Model-on-Demand MATLAB Toolbox for Fault Diagnosis

HAJRUDIN EFENDIC
Siemens AG
VOEST-Alpine Industrieanlagenbau (VAI)
Turmstraße 44, A-4031 Linz
AUSTRIA

Abstract: - The modeling and fault diagnosis in complex systems must involve a kind of trade-off between computational efficiency and demands for precision of fault diagnosis. In this paper a software prototype for fault diagnosis of complex systems, utilizing an approach which deals with those two opposite requests, has been presented.

Key-Words: - Complex Systems, Modeling, Fault Diagnosis, Structural Analysis, Model-on-Demand

1 Introduction

Fault diagnosis in complex systems has to deal with significant problems, primarily with large number of variables which have to be analyzed, and which are described by large amounts of measurements. Fault diagnosis in such systems can be achieved only through an automatic, systematic and computationally effective approach, capable to cope with complexity of the systems and to produce satisfying diagnosis results at the same time.

The problem of systematic approach to fault detection and isolation is researched for a long time. The examples discussing problems of isolability, sensitivity to failures and conflicting diagnostic outcomes can be found in [1] and [2], and the utilization of structural analysis in diagnosis for example in [3], [4] and [5]. Also, a number of software tools for fault diagnosis in technical systems are developed, the latest examples being DTSA (Design Tool for Structural Analysis) [6], SaTool [7] and AMandD (Advanced Monitoring and Diagnostic System) [8].

But all these approaches which utilize structural analysis for fault diagnosis start from a known model of the system under consideration, and then translate that model into a structural model. The automatic model generation for use in structural analysis for fault diagnosis is seldom discussed. Furthermore, neither qualitative characteristics of modeling process for fault diagnosis nor qualitative characteristics of fault diagnosis itself are considered.

The automatic derivation of the process models from the measurement data for fault diagnosis can however be computationally expensive and very difficult especially if the data carry insufficient information about the process. In this paper an approach which combines efficiency in automatic modeling of a complex process with precision of fault diagnosis is presented. The Model-on-Demand (MoD) MATLAB Toolbox, a prototype of software for fault diagnosis in complex industrial systems, is then programmed using the proposed approach.

The paper is organized as follows: first the problem statement and the basics of the theoretical solution are given, followed by the design principles of MoD Toolbox and an experimental example.

2 Elements of the Solution Algorithm

An iterative multilayer approach to fault diagnosis in complex systems has been discussed in [9], [10] and [11]. The algorithmic structure proposed there consists of four main steps: data acquisition, data preprocessing, data-based modeling and fault diagnosis. The main feature of the proposed algorithm is complexity management in order to have a compromise between fast and accurate fault diagnosis in complex systems.

The modeling for fault diagnosis is separated into two segments: simple modeling and sophisticated modeling. Simple modeling includes derivation of the process models for the “neighboring problem” (the simplification of the original process under consideration). The sophisticated modeling includes refining of the modeling process in order to improve both detectability (by increasing the model quality) and isolability (by forming isolable common model structures).

Due to the length limitations, only the necessary extensions of the proposed approach are given in this paper. For more detailed description of the previously proposed approach the readers are kindly directed to the referred publications.
2.1 Modeling of Industrial Processes

In modeling of a technical process either first-principle mathematics or measurement data-driven modeling approaches may be used. First-principles models provide important insight into system behavior, but may lack accuracy. Data-driven models on the other hand can provide accurate results, but tend to offer limited understanding of the physics of the system.

The measurement and modeling processes can be schematically shown as in Figure 1. The output from the measurement system of the process is used in the modeling. Note that a measurement is always an approximation of the real values of variables, since the exact values are never accessible. Therefore, to distinguish from the exact value, the measurements of vectors of process variables \(\mathbf{u}\) and \(\mathbf{y}\) in Figure 1 are denoted as vectors of measurements \(\mathbf{\tilde{u}}\) and \(\mathbf{\tilde{y}}\) respectively. The measurement vectors \(\mathbf{\tilde{u}}\) and \(\mathbf{\tilde{y}}\) are composed from measurements of single variables measured at \(N\) time instants.

The process \(\Sigma\) with \(m\) input and \(k\) output measured variables is in general represented by the following mapping:

\[ M : \mathbb{R}^{N \times m} \rightarrow \mathbb{R}^{N \times k} \]  
(1)

In the general case, functional dependency \(M(\cdot)\) in previous MIMO mapping is a nonlinear function.

The use of models with multiple outputs in fault diagnosis reduces flexibility to some degree. A more suitable “target variable” approach to modeling is utilized in this paper. A target variable can be selected both from input or output vector. If suppose that, without loss of generality, a target channel is chosen from the output vector:

\[ \hat{y}_i \in \mathbf{\tilde{y}} \]  
(2)

a model of the target variable is built on the basis of remaining inputs and outputs:

\[ M : \mathbb{R}^{N \times m+1-k} \rightarrow \mathbb{R}^{N \times 1} \]

\[ \hat{y}_i = M(\mathbf{\tilde{u}}, \mathbf{\tilde{y}} / \{\hat{y}_i\}), i \in \mathbb{N}, 0 < i \leq k \]  
(3)

Selecting consecutively each process variable from the data set as the target variable once, the complete data set can be analyzed automatically.

Modeling of a target variable using all available process variables is not reasonable. Only in rare cases of simple technical systems, a process variable indubitably depends on all other process variables. Therefore, a selection of dependent variables before modeling can help to reduce the complexity of the modeling process itself. The variable selection has two main goals:

1. to reduce the number of inputs for each target variable;
2. to arrange the selected inputs in the descending order in accordance to the degree of influence on the target variable.

Note that the degree of influence level determined in the variable selection step is not equal or equivalent to the model sensitivity determined in the later modeling step.

The reduction of the number of inputs for a model of the target variable results in the following notation:

\[ M : \mathbb{R}^{N \times v} \rightarrow \mathbb{R}^{N \times 1}; \ v \leq n = m + k \]  
(4)

2.2 Solution of a Neighboring Problem

The solution of a fault diagnosis in a complex industrial system by utilization of a neighboring problem has been discussed in details in [12]. It is assumed that for all specified target variables \(\hat{y}_i\) exists a mapping:

\[ \hat{y}_i = f_i(\tilde{x}_j) = L_i \cdot \tilde{x}_j \ \forall t \in I, \forall i, j \in \{1,...,n\}, i \neq j, \]  
(5)

where \(\tilde{x}_j\) denotes the relevant input variables for the target variable, and

\[ \hat{y}_i \in \mathbb{R}^N, \tilde{x}_j \in \mathbb{R} \]  

\[ \{\hat{y}_i \cup \tilde{y}\} \in \mathbb{R}^{N \times v}, L_i : \mathbb{R}^{N \times v} \rightarrow \mathbb{R}^N \]

In view of the complexity of production systems, the limitations of the measurement systems and the noise affecting it, any modeling effort will have to focus on an approximated problem. The important condition for approximation is that the perturbation of the data has to be as small as possible:

\[ \|\hat{y}_i - \tilde{y}_i\| \leq \xi, \forall t \in I = \{t_1,...,t_n\}, \forall j \in \{1,...,n\}, \xi \in \mathbb{R}^N \]  
(6)

where \(\tilde{x}_j \in \mathbb{R}^N\) represents the exact process variable which is not accessible. This condition can be achieved through preprocessing of measurement data by applying filtering, outlier removal and time delay detection and compensation.
The goal of the simplification of the original complex problem is to find a corresponding neighboring problem, which is easier to solve, so that both computational effort and modeling complexity are reduced. The simplification of the original problem can be done in two ways: by reduction in time behavior or by reduction of dimensionality.

The partial reduction in time behavior can be achieved by identification of stationary parts of variables. In some industrial plants, typical example being metallurgical processes, the transients are not as important as the stationary parts of the processes are. Then, if all the transients in the data are neglected, only static models can be used to describe behavior in the parts of the processes of interest.

The dimension of the search space is reduced by redundancy analysis in the first step and after that by variable selection for modeling. The redundant variables can be excluded from the set of possible inputs, but they can be used directly in fault diagnosis. After the simplification, the data set is represented with:

$$\tilde{x}_i(t'), \ i \in \{1,...,\lambda\}, t' \in I' = \{t'_1,...,t'_N\},$$

$$\lambda,N_S \in \mathbb{N}\{0\}, \lambda \leq n, N_S \leq N$$

where $\tilde{x}_i$ denotes the variables associated with the system $\Sigma'$, which is an approximation of the original system $\Sigma$ (Figure 1). $\lambda$ denotes the number of non-redundant variables and $N_S$ denotes the measurement samples belonging to the stationary segments.

Now it can be assumed that for all specified target variables $\tilde{y}_j$ exists a mapping:

$$\tilde{y}_j = f_j(\tilde{x}_i) = L_i \cdot \tilde{x}_j, \ \forall t \in I, \forall i,j \in \{1,...,\lambda\}, i \neq j,$$

where $\tilde{x}_j$ denotes the relevant input variables for the target variable, and

$$\tilde{x}_j \in \mathbb{R}^{N_{x,j}}, \ \lambda < n, \ \tilde{y}_j \in \mathbb{R}^{N_y}, \ L_i : \mathbb{R}^{N_{x,j}} \rightarrow \mathbb{R}^{N_y}.$$

Assuming the necessary information about the process is within the recorded data it can be shown that the following is valid:

$$\|L_i \tilde{x}_j - L_i \tilde{x}_j\| < \delta,$$

with $\delta \in \mathbb{R}^*$ sufficiently small.

### 2.3 Modeling for Fault Diagnosis

If two different models are compared, the following differences between them can be seen:

- Differences in model structure;
- Differences in model quality;
- Differences in sensitivity to input variables.

The use of several different modeling methods leads to formation of a pool of models, which represents analytical redundancy for the target channel $\tilde{y}_j$. Since $n$ process variables are measured, there are $n$ potential target variables, each with own analytical redundancy:

$$\Phi_j = \{M_{y_j}, M_{y_j}, ..., M_{y_j}\}, t, l \in \mathbb{N}/\{0\}, 0 < t \leq n$$

(10)

In general, the output from such a modeling process consists of a union of models calculated in each of $z$ used modeling methods, or:

$$\Phi = \bigcup_{i=1}^{z} \Phi_i$$

(11)

Use of analytical redundancy is necessary in fault diagnosis, but different levels of it are used at different steps in fault diagnosis. Fault detection demands for use of simple analytical redundancy, since it needs the existence of only one calculated value of the observed variable. Fault isolation on the other hand asks for functional redundancy, meaning several process models, structured in a way that isolation of faults can be achieved.

The principal goal of fault detection is to detect the existence of faults in the system, without consideration about its origin. The detection has to be done with minimum errors caused by inadequate prediction ability of used models. This means that prediction variance of the models has to be minimized, or speaking in terms of model quality, quality of the models has to be maximized. If define model quality measure $Q$ as quantitative value subject to $0 < Q < 1$, this leads to the following constrained optimization problem:

$$1 - Q(M_{y_j}) \rightarrow \min_{M_{y_j} \in \Phi} \ Q(M_{y_j}) \geq Q_{\text{min}}$$

(12)

The set of all feasible models from a model pool of the target variable is therefore:

$$Z_j := \{M_{y_j} \in \Phi | Q(M_{y_j}) \geq Q_{\text{min}}\}$$

(13)

It is clear that this optimization problem shall lead to a simple analytical redundancy, ending with a single, optimal model of the target variable:

$$\overline{M}_j \leftarrow \{M_{y_j} \in Z_j | Q(M_{y_j}) = \max_{M_{y_j} \in Z_j}\}$$

(14)

Note that the structural differences between models are not important for fault detection. Differently to fault detection, model based fault isolation requires (besides the requests on
appropriate model quality) structurally different models (to the certain extent). The modeling problem for fault isolation can be defined as:

$$M = \{ M \in \Phi | S(M) \text{ is isolable}, \forall x_i \in X \exists M \in M \},$$

(15)

Here $M$ represents a set of models for the system under investigation or any part thereof. $S(M)$ represents the structure of the set of models $M$.

3 Model-on-Demand Toolbox Design

The prototype of the MoD Toolbox is programmed in MATLAB, the engineering tool which has become almost a development standard. The prototype can be easily used via MATLAB based graphical user interface (GUI), which main menu is shown in Figure 2.

The structure of the toolbox follows the four functional blocks, as outlined in [11]: data acquisition, data preprocessing, data-based modeling and fault diagnosis.

The data acquisition block itself is not a part of the toolbox. The appropriate data interface serves as a connector between the data acquisition block and the MoD Toolbox.

The approach to fault diagnosis is roughly divided into two parts [11]: fast fault diagnosis which uses the approximated neighboring problem and precise fault diagnosis.

The function of the data preprocessing block is twofold: on one side conditioning of the raw process data in order to eliminate the influence of disturbances, to identify redundant variables and to eliminate variables carrying no useful process information and on another side to extract the neighboring approximation of the original process.

The selection of model structures, automatic data-based modeling and the following fault diagnosis are done using the neighboring problem, an approximation of the original system. Besides automatically generated data based models, also other process models like first principle models or models based on expert knowledge can be used. These models can also be used in fault diagnosis, with certain restrictions. In the fast fault diagnosis, residual based fault detection and weighted fault isolation, as outlined in [11], are used.

The whole data analysis process can be conducted either step-by-step or automatically. In the case of automatic conduction the general algorithm is given as follows:

*Algorithm 1: Initialize* (MoD Toolbox)

- define training data;
- define test data;
- define variable subset for analysis;

while (not end of variable set)

- run model training procedure;
- run fault diagnosis procedure;

if (necessary to refine fault diagnosis output)

- run MoD specific procedures;

end

end (of Algorithm 1)

The precise fault diagnosis tends to increase the correctness of the fast fault diagnosis, what can be done in two ways: improving fault detection and/or improving fault isolation. In order to do so with affordable computation effort, it is necessary to apply MoD specific procedures, which include extension of the modeling with respect to both extended measurement data base (limited move back from the neighboring problem) and structural demands for fault isolation.

The structural demands for fault isolation are reached through the orthogonalization and structura-

tlization procedures. The orthogonalization is applied in cases with multiple indistinctive fault isolations (after fast fault diagnosis) to variables which appear together in a single model. The aim of orthogonalization is to produce models which are structurally orthogonal to the original model with respect to the variables with indecisive diagnosis results. Since additional modeling for precise isolation inevitably leads to increased model base, it is necessary to apply the structuralization procedure in order to identify isolable substructures with respect to the variables with indecisive diagnosis results. The orthogonalization and the structuralization procedures are outlined in the following algorithms:
Algorithm 2: Initialize: (Orthogonalization)

- open model database;
- open result output file;

for (all models with multiple indistinctive isolations)
  for (all indistinctively isolated variables in that model)
    - precise modeling using orthogonal structure w.r.t. variable;
    if (model quality ≥ threshold)
      - add model to the model base;
  end
end

(end of Algorithm 2)

Algorithm 3: Initialize: (Structuralization)

- open model database;

while (not end of Algorithm 3)
  - form structural representation of the models from the model database;
  - decompose the common structure into isolable substructures with minimum dimensionality;
    for (each indistinctly isolated variable)
      - find isolable substructures with minimum dimensionality containing variable;
      - calculate substructure sensitivity parameter with respect to the variable;
      if (sensitivity parameter ≥ threshold)
        - add substructure to the substructure base;
    end
  end
while (other possible isolable n x m structures)
  - form n x m isolable structures from substructure basis with respect to the selected variable;
  if (n x m isolable structure is not unique)
    - additional orthogonal modeling;
    - additional structuralization;
  end
end

(end of Algorithm 3)

4 An Experimental Example

The functionality of the MoD MATLAB Toolbox will be illustrated on an example using measured process data from a steel plant. Because of confidentiality, the variables will be referenced with their numerical denotations used in measurement system rather than with their real names.

Data based models for various target channels are generated using the training data set. To avoid overfitting in model generation, the training data are split in proportion 1:2, where 1/3 of the training data was used for validation of models.

Since in examined steel plant (quasi)stationary parts of the process are much more important than short transients, the solution of the neighboring problem is satisfying regarding the process modeling, as well as the fast fault diagnosis.\[12\] Therefore, the example given here will focus on improvement of the fault diagnosis outcome, by applying the MoD procedures necessary to achieve improved structural isolability in the system.

For this example three automatically generated and selected data based models are used:

- Model 5: $[1:0] = f([2:1],[6:27],[10:16])$
- Model 2: $[2:0] = f([4:8],[5:22],[7:26])$
- Model 3: $[7:26] = f([4:13],[7:27],[10:16])$

All three models are selected according to their high model quality, so that they are all suitable for fault detection as well as for fault isolation. However, even the fault in Model 2 has been detected, the unique fault isolation was impossible. If the structural matrix given in Table 1 is checked, it can be easily seen that the fault signature $[M5,M2,M3] = [0,1,0]$ corresponds to faults in variables $[2:0]$, $[4:8]$ and $[5:22]$ simultaneously. Therefore, additional MoD was necessary. The orthogonalization of the Model 2, with respect to the variable $[5:22]$ as the target variable resulted in sufficiently Model 6 with sufficient model quality:

$[5:22] = f([1:0],[3:1],[4:0],[4:1],[10:16])$

The complete structural matrix (with Model 6 included) is given in Table 1. Fault detection using the Model 6 is shown in Figure 3, as well as the residuals from models $M5$, $M2$, $M3$ and $M6$.

Automatic decomposition of the model system with structural matrix given in Table 2 gives a total of 71 minimum strongly isolable 2 x 2 subsystems, 12 of them being related to the each variable under focus (i.e. $[2:0]$, $[4:8]$ and $[5:22]$). Not a single one of them could produce a unique fault signature. In other word, not a single subsystem could give unambiguous fault isolation. Therefore, it was necessary for each of these three variables to compose $n \times m$ strongly isolable subsystems, using as the basis the decomposition into 2 x 2 subsystems.

Table 1: Structural Matrix for Models $M5$, $M2$, $M3$ and $M6$

<table>
<thead>
<tr>
<th>$[1:0]$</th>
<th>$[2:0]$</th>
<th>$[3:1]$</th>
<th>$[4:0]$</th>
<th>$[4:1]$</th>
<th>$[4:8]$</th>
<th>$[5:22]$</th>
<th>$[6:27]$</th>
<th>$[7:26]$</th>
<th>$[10:16]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M5$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$M2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$M3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$M6$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2: Example decomposition into strongly isolable sub-systems

<table>
<thead>
<tr>
<th>$[1:0]$</th>
<th>$[2:0]$</th>
<th>$[3:1]$</th>
<th>$[4:0]$</th>
<th>$[4:1]$</th>
<th>$[4:8]$</th>
<th>$[5:22]$</th>
<th>$[6:27]$</th>
<th>$[7:26]$</th>
<th>$[10:16]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M5$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$M2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$M3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$M6$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Fig. 3 Fault detection using Models M5, M2, M3 and M6

The structural composition procedure for variables [2:0] and [4:8] did not result in a unique $n \times m$ subsystem, simply because the both channels share the same structural signature (Table 1). The structural composition for variable [5:22] resulted in a unique 4 x 3 strongly isolable subsystem (Table 2), which could be used to unambiguously isolate the fault in variable [5:22] through the fault signature $[\text{M5,M2,M3,M6}] = [0,1,0,1]$.

Table 2: 4x3 strongly isolable subsystem for Models M5, M2, M3 and M6

<table>
<thead>
<tr>
<th></th>
<th>1:0</th>
<th>5:22</th>
<th>7:26</th>
</tr>
</thead>
<tbody>
<tr>
<td>M5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>M2</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>M3</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>M6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

5 Conclusion

The complexity of modern technical systems is reflected also in the quantity of the measurement data recorded by their accompanied measurement systems. Nevertheless, even the quantity of measurements can be huge the essential information content in the measurements can sometimes be relatively low, making it necessary to analyze all available data in search for useful results. The difficulty here is to find a balance between demands and capability to analyze all available measurement data and accordingly to diagnose the state of the system in adequate time. A software prototype for fault diagnosis in complex industrial processes utilizing Model-on-Demand approach presented in this paper is aimed to be a solution for this difficulty. Experiments showed the ability of the software prototype to automatically analyze and diagnose complex systems, by applying a trade-off between computational efficiency and demands on precision of fault diagnosis in such systems.

Acknowledgement

The author thanks Prof. Dr Luigi del Re for helpful discussions about the presented topic. Also, the work of Mr. Martin Bernauer in programming of MoD MATLAB software is gratefully acknowledged.

References:

[7] Lorentzen, T. and Blanke, M., SaTool manual, Ver.1.0, Section of Automation at Ørsted•DTU, Techn.Univer. of Denmark, Kgs. Lyngby, Denmark, April 2004.