© IFAC Fault Detection, Supervision and Safety of Technical Processes, Beijing 2006

ELSEVIER
IFAC
PUBLICATIONS

AN INTRODUCTION TO A MATLAB-BASED FDI-TOOLBOX

S. X. Ding * **E. Atlas** *,** **S. Schneider** *,[1] **Y. Ma** *,***
T. Jeinsch *** **E. L. Ding** **

* *Institute for Automatic Control and Complex Systems, University of Duisburg-Essen , 47057 Duisburg, Germany*
** *Dept. of Physical Engineering, University of Applied Sciences Gelsenkirchen, 45877 Gelsenkirchen, Germany*
*** *Department of Powertrain Mechatronics Development Gasoline Engines, IAV GmbH, 38518 Gifhorn*

Abstract: In this paper, an FDI-Toolbox developed in the Matlab© programming enviroment is introduced. It includes a number of functions for the design of observer-based and parity space FDI systems including both residual generation and evaluation. The application of the toolbox is illustrated by a number of examples. *Copyright © 2006 IFAC*

Keywords: Matlab© Toolbox; Fault diagnosis; Observer based FDI; Parity Space; Unified Solution

1. INTRODUCTION

The study on model-based fault diagnosis began in the early 1970s. Strongly stimulated by the newly established observer theory at that time, the first model-based fault detection method, the so-called fault detection filter, was proposed by Beard and Jones (Gertler, 1998; Chen and Patton, 1999). Since then, the model-based *F*ault *D*etection and *I*solation (FDI) theory and technique went through a dynamic and rapid development and is currently becoming an important field of automatic control theory and engineering.

In the first decade of the young history of the model-based FDI technique, various methods were developed. During that time the framework of the model-based FDI technique had been established step by step. In his survey paper in *Automatica*, Frank (Frank, 1990) summarised the major results achieved in the first fifteen years of the model-based FDI technique, clearly sketched its framework and classified the studies on model-based fault diagnosis into

- observer-based methods
- parity space methods and
- parameter estimation based methods.

Led by Prof. Frank, our institute has been intensively involved in the development of observer-based and parity space methods since 1985. During the past five years, applications of observer-based and parity space FDI technique in different industrial sectors have built a major R&D-focus of our institute. We noticed that there is a lack of a flexible software tool for the design of model-based FDI systems, even for some well-established and standardised schemes. Also, our industrial cooperation partners asked frequently for such a software tool. These motivated us to start a R&D project, since 2005, aiming at the development of an FDI-Toolbox.

Considering that the software program Matlab© is now widely accepted as a standard system design tool at universities as well as in industry, and moreover in our institute there exist a number

[1] supported in part by EU grant NeCST

of function blocks in Matlab© that have been developed for the design of different FDI systems (Jeinsch *et al.*, 1998), it has been decided to develop the FDI-Toolbox in the Matlab© programming environment.

The main objective of this paper is to give a brief introduction to the FDI-Toolbox developed in our institute. The paper is organised as follows. First, the basic concepts and structure of the FDI-Toolbox are described. It is followed by a brief introduction to the basic functions of the toolbox. By means of a number of academic examples, the application of the FDI-Toolbox is finally illustrated and demonstrated.

2. CONCEPTS AND STRUCTURE OF THE FDI-TOOLBOX

2.1 Basic ideas, concepts and some features

The major objective of the FDI-Toolbox is to aid academics and engineers by the design of (standard) observer-based and parity space FDI systems. Considering the intimate relationships between the observer-based FDI methods and control theory, the development of the FDI-Toolbox is based on the program Matlab© and the Matlab© Control System and Robust Control Toolbox. In the actual development phase, only well-established, standard FDI schemes are integrated into the toolbox. On the other side, in order to ensure a high acceptance, no additional toolboxes, besides of the above-mentioned Control System and Robust Control Toolbox, are needed for the implementation of the FDI-Toolbox.

In the past, the residual generation and evaluation problems were often separately handled and the major research focus was on the residual generation schemes. It has been recognised that a more efficient way to solve FDI problems is an integrated design of residual generation and evaluation (Frank, 1994; Ding and Guo, 1996). It is one of the technical features of this FDI-Toolbox that both residual generation and residual evaluation functions are integrated into the toolbox.

In the early nineties, great efforts have been made to establish relationships between the observer and parity relation based methods. Several authors from different research groups, in parallel and from different aspects, demonstrated that the parity space methods lead to certain types of observer structures and are therefore structurally equivalent to the observer-based ones even though the design procedures differ. It is another important technical feature of this FDI-Toolbox is a function block of the toolbox by which one can transfer a parity space FDI system into an observer-based (Frank and Wuennenberg, 1989).

2.2 Structure of the FDI-Toolbox

The FDI-Toolbox includes a number of functionalities, from system definition over design of residual generators to the design of residual evaluation leading to an optimal FDI-System (Ding *et al.*, 2000*a*; Ding *et al.*, 2000*b*). The steps to a complete design of an FDI-System using the FDI-Toolbox can be divided into three parts:

- modelling and definition of data structure,
- residual generation and
- residual evaluation.

2.3 Modelling and definition of data structure

The first step in designing an FDI-System using the FDI-Toolbox is the modelling and definition of the data/system structure. Due to the intimate relationship to the linear control theory, standard system description forms and modelling schemes adopted in the Matlab© Control System Toolbox are used in the FDI-toolbox. Moreover, the dynamics due to faults, model uncertainties and disturbances as well as their statistical behavior are also modelled. For this purpose, a new state space object, based on the Matlab© Control System Toolbox object *ss*, has been developed and integrated into the FDI-Toolbox (see Sec. 3.1).

2.4 Residual generation concept

In the framework of the FDI-Toolbox several analytical model based residual generators are developed, like observer-based and parity space approaches to consider deterministic disturbances and Kalman filter approaches to consider stochastic disturbances. In case the disturbance can not be decoupled from the faults robust fault detection algorithms are included in the FDI-Toolbox.

An elegant and optimal solution using the Unified Solution proposed by Ding (Ding et al., 2000a) is implemented in the FDI-Toolbox to overcome the mentioned robustness problems.

A unified and systematic way for residual generators is one of the cores of the FDI-Toolbox. As depicted in Fig. 1 most of the residual generator methods try to determine the best possible fault detection system.

2.5 Residual evaluation concept

The step of residual generation is followed by a sufficient selection of a residual evaluation method. As introduced in (Frank *et al.*, 2000), the basic forms of residual generator for model-based FDI are fault detection filter (FDF), diagnostic

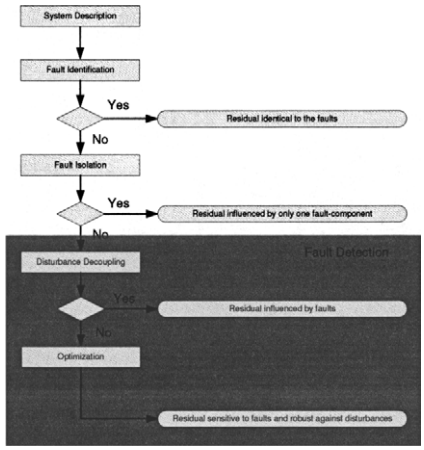

Fig. 1. Residual generation strategy

observer (DO) and parity space (PS) approach, any type of LTI residual generator is only a variation of one of the basic forms. Therefore in the residual evaluation concept, only the evaluation algorithms for these basic forms of residual generators have been developed.

Fig.2 gives the structure of the residual evaluation concept. According to the type of residual generator (FDF, DO or PS), the corresponding residual evaluation function (*redfh2()*, *redoh2()* or *repsh2()*) is called by *reh2()*, which is the core of residual evaluation. Depending on the information about stochastic disturbance the result of the mentioned sub-functions is either a threshold with ($n = 0$) or without covariance matrix ($n \neq 0$).

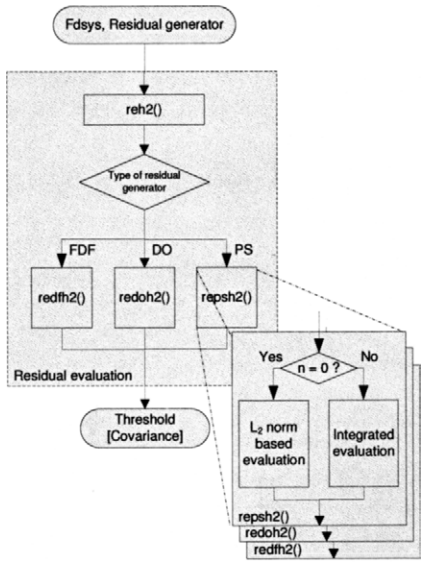

Fig. 2. Residual evaluation strategy

3. BASIC FUNCTIONS AND IMPLEMENTATION

Following the basic functions of the FDI-Toolbox including the definition of system structure, residual generation and evaluation is introduced.

3.1 System structure definition

Matlab© supports object oriented programming (OOP), including overloading [2], overriding [3], encapsulation and inheritance as well as public and private methods. This section is drawing parallels between the ideas of OOP and the implementation of the basic functions of the FDI-Toolbox.

A key of the implementation of the FDI-Toolbox is a new object called *fdss* (fault detection state space). This new object is based on the state space object *ss* of the Control System Toolbox. It is derived by inheritance, one of the basic concepts of OOP (Eckel, 2000), from the *ss*-object. Base class of the objects *tf, zpk, frd, ss* and also the new *fdss* is the *lti*-class. Fig. 3 illustrates the procedure of inheritance for the *fdss*-object.

The advantage using OOP techniques is the ability of using the fundamental structure and public methods of the base-class and the parent object (*ss*) for the new *fdss*. Only methods using additional properties or object specific methods have to be implemented. In sense of OOP methods of the class and/or the object can be overloaded and/or overridden. An example for overriding

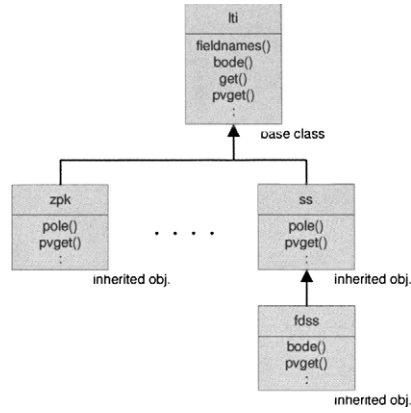

Fig. 3. Inheritance for *fdss* objects

methods of the base class is the method *bode()*, the method *pole()* overrides a method of the parent object. The method *bode()* also gives an example for overloading methods. Based on following code

[2] overloading - having more than one function with the same name in the same scope or having more than one operator with the same name in the same scope.

[3] overriding - declaring a function in derived class with same name and matching type as virtual function in a base class. Argument types must match(Stroustrup, 1994).

input	method	result
bode(sys1)	lti/bode.m	bode plot from inputs to outputs
bode(sys2)	fdss/bode.m	inputs incl. faults
bode(sys1, Ed, ...)		and disturbances to outputs
pole(sys1)	ss/pole.m	eigenvalues of A
pole(sys2)	ss/pole.m	eigenvalues of A
fieldnames(sys)	lti/fieldnames.m	all fieldnames

Table 1. calling functions

the behavior of Matlab© calling the different methods will be described.

```
1 sys1 = ss(A,B,C,D);
2 sys2 = fdss(sys1, Ed, Fd, Ef, Ff, En, Fn, Sn);
3 bode(sys1); bode(sys2);
4 pole(sys1); pole(sys2);
5 fieldnames(sys2);
6 bode(sys1, Ed, Fd, Ef, Ff);
```

In the first two rows two different systems are defined as *ss*-object and *fdss*-object respectively. In line 3 bode is called for the different systems. Depending on type of object given as argument either *bode()* of the base class or the overridden function of the *fdss*-object is called. Due to limited space further method calls are shown in Tab. 1.

3.2 Residual generation

Based on the following system description

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + E_f f(k) + E_d d(k) \\ y(k) &= Cx(k) + Du(k) + F_f f(k) + F_d d(k) \end{aligned} \quad (1)$$

where $f(k), d(k)$ denote the fault and unknown input vectors respectively with $d \in R^{k_d}, f \in R^{k_f}$ and matrices $A, B, C, D, E_f, E_d, F_f, F_d$ are known and of appropriate dimensions, different residual generation methods are described below.

3.2.1. Residual generation by observer structures

Consider system model (1) which can also be reformulated as:

$$y(z) = G_u(z)u(z) + G_f(z)f(z) + G_d(z)d(z).$$

An observer-based residual generator is a dynamic system which can be generally described by (Chen and Patton, 1999; Patton *et al.*, 2000; Blanke *et al.*, 2003):

$$\begin{aligned} z(k+1) &= Fz(k) + Gy(k) + Hu(k) \\ r(k) &= Wz(k) + Vy(k) + Qu(k) \end{aligned} \quad (2)$$

where $r(k)$ denotes the residual vector whose dimension depends on the FDI requirements (early detection, low missed-detection rate etc.) and matrices F, G, H, Q, V, W satisfy the so-called Luenberger equations:

$$\begin{aligned} TA - FT = GC, \quad VC + WT = 0 \\ H = TB - GD, \quad Q + VD = 0 \end{aligned} \quad (3)$$

In (3), matrix T stands for a state transformation. Now write residual generator (2) in a more general form:

$$r(z) = \begin{bmatrix} R_y(z) & R_u(z) \end{bmatrix} \begin{bmatrix} y(z) \\ u(z) \end{bmatrix} = R(z) \begin{bmatrix} y(z) \\ u(z) \end{bmatrix} \quad (4)$$

The dynamics of residual generator (4) is governed by:

$$\begin{aligned} r(z) &= R(z) \begin{bmatrix} y(z) \\ u(z) \end{bmatrix} \\ &= R(z) \underbrace{\begin{bmatrix} G_f(z) & G_d(z) & G_u(z) \\ 0 & 0 & I \end{bmatrix}}_{\Gamma(z)} \begin{bmatrix} f(z) \\ d(z) \\ u(z) \end{bmatrix} \end{aligned} \quad (5)$$

The residual generator design problems can then be formulated as finding a stable postfilter R(z) such that fulfills for:

- fault identification

$$R(z)\Gamma(z) = \begin{bmatrix} I_{k_f \times k_f} & 0 & 0 \end{bmatrix}$$

- fault isolation

$$R(z)\Gamma(z) = \begin{bmatrix} diag(\hat{g}_1(z), ..., \hat{g}_{k_f}(z) & 0 & 0 \end{bmatrix}$$

- fault decoupling

$$R(z)\Gamma(z) = \begin{bmatrix} \hat{G}_f(z) & O & O \end{bmatrix}$$

- residual generation (neglecting unknown inputs)

$$R(z) \begin{bmatrix} G_f(z) & G_u(z) \\ O & I \end{bmatrix} = \begin{bmatrix} \hat{G}_f(z) & O \end{bmatrix}$$

The functions *obside()*, *obsiso()* and *obsdec()*, as mentioned in Tab.2, are developed for the task of fault identification, isolation and decoupling. The function *obsresgen()* computes the postfilter $R(z)$. The header function *obsopt()* determines the best possible observer for the given system.

3.2.2. Residual generation by Parity Space

Consider the linear discrete time-invariant system (1) a parity relation based residual generator is expressed by

$$r(k) = v_p^{\mathrm{T}}(y_s(k) - H_{u,s}u_s(k)) \quad (6)$$

with the so-called parity vector

$$v_p^{\mathrm{T}} = [v_{p,0}^{\mathrm{T}}, \cdots, v_{p,s}^{\mathrm{T}}] \in P_s$$

where $P_s = \{v_p^{\mathrm{T}} | v_p^{\mathrm{T}} H_{0,s} = 0\}$ is the parity space and s is called the order of the parity vector and

$$\begin{aligned} H_{u,s} &= \begin{bmatrix} D & 0 & \dots & 0 \\ CB & D & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{s-1}B & CA^{s-2}B & \dots & D \end{bmatrix}, \\ u_s(k) &= \begin{bmatrix} u^{\mathrm{T}}(k-s) & \dots & u^{\mathrm{T}}(k) \end{bmatrix}^{\mathrm{T}}, \\ y_s(k) &= \begin{bmatrix} y^{\mathrm{T}}(k-s) & \dots & y^{\mathrm{T}}(k) \end{bmatrix}^{\mathrm{T}}. \end{aligned} \quad (7)$$

The dynamics of the residual generator (6) is governed by

$$r(k) = v_p^{\mathrm{T}}(H_{0,s}x(k-s) + H_{d,s}d_s(k) + H_{f,s}f_s(k))$$

with $H_{d,s}, H_{f,s}, d_s(k), f_s(k)$ analog to (7) substituting B, D with E_d, F_d and E_f, F_f respectively.

A significant advantage of the parity space methods is that only computation of some well-defined algebraic equations is involved for the FDI system design. On the other hand this method causes a high implementation effort which is noticeable in a high amount of computation and memory effort. For this reason a function *ps2obs()* is included in the FDI-Toolbox to transform the parity space based residual generator into a Diagnostic Observer (DO)(Frank and Wuennenberg, 1989).

3.3 Residual evaluation

The norm-based residual evaluation scheme is a practical way to approach the decision-making problem, in which a certain mathematical norm is chosen as the residual evaluation function and, based on it, the threshold is established. The mostly used norm for the purpose of residual evaluation is the L_2 norm. The L_2 norm can be interpreted as an evaluation of the energy level in a signal. The L_2 norm based evaluation algorithm has been realized in every sub-function of the evaluation module. The structure of sub-function can be found in Fig. 2.

For systems with deterministic and stochastic disturbance, the generated residual is also a random signal. Therefore the integrated evaluation algorithm, which combines the L_2 norm and statistical testing based evaluation, should be used to evaluate the residual to achieve satisfactory fault detection performance and robustness against the influence of deterministic disturbances (Ma, 2005).

4. EXAMPLES

To demonstrate the capabilities of the FDI-Toolbox an example for residual generation in the following form can be taken. Starting with the system description in the form of (1) the matrices of the system are defined as

$$A = \begin{bmatrix} -4 & 0.4 \\ 0.4 & -0.57 \end{bmatrix}, B = \begin{bmatrix} -0.74 & 0.64 \\ -1.05 & 0.78 \end{bmatrix},$$
$$C = \begin{bmatrix} -1.12 & 0.72 \\ 0 & 0.67 \\ -0.23 & 0.03 \end{bmatrix}, D = \begin{bmatrix} 0.19 & 0 \\ 0 & -0.7 \\ 0 & 0 \end{bmatrix},$$
$$E_f = \begin{bmatrix} 0 & 1.85 \\ 0.16 & -2.28 \end{bmatrix}, F_f = \begin{bmatrix} 0 & 0 \\ 0 & 0.17 \\ -2.46 & 0.78 \end{bmatrix} \quad (8)$$

4.1 Observer-based residual generation

Starting point for the observer-based residual generation are the system matrices (8) and additional disturbance distribution matrices:

$$E_d = \begin{bmatrix} 0.58 & -0.78 \\ 0 & 1.09 \end{bmatrix}, F_d = \begin{bmatrix} 0 & -0.33 \\ 0.35 & -2.17 \\ 2.81 & 0 \end{bmatrix}$$

By using *obsopt()* to generate the best possible observer following results can be obtained:

>> sys3 = *ss*(A,B,C,D);
>> sys4 = *fdss*(sys3, Ed, Fd, Ef, Ff);
>> [Aobs, Bobs, Cobs, Dobs] = *obsopt*(sys4);

$$A_{obs} = \begin{bmatrix} -1.88 & 2.82 \\ 1.14 & -4.39 \end{bmatrix}, C_{obs} = \begin{bmatrix} -4.76 & 4.36 \end{bmatrix},$$
$$B_{obs} = \begin{bmatrix} -1.3 & 0.59 & -0.02 & 0.67 & 0.13 \\ 0.59 & -0.36 & -0.12 & 0.52 & -0.79 \end{bmatrix},$$
$$D_{obs} = \begin{bmatrix} -4.19 & 0.65 & -0.08 & 0.8 & 0.45 \end{bmatrix}$$

which fulfills requirements for fault decoupling.

4.2 Optimal robust fault detection

An example of the design procedure for an Unified Solution based fault detection system is given. The resulting residual generator is given in form of a fault detection filter, and the residual is evaluated by the L_2 norm based evaluator.

Given system (1), matrices (8) and the distribution matrices of disturbances

$$E_d = \begin{bmatrix} 0.58 & -0.78 & 0 & 0 \\ 0 & 1.09 & 0 & 0.76 \end{bmatrix},$$
$$F_d = \begin{bmatrix} 0 & -0.33 & 0 & 1.02 \\ 0.35 & -2.17 & -1.27 & -0.84 \\ 2.81 & 0 & -2.01 & 0.34 \end{bmatrix}$$

no full decoupling of disturbances can be achieved. This can be verified by using function *fdichk()*.

>> fdichk(A, B, C, D, Ef, Ed, Ff, Fd)

The Unified Solution can be used to design the optimal robust fault detection system with:

>> [Gr, L, V] = optus(A, B, C, D, Ts, Ed, Fd).

The resulting observer gain L and weighting matrix V of the FDF based residual generator are

$$L = \begin{bmatrix} 0.18 & 0.24 & 0.07 \\ 0.40 & -0.46 & 0.14 \end{bmatrix}, V = \begin{bmatrix} -0.01 & -0.12 & -0.24 \\ 0.03 & -0.38 & 0.198 \\ -0.94 & -0.04 & 0.04 \end{bmatrix}$$

Assume that the upper bound of the L_2 norm of disturbances in the evaluation window $[t_1, t_2]$ is given by $\Delta_d^2 = 0.1$. The threshold for residual evaluation can be computed in the following form:

>> sys = fdss(A, B, C, D, Ed, Fd, Ef, Ff);

>> [Jth] = reh2(sys, L, V, 0.1)

or directly using the sub-function of evaluation module *redfh2()* to compute the threshold as:

>> s1 = ss (A, B, C, D)

>> [Jth] = redfh2(s1, Ed, Fd, 0.1)

This calculation results in threshold $J_{th} = 0.1$.

5. CONCLUSION

In this paper the Matlab-based FDI-Toolbox has been introduced. The FDI-Toolbox provides a set of functions for the design of LTI FDI systems.

Many functions of the toolbox have been successfully tested in industrial applications such as in the automotive or metals industry.

The toolbox itself is developed under Matlab© (R2006a) and is downward compatible till release R13. Beside Matlab© the Control System and Robust Control Toolbox is needed.

6. DEDICATION

This paper is dedicated to the memory of Prof. P.M. Frank.

REFERENCES

Blanke, M., M. Kinnaert, J. Lunze and M. Staroswiecki (2003). *Diagnosis and Fault-Tolerant Control.* Springer-Verlag.

Chen, J. and R. J. Patton (1999). *Robust Model-Based Fault Diagnosis for Dynamic Systems.* The Kluwer International Series on Asian Studies in Computer and Information Science. Kluwer Academic Publishers.

Ding, S. X. and L. Guo (1996). Observer-based fault detection optimized in the frequency domain. In: *Proc 13th IFAC World Congress.*

Ding, S. X., E.L. Ding, T. Jeinsch and P. Zhang (2000*a*). An approach to a unified design of fdi systems. *Asian Control Conference, Shanghai, China.*

Ding, S. X., T. Jeinsch, P. M. Frank and E. L. Ding (2000*b*). A unified approach to the optimization of fault detection systems. *International Journal of adaptive control and signal processing.*

Eckel, B. (2000). *Thinking in C++ Second Edition, Volume 1: Introduction to Standard C++.* Vol. 2. Prentice Hall.

Frank, P. M. (1990). Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy-a survey and some new results. *Automatica* **26**, 459–474.

Frank, P. M. (1994). Enhancement of robustness in observer-based fault detection. *International Journal of Control* **59 (4)**, 955–981.

Frank, P. M. and J. Wuennenberg (1989). *Fault Diagnosis in Dynamic Systems.* Chap. Robust fault diagnosis using unknown input observer schemes, pp. 46–98. Prentice Hall.

Frank, P. M., S. X. Ding and T. Marcu (2000). Model based fault diagnosis in technical processes. *Trans. Inst. MC / Millennium issue* **22 (1)**, 57–101.

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

Jeinsch, T., X. Ding and S. Mueller (1998). FDI-toolbox fuer MATLAB. *Automatiserungstechnik* **46 (1998) 7**, 355–356.

Ma, Y. (2005). Integrated Design of Observer-Based Fault Diagnosis Systems and its Application to Vehicle Lateral Dynamic Control Systems. PhD thesis. Unversity of Duisburg-Essen.

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

Stroustrup, B. (1994). *The Design and Evolution of C++.* Addison Wesley. German translation, Addison-Wesley Germany, Bonn. 1994. ISBN 3-89319-755-9.

APPENDIX:

General	
fdichk	Check existence conditions
fdicon	Convert fdsys to Gu, Gf and Gd
fdinv	Inversion of dynamical systems
ps2obs	Convert parity vector to observer
bode	Bode plot of the system to be supervised
psmat	Generation of parity space matrices
Residual Generation	
kald	Design of fault detection filter based residual generator with Kalman filter algorithm
psdec	Fault Detection
psiso	Fault Isolation
obsdec	Fault Detection
obsiso	Fault Isolation
obside	Fault Identification
obsresgen	Post-filter generation
Optimization	
optus	Optimization of fault detection filter based residual generator with Unified Solution
psopt	Optimization via multiobjective performance index
obsopt	Best possible diagnostic observer
Residual evaluation	
reh2	L_2 norm based residual evaluation of model-based fault detection system
redfh2	L_2 norm based residual evaluation for fault detection filter based residual generator
redoh2	L_2 norm based residual evaluation for diagnostic observer based residual generator
repsh2	L_2 norm based residual evaluation for diagnostic observer based residual generator
vcov	Compute the covariance matrix of the output vector $Y_{k-s,k} = [y(k-s)^T, y(k-s+1)^T, ..., y(k)^T]^T$ for discrete system with white zero-mean Gaussian input

Table 2. List of implemented functions