the FDI problem approximately by minimizing the $\mathcal{H}_2$-norm of the associated residual

$$\mathcal{R}(\lambda) := M(\lambda)F(\lambda) - Q(\lambda)G_e(\lambda) \qquad (15)$$

where $G_e(\lambda)$ is defined in (11) and $F(\lambda) = [\,I_{m_f}\ O\ O\,]$. The solution involves choosing an appropriate $M(\lambda)$ (i.e., stable, proper, diagonal and invertible) such that $\|\mathcal{R}(\lambda)\|_2$ is finite and the resulting $Q(\lambda)$ is stable and proper.

The function **`fdih2`** implements the general algorithm recently proposed by the author in [6] which also involves the selection of an appropriate $M(\lambda)$ to guarantee the existence of a solution. In contrast, the applicability of standard solution techniques based on solving an $\mathcal{H}_2$-filtering problem (see for example [3]) is conditioned by technical assumptions, as for example, full column rank of $G_e(\lambda)$ and lack of zeros on the extended imaginary axis in the continuous-time or on the unit-circle in the discrete-time. Although these conditions are not necessary for the existence of a solution, the standard approach still fails when they are not fulfilled. Since the filtering-based approach provides no clear guidance how to choose appropriate filter specification for successful design, the whole filter design reduces to an *ad-hoc* trial-and-error procedure [13].

The approach [6] has two main computational stages. The first stage basically achieves the reduction of the original problem to a simpler one for which, in the second step, the exact algorithm of [5] is used to solve the $\mathcal{H}_2$-norm minimization problem.

The main computation in the first stage is the determination of the *quasi* co-inner–co-outer of factorization

$$G_e(\lambda) = [\,G_{o,1}(\lambda)\ O\,] \left[\begin{array}{c} G_{i,1}(\lambda) \\ G_{i,2}(\lambda) \end{array} \right] := G_o(\lambda)G_i(\lambda),$$

where $G_i(\lambda)$ is square and co-inner and $G_{o,1}(\lambda)$ has full column rank. Recall that $G_i(\lambda)$ is *co-inner* if it has only stable poles and satisfies $G_i(\lambda)G_i^*(\lambda) = I$, where $G_i^*(s) := G_i^T(-s)$ in a continuous-time setting and $G_i^*(z) := G_i^T(1/z)$ in a discrete-time setting. The computation of this factorization is done using the first step of the dual of the inner-outer factorization algorithm proposed in [14].

Using (15), we obtain that

$$\|\mathcal{R}(\lambda)\|_2 = \left\| \left[\begin{array}{c|c} M(\lambda)\widehat{F}_1(\lambda) - Q(\lambda)G_{o,1}(\lambda) & M(\lambda)\widehat{F}_2(\lambda) \end{array} \right] \right\|_2$$

where $\widehat{F}_1(\lambda) = F(\lambda)G^*_{i,1}(\lambda)$ and $\widehat{F}_2(\lambda) = F(\lambda)G^*_{i,2}(\lambda)$.

If $Q(\lambda)$ is an *exact* solution of the equation

$$Q(\lambda)G_{o,1}(\lambda) = M(\lambda)\widehat{F}_1(\lambda) \tag{16}$$

then

$$\|\mathcal{R}(\lambda)\|_2 = \|M(\lambda)\widehat{F}_2(\lambda)\|_2$$

Since $G_{o,1}(\lambda)$ has full column rank, the system (16) has a solution which can be made proper and stable by appropriately selecting $M(\lambda)$ using the algorithm proposed in [5]. The general solution of (16) can be expressed as

$$Q(\lambda) = Q_0(\lambda) + Y(\lambda)Q_N(\lambda), \tag{17}$$

where $Q_0(\lambda)$ is a particular solution of (16) and $Q_N(\lambda)$ is a rational basis matrix for the left nullspace of $G_{o,1}(\lambda)$. The parametrization (17) of all $\mathcal{H}_2$-optimal solutions allows to determine suitable $Y(\lambda)$ leading to a solution of least McMillan degree. The choice of $M(\lambda)$ must guarantee that the resulting $Q(\lambda)$ is proper and stable and the residual norm is finite. Therefore, $M(\lambda)$ is chosen to additionally ensure that $\widehat{\mathcal{R}}(\lambda) := M(\lambda)\widehat{F}_2(\lambda)$ is stable and strictly proper in the continuous-time case, or stable and only proper in the discrete-time case. The computation of appropriate $M(\lambda)$ is done automatically using the stable and proper coprime factorization algorithm of [15].

F. $\mathcal{H}_\infty$-design of fault detection and isolation filters

The function **`fdihinf`** implements the general algorithm recently proposed by the author in [6] to solve the FDI problem approximately by minimizing the $\mathcal{H}_\infty$-norm of the residual (15). The solution has two main computational stages and involves choosing an appropriate $M(\lambda)$ such that $\|\mathcal{R}(\lambda)\|_\infty$ is finite and the resulting $Q(\lambda)$ is stable and proper. The first stage is identical to the first stage of solving the $\mathcal{H}_2$-norm minimization problem. In the second stage the two-blocks minimal distance problem

$$\gamma_{opt} = \inf \left\| \left[\; M(\lambda)\widehat{F}_1(\lambda) - Y(\lambda) \,\middle|\, M(\lambda)\widehat{F}_2(\lambda) \; \right] \right\|_\infty$$

is solved for $Y(\lambda)$ and the solution $Q(\lambda)$ is obtained by solving $Y(\lambda) = Q(\lambda)G_{o,1}(\lambda)$. In this phase an appropriate $M(\lambda)$ is also chosen to ensure that the above infimum exists. This implies that $M(\lambda)\widehat{F}_1(\lambda)$ and $M(\lambda)\widehat{F}_2(\lambda)$ must be proper and stable.

With the following lower and upper bounds for γ_{opt}

$$\gamma_l = \|M(\lambda)\widehat{F}_2(\lambda)\|_\infty,\ \gamma_u = \left\| \left[M(\lambda)\widehat{F}_1(\lambda) \middle| M(\lambda)\widehat{F}_2(\lambda) \right] \right\|_\infty$$

the optimal solution $Y(\lambda)$ is computed using the bisection-based γ-iteration approach [16].

For a given $\gamma > \|M(\lambda)\widehat{F}_2(\lambda)\|_\infty$ (e.g., $\gamma = (\gamma_l + \gamma_u)/2$), consider the solution of the suboptimal two-blocks problem

$$\left\| \left[\; M(\lambda)\widehat{F}_1(\lambda) - Y(\lambda) \,\middle|\, M(\lambda)\widehat{F}_2(\lambda) \; \right] \right\|_\infty \le \gamma \tag{18}$$

First we compute the right spectral factorization (see [17])

$$\gamma^2 I - M(\lambda)\widehat{F}_2(\lambda)\widehat{F}_2^*(\lambda)M^*(\lambda) = W(\lambda)W^*(\lambda) \tag{19}$$

where by construction, $W(\lambda)$ is biproper, stable and minimum-phase. Further, we compute the additive decomposition

$$L_s(\lambda) + L_u(\lambda) = W^{-1}(\lambda)M(\lambda)\widehat{F}_2(\lambda) \tag{20}$$

where $L_s(\lambda)$ is the stable part and $L_u(\lambda)$ is the unstable part.

If $\gamma > \gamma_{opt}$, the two-blocks problem (18) is equivalent to the one-block problem (see [16, Theorem 1, page 106])

$$\left\| W^{-1}(\lambda) \left(M(\lambda)\widehat{F}_1(\lambda) - Y(\lambda) \right) \right\|_\infty \le 1 \tag{21}$$

and $\gamma_H := \|L_u^*(\lambda)\|_H < 1$ ($\|\cdot\|_H$ denotes the Hankel norm of a stable TFM). In this case we readjust $\gamma_u = \gamma$. Otherwise (i.e., $\gamma_H \ge 1$), we readjust $\gamma_l = \gamma$. Then, for $\gamma = (\gamma_l + \gamma_u)/2$ we redo the factorization (19) and decomposition (20). This process is repeated until $\gamma_u - \gamma_l \le \varepsilon$ (a given tolerance).

If $\gamma_u \ge \gamma > \gamma_{opt}$, the stable solution of (21) can be expressed as

$$Y(\lambda) = W(\lambda)(L_s(\lambda) + Y_s(\lambda)),$$

where, for $1 \ge \gamma_1 > \gamma_H$, $Y_s(\lambda)$ is a stable solution of the γ_1-suboptimal Nehari problem

$$\|L_u(\lambda) - Y_s(\lambda)\|_\infty < \gamma_1 \tag{22}$$

The $\mathcal{H}_\infty$-solution $Q(\lambda)$ is the exact least McMillan degree solution of the linear rational equation $Q(\lambda)G_{o,1}(\lambda) = Y(\lambda)$. Since $G_{o,1}(\lambda)$ is only a quasi co-outer factor, it can still have unstable or infinite zeros. In the case when these zeros are not cancelled in the solution, the resulting $Q(\lambda)$ can be replaced by $\widetilde{M}(\lambda)Q(\lambda)$, where $\widetilde{M}(\lambda)$ is chosen such that $\widetilde{M}(\lambda)Q(\lambda)$ is proper and stable, and the norm condition (18) is still fulfilled when replacing $Y(\lambda)$ by $\widetilde{M}(\lambda)Y(\lambda)$. For example, to ensure properness, $\widetilde{M}(\lambda)$ are chosen diagonal with the diagonal terms of the form

$$\widetilde{M}_i(s) = \frac{1}{(\tau s + 1)^{k_i}} \quad \text{or} \quad \widetilde{M}_i(z) = \frac{1}{z^{k_i}}$$

for continuous- or discrete-time settings, respectively. Note that these factors have unit $\mathcal{H}_\infty$-norm.

IV. Implementation aspects

The main emphasis when implementing the function of the toolbox was to provide user-friendly tools which allow to solve challenging FD/FDI problems in a numerically reliable way, by exploiting completely all available parametric freedom in the problems. The underlying computational algorithms are based on descriptor system representations and rely on orthogonal matrix pencil reductions or orthogonal similarity transformations. For all basic computations, robust numerical software tools available in the Descriptor Systems Toolbox have been employed. Among the employed higher level *m*-functions and corresponding algorithms we mention `snull` – to compute rational or polynomial nullspaces [4], `kronscf` – to compute various Kronecker-like forms of system pencils [18], `sqrfac` – to compute quasi inner-outer factorizations [14], `smcover1`/`smcover2` – to compute type 1/2

minimum dynamic covers [19], [20], [11], `srsol` – to solve linear rational equations [5], [19], `dsminreal` – to compute minimal realizations of descriptor systems [21], `gstab` – to perform stabilization of generalized systems [22], `srcf/slcf` – to perform various right/left coprime factorizations for descriptor systems [15], [23], `dss2tm` – to convert descriptor system models to TFM form [24], etc. To implement the $\mathcal{H}_\infty$-norm solution, two functions from the HTOOLS Toolbox [25] have been upgraded and included in the toolbox: `lsfg2` – to compute left spectral factorizations, and `nehari` – to solve optimal or suboptimal Nehari problems.

There are several features which allow an user friendly operation with the functions of the toolbox. One aspect is using of system objects for which the grouping of input and output variables is used to label different categories of inputs (control, disturbance, fault) or outputs. Another important aspect is a unified way to define problem and user options. To manage the multitude of user options, a unique option structure has been defined, which can be set up and accessed with special functions. The considerable flexibility achieved in this way allows to exploit all existing parametric freedom when solving FD/FDI synthesis problems, as for example, choosing of arbitrary dynamics or exploiting non-uniqueness of solution by computing FD filter of least dynamical order.

V. Conclusions

The new Fault Detection Toolbox for MATLAB is based on new computational methods to solve FD/FDI filter design problems in the most general setting. A set of functions dedicated to solve FD filter design problems relies on rational nullspace computations. The new approaches to design FDI filters reformulate the filter design problems as equivalent model matching problems for which recently developed algorithms are employed to solve these problems exactly or approximately. Since the determination of suitable filter specification is part of the design, the technical difficulties often encountered when using standard methods (e.g., when trying to apply standard $\mathcal{H}_2$- or $\mathcal{H}_\infty$-norm synthesis tools) are completely avoided. For example, the presence of zeros or poles on the boundary of stability domains or problems with non-full rank and even improper transfer-function matrices can be easily handled.

The FD Toolbox basically covers the deterministic framework of residual generation. However, for systems having additionally white noise inputs, a stochastic fault detection approach based on whitening filters is more appropriate (see for example [26]). To detect faults hypothesis testing is performed for zero mean and whiteness of the residual signal. The design of whitening filters involves the solution of a spectral factorization problem. This supplementary functionality is provided by the functions **`fd`** and **`fdbank`**.

Further developments of the Fault Detection Toolbox are intended. These developments depend partly on availability of algorithms to solve problems like FD filter design problems in $\mathcal{H}_2$- or $\mathcal{H}_\infty$ sense or solving problems with mixed FD/FDI specifications. The computation of least order FD filters for a given fault influence matrix also appears to be still an open problem. Other developments will address explicitly robustness related issues, as for example, design of robust FD/FDI filters in the presence of parametric uncertainties, determination of appropriate thresholds to achieve a trade-off between false alarms and missed detections.

References

[1] J. Gertler, *Fault Detection and Diagnosis in Engineering Systems*. New York: Marcel Dekker, 1998.

[2] J. Chen and R. J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Kluwer Academic Publishers, London, 1999.

[3] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*. Springer-Verlag, Berlin, 2003.

[4] A. Varga, "On computing least order fault detectors using rational nullspace bases," in *Proc. of IFAC Symp. SAFEPROCESS'2003, Washington D.C.*, 2003.

[5] ——, "New computational approach for the design of fault detection and isolation filters," in *Advances in Automatic Control*, ser. The Kluwer International Series in Engineering and Computer Science, M. Voicu, Ed., vol. 754. Kluwer Academic Publishers, 2004, pp. 367–381.

[6] ——, "Numerically reliable methods for optimal design of fault detection filters," in *Proc. of CDC'05, Seville, Spain*, 2005.

[7] ——, "A Descriptor Systems toolbox for MATLAB," in *Proc. CACSD'2000 Symposium, Anchorage, Alaska*, 2000.

[8] ——, "Descriptor Systems Toolbox, V1.05, 1 October 2005," http://www.dlr.de/rm/desctool/.

[9] M. Nyberg, "Criterions for detectability and strong detectability of faults in linear systems," *Int. J. Control*, vol. 75, pp. 490–501, 2002.

[10] E. Frisk and M. Nyberg, "A minimal polynomial basis solution to residual generation for fault diagnosis in linear systems," *Automatica*, vol. 37, pp. 1417–1424, 2001.

[11] A. Varga, "Reliable algorithms for computing minimal dynamic covers," in *Proc. of CDC'2003, Maui, Hawaii*, 2003.

[12] P. M. Frank and X. Ding, "Frequency domain approach to optimally robust residual generation and evaluation for model-based fault diagnosis," *Automatica*, vol. 30, pp. 789–804, 1994.

[13] H. Niemann and J. Stoustrup, "Desifgn of fault detectors using H_∞ optimization," in *Proc. of CDC'00, Sydney, Australia*, 2000, pp. 4237–4238.

[14] C. Oară and A. Varga, "Computation of general inner-outer and spectral factorizations," *IEEE Trans. Automat. Control*, vol. 45, pp. 2307–2325, 2000.

[15] A. Varga, "Computation of coprime factorizations of rational matrices," *Lin. Alg. & Appl.*, vol. 271, pp. 83–115, 1998.

[16] B. A. Francis, *A Course in H^∞ Theory*. Springer-Verlag, New York, 1987.

[17] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*. Prentice Hall, 1996.

[18] T. Beelen and P. Van Dooren, "An improved algorithm for the computation of Kronecker's canonical form of a singular pencil," *Lin. Alg. & Appl.*, vol. 105, pp. 9–65, 1988.

[19] A. Varga, "Computation of least order solutions of linear rational equations," in *Proc. of MTNS'04, Leuven, Belgium*, 2004.

[20] ——, "Reliable algorithms for computing minimal dynamic covers for descriptor systems," in *Proc. of MTNS'04, Leuven, Belgium*, 2004.

[21] ——, "Computation of irreducible generalized state-space realizations," *Kybernetika*, vol. 26, pp. 89–106, 1990.

[22] ——, "On stabilization of descriptor systems," *Systems & Control Letters*, vol. 24, pp. 133–138, 1995.

[23] C. Oară and A. Varga, "Minimal degree coprime factorization of rational matrices," *SIAM J. Matrix Anal. Appl.*, vol. 21, pp. 245–278, 1999.

[24] A. Varga, "Computation of transfer function matrices of generalized state-space models," *Int. J. Control*, vol. 50, pp. 2543–2561, 1989.

[25] A. Varga and V. Ionescu, "HTOOLS - A Toolbox for solving H_∞ and H_2 synthesis problems," in *Proc. of IFAC/IMACS Symp. on Computer Aided Design of Control Systems, Swansea, UK*, July 1991, pp. 508–511.

[26] R. Nikoukhah, "Innovations generation in the presence of unknown inputs: application to robust failure detection," *Automatica*, vol. 30, pp. 1851–1867, 1994.