Linköping Studies in Science and Technology
Thesis No. 1406

Residual Generation Methods for Fault Diagnosis with Automotive Applications

Carl Svärd

Linköpings universitet
INSTITUTE OF TECHNOLOGY

Department of Electrical Engineering
Linköpings universitet, SE–581 83 Linköping, Sweden

Linköping 2009

Residual Generation Methods
for Fault Diagnosis with Automotive Applications

carl@isy.liu.se
http://www.vehicular.isy.liu.se
Department of Electrical Engineering,
Linköpings universitet,
SE–581 83 Linköping,
Sweden.

ISBN 978-91-7393-608-8
ISSN 0280-7971
LIU-TEK-LIC-2009:14

Printed by LiU-Tryck, Linköping, Sweden 2009

Abstract

The problem of fault diagnosis consists of detecting and isolating faults present in a system. As technical systems become more and more complex and the demands for safety, reliability and environmental friendliness are rising, fault diagnosis is becoming increasingly important. One example is *automotive systems*, where fault diagnosis is a necessity for low emissions, high safety, high vehicle uptime, and efficient repair and maintenance.

One approach to fault diagnosis, providing potentially good performance and in which the need for additional hardware is minimal, is model-based fault diagnosis with *residuals*. A residual is a signal that is zero when the system under diagnosis is fault-free, and non-zero when particular faults are present in the system. Residuals are typically generated by using a mathematical model of the system and measurements from sensors and actuators. This process is referred to as *residual generation*.

The main contributions in this thesis are two novel methods for residual generation. In both methods, systems described by Differential-Algebraic Equation (DAE) models are considered. Such models appear in a large class of technical systems, for example automotive systems. The first method consider *observer-based residual generation* for linear DAE-models. This method places no restrictions on the model, such as e.g. observability or regularity, in comparison with other previous methods. If the faults of interest can be detected in the system, the output from the design method is a residual generator, in state-space form, that is sensitive to the faults of interest. The method is iterative and relies on constant matrix operations, such as e.g. null-space calculations and equivalence transformations.

In the second method, non-linear DAE-models are considered. The proposed method belongs to a class of methods, in this thesis referred to as *sequential residual generation*, which has shown to be successful for real applications. This method enables simultaneous use of integral and derivative causality, and is able to handle equation sets corresponding to algebraic and differential loops in a systematic manner. It relies on a formal framework for computing unknown variables in the model according to a computation sequence, in which the analytical properties of the equations in the model as well as the available tools for equation solving are taken into account. The method is successfully applied to complex models of an automotive diesel engine and a hydraulic braking system.

Acknowledgments

This work has been performed as a part of a collaborative industrial research project between Scania CV AB in Södertälje and the division of Vehicular Systems, Department of Electrical Engineering, Linköping University.

First of all, I would like to express my gratitude to Dr. Mattias Nyberg, my supervisor, for great guidance into the world of research, his never-ending enthusiasm, and for always taking his time for discussions. I would also like to thank Dr. Erik Frisk, my assistant supervisor, for giving and interesting discussions, proof-reading, and help with numerous LaTeX issues. Professor Lars Nielsen is acknowledged for letting me join his research group Vehicular Systems.

Thanks also goes to all colleagues at Vehicular Systems and NESD for inspiring working atmospheres and nice coffee-breaks. Furthermore, I would like to thank Anna Pernestål for proof-reading parts of this manuscript, and Anders Eriksson, Peter Madsen, and Peter Vansölin, my managers at Scania, for letting me be a part of this project and do research.

This work is jointly financed by Scania CV AB and VINNOVA, Swedish Governmental Agency for Innovation Systems, who are also acknowledged.

Finally, I would like to thank my parents, Åsa and Kjell, my sister Anna and my friends for their support and encouragement.

Carl Svärd
Linköping, May 2009

Contents

Part I

Introduction to Model-Based Fault Diagnosis

1

Introduction

Fault diagnosis is the act of detecting and isolating faults present in a system. With the rising demand for safety and reliability of technical systems, driven by economical and environmental incentives, fault diagnosis has become increasingly important. One example is automotive systems, and in particular engines, that are by regulations required to have on-board diagnosis of all faults that may lead to increased emissions, see e.g. [United Nations, 2008]. In addition, fault diagnosis in automotive systems is essential to maintain high vehicle uptime, low fuel consumption, high safety, and efficient service and maintenance.

One approach to fault diagnosis that provides potentially good performance and in which the need for additional hardware is avoided, is model-based fault diagnosis with *residuals*. A residual is a signal that is zero when the system under diagnosis is fault-free, and non-zero when particular faults are present in the system. Residuals are often generated by utilizing a mathematical model of the system under diagnosis and measurements from sensors and actuators, a process referred to as *residual generation*. To enable fault isolation, a diagnosis system typically contains a set of residuals designed to respond to different subsets of faults. Meaning that some faults in a residual must be *decoupled*. Decoupling of faults in residuals is thus a fundamental problem in residual generation for fault isolation.

One important class of residual generation methods is *observer-based residual generation*. In these methods, the approach is to base residual generators on state-observers. A state-observer utilizes a model of the system and measurements to obtain an estimate of the states in the system. A residual can then

be formed as the difference between estimated and measured states. Several methods exists for design of observers for both linear and non-linear state-space models, i.e. ordinary differential equations with additional equations relating the states and measurements, as well as for linear and non-linear Differential-Algebraic Equation (DAE) models. DAE-models contains both differential and algebraic equations, and are of interest since these general models appear in a large class of technical systems, e.g. electrical-, mechanical-, and chemical systems. DAE-models are also the result when using object-oriented modeling tools, e.g. Modelica. In most of the observer-based residual generation methods, for both state-space and DAE-models, decoupling of faults is obtained by transforming the original model into a sub-model where only the faults of interest are present.

Another class of residual generation methods, that has shown to be successful for real applications, is in this thesis referred to as *sequential residual generation*. In sequential residual generation, the unknown variables in a model, or sub-model, are computed by solving equations one at a time in a sequence and a residual is then obtained by evaluating a redundant equation. In this class of methods, the original model is often divided into sub-models with specific properties and then residual generators are designed for each sub-model. Since a residual generator is only sensitive to those faults affecting its corresponding sub-model, all other faults are decoupled. Sequential residual generation methods has the potential to be automated to an high extent, making them especially important for the automotive applications studied in this thesis.

1.1 Overview and Contributions

Chapter 2 gives a brief introduction of theoretical concepts in model-based fault diagnosis that are central in this thesis. The aim of Chapter 2 is to provide a theoretical background to the rest of the thesis and to place its contributions in a context. Chapter 3 focuses on model-based fault diagnosis in automotive systems and intends to give an application oriented background and motivation to this work. Two papers are enclosed in Part II. These constitute the main contributions and are summarized below.

1.1.1 Paper 1 - Linear Observer-Based Residual Generation

In Paper 1, residual generation for linear DAE-models is considered. The main contribution is a new systematic design method for observer-based residual generation for systems described by linear DAE-models. By constant matrix operations, the original DAE-model is transformed into a sub-model in state-space form, of lower dimension than the DAE-model, where only faults that should be detected are present. Thus, faults not present in the transformed sub-model are decoupled. The transformation is iterative and straightforward

to implement. In contrast to other methods no restrictions, such as e.g. observability or regularity, are placed on the model of the system to be diagnosed. An illustrative numerical example is included, where the design method is applied to a non-observable model of a robot manipulator.

Paper 1 has been submitted to European Journal of Control. The paper is based on [Svärd and Nyberg, 2008c]:

> Svärd, C. and Nyberg, M. (2008). Observer-based residual generation for linear differential-algebraic equation systems. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea.

The work in the above conference paper has also been presented at *Reglermöte 2008*, Luleå, Sweden, [Svärd and Nyberg, 2008d].

1.1.2 Paper 2 - Non-Linear Sequential Residual Generation

The main contribution of Paper 2 is a novel method for sequential residual generation for non-linear DAE-models. The method relies on a formal framework for computing unknown variables according to a computation sequence, in which the analytical properties of the equations in the model and the available tools for algebraic equation solving are taken into account. An initial step in the method is to divide the original model into sub-models with specific properties, and residual generators for each sub-model are then designed. In this way, all faults not affecting the sub-model are decoupled in the corresponding residual generator. The proposed method is successfully applied to two models of automotive systems, a Scania diesel engine and a hydraulic braking system.

Paper 2 has been submitted to IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans. The paper is an extended and revised version of the work presented in [Svärd and Nyberg, 2008a]:

> Svärd, C. and Nyberg, M. (2008a). A mixed causality approach to residual generation utilizing equation system solvers and differential-algebraic equation theory. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, Blue Mountains, Australia.

An extended version of the above conference paper can be found in [Svärd and Nyberg, 2008b].

2

Model-Based Fault Diagnosis

The aim of this chapter is to introduce some theoretical concepts in model-based fault diagnosis that are central in this thesis, and to place the contributions presented in Part II in a context.

2.1 Models

To perform model-based diagnosis, a model of the system under diagnosis is needed. In this thesis, a model is a set of equations relating sets of unknown and known variables. The equations may be linear or non-linear, static or dynamic. That is, linear and non-linear Differential-Algebraic Equation (DAE) models are considered. Typically, faults that may affect the system are also included in the model. Faults are often classified into behavioral modes. For example, behavioral modes for a simple system containing one sensor and one actuator may be "sensor fault", "actuator fault", and "no fault". Behavioral modes are usually assigned to components, here we instead use them for systems.

2.2 Diagnostic Tests

A typical diagnosis system consists of a set of diagnostic tests and a fault isolation scheme, see Figure 2.1. A diagnostic test utilizes observations, i.e. measurements, from the system under diagnosis to determine if a specific behavioral mode is present in the system or not. A diagnostic test δ_i, can be viewed

as a *hypothesis test* [Berger, 1985] with the hypothesis

$$\begin{aligned} H_i^0 : &\quad F_p \in B_i \\ H_i^1 : &\quad F_p \in B_i^C \end{aligned}$$

where F_p denotes the present behavioral mode in the system, B_i a set of behavioral modes corresponding to faults not monitored by δ_i, and B_i^C the complement of B_i, see e.g. [Nyberg, 1999]. The common convention used is that when the hypothesis H_i^0 is rejected, it is assumed that H_i^1 is true. When H_i^0 is not rejected, nothing is assumed which means that the present behavioral mode can be any of the behavioral modes for the system under diagnosis. The outcome of the diagnostic test δ_i is thus a decision

$$S_i = \begin{cases} S_i^1 = B_i^C & \text{if } H^0 \text{ is rejected} \\ S_i^0 = \Omega & \text{if } H^0 \text{ is not rejected} \end{cases} \tag{2.1}$$

where Ω denotes all behavioral modes for the system.

Common traditional approaches for construction of diagnostic tests are for example *limit checking*, i.e. to check if a sensor is within its normal operating range, or to employ *hardware redundancy*. For instance, if two sensors are used to measure the same physical quantity, it is possible to test if one of the sensors is faulty by comparing the values of the sensors. Another approach, providing potentially increased diagnosis performance and in which the need of additional, redundant, hardware is avoided, is to use diagnostic tests based on *residuals*.

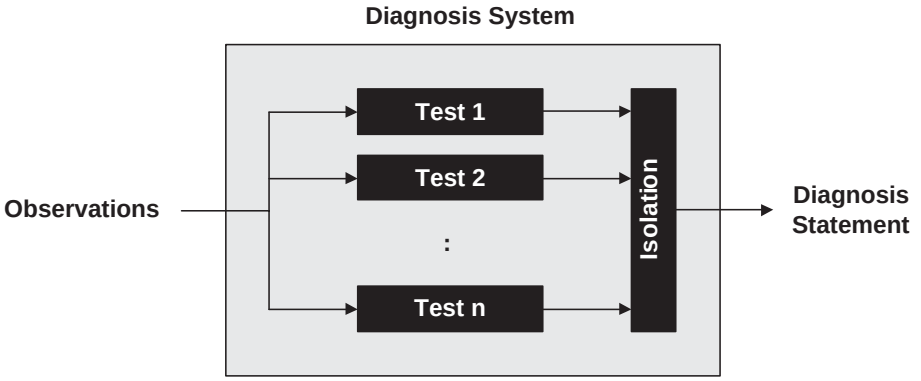


Figure 2.1: A typical diagnosis system consists of a set of diagnostic test and a fault isolation scheme.

2.3 Diagnostic Tests Based on Residuals

A residual is a signal ideally zero in the non-faulty case and non-zero else. A *residual generator* takes measurements from the system under diagnosis as input, and produces a residual as output, see Figure 2.2. Residual generators are typically constructed by using a mathematical model of the system. For instance, a residual can be obtained as the comparison between a value estimated by a model and the corresponding measured quantity. The residual generator consists in this case of the model used for the estimation and the equation describing the comparison, referred to as the *residual equation*. Two methods for residual generation are presented in this thesis. The method presented in Paper 1 handles linear DAE-models, and the method in Paper 2 non-linear DAE-models.

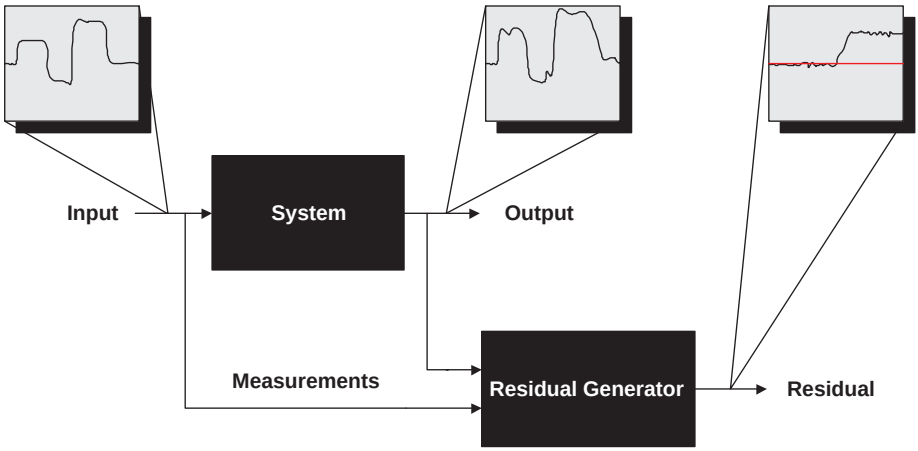


Figure 2.2: A residual can be generated by utilizing a mathematical model of the system under diagnosis and measurements.

2.3.1 Test Quantity

A common way to construct a diagnostic test based on a residual is to form a *test quantity* from the residual, and then threshold the test quantity, see Figure 2.3. A test quantity is a constant value, in comparison with a residual which is a trajectory, i.e. a function of time. A test quantity can for example be formed as the mean-effect or mean-value of the residual in some time-window, or just as a sample of the residual at a specific time. Simply, given a residual r, generated by using a model and measurements $\mathbf{z}$, a diagnostic test δ_i constructed

via a test quantity T, based on r, is defined as

$$S_i = \delta(\mathbf{z}) = \begin{cases} S_i^1 & \text{if } T(r(\mathbf{z})) \geq J \\ S_i^0 & \text{if } T(r(\mathbf{z})) < J \end{cases}$$

where J is a threshold. Typically, residuals are not perfectly zero in the non-faulty case due to for example noisy measurements and modeling errors. Thus, the approach used to form the test quantities and the thresholds are important design parameters in a diagnosis system.

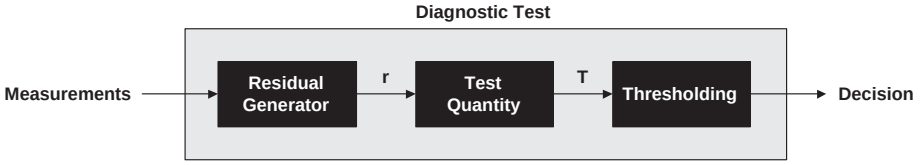


Figure 2.3: A diagnostic test based on a residual via a test quantity.

2.4 Fault Isolation

There are several approaches for fault isolation, most originating from the field of Artificial Intelligence (AI), see e.g. [de Kleer and Williams, 1987]. Another approach is Bayesian fault isolation, see e.g. [Pernestål, 2007]. Here, in order to briefly illustrate the concept of fault isolation, we will use a straight-forward method referred to as *structured residuals*, [Gertler, 1991], or *structured hypothesis tests*, [Nyberg, 1999].

To enable isolation of faults, the diagnostic tests used in a diagnosis system are designed to test different behavioral modes. Consider a diagnosis system containing the diagnostic tests $\{\delta_1, \delta_2, \ldots, \delta_n\}$. The outcome of the diagnostic test δ_i is a decision S_i, according to (2.1). Under a single fault assumption, we simply obtain the total diagnosis statement S as

$$S = \bigcap_{i=1}^{n} S_i,$$

for multiple faults please refer to e.g. [de Kleer and Williams, 1987]; [Reiter, 1987]; [Greiner et al., 1989].

For an example, consider a set of tests, $\{\delta_1, \delta_2, \delta_3\}$, constructed to detect and isolate three faults, $\{f_1, f_2, f_3\}$. The following *fault signature matrix*,

	f_1	f_2	f_3
δ_1		1	1
δ_2	1		1
δ_3	1	1	

(2.2)

shows which tests that are sensitive to which faults, i.e. test δ_1 is sensitive to faults f_2 and f_3, and so on. Now assume a situation where tests δ_1 and δ_2, but not δ_3 has reacted. We then obtain the decisions $S_1 = \{f_2, f_3\}$, $S_2 = \{f_1, f_3\}$, and $S_3 = \{f_1, f_2, f_3, NF\}$, where NF is used to denote the behavioral mode corresponding to that no faults are present. The diagnosis statement thus becomes

$$S = S_1 \cap S_2 \cap S_3 = \{f_2, f_3\} \cap \{f_1, f_3\} \cap \{f_1, f_2, f_3, NF\} = f_3$$

and we can conclude that fault f_3 is present.

2.5 Fault Decoupling

To achieve a specific fault signature matrix, for example one similar to (2.2), *decoupling* of faults in diagnostic tests is needed. The faults that are decoupled in a test are often referred to as *non-monitored faults*, whereas the faults not decoupled are called *monitored faults*. In the example above, fault f_1 is decoupled in test δ_1, which means that for δ_1, fault f_1 is a non-monitored fault and f_2 and f_3 are monitored faults. Decoupling of faults in a set of tests based on residuals, means that the residuals must respond to, or similarly be sensitive to, different subsets of faults. Thus, fault decoupling is a fundamental problem in residual generation for fault isolation.

2.6 Residual Generation

In this thesis, two classes of residual generation methods are considered, observer-based residual generation and sequential residual generation. These both classes have the potential to handle DAE-models, and to handle fault decoupling in a systematic manner. DAE-models are of interest since such models appear in a large class of technical system, e.g. automotive systems, and also are the result when using object-oriented modeling tools such as e.g. Modelica, [Fritzon, 2004].

2.6.1 Observer-Based Residual Generation

A common approach is, as said in Section 1, to base residual generators on state-observers. A residual is in this case formed as the difference between estimated and measured states. Several methods exists for design of observers for state-space models, see e.g. [Kailath et al., 2000] for linear models, and [Hendeby, 2008]; [Misawa and Hedrick, 1989]; [Walcott et al., 1987]; [Slotine et al., 1987]; [Khalil, 1999] for non-linear models. For linear DAE-models see e.g. [Hou and Müller, 1999]; [Hou and Müller, 1995]; [Darouach and Boutayeb, 1995];

[Müller and Hou, 1993]; [Shields, 1992]; [Dai, 1989]. For non-linear DAE-models the list of works is not that extensive, but includes for example [Åslund and Frisk, 2006]; [Becerra et al., 2001]; [Zimmer and Meier, 1997]. For the specific application of using the observer for diagnosis, see for example [Massoumnia, 1986]; [Massoumnia et al., 1989]; [Hammouri et al., 2001] for linear state-space models, and [Hammouri et al., 1999]; [De Persis and Isidori, 2001]; [Martínez-Guerra et al., 2005]; [Kaboré et al., 2000] for non-linear state-space models. Several methods also exists for observer-based residual generation in linear DAE-models, for example [Hou, 2000]; [Patton and Hou, 1998]; [Shields, 1994]; [Marx et al., 2003] and some for non-linear DAE-models e.g. [Gao and Ding, 2007]; [Vemuri et al., 2001]; [Shields, 1997]. In most of the works above, for both state-space models and DAE-models, decoupling of faults is obtained by transforming the original model into a sub-model where only the faults of interest are present. Observer-based residual generation for linear DAE-models is considered in Paper 1.

2.6.2 Sequential Residual Generation

Sequential residual generation, [Staroswiecki and Declerck, 1989], is of interest since it has shown to be successful for real applications, [Dustegor et al., 2004]; [Izadi-Zamanabadi, 2002]; [Cocquempot et al., 1998]; [Hansen and Molin, 2006]; [Kingstedt and Johansson, 2008]; [Dagson and Nissilä-Källström, 2009], and in addition also has the potential to be automated to a high extent, [Frisk et al., 2006]; [Einarsson and Arrhenius, 2005]; [Krigsman and Nilsson, 2005]; [Eriksson, 2005]; [Svärd and Wassén, 2006]. In sequential residual generation, the unknown variables in a model, or sub-model, are computed by solving equations one at a time in a sequence and a residual is then obtained by evaluating a redundant equation. Similar approaches as in [Staroswiecki and Declerck, 1989], have been described and exploited in e.g. [Staroswiecki, 2002]; [Blanke et al., 2003]; [Pulido and Alonso-González, 2004]. In this class of methods, the original model is often divided into sub-models with specific properties and residual generators are then designed for each sub-model. Since a residual generator is only sensitive to those faults affecting its corresponding sub-model, all other faults are decoupled. Sequential residual generation is considered in Paper 2.

3

Model-Based Fault Diagnosis in Automotive Systems

Modern automotive systems are complex. One example is automotive diesel engines, see Figure 3.1, that in order to have low fuel consumption, produce low emissions, and offer good driveability, are equipped with for example Exhaust Gas Recirculation (EGR) and a Variable Geometry Turbocharger (VGT). To purify exhausts, diesel engines interact with and are dependent on one or several advanced after-treatment systems such as a Diesel Particulate Filter (DPF), and a Selective Catalytic Reduction (SCR) system, see Figure 3.2(b). In addition, to provide optimum fuel economy, good safety, and further increase driveability, they interact with other complex systems in the powertrain like an automatic gearbox and an auxiliary hydraulic braking system, see Figure 3.3. Even small faults in the engine or in any of the systems mentioned above may have undesirable effects, such as increased emissions or reduced safety. The objectives of this chapter are to provide a background and motivation to this thesis and to place its contributions into an application oriented context.

3.1 Why Fault Diagnosis is Important

Faults affecting the engine or any of the systems mentioned above may lead to

- increased emissions,
- decreased safety,
- increased fuel consumption,

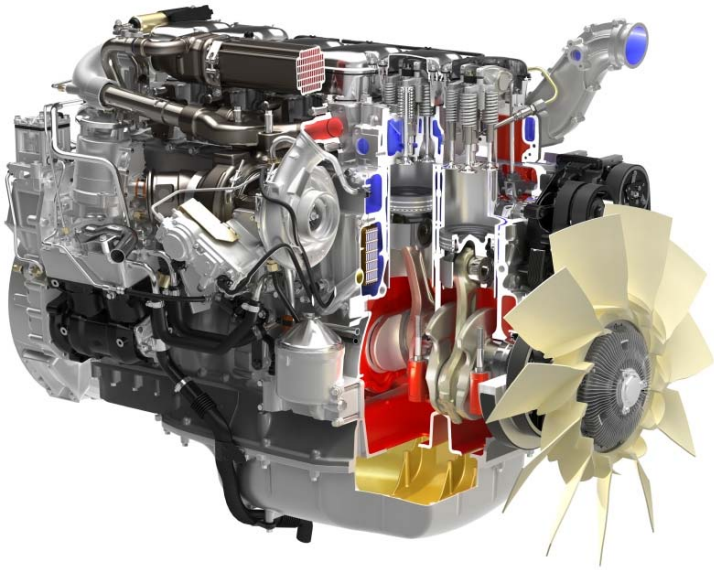


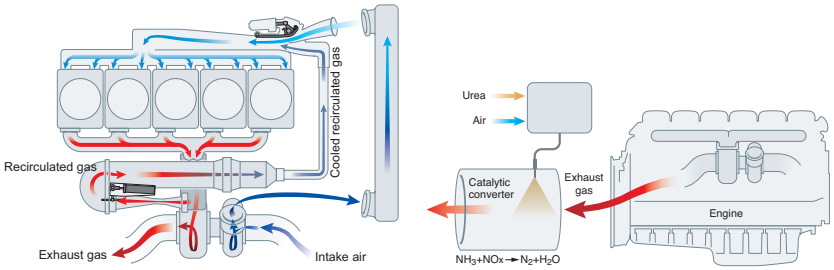
Figure 3.1: A Scania 13-liter, 6-cylinder diesel engine equipped with EGR and VGT. Illustration is due to Semcon Informatic Graphic Solutions.

- decreased driveability, or
- vehicle off-road.

These consequences may be prevented, or at least reduced, if faults can be detected and isolated in time. In addition, beside these more or less obvious gains, good diagnosis is a requirement for high vehicle uptime and efficient maintenance, regarding both cost and time. These aspects are further discussed below.

3.1.1 Emissions

Automotive engines are by regulations required to have high-precision On-Board Diagnosis (OBD) of faults that are harmful for the environment, see e.g. [United Nations, 2008]. The legislations states that all manufactured vehicles must be equipped with an OBD-system capable of detecting faults in all components that, if broken, leads to emissions over pre-defined OBD thresholds during a specific driving cycle. For heavy-duty trucks, emissions of especially nitrogen oxides (NOx) and particulate matter (PM) are crucial. Coming legislations in the European Union, EUVI, require substantially lowered emission and OBD thresholds, see Figures 3.4 and 3.5, and in addition that faults leading to increased emissions can be isolated.



(a) EGR system schematic. (b) SCR system schematic.

Figure 3.2: Usage of EGR and/or SCR in diesel engine reduces the generation of NOx. Illustrations are due to Semcon Informatic Graphic Solutions.

3.1.2 Vehicle Uptime

To reduce, or preferably eliminate, the impact of faults by taking appropriate actions on the road, refereed to as *fault tolerant control*, see e.g. [Blanke et al., 2003], on-board diagnosis is essential. For example, if a fault occurs but the fault can be detected and isolated on-board so that the effects of the fault can be eliminated, the vehicle can continue on its driving mission and stop by the workshop later. On-board diagnosis therefore increases the vehicle uptime. Vehicle uptime is important for vehicle owners, since even a stationary vehicle costs money, and can not be used to earn money.

3.1.3 Efficient Repair and Maintenance

On-board diagnosis of faults is also important to provide efficient service when the vehicle visits the workshop. If faults have been correctly detected and isolated, additional troubleshooting at the workshop is unnecessary. However, as automotive systems become more and more complex it is utopian that all necessary fault detection and isolation can be performed on-board the vehicle. Therefore *off-board* fault detection and isolation of faults, i.e. at the workshop, is becoming more and more important. Due to hardware limitations on-board the vehicle and the ability to actively excite systems when the vehicle is at the workshop, off-board fault detection and isolation of faults may also give better and more precise results. Nevertheless, fault detection and isolation, on- or/and off-board, decreases the repair and maintenance costs for the vehicle, since the time at the workshop is minimized and no unnecessary parts are changed.

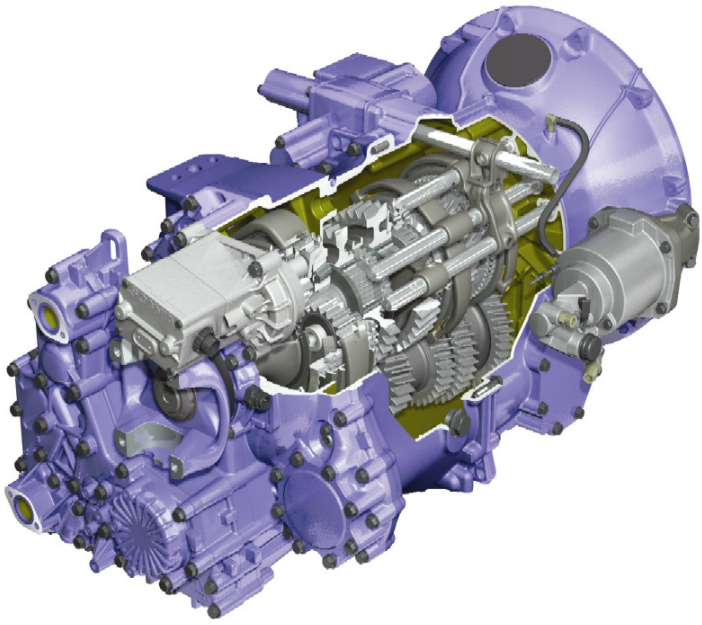


Figure 3.3: Scania GR875R 8-speed gearbox with a retarder. The retarder is a hydraulic braking system used for on heavy duty trucks for long continuous braking, for example to maintain constant speed down a slope. Illustration is due to Semcon Informatic Graphic Solutions.

3.2 Faults to Diagnose

To investigate which faults that need to be considered, Failure Mode Effect Analysis (FMEA) [Stamatis, 1995] and Fault Tree Analysis (FTA) [Haasl et al., 1981] may be successful approaches. Furthermore, as said above, legislations require that all faults in the engine, or in its surroundings, that results in increased emissions must be detected and in some cases also isolated. Much effort is therefore also spent on testing engines in test-cells where faults can be injected and emissions measured, with the objective to see which faults that may lead to increased emissions.

3.2.1 Fault Types

Faults that must be diagnosed in, or around, the engine are for instance faults affecting the fuel injection system, the cooling system, and the gas-flow system, faults in all sensors and actuators, and faults affecting after-treatment systems like the SCR-system and the DPF. Common fault types are electrical faults,

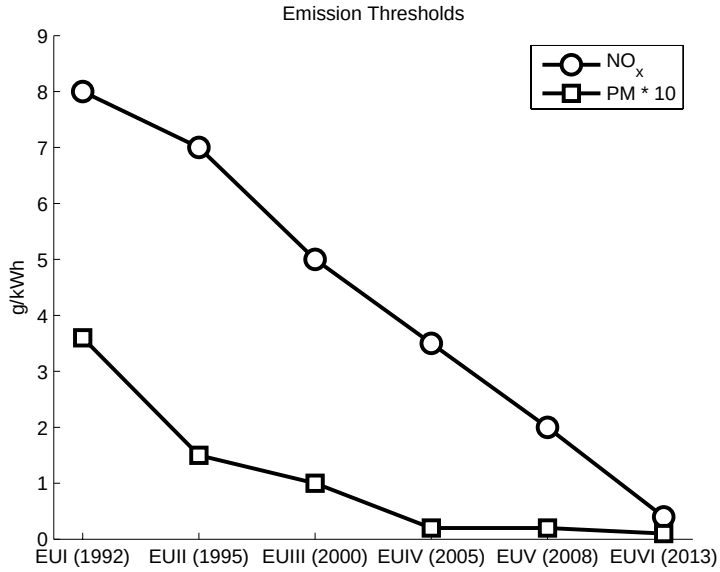


Figure 3.4: Legislations require lowered emission thresholds for heavy-duty trucks in the European Union. The line with circle markers shows NOx emission thresholds. The line with dotted markers shows thresholds for PM emissions scaled with a factor 10.

plausibility faults, and functional faults. A plausibility fault is for example a sensor giving wrong value, possibly caused by a bias or gain. A functional fault may for instance be a non-working feedback control loop. Most fault types are however specific for each system, for instance air-leakage in the VGT- or EGR-system, bad UREA quality in the SCR-system, and broken or missing filter substrate in the DPF. Sensors and actuators are in general complex electro-mechanical systems. These systems are particularly sensitive to faults, in comparison with for example purely mechanical systems. Hence, it is important that especially faults in sensors and actuators in automotive systems can be detected and isolated.

3.3 Residual Generation for Automotive Systems

Due to economical reasons and space limitations, it is not a desired option to mount additional hardware in order to diagnose faults. As said in Chapter 2, one approach to fault diagnosis, providing potentially good performance and in which no additional hardware is needed, is model-based fault diagnosis with residuals.

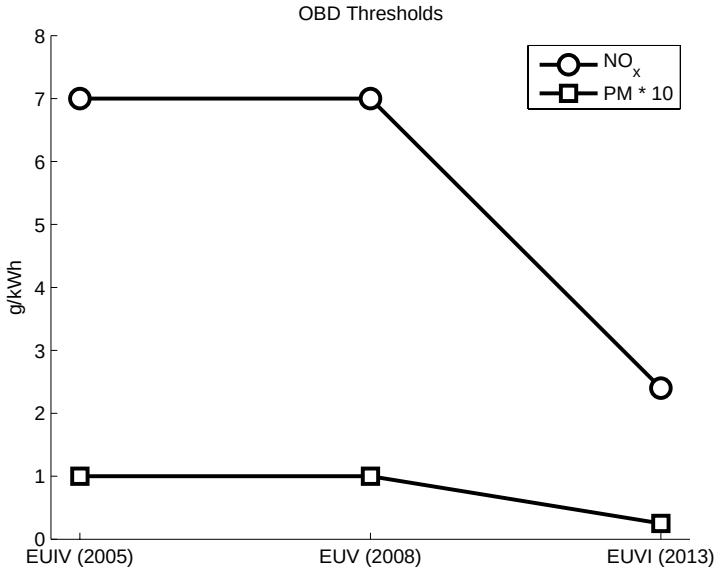


Figure 3.5: Legislations require lowered OBD thresholds for heavy-duty trucks in the European Union. The line with circle markers shows NOx emission thresholds. The line with dotted markers shows thresholds for PM emissions scaled with a factor 10.

3.3.1 Models

For model-based fault diagnosis, a model of the system under diagnosis is needed. Above we concluded that modern automotive diesel engines as well as their surrounding systems are complex. To describe the dynamic courses in these systems, physical modeling is an often utilized approach, in which models are based on first principles of physics. One popular and successful approach is to use physical object-oriented modeling tools, e.g. Modelica [Fritzon, 2004]. For a large class of technical systems, such as mechanical-, electrical, and chemical systems, these approaches generally results in complex non-linear equation systems containing both algebraic and differential equations, i.e. non-linear DAE-systems. When considering complex automotive systems, it is thus important that the residual generation method is able to handle such models.

3.3.2 Design Process

The view taken in this thesis, as in e.g. [Nyberg and Krysander, 2008], [Nyberg, 1999], is that design of a diagnosis system is a two-step approach, see

Figure 3.6. In a first step, a large number of candidate residual generators are found, and in a second step the set of residual generators most suitable to be included in the final diagnosis system is picked out. The set of residual generators to be used in the final diagnosis system must in the second step be chosen so that desired fault detection and isolation performance is achieved. To do this, it is necessary to evaluate many candidate residual generators with real measurement data in order to investigate sensitivity to faults in the presence of disturbances, modeling errors, measurement noise, etc. Therefore it is for the second step important that there is a large selection of different candidate residual generators to choose between. Thus, the initial set of candidate residual generators should be as large as possible.

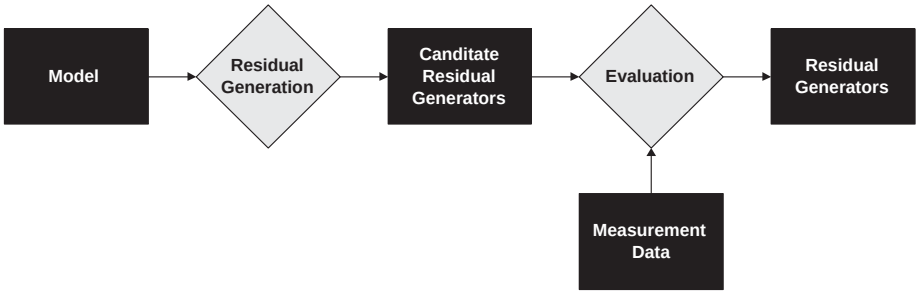


Figure 3.6: Design of a diagnosis system is a two-step approach. In the first step, a large number of candidate residual generators are found, and in the second step the set of residual generators to be used in the final diagnosis system is picked out. This is done by evaluating candidate residual generators with measurement data.

3.3.3 Methods

As argued above, it is desirable that a method for residual generation intended to be used for automotive systems is able to handle DAE-models. In addition, as said in Section 2.5, decoupling of faults is a fundamental problem in residual generation. When aiming at finding as many candidate residual generators as possible, it is also highly desirable that the method used for residual generation is automated. One class of residual generation methods having the potential to handle all of these issues, is sequential residual generation. Sequential residual generation with application to automotive systems is considered in Paper 2.

3.4 Industrial Relevance

As said earlier, model-based diagnosis with residuals is one of many approaches for design of diagnosis systems. The main argument for not using model-

based approaches is the lack of adequate accurate models. It is true that modeling may be time-consuming and also that once a model is created, it must be validated and tuned which may require additional effort and access to measurement data. In addition, models also need to be kept updated to be useful. Therefore, model-based approaches require a well defined engineering process that supports this way of working and takes mentioned aspects into account. However, there are efficient object-oriented modeling tools such as Simulink or Modelica that can be used to facilitate this process, and several established engineering tools supporting model-based development, e.g. Real-Time Workshop. Furthermore, as systems become more and more complex, models are needed for other purposes than diagnosis system development, for example simulation and development of control systems. These models can likewise, perhaps with small modifications, be used for development of diagnosis systems as long as they describe the system under diagnosis. This is for example the case for the models used in the application study in Paper 2, which are developed for simulation purposes.

Another argument is that model-based diagnostic tests based on residuals tend to be hard to run in real-time in computers on-board for example trucks. This is due to severe hardware limitations, in terms of CPU power and memory. The matter of the fact is however that technical systems become more and more advanced and complex. It is therefore reasonable that the hardware on-board these systems evolves in the same pace. The method presented in Paper 2 gives residual generators where variables are computed sequentially. Computing variables in this way is suitable for real-time execution. Further, it is not necessary to run all tests in a diagnosis system at the same time or even in real-time. For example, one set of tests can be run for fault detection and once a fault is detected, another set of tests can be run for fault isolation. If there is memory available, data can be saved and the isolation tests can as well be run later, i.e. not in real-time. In addition, for a given detection and isolation performance, it is not that certain that a diagnosis system developed with a systematic model-based approach requires more CPU power and memory, in comparison with a diagnosis system developed through some "ad-hoc" approach. It is likely that the set of tests contained in the model-based system can be more tailor made, through for example fault decoupling. Moreover, for a given detection and isolation performance, a model-based diagnosis system would probably require fewer sensors than a non model-based diagnosis system, since models instead of hardware are used to provide necessary redundancy. This means an over-all cost reduction. Another aspect is that it is not necessary that all diagnosis is performed on-board, see Section 3.1.3. When diagnosis instead is done off-board, diagnostic tests can be run in ordinary computers stationed in the workshop, and thus limitations in hardware are not an issue.

It is the author's strong belief that model-based fault diagnosis, or model-based development in general, is a necessity for being able to meet future de-

mands on safety, reliability, environmental friendliness, and performance of automotive systems. It is believed that usage of model-based approaches for design of diagnosis systems increases productivity and simplifies the overall design process for diagnosis systems, since many steps in the process can be automated. For instance, a large set of diagnostic tests, or residual generators, can be automatically generated given a model with the method presented in Paper 2. If conditions are changed, the model can be updated and a new set of tests can easily be generated, meaning reconfigurability.

Bibliography

[Åslund and Frisk, 2006] Åslund, J. and Frisk, E. (2006). An observer for non-linear differential-algebraic systems. *Automatica*, 42(6):959–965.

[Becerra et al., 2001] Becerra, V. M., Roberts, P. D., and Griffiths, G. W. (2001). Applying the extended kalman filter to systems described by nonlinear differential-algebraic equations. *Control Engineering Practice*, 9(3):267 – 281.

[Berger, 1985] Berger, J. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer.

[Blanke et al., 2003] Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2003). *Diagnosis and Fault-Tolerant Control*. Springer.

[Cocquempot et al., 1998] Cocquempot, V., Izadi-Zamanabadi, R., Staroswiecki, M., and Blanke, M. (1998). Residual generation for the ship benchmark using structural approach. In *Proceedings of the UKACC International Conference on Control '98*, pages 1480–1485.

[Dagson and Nissilä-Källström, 2009] Dagson, J. and Nissilä-Källström, S. (2009). Air leakage diagnosis in heavy duty truck engines with egr and vgt. Master's thesis, Linköpings Universitet, SE-581 83 Linköping. LITH-ISY-EX-09/4210-SE.

[Dai, 1989] Dai, L. (1989). *Singular Control Systems*. Springer-Verlag.

[Darouach and Boutayeb, 1995] Darouach, M. and Boutayeb, M. (1995). Design of observers for descriptor systems. *IEEE Transactions on Automatic Control*, 40(7):1323–1327.

[de Kleer and Williams, 1987] de Kleer, J. and Williams, B. C. (1987). Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130.

[De Persis and Isidori, 2001] De Persis, C. and Isidori, A. (2001). A geometric approach to nonlinear fault detection and isolation. *IEEE Transactions on Automatic Control*, 46:853–865.

[Dustegor et al., 2004] Dustegor, D., Cocquempot, V., and Staroswiecki, M. (2004). Structural analysis for residual generation: Towards implementation. In *Proceedings of the 2004 IEEE Inter. Conf. on Control App.*, pages 1217–1222.

[Einarsson and Arrhenius, 2005] Einarsson, H. and Arrhenius, G. (2005). Automatic design of diagnosis systems using consistency based residuals - optimizing isolation and computational load. Master's thesis, Uppsala University. UPTEC F 1401-5757 ; 05032.

[Eriksson, 2005] Eriksson, L. (2005). Structural algorithms for diagnostic system design using simulink models. Master's thesis, Linköpings Universitet, SE-581 83 Linköping. LiTH-ISY-EX-3601-2004.

[Frisk et al., 2006] Frisk, E., Krysander, M., Nyberg, M., and Åslund, J. (2006). A toolbox for design of diagnosis systems. In *Proceedings of IFAC Safeprocess'06*, Beijing, China.

[Fritzon, 2004] Fritzon, P. (2004). *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. IEEE Press.

[Gao and Ding, 2007] Gao, Z. and Ding, S. X. (2007). Actuator fault robust estimation and fault-tolerant control for a class of nonlinear descriptor systems. *Automatica*, 43(5):912 – 920.

[Gertler, 1991] Gertler, J. (1991). Analytical redundancy methods in fault detection and isolation; survey and analysis. In *IFAC Fault Detection, Supervision and Safety for Technical Processes*, pages 9–21, Baden-Baden, Germany.

[Greiner et al., 1989] Greiner, R., Smith, B. A., and Wilkerson, R. W. (1989). A correction to the algorithm in reiter's theory of diagnosis. *Artificial Intelligence*, 41:79–88.

[Haasl et al., 1981] Haasl, D., Roberts, N., Vesely, W., and Goldberg, F. (1981). *Fault Tree Handbook*. U.S. Nuclear Regulatory Commission.

[Hammouri et al., 2001] Hammouri, H., Kabore, P., and Kinnaert, M. (2001). A geometric approach to fault detection and isolation for bilinear systems. *IEEE Transactions on Automatic Control*, 46(9):1451–1455.

[Hammouri et al., 1999] Hammouri, H., Kinnaert, M., and El Yaagoubi, E. H. (1999). Observer-based approach to fault detection and isolation for nonlinear systems. *IEEE Transactions on Automatic Control*, 44(10):1879–1884.

[Hansen and Molin, 2006] Hansen, J. and Molin, J. (2006). Design and evaluation of an automatically generated diagnosis system. Master's thesis, Linköpings Universitet, SE-581 83 Linköping.

[Hendeby, 2008] Hendeby, G. (2008). *Performance and Implementaion Aspects of Nonlinear Filtering*. PhD thesis, Linköpings universitet.

[Hou, 2000] Hou, M. (2000). *Fault detection and isolation for descriptor systems*, chapter 5. Issues of Fault Diagnosis for Dynamic Systems. Springer-Verlag.

[Hou and Müller, 1999] Hou, M. and Müller, P. (1999). Observer design for descriptor systems. *IEEE Transactions on Automatic Control*, 44(1):164–168.

[Hou and Müller, 1995] Hou, M. and Müller, P. C. (1995). Design of a class of luenberger observers for descriptor systems. *IEEE Transactions on Automatic Control*, 40(1):133–136.

[Izadi-Zamanabadi, 2002] Izadi-Zamanabadi, R. (2002). Structural analysis approach to fault fiagnosis with application to fixed-wing aircraft motion. In *Proceedings of the 2002 American Control Conference*, volume 5, pages 3949–3954.

[Kaboré et al., 2000] Kaboré, P., Othman, S., McKenna, T. F., and Hammouri, H. (2000). Observer-based fault diagnosis for a class of non-linear systems - application to a free radical copolymerization reaction. *International Journal of Control*, 73(9):787–803.

[Kailath et al., 2000] Kailath, T., Sayed, A. H., and Hassibi, B. (2000). *Linear Estimation*. Prentice-Hall, Inc, Upper Saddle River, New Jersey 07458, 2 edition.

[Khalil, 1999] Khalil, H. (1999). *New Directions in Nonlinear Observer Design (Lecture Notes in Control and Information Sciences)*, chapter High-gain observers in Nonlinear Feedback Control. Springer.

[Kingstedt and Johansson, 2008] Kingstedt, J. and Johansson, M. (2008). Methods for residual generation using mixed causality in model based diagnosis. Master's thesis, Linköpings Universitet, SE-581 83 Linköping.

[Krigsman and Nilsson, 2005] Krigsman, K. and Nilsson, J. (2005). Code generation for efficient real-time execution of diagnostic residual generators. Master's thesis, Dept. of Microelectronics and Information Technology, KTH. IMIT/LECS-2004-66.

[Martínez-Guerra et al., 2005] Martínez-Guerra, R., Garrido, R., and Osorio-Miron, A. (2005). The fault detection problem in nonlinear systems using residual generators. *IMA Journal of Mathematical Control and Information*, 22(2):119–136.

[Marx et al., 2003] Marx, B., Koenig, D., and Georges, D. (2003). Robust fault diagnosis for linear descriptor systems using proportional integral observers. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 457–462, Maui, Hawaii, USA.

[Massoumnia, 1986] Massoumnia, M. (1986). A geometric approach to the synthesis of failure detection filters. *IEEE Transactions on Automatic Control*, 31(9):839–846.

[Massoumnia et al., 1989] Massoumnia, M., Verghese, G., and Willsky, A. (1989). Failure detection and isolation. *IEEE Transactions on Automatic Control*, 34(3):316–321.

[Misawa and Hedrick, 1989] Misawa, E. A. and Hedrick, J. K. (1989). Nonlinear observers - a state of the art survey. *Transactions of the ASME*, 111(344).

[Müller and Hou, 1993] Müller, P. and Hou, M. (1993). On the observer design for descriptor systems. *IEEE Transactions on Automatic Control*, 38(11):1666–1670.

[Nyberg, 1999] Nyberg, M. (1999). Automatic design of diagnosis systems with application to an automotive engine. *Control Engineering Practice*, 87(8):993–1005.

[Nyberg and Krysander, 2008] Nyberg, M. and Krysander, M. (2008). Statistical properties and design criterions for AI-based fault isolation. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea.

[Patton and Hou, 1998] Patton, R. J. and Hou, M. (1998). Design of fault detection and isolation observers: A matrix pencil approach. *Automatica*, 34(9):1135–1140.

[Pernestål, 2007] Pernestål, A. (2007). *A Bayesian Approach to Fault Isolation with Application To Diesel Engine Diagnosis*. Lic. Thesis, Royal Institute of Technology, Stockholm, Sweden.

[Pulido and Alonso-González, 2004] Pulido, B. and Alonso-González, C. (2004). Possible conflicts: a compilation technique for consistency-based diagnosis. *"IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics", Special Issue on Diagnosis of Complex Systems*, 34(5):2192–2206.

[Reiter, 1987] Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95.

[Shields, 1994] Shields, D. (1994). Robust fault detection for generalized state space systems. In *Proceedings International Conference Control*, pages 1335–1339.

[Shields, 1992] Shields, D. N. (1992). Observers for descriptor systems. *International Journal of Control*, 55(1):249–256.

[Shields, 1997] Shields, D. N. (1997). Observer design and detection for nonlinear descriptor systems. *International Journal of Control*, 67(2):153–168.

[Slotine et al., 1987] Slotine, J., Hedrick, J. K., and Misawa, E. A. (1987). On sliding observers for nonlinear systems. *Journal of Dynamic Systems Measurement and Control*, 109:245–252.

[Stamatis, 1995] Stamatis, D. (1995). *Failure Mode and Effect Analysis: FMEA from Theory to Execution*. ASQ Quality Press.

[Staroswiecki, 2002] Staroswiecki, M. (2002). *Fault Diagnosis and Fault Tolerant Control*, chapter Structural Analysis for Fault Detection and Isolation and for Fault Tolerant Control. Encyclopedia of Life Support Systems, Eolss Publishers, Oxford, UK.

[Staroswiecki and Declerck, 1989] Staroswiecki, M. and Declerck, P. (1989). Analytical redundancy in non-linear interconnected systems by means of structural analysis. In *Proceedings of IFAC AIPAC'89*, pages 51–55, Nancy, France.

[Svärd and Nyberg, 2008a] Svärd, C. and Nyberg, M. (2008a). A mixed causality approach to residual generation utilizing equation system solvers and differential-algebraic equation theory. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, Blue Mountains, Australia.

[Svärd and Nyberg, 2008b] Svärd, C. and Nyberg, M. (2008b). A mixed causality approach to residual generation utilizing equation system solvers and differential-algebraic equation theory. Technical Report LiTH-ISY-R-2854, Department of Electrical Engineering, Linköpings Universitet, SE-581 83 Linköping, Sweden.

[Svärd and Nyberg, 2008c] Svärd, C. and Nyberg, M. (2008c). Observer-based residual generation for linear differential-algebraic equation systems. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea.

[Svärd and Nyberg, 2008d] Svärd, C. and Nyberg, M. (2008d). Observer-based residual generation for linear differential-algebraic equation systems. In *Proceedings of Reglermöte 2008*, Luleå, Sweden.

[Svärd and Wassén, 2006] Svärd, C. and Wassén, H. (2006). Development of methods for automatic design of residual generators. Master's thesis, Linköpings Universitet, SE-581 83 Linköping.

[United Nations, 2008] United Nations (2008). Regulation no. 49: Uniform provisions concerning the measures to be taken against the emission of gaseous and particulate pollutants from compressionignition engines for

use in vehicles, and the emission of gaseous pollutants from positive-ignition engines fuelled with natural gas or liquefied petroleum gas for use in vehicles. ECE-R49.

[Vemuri et al., 2001] Vemuri, A., Polycarpou, M., and Ciric, A. (2001). Fault diagnosis of differential-algebraic systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(2):143–152.

[Walcott et al., 1987] Walcott, B. L., Corless, M. J., and Zak, S. H. (1987). Comparitive study of non-linear state-observation techniques. *Internatinal Journal of Control*, 45(6):2109–2132.

[Zimmer and Meier, 1997] Zimmer, G. and Meier, J. (1997). On observing nonlinear descriptor systems. *System and Control Letters*, 32(1).

Part II

Papers

PAPER 1

1

An Observer-Based Residual Generation Method for Linear Differential-Algebraic Equation Systems[1]

Carl Svärd and Mattias Nyberg

Vehicular Systems, Department of Electrical Engineering,
Linköping University, S-581 83 Linköping,
Sweden.

Abstract

Residual generation for linear differential-algebraic systems is considered. A new systematic method for observer-based residual generation is presented. The proposed design method places no restrictions on the system to be diagnosed. If the fault of interest can be detected in the system, the output from the design method is a residual generator in state-space form that is sensitive to the fault of interest. The method is iterative and relies only on constant matrix operations such as multiplications, null-space calculations and equivalence transformations, and thereby straightforward to implement. An illustrative numerical example is included, where the design method is applied to a non-observable model of a robot manipulator.

[1]This paper has been submitted to European Journal of Control. It is based on [Svärd and Nyberg, 2008].

1 Introduction

The aim of fault diagnosis is to detect and isolate faults present in a system. With the rising demand for reliability and safety of technical systems, fault diagnosis has become increasingly important. One approach is to generate a set of residuals where different subsets of residuals respond to different subsets of faults. For this reason decoupling of faults in residuals is fundamental. Furthermore, decoupling can also be used to handle disturbances or unknown inputs.

Differential-algebraic equation (DAE) systems, or descriptor systems, are important in the residual generation context since DAE-systems appear in large classes of technical systems like mechanical-, electrical-, and chemical systems. Further, DAE-systems are also the result when using physically based object-oriented modeling tools, e.g. Modelica, [Mattson et al., 1998].

For the class of linear state-space systems, residual generation is an extensively studied area. Main approaches are for example the parity-space method, [Chow and Willsky, 1984], the factorization approach e.g. [Frank and Ding, 1994], and different observer-based methods, [Chen and Patton, 1999], [Massoumnia et al., 1989], [Hou and Müller, 1994]. For the more general class of linear DAE-systems, the list of previous works is not as extensive but includes parity-space approaches, [Sauter et al., 1996], [Maquin et al., 1993], parity-space-like approaches, [Nyberg and Frisk, 2006], [Varga, 2003], a parametric approach, [Duan et al., 2002], and several observer-based methods, [Hou, 2000], [Shields, 1994], [Marx et al., 2003].

Several of the above mentioned residual generation methods for DAE-systems have limitations since they have restrictions on the system to be diagnosed. The observer-based methods [Shields, 1994] and [Marx et al., 2003], both assume observability and so does the parity-space method, [Sauter et al., 1996]. In addition, [Marx et al., 2003], does not handle decoupling in the measurement equation. Observability is not assumed in [Maquin et al., 1993], but instead decoupling is not considered.

The main contribution in this paper is a new observer-based method for residual generation in linear DAE-systems. In contrast to the above mentioned methods, no restrictions are placed on the system to be diagnosed. This means that if the fault of interest is possible to detect, a residual generator can be designed with the proposed method. The method is based only on constant matrix operations such as multiplications, null-space calculations and equivalence transformations, and thereby straightforward to implement.

The paper is organized as follows. Section 2 presents preliminaries and states the problem formulation. Section 3, outlines the principles of the design method. Section 4 verifies in two theorems that the stated objective is met with the proposed design method. In Section 5 the method is applied to a non-observable DAE-model of a robot manipulator and Section 7 concludes the paper. The design method is summarizes as a ready-to-implement algorithm in Appendix 5.

2 Preliminaries and Problem Formulation

Consider the linear time-invariant differential-algebraic equation (DAE) system described by

$$E\dot{x} = Ax + Bu + Fd + Hf \tag{1a}$$
$$y = Cx + Du + Gd + Jf \tag{1b}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^p$, $y \in \mathbb{R}^m$, $d \in \mathbb{R}^q$, and $f \in \mathbb{R}^s$ are vectors of the states, inputs, outputs, disturbances, and faults of interest respectively. The inputs and outputs are considered as known variables and the states, disturbances and faults as unknowns. The matrix $E \in \mathbb{R}^{k\times n}$ may be singular and the disturbance-vector d consists of faults and unknown inputs that are to be decoupled. The matrices A, B, F, H, C, D, G, and J are all constant real-coefficient matrices of appropriate dimensions.

Before stating our main objective, the notions of fault detectability and fault sensitivity are needed. First, let $\mathcal{O}_{NF}$ denote the set of all known trajectories u and y consistent with the DAE-system (1) under the presence of no faults, i.e.

$$\mathcal{O}_{NF} = \{[u, y] | \exists x, d; E\dot{x} = Ax + Bu + Fd, y = Cx + Du + Gd\}. \tag{2}$$

Note that u, y, x, and d are here considered to be trajectories. In a similar way, $\mathcal{O}_f$ is defined as the corresponding set when the fault f is allowed to be non-zero, i.e.

$$\mathcal{O}_f = \{[u, y] | \exists x, d, f; E\dot{x} = Ax + Bu + Fd + Hf, y = Cx + Du + Gd + Jf\}. \tag{3}$$

The sets $\mathcal{O}_{NF}$ and $\mathcal{O}_f$ will in the sequel be referred to as observation sets. With $\mathcal{O}_{NF}$ and $\mathcal{O}_f$ defined, fault detectability can now be defined, see also [Nyberg and Frisk, 2006].

Definition 1 (Fault Detectability)**.** *Fault f is* detectable *in* (1) *if* $\mathcal{O}_f \not\subseteq \mathcal{O}_{NF}$.

It may be noted that fault detectability is a system property.

To check if given trajectories of u and y belongs to the observation set $\mathcal{O}_{NF}$ or not, i.e. if a fault is present in the system, residuals can be used. In this work, only residuals that are outputs from state-space systems are considered, leading to the following definition.

Definition 2 (Residual Generator)**.** *The linear time-invariant state-space system*

$$\dot{\xi} = \bar{A}\xi + \bar{B}u + \bar{M}y \tag{4a}$$
$$r = \bar{C}\xi + \bar{D}u + \bar{N}y \tag{4b}$$

is a residual generator *for (1) and r is a* residual *if*

$$[u, y] \in \mathcal{O}_{NF} \Rightarrow \lim_{t\to\infty} r = 0. \tag{5}$$

Note that r may here be multi-dimensional.

The problem that we consider can now be formulated as follows. Given the system (1), where it is assumed that the fault f is detectable, the objective is to create a residual generator for (1) where the residual is sensitive to f, that is, the transfer function from fault to residual is non-zero.

3 Outline of the Design Method

As stated in the problem formulation, the input to the design method is assumed to be a DAE-system on the form (1), where f is detectable. The design method consists of two main parts. First, a system in state-space form with no disturbances present is extracted from the input system. This is done iteratively, where disturbances are decoupled and the dimension of the system is reduced in each step. Second, a residual generator based on the decoupled system is designed. The steps of the design method are outlined below.

Step 1: Write the system on the form

$$\begin{bmatrix} E \\ 0 \end{bmatrix} \dot{x} = \begin{bmatrix} A \\ C \end{bmatrix} x + \begin{bmatrix} B \\ D \end{bmatrix} u + \begin{bmatrix} M \\ N \end{bmatrix} y + \begin{bmatrix} F \\ G \end{bmatrix} d + \begin{bmatrix} H \\ J \end{bmatrix} f \tag{6}$$

Step 2: Let

$$r = \text{rank} \begin{bmatrix} F \\ G \end{bmatrix}, \tag{7}$$

and

$$P = \begin{bmatrix} P_1 & P_2 \\ P_3 & P_4 \end{bmatrix}, \tag{8}$$

with $P_1 \in \mathbb{R}^{(k+m-r)\times k}$, $P_2 \in \mathbb{R}^{(k+m-r)\times m}$, $P_3 \in \mathbb{R}^{r\times k}$, $P_4 \in \mathbb{R}^{r\times m}$ chosen such that the rows of $\begin{bmatrix} P_1 & P_2 \end{bmatrix}$ form a basis for the left null-space of $\begin{bmatrix} F \\ G \end{bmatrix}$, and the rows of $\begin{bmatrix} P_3 & P_4 \end{bmatrix}$ form a basis for the image of $\begin{bmatrix} F \\ G \end{bmatrix}$. This implies that

$$\text{rank } P = k + m, \tag{9}$$

$$P_1 F + P_2 G = 0, \tag{10}$$

$$\text{rank } (P_3 F + P_4 G) = r. \tag{11}$$

Step 3: Pre-multiply (6) with the full-rank matrix P. Since (10) holds, the result becomes

$$P_1E\dot{x} = (P_1A + P_2C)x + (P_1B + P_2D)u + (P_1M + P_2N)y + (P_1H + P_2J)f \tag{12a}$$

$$P_3E\dot{x} = (P_3A + P_4C)x + (P_3B + P_4D)u + (P_3M + P_4N)y + (P_3F + P_4G)d + (P_3H + P_4J)f, \tag{12b}$$

where d not is present in (12a).

Step 4: Due to (11), the matrix $(P_3F + P_4G)$ has full row-rank. This implies that (12b) can always be fulfilled, since d can be arbitrarily chosen. Hence, the equation (12b) does not contain any usable information and is discarded.

Step 5: Let $t = \text{rank } (P_1E)$. If $t = n$, go to step 8, otherwise continue to step 6.

Step 6: Find, by e.g. singular-value decomposition, non-singular matrices U and V such that

$$U\,(P_1E)\,V = \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix}, \tag{13}$$

where $\Sigma \in \mathbb{R}^{t\times t}$ is a non-singular matrix.

Step 7: Pre-multiply (12a) with U, then introduce the non-singular state-transformation

$$w = V^{-1}x, \quad w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}, \tag{14}$$

where $w_1 \in \mathbb{R}^t$ and $w_2 \in \mathbb{R}^{(n-t)}$ to obtain

$$\begin{bmatrix} \Sigma \\ 0 \end{bmatrix} \dot{w}_1 = \begin{bmatrix} A_1 \\ A_3 \end{bmatrix} w_1 + \begin{bmatrix} A_2 \\ A_4 \end{bmatrix} w_2 + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u + \begin{bmatrix} M_1 \\ M_2 \end{bmatrix} y + \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} f, \tag{15}$$

where

$$\begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} = U(P_1A + P_2C)V, \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = U(P_1B + P_2D),$$
$$\begin{bmatrix} M_1 \\ M_2 \end{bmatrix} = U(P_1M + P_2N), \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} = U(P_1H + P_2J), \tag{16}$$

and $A_1 \in \mathbb{R}^{t\times t}$, $A_4 \in \mathbb{R}^{(k+m-r-t)\times(n-t)}$, $B_1 \in \mathbb{R}^{t\times p}$, $M_1 \in \mathbb{R}^{t\times m}$, and $H_1 \in \mathbb{R}^{t\times s}$. We can now consider w_2 to be a disturbance, and hence the system (15) is on the same form as (6). Thus, we return to step 1 with the system (15) as input.

Step 8: Find a non-singular matrix U such that

$$U\left(P_1 E\right) = \begin{bmatrix} \Pi \\ 0 \end{bmatrix}, \tag{17}$$

where $\Pi \in \mathbb{R}^{n\times n}$ is non-singular.

Step 9: Pre-multiply (12a) with U, then multiply the dynamic part of the result with Π^{-1}, to obtain

$$\dot{x} = \bar{A}_1 x + \bar{B}_1 u + \bar{M}_1 y + \bar{H}_1 f \tag{18a}$$
$$0 = A_2 x + B_2 u + M_2 y + H_2 f \tag{18b}$$

where

$$\begin{aligned} \bar{A}_1 = \Pi^{-1} A_1, \quad \bar{B}_1 = \Pi^{-1} B_1, \\ \bar{M}_1 = \Pi^{-1} M_1, \quad \bar{H}_1 = \Pi^{-1} H_1, \\ \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} = U(P_1 A + P_2 C), \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = U(P_1 B + P_2 D), \\ \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} = U(P_1 H + P_2 J), \begin{bmatrix} M_1 \\ M_2 \end{bmatrix} = U(P_1 M + P_2 N). \end{aligned} \tag{19}$$

Step 10: Find a matrix $L \in \mathbb{R}^{n\times m}$ such that all eigenvalues of the matrix $(\bar{A}_1 + LA_2)$ have negative real-parts. Pre-multiply (18) with the non-singular matrix

$$Q = \begin{bmatrix} I & L \\ 0 & I \end{bmatrix} \tag{20}$$

to obtain

$$\dot{x} = (\bar{A}_1 + LA_2)x + (\bar{B}_1 + LB_2)u + (\bar{M}_1 + LM_2)y + (\bar{H}_1 + LH_2)f \tag{21a}$$
$$0 = A_2 x + B_2 u + M_2 y + H_2 f. \tag{21b}$$

Step 11: Design the residual generator as

$$\dot{\xi} = (\bar{A}_1 + LA_2)\xi + (\bar{B}_1 + LB_2)u + (\bar{M}_1 + LM_2)y \tag{22a}$$
$$r = A_2 \xi + B_2 u + M_2 y. \tag{22b}$$

The design method is summarized as a ready-to-implement algorithm in Appendix A.

Remark 1. *Step 10 requires that* (18) *is observable or at least detectable, see e.g. [Rugh, 1996]. If this is not the case, the canonical structure theorem, e.g. [Gilbert, 1963], can be used to extract the observable subsystem from* (18)*, which instead is used in step 10.*

Remark 2. *The states ξ, in* (22) *are actually an estimate of a linear combination of the states x in* (1)*, and* (22) *is sometimes referred to as a FDI (Fault Detection and Isolation) observer, see e.g. [Hou and Müller, 1994]. It may also be noted that observer-based residual generation has strong connections with the design of unknown-input observers, see e.g. [Müller and Hou, 1994] for state-space systems and e.g. [Sun and Cheng, 2004] for DAE-systems. The aim in these works is to estimate the states of the system and not generate a residual suitable for fault detection. However, if an observer for a system can be designed, a residual can be created as the difference between measurements and estimated states.*

Remark 3. *Throughout this work, it is assumed that the system to be diagnosed is a DAE-system and the design method is described in this framework. Still, the method can likewise be applied to a state-space system, i.e. a system where $E = I$.*

4 Correctness of the Design Method

In this section we verify that the objective stated in Section 2 is met. That is, that the output from the proposed design method is a residual generator for (1), and that the corresponding residual is sensitive to the fault f.

Since the design method (or algorithm, as in Appendix A) is iterative, the following result is needed.

Lemma 1. *With* (1) *as input, the design method terminates.*

Proof. The system (1) has $k + m$ equations. In step 4 at least one equation is removed. Since $t \geq 0$ in step 5, the algorithm will terminate, if not earlier, after at most $k + m$ iterations when the remaining system is of zero dimension. □

4.1 Residual Generator Property

The output from the design method is (22) which is based on the system (21), obviously different from (1). A key property for (22) to be a residual generator for (1) is that the systems (1) and (21) have equal observations sets. This means that designing a residual generator for (1) is equivalent to designing a residual generator for (21). This property is the result of the following lemma.

Lemma 2. *Let* (1) *be the input to the design method, $\mathcal{O}_{NF}$ defined by* (2) *and $\mathcal{O}_f$ by* (3)*. Let $\mathcal{O}'_{NF}$ be the set of trajectories u and y consistent with* (21) *when $f = 0$ and $\mathcal{O}'_f$ the corresponding sets when f is allowed to be non-zero. It holds that $\mathcal{O}_{NF} = \mathcal{O}'_{NF}$ and $\mathcal{O}_f = \mathcal{O}'_f$.*

Proof. Given (1) as input to the design method, Lemma 1 states that the method will terminate. Two scenarios of execution of the steps 1 to 11 are possible. Either steps 1-5 followed by steps 8-11 is performed directly, else steps 1-7 will be iterated until the condition in step 5 holds, and then steps 8-11 will

be performed. In both cases, steps 3, 9, and 10 consist of multiplication with non-singular matrices and does not change the sets $\mathcal{O}_f$ and $\mathcal{O}_{NF}$ in any of the execution cases. The same holds for step 7 in the second case. Hence, the critical part is step 4, where equation (12b) is discarded. For the first execution case we must show that the observation sets are equal for (12) and (15) and for the second case that the same holds for (12) and (18). Or in other words for both cases, that (12b) can be discarded without loosing any usable information. Here, we will consider the second case and the first case can be shown in the same manner. Since it is trivial that the observation sets for (12) are subsets of the observation sets for (18), only the reverse inclusion is shown. Let $\tilde{x}$, $\tilde{u}$, and $\tilde{y}$ be trajectories satisfying (18) when $f = 0$. Since (12a) and (18) are related by a non-singular transformation, $\tilde{x}$, $\tilde{u}$, and $\tilde{y}$ also satisfies (12a). As a consequence of step 2, (11) holds. This implies that the matrix $(P_3F + P_4G)$ has full row-rank and hence the matrix has a right-inverse. Denote this right-inverse R and choose

$$\tilde{d} = RP_3E\dot{\tilde{x}} - R(P_3A + P_4C)\tilde{x} - R(P_3B + P_4D)\tilde{u} - R(P_3M + P_4N)\tilde{y}. \quad (23)$$

With $\tilde{d}$ and the previously defined $\tilde{x}$, $\tilde{u}$, and $\tilde{y}$, the equation (12b) is satisfied. This shows that (12b) does not contain any usable information and can be discarded. The reasoning can be repeated for the case when f is allowed to be non-zero to show that the observation sets are equal for (12) and (18), which completes the proof.

□

With help of Lemma 2, we can show that the first part of the stated objective is met.

Theorem 1. *Let* (1) *be the input to the design method and* (22) *the output. The system* (22) *is a residual generator for* (1) *and r in* (22b) *is a residual.*

Proof. Assume $f = 0$ and let $[u, y] \in \mathcal{O}_{NF}$, where $\mathcal{O}_{NF}$ is the set defined in (2). Lemma 2 then implies that u and y also satisfy (21). By subtracting (21a) from (22a) and (21b) from (22b) we obtain the autonomous system

$$\dot{\lambda} = (\bar{A}_1 + LA_2)\lambda \quad (24a)$$

$$r = A_2\lambda, \quad (24b)$$

where $\lambda = \xi - x$. Since, according to step 10, the matrix L is chosen such that all eigenvalues of $(\bar{A}_1 + LA_2)$ have negative real-parts, it follows directly that $\lim_{t\to\infty} r = 0$ and hence (22) is a residual generator for (1) and (22b) is a residual. □

4.2 Fault Sensitivity

The aim of this section is to show that the residual generator (22) is sensitive to the fault f, i.e. that the transfer function from f to the residual r is non-zero. However, the residual generator (22) is written in a form without faults.

By again, as in the proof to Theorem 1, subtracting (21a) from (22a) and (21b) from (22b), the relation between f and r can be described as

$$\dot{\lambda} = (\bar{A}_1 + LA_2)\lambda - (\bar{H}_1 + LH_2)f \tag{25a}$$
$$r = A_2\lambda - H_2 f, \tag{25b}$$

where $\lambda = \xi - x$.

From (25), the transfer function from fault to residual can be written as

$$G_{rf}(s) = A_2\left(-sI + \bar{A}_1 + LA_2\right)^{-1}(\bar{H}_1 + LH_2) - H_2. \tag{26}$$

The result that verifies that the second part of the objective is met with the design method here follows.

Theorem 2. *Let* (1) *be the input to the design method and* (25) *the output. If f is detectable in* (1)*, the transfer function from fault to residual* (26) *is non-zero.*

Proof. The transfer function (26) can by power-series expansion of

$$\left(-sI + \bar{A}_1 + LA_2\right)^{-1}$$

be written as

$$G_{rf}(s) = A_2\left(-sI + \bar{A}_1 + LA_2\right)^{-1}(\bar{H}_1 + LH_2) - H_2 = \\ -\sum_{i=1}^{\infty} A_2\left(\bar{A}_1 + LA_2\right)^{i-1}(\bar{H}_1 + LH_2)s^{-i} - H_2. \tag{27}$$

To show the contrary of the claim, i.e. that $G_{rf}(s) = 0$ implies $\mathcal{O}_f \subseteq \mathcal{O}_{NF}$, we assume $G_{rf}(s) = 0$. Using (27), $G_{rf}(s) = 0$ is equivalent to

$$H_2 = 0, \tag{28}$$
$$A_2\left(\bar{A}_1 + LA_2\right)^{i-1}(\bar{H}_1 + LH_2) = 0, \quad i = 1, \ldots, \infty. \tag{29}$$

As a consequence of the Cayley-Hamilton theorem,
$A_2\left(\bar{A}_1 + LA_2\right)^{i-1}$, for $i \geq n+1$, can be written as a linear combination of

$$A_2, A_2\left(\bar{A}_1 + LA_2\right), \ldots, A_2\left(\bar{A}_1 + LA_2\right)^{n-1}, \tag{30}$$

therefore it is sufficient to consider the matrix

$$\Omega = \begin{bmatrix} A_2 \\ A_2\left(\bar{A}_1 + LA_2\right) \\ \vdots \\ A_2\left(\bar{A}_1 + LA_2\right)^{n-1} \end{bmatrix}. \tag{31}$$

The condition (29) clearly implies $(\bar{H}_1 + LH_2) \in \text{Ker}\,\Omega$ and the two cases rank $\Omega = n$ and rank $\Omega < n$ will now be studied separately.

For the first case, i.e. when (22) and (25) are both observable, dim Ker $\Omega = 0$ which implies $(\bar{H}_1 + LH_2) = 0$. From (28), $H_2 = 0$ and it must hold that $\mathcal{O}_f = \mathcal{O}_{NF}$.

For the second case, let $[\tilde{u}, \tilde{y}] \in \mathcal{O}_f$. This means that there exist trajectories, say $\tilde{f}$ and $\tilde{x}$ with $\tilde{x}(t_0) = \tilde{x}_0$, such that

$$\dot{\tilde{x}} = (\bar{A}_1 + LA_2)\tilde{x} + (\bar{B}_1 + LB_2)\tilde{u} + (\bar{M}_1 + LM_2)\tilde{y} + (\bar{H}_1 + LH_2)\tilde{f} \tag{32a}$$

$$0 = A_2\tilde{x} + B_2\tilde{u} + M_2\tilde{y} + H_2\tilde{f}. \tag{32b}$$

We will now show that there exists a trajectory ζ that along with the trajectories $\tilde{u}$ and $\tilde{y}$ satisfies (32) when $\tilde{f} = 0$. Consider the residual generator (25) and let $\tilde{\lambda}(t_0) = \tilde{\lambda}_0 \in \text{Ker}\,\Omega$. This implies that $\tilde{\lambda}$ will be a trajectory in the non-empty unobservable subspace of (25). Evaluation of (25) with the initial state $\tilde{\lambda}_0$, together with (28) and (29), yields $r \equiv 0$ independent of f. In particular, this holds for $f = \tilde{f}$ and hence

$$\dot{\tilde{\lambda}} = (\bar{A}_1 + LA_2)\tilde{\lambda} - (\bar{H}_1 + LH_2)\tilde{f} \tag{33a}$$

$$0 = A_2\tilde{\lambda} - H_2\tilde{f}. \tag{33b}$$

Now form $\zeta = \tilde{x} + \tilde{\lambda}$, $\zeta(t_0) = \tilde{x}_0 + \tilde{\lambda}_0$, and combine (32) with (33) to obtain

$$\dot{\zeta} = (\bar{A}_1 + LA_2)\zeta + (\bar{B}_1 + LB_2)\tilde{u} + (\bar{M}_1 + LM_2)\tilde{y} \tag{34a}$$

$$0 = A_2\zeta + B_2\tilde{u} + M_2\tilde{y}. \tag{34b}$$

Thus, there exists a trajectory ζ satisfying the fault-free system (34) so that $[\tilde{u}, \tilde{y}] \in \mathcal{O}_{NF}$, implying $\mathcal{O}_f \subseteq \mathcal{O}_{NF}$ and the proof is complete.

□

Remark 4. *The two systems* (22) *and* (25) *are two ways of writing a residual generator. The form* (22) *is the so called computational form, and* (25) *is usually referred to as internal form.*

5 Application Example

To illustrate the design method, we apply it to a DAE-model of a three-link planar manipulator from [Hou, 2000] and [Hou and Müller, 1996], see Figure 1. The objective of the manipulator is to apply a constant horizontal force in the region between point A and B, e.g. for cleaning the region. The manipulator consists of an end-effector, three rods, and three joints. Via actuators at every joint, a torque can be applied to move the effector repeatedly between A and B. The manipulator is equipped with four sensors measuring the height of the end-effector, the contact force in the horizontal direction, and tracking

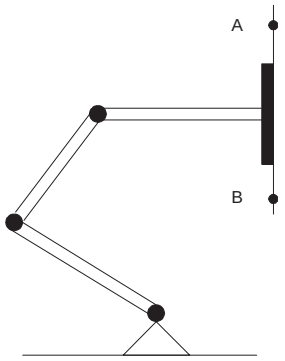


Figure 1: The three-link planar manipulator.

signals. The DAE-model has three states for the Cartesian coordinates of the end-effector, three states for the derivatives of the Cartesian coordinates, two states for Lagrangian multipliers, and three states for the controller, altogether 11 states. In this example, the original fault model has been extended with a sensor fault. The process is subjected to 3 faults. Fault f_1 represents a fault in actuator 1, f_2 a fault in the tracking reference signal, and f_3 a fault in sensor 4. Hence, the form of the DAE is

$$E\dot{x} = Ax + Bu + Hf \tag{35a}$$

$$y = Cx + Jf, \tag{35b}$$

where $x \in \mathbb{R}^{11}$, $u \in \mathbb{R}^3$, $y \in \mathbb{R}^4$, and $f \in \mathbb{R}^3$. Numerical values of the matrices E, A, B, C, H, and J can be found in Appendix B. The matrix E is square with rank $E = 9$ and (35) is regular. Further, the system (35) is not impulse observable ([Dai, 1989]), since

$$\text{rank} \begin{bmatrix} E & A \\ 0 & E \\ 0 & C \end{bmatrix} = 18 \neq \text{rank}\, E + n = 9 + 11 = 20. \tag{36}$$

This means that methods assuming observability, for example [Shields, 1994], [Marx et al., 2003], and [Sauter et al., 1996], can not be applied to the system.

Our design objective is to create three residual generators for (35) whose residuals r_1, r_2, and r_3, should monitor each of the faults f_1, f_2, and f_3, according to the following fault signature matrix

	f_1	f_2	f_3
r_1	1		
r_2		1	
r_3			1

In residual generator 1, the transfer function from fault f_1 to r_1 should be non-zero, and the same should hold for the transfer function from f_2 to r_2 in residual generator 2, and for the transfer function from f_3 to r_3 in residual generator 3. Since each residual generator should monitor only one fault, two faults need to be decoupled in each residual generator. This means that f_2 and f_3 are seen as disturbances in residual generator 1 and the matrix $F_1 = \begin{bmatrix} H_2 & H_3 \end{bmatrix}$, and $G_1 = \begin{bmatrix} J_2 & J_3 \end{bmatrix}$ can be formed, where H_i and J_i denotes the i:th column of the matrices H and J respectively. In the same way the matrices F_2, G_2, F_3, and G_3, with the columns from H and J corresponding to the faults to be decoupled in each residual generator, are created.

Performing the design according to the method in Section 3 with the three different configurations of system (35) as input, three disturbance decoupled systems on the form (18) with 6, 5, and 5 states respectively are obtained. For all three input systems, the algorithm terminates after 3 iterations. The three systems are all observable, and hence it is straightforward to perform step 10. For all three residual generators, the poles are placed in -1.

All three residual generators have two-dimensional residuals. By calculating the transfer functions from fault to residual for each residual generator, it can be verified that $G_{rf_i}(s) = 0$, when $i = 2, 3$ for residual generator 1, when $i = 1, 3$ for residual generator 2, and when $i = 1, 2$ for residual generator 3. To verify that the design objective is met, the transfer functions from the monitored faults to the residual for each residual generator is shown in Figures 3.2(a), 3.2(b), and 3.2(c). It is clear that all transfer functions are non-zero.

6 Conclusions

We have considered the problem of residual generation for linear DAE-systems. A new systematic method for observer-based residual generation has been presented. In contrast to several previous methods, no restrictions such as observability are placed on the system to be diagnosed. This means that if the fault of interest is detectable in the system to be diagnosed, a residual generator can be designed with the design method in this paper. It has been verified in Theorem 1 and 2 that the output from the design method is indeed a residual generator, and that the corresponding transfer function from fault to residual is non-zero. Finally note that even though the design method has been described in the framework of DAE-systems, it can likewise be applied to state-space systems.

References

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

[Chow and Willsky, 1984] Chow, E. and Willsky, A. (1984). Analytical redundancy and the design of robust failure detection systems. *IEEE Transactions on Automatic Control*, 29(7):603–613.

[Dai, 1989] Dai, L. (1989). *Singular Control Systems*. Springer-Verlag.

[Duan et al., 2002] Duan, G., Howe, D., and Patton, R. (2002). Robust fault detection in descriptor linear systems via generalized unknown input observers. *International Journal of Systems Science*, 33(5):369–377.

[Frank and Ding, 1994] Frank, P. and Ding, X. (1994). Frequency domain approach to optimally robust residual generation and evalutaion for model-based fault diagnosis. *Automatica*, 30(4):789–804.

[Gilbert, 1963] Gilbert, E. (1963). Controllability and observability in multivariable control systems. *SIAM Journal on Control and Optimization*, 1(2):128–152.

[Hou, 2000] Hou, M. (2000). *Fault detection and isolation for descriptor systems*, chapter 5. Issues of Fault Diagnosis for Dynamic Systems. Springer-Verlag.

[Hou and Müller, 1994] Hou, M. and Müller, P. (1994). Fault detection and isolation observers. *International Journal of Control*, 60(5):827–846.

[Hou and Müller, 1996] Hou, M. and Müller, P. (1996). Tracking control for a class of descriptor systems. In *Proceedings 13th IFAC World Congress*, pages 109–114, San Fransisco, USA.

[Maquin et al., 1993] Maquin, D., Gaddouna, B., and Ragot, J. (1993). Generation of parity equations for singular systems. In *Proc. Int. Conf. Syst., Man Cybern.*, volume 3, pages 400–405.

[Marx et al., 2003] Marx, B., Koenig, D., and Georges, D. (2003). Robust fault diagnosis for linear descriptor systems using proportional integral observers. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 457–462, Maui, Hawaii, USA.

[Massoumnia et al., 1989] Massoumnia, M., Verghese, G., and Willsky, A. (1989). Failure detection and isolation. *IEEE Transactions on Automatic Control*, 34(3):316–321.

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

[Müller and Hou, 1994] Müller, P. and Hou, M. (1994). Disturbance decoupled observer design: A unified viewpoint. *IEEE Transactions on Automatic Control*, 39(6):1338–1341.

[Nyberg and Frisk, 2006] Nyberg, M. and Frisk, E. (2006). Residual generation for fault diagnosis of systems described by linear differential-algebraic equations. *IEEE Transactions on Automatic Control*, 51(12):1995–2000.

[Rugh, 1996] Rugh, W. (1996). *Linear System Theory*, chapter 18. Prentice Hall Information and System Sciences.

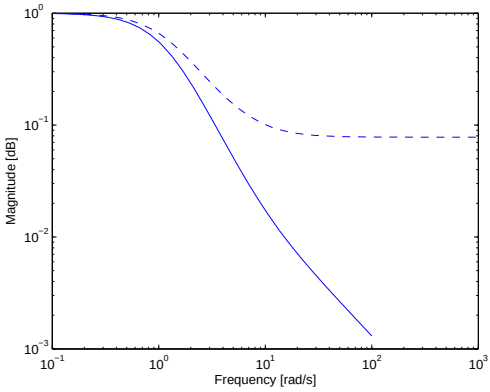
[Sauter et al., 1996] Sauter, D., Noura, H., Hamelin, F., and Theilliol, D. (1996). Parity space approach for fault diagnosis in descriptor systems. In *Proc. CESA '06 IMACS Multiconf. Comput. Eng. Syst. Appl.*, volume 1, pages 380–383.

[Shields, 1994] Shields, D. (1994). Robust fault detection for generalized state space systems. In *Proceedings International Conference Control*, pages 1335–1339.

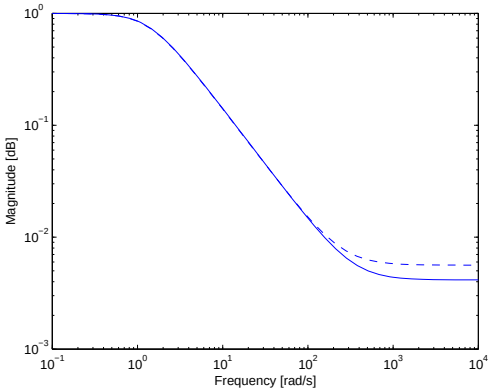
[Sun and Cheng, 2004] Sun, L. and Cheng, Z. (2004). State and input estmitation for descriptor systems. In *Proceedings of the 2004 American Control Conference*, Boston, Massachusetts.

[Svärd and Nyberg, 2008] Svärd, C. and Nyberg, M. (2008). Observer-based residual generation for linear differential-algebraic equation systems. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea.

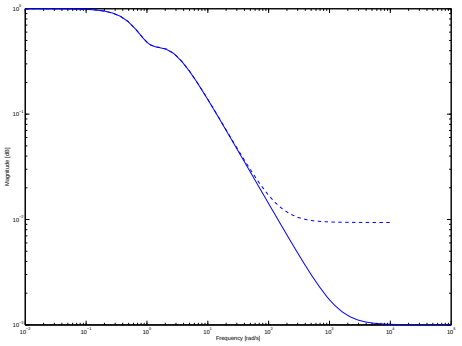
[Varga, 2003] Varga, A. (2003). On computing least order fault detectors using rational nullspace bases. In *Proc. Safeprocess 2003*, pages 229–234, Washington DC.



(a) Transfer functions from fault f_1 to the two residuals in residual generator 1.



(b) Transfer functions from fault f_2 to the two residuals in residual generator 2.



(c) Transfer functions from fault f_3 to the two residuals in residual generator 3.

Figure 2: Transfer functions from monitored faults to residuals in the obtained residual generators.

A Design Algorithm

The design method is below summarized as the function DESIGNRESIDUALGENERATOR, taking matrices E, A, B, F, H, C, D, G, and J corresponding to a system on the form (1) where $E \in \mathbb{R}^{k \times n}$, as input. The output consists of the matrices $\bar{A}$, $\bar{B}$,$\bar{M}$, $\bar{C}$, $\bar{D}$, and $\bar{N}$ corresponding to a residual generator on the form (4). The function uses the following sub-functions

- NULL computes a basis for the null-space of a matrix.
- SVD performs a singular-value decomposition.
- RANK computes the rank of a matrix.
- STABILIZE computes a feedback gain such that all eigenvalues of the resulting matrix have negative real-parts.

In addition, we have used the notation $(A, B; C, D)$ to denote the matrix $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$.

function DESIGNRESIDUALGENERATOR($E, A, B, F, H, C, D, G, J$)
  $N := I$
  $t := 0$
  while $t \neq n$ **do**
    $(P_1, P_2) := \text{NULL}((F; G)^T)^T$
    $t := \text{RANK}(P_1 E)$
    $(U, \Sigma, V) := \text{SVD}(P_1 E)$
    $(B; D) := U(P_1 B + P_2 D)$
    $(H; J) := U(P_1 H + P_2 J)$
    $(M; N) := U(P_1 M + P_2 N)$
    if $t \neq n$ **then**
      $E := \Sigma$
      $(A, F; C, G) := U(P_1 A + P_2 C)V$
    end if
  end while
  $(A; C) := U(P_1 A + P_2 C)$
  $L := \text{STABILIZE}(\Sigma^{-1} A, C)$
  $\bar{A} := \Sigma^{-1} A + LC$
  $\bar{B} := \Sigma^{-1} B + LD$
  $\bar{M} := \Sigma^{-1} M + LN$
  $\bar{C} := C$
  $\bar{D} := D$
  $\bar{N} := N$
end function

B Matrices for Application Example

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 18.75 & -7.95 & 7.95 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -7.95 & 31.82 & -26.82 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7.95 & -26.82 & 26.82 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -68.7 & 98.96 & -77.74 & -601.16 & 43.92 & -107.77 \\ 45.23 & -402.43 & 337.54 & -906.97 & -177.27 & 179.24 \\ 4.48 & 339.82 & -219.17 & 697.11 & 149.56 & -360.37 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -68.83 & -34.83 & -6.41 \\ 0 & 0 & 280.46 & -58.29 & 24.22 \\ 0 & 1 & -236.89 & 48.76 & -69.93 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T,$$

$$H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -36.334 & 76.914 & -76.914 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T,$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad J = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

PAPER 2

Residual Generators for Fault Diagnosis using Computation Sequences with Mixed Causality Applied to Automotive Systems[1]

Carl Svärd and Mattias Nyberg

Vehicular Systems, Department of Electrical Engineering,
Linköping University, S-581 83 Linköping,
Sweden.

Abstract

Rising demands for reliability and safety of complex technical systems, e.g. automotive systems, has made model-based fault diagnosis increasingly important. An essential step in the design of a model-based diagnosis system is to find a set of residual generators fulfilling stated fault detection and isolation requirements. To be able to perform a good selection, it is desirable that the method used for residual generation gives as many candidate residual generators as possible, given a model. This paper presents a novel method for residual generation that enables simultaneous use of integral and derivative causality, i.e. mixed causality, and is able to handle equation sets corresponding to algebraic and differential loops in a systematic manner. The method relies on a formal framework for computing unknown variables according to a computation sequence, in which mixed causality is utilized and the analytical properties of the equations in the model and the available tools for algebraic equation solving are taken into account. The proposed method is applied to two models of automotive systems, a Scania diesel engine and a hydraulic braking system. Significantly more residual generators are found with the proposed method, in comparison with methods using solely integral or derivative causality.

[1]This paper has been submitted to IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans. It is an extended and revised version of the work presented in [Svärd and Nyberg, 2008].

1 Introduction

Fault diagnosis of technical systems has become increasingly important with the rising demand for reliability and safety, driven by environmental and economical incentives. One example is automotive engines that are by regulations required to have high precision on-board diagnosis of failures that are harmful for the environment [United Nations, 2008].

To obtain good detection and isolation of faults, model-based fault diagnosis is necessary. In the Fault Detection and Isolation (FDI) approach to model-based fault diagnosis, *residuals*, signals ideally zero in the non-faulty case and non-zero else, are used to detect and isolate faults present in the system, see e.g. [Blanke et al., 2003]. Residuals are typically generated by utilizing a mathematical model of the system and measurements.

In this paper, we have the view that design of diagnosis systems is a two-step approach, as elaborated in [Nyberg and Krysander, 2008], [Nyberg, 1999], where in the first step a large number of candidate residual generators are found, and in the second step the residual generators most suitable to be included in the final diagnosis system are picked out. Since different residual generators have different properties regarding fault and noise sensitivities, it is for the second step important that there is a large selection of different residual generator candidates to choose between. Thus, the initial set of candidate residual generators should be as large as possible.

One approach residual generator design [Staroswiecki and Declerck, 1989], which has shown to be successful in real applications, [Dustegor et al., 2004], [Izadi-Zamanabadi, 2002], [Cocquempot et al., 1998], [Svärd and Wassén, 2006], [Hansen and Molin, 2006], is to compute unknown variables in the model by solving equations one at a time in a sequence, i.e. according to a computation sequence, and then evaluate a redundant equation to obtain a residual. To determine from which equations and in which order the unknown variables should be computed, *structural analysis* is utilized. In addition to [Staroswiecki and Declerck, 1989], similar approaches have been described and exploited in e.g. [Cassar and Staroswiecki, 1997], [Staroswiecki, 2002], [Blanke et al., 2003], [Pulido and Alonso-González, 2004], [Ploix et al., 2005], and [Travé-Massuyès et al., 2006].

In the works mentioned above, the approach is to apply either integral or derivative causality [Blanke et al., 2003] for differential equations. However, as will be illustrated in this paper through application studies, it is advantageous to allow simultaneous use of integral and derivative causality, i.e. mixed causality. Furthermore, real-world applications involves complex models that gives rise to algebraic and differential loops or cycles [Blanke et al., 2003], [Katsillis and Chantler, 1997], which correspond to sets of equations that have to be treated simultaneously. Thus, it is desirable that a method for residual generation is able to handle mixed causality and equation sets corresponding to algebraic and differential loops. The intention with the following simple example is to illustrate these issues. Consider the set of differential-algebraic

equations

$$\begin{aligned} e_1: &\quad \dot{x}_1 - x_2 = 0 \\ e_2: &\quad \dot{x}_3 - x_4 = 0 \\ e_3: &\quad \dot{x}_4 x_1 + 2x_2 x_4 - y_1 = 0 \\ e_4: &\quad x_3 - y_3 = 0 \\ e_5: &\quad x_2 - y_2 = 0, \end{aligned} \tag{1}$$

which is a subsystem of a model describing the planar motion of a point-mass satellite [Brockett, 1970], [De Persis and Isidori, 2001], and where x_1, x_2, x_3, x_4 are unknown variables and y_1, y_2, y_3 known variables. Assume that we want to use equation e_5 as residual. This implies that the unknown variables x_1, x_2, x_3, and x_4 must be computed from the equations e_1, e_2, e_3,and e_4. A *structure*, i.e. which unknown variables that are contained in which equation, of the equation set $\{e_1, e_2, e_3, e_4\}$ with respect to $\{x_1, x_2, x_3, x_4\}$, in permuted form, is depicted in (2).

	x_3	x_4	x_2	x_1
e_4	**1**			
e_2	1	**1**		
e_3		1	**1**	**1**
e_1			**1**	**1**

(2)

This structure reveals the order and from which equations, marked with bold, the unknown variables should be computed. It is clear that computation of the variables will involve handling the differential loop arising in the equation set $\{e_1, e_2\}$, since to compute x_2 the value of x_1 is needed and vice versa. Furthermore, computation of the variables according to (2) will require use of both causality approaches, derivative causality when solving for x_4 in e_2 and integral causality when solving for x_1 in e_1, i.e. mixed causality.

The main contribution in this paper is a novel method for residual generation that enables simultaneous use of integral and derivative causality, and is able to handle equation sets corresponding to algebraic and differential loops in a systematic manner. In this sense, the proposed method also generalizes previous methods for residual generation, e.g. [Staroswiecki and Declerck, 1989]– [Travé-Massuyès et al., 2006]. To achieve this, a formal framework for sequential computation of variables is presented. In this framework, tools for equation solving and approximate differentiation, as well as analytical and structural properties of the equations in the model are important.

In Section 2 some preliminaries, basic theories and references regarding structural analysis and differential-algebraic equation systems are given. Section 3 presents the framework for sequential computation of variables, in which the concepts *BLT semi-explicit DAE form*, *tools* and *computation sequence* are important. Tools, or more precisely algebraic equation solving tools, are essential for the ability to handle loops. In Section 4, it is shown how a computation

sequence is utilized for residual generation. The resulting residual generator is referred to as a *sequential residual generator*. Motivated by implementation aspects, the concept of *proper sequential residual generator* is introduced as a sequential residual generator in which no unnecessary variables are computed and in which computations are performed from as small equation sets as possible. A necessary condition for the existence of a proper sequential residual generator is derived, connecting proper sequential residual generators with *minimal structurally over-determined* (MSO) equation sets [Krysander et al., 2008]. An algorithm able to find proper sequential residual generators, given a model and a set of tools, is outlined. A key step in the algorithm is to find minimal and irreducible computation sequences, which is considered in Section 5. In Section 6, the proposed method for residual generation is applied to models of an automotive diesel engine and an auxiliary hydraulic braking system. The application studies clearly show the benefits, in the sense that more residual generators are found, of using a mixed causality approach and handling algebraic and differential loops. Finally, Section 7 concludes the paper. For readability, proofs to all lemmas and theorems are collected in Appendix A.

2 Preliminaries and Background Theory

Consider a *model*, $M(E, X, Y)$, or M for short, consisting of a set of equations $E = \{e_1, e_2, \ldots, e_m\}$ relating a set of unknown variables $X = \{x_1, x_2, \ldots, x_n\}$, and a set of known, i.e. measured, variables $Y = \{y_1, y_2, \ldots, y_r\}$. Introduce a third variable set $D = \{\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_n\}$, containing the (time) derivatives of the variables in X. Without loss of generality, it is assumed that the equations in E are in the form

$$e_i : \quad f_i(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}) = 0, \quad i = 1, 2, \ldots, m \tag{3}$$

where $\dot{\mathbf{x}} = (\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_n)$ is a vector of the variables in D, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ a vector of the variables in X, and $\mathbf{y} = (y_1, y_2, \ldots, y_r)$ a vector of the variables in Y. Also without loss of generality, it is assumed that each equation $e_i \in E$ contains, at most, one differentiated variable $\dot{x}_j \in D$ and that $\dot{x}_j$ is contained only in one equation.

Define the set of trajectories of the variables in Y that are *consistent* with the model $M(E, X, Y)$ as

$$\mathcal{O}(M) = \{\mathbf{y} : \exists \mathbf{x}; f_i(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}) = 0, i = 1, 2, \ldots, m\}. \tag{4}$$

The set $\mathcal{O}(M)$ is the *observation set* of the model M. We formally define a residual generator as follows.

Definition 1 (Residual Generator)**.** *A system with input* $\mathbf{y}$ *and output* r *is a* residual generator *for the model* $M(E, X, Y)$ *and* r *is a residual if*

$$\mathbf{y} \in \mathcal{O}(M) \Rightarrow \lim_{t \to \infty} r \to 0$$

2.1 Integral and Derivative Causality

In the context of the methods for residual generation mentioned in Section 1, there are two approaches for handling differential equations, referred to as *integral* and *derivative* causality, see e.g. [Blanke et al., 2003]. When adopting integral causality, the differentiated variables, or states, of a differential equation can be computed. The use of integral causality hence relies on the assumption that ordinary differential equations can be solved, i.e. integrated, which in general requires that initial conditions of the states are known. Integral causality is used in for example [Pulido et al., 2008] and [Pulido and Alonso-González, 2004].

If instead derivative causality is applied, a differential equation is interpreted as an algebraic equation and only un-differentiated, i.e. algebraic, variables can be computed. Usage of derivative causality thus relies on the assumption that values of the differentiated variables in a differential equation are available. This requires in general that derivatives of known, or previously computed, variables can be computed or estimated. Derivative causality is used in [Staroswiecki, 2002], and also adopted in e.g. [Dustegor et al., 2004]. The difference between integral and derivative causality is discussed in [Pulido et al., 2007] and from a simulation point of view in [Cellier and Elmqvist, 1993].

The chosen causality approach naturally influences which variables that can be computed from an equation set. For instance, consider the differential equation $e_1 : \dot{x}_1 - x_2 = 0$ from (1), where both x_1 and x_2 are unknown variables. If integral causality is used, x_1 can be computed from e_1 but if instead derivative causality is used, x_2 can be computed from e_1.

2.2 Structure of Equation Sets

To study which unknown variables that are contained in a set of equations, a structural representation of the equation set will be used. Let $E' \subseteq E$ and introduce the notations

$$\mathrm{var}_X(E') = \left\{ x_j \in X : \exists e_i \in E', \frac{\partial f_i}{\partial x_j} \not\equiv 0 \vee \frac{\partial f_i}{\partial \dot{x}_j} \not\equiv 0 \right\},$$

$$\mathrm{var}_D(E') = \left\{ \dot{x}_j \in D : \exists e_i \in E', \frac{\partial f_i}{\partial \dot{x}_j} \not\equiv 0 \right\}.$$

Consider the model (1) and let $X = \{x_1, x_2, x_3, x_4\}$ and $D = \{\dot{x}_1, \dot{x}_2, \dot{x}_3, \dot{x}_4\}$. For instance, it holds that

$$\mathrm{var}_X(\{e_3\}) = \{x_1, x_2, x_4\}. \tag{5}$$

Let $G = (E, X, A)$ be a bi-partite graph where E and X are the (disjoint) sets of vertices, and

$$A = \left\{ (e_i, x_j) \in E \times X : x_j \in \mathrm{var}_X(\{e_i\}) \right\},$$

the set of arcs. We will call the bi-partite graph G the *structure* of the equation set E with respect to X. Note that with this representation, there is no structural difference between the variable x_j and the differentiated variable $\dot{x}_j$. An equivalent representation of G is the $m \times n$ bi-adjacency matrix B defined as

$$B_{ij} = \begin{cases} 1 & \text{if } (e_i, x_j) \in A \\ 0 & \text{otherwise} \end{cases}$$

Return to the model (1). The structure of the equation set $\{e_1, e_2, e_3, e_3\}$ with respect to $\{x_1, x_2, x_3, x_4\}$ is given by the bi-adjacency matrix (2). The result in (5) corresponds to the third row of (2)

We will also consider the structure of E with respect to D which refers to the bi-partite graph $\bar{G} = (E, D, \bar{A})$, where

$$\bar{A} = \left\{ (e_i, \dot{x}_j) \in E \times D : \dot{x}_j \in \mathrm{var}_D(\{e_i\}) \right\}.$$

2.3 Structural Decomposition

A *matching* on the bi-partite graph $G = (E, X, A)$ is a subset of A such that no two arcs have common vertices. A matching with maximum cardinality is a *maximum matching*. A matching is a *complete matching* with respect to E (or X), if the matching covers every vertex in E (or X). By directing the arcs contained in a matching on the bi-partite graph G in one direction, and the remaining arcs in the opposite direction, a *directed graph* can be obtained from G, see for example [Asratian et al., 1998]. A directed graph is said to be *strongly connected* if for every pair of vertices x_i and x_j there is a directed path from x_i to x_j. The maximal strongly-connected subgraphs of a directed graph are called its *strongly-connected components* (SCC).

There exists a unique structural decomposition of the bi-partite graph $G = (E, X, A)$, referred to as the Dulmage-Mendelsohn (DM) decomposition, [Dulmage and Mendelsohn, 1958], [Murota, 1987]. It decomposes G into irreducible bi-partite subgraphs $G^+ = \left(E^+, X^+, A^+\right)$, $G_i^0 = \left(E_i^0, X_i^0, A_i^0\right), i = 1, 2, \ldots, s$, and $G^- = (E^-, X^-, A^-)$, called DM-components, see Figure 1. The component G^+ is the over-determined part of G, $G^0 = \bigcup_{i=1}^{s} G_i^0$ the just-determined part, and G^- the under-determined part. The DM-components $G_i^0 = \left(E_i^0, X_i^0, A_i^0\right)$ correspond to the SCCs of the directed graph induced by any complete matching on the bi-partite graph G^0, [Murota, 1987]. The equation set $E^0 = \bigcup_{i=1}^{s} E_i^0$ is said to be a just-determined equation set with respect to the variables $X^0 = \bigcup_{i=1}^{s} X_i^0$.

Algebraic and Differential Loops

If the structure of an equation set, with respect to a set of unknown variables, contains SCCs of larger size than one, the equation set contains *loops* or *cycles*, see e.g. [Blanke et al., 2003], [Katsillis and Chantler, 1997], [Pulido et al.,

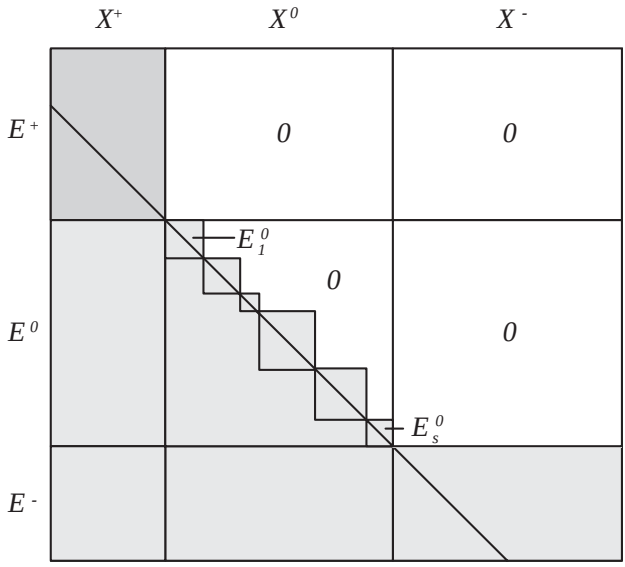


Figure 1: DM-decomposition of the bi-partite graph $G = (E, X, A)$. The DM-components $G_i^0 = (E_i^0, X_i^0, A_i^0)$ correspond to the SCCs of the structure of E^0 with respect to X^0.

2007]. If the equation set contains cyclic dependencies including unknown differentiated variables, the loop is said to be *differential*, else *algebraic*. In the example outlined in Section 1, the structure (2), which in fact is the result of a DM-decomposition, revealed three SCCs which are bold-marked. The SCCs are $(\{e_4\}, \{x_3\})$,$(\{e_2\}, \{x_4\})$, and $(\{e_2, e_1\}, \{x_1, x_2\})$ of size 1, 1, and 2 respectively. The latter corresponds to a differential loop.

2.4 Differential-Algebraic Equation Systems

Due to its general form, it is assumed that the model (3) contains both differential and algebraic equations, i.e. it is a differential-algebraic equation (DAE) system, or descriptor system. The most general form of a DAE is $\mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}) = 0$, where $\mathbf{f}$ is some vector-valued function, cf. (3). DAEs appear in large classes of technical systems like mechanical-, electrical-, and chemical systems. Further, DAEs are also the result when using physically based object-oriented modeling tools, e.g. Modelica [Mattson et al., 1998].

Differential Index

A common approach when analyzing and solving general DAE-systems, is to seek a reformulation of the original DAE into a simpler and well-structured

description with the same set of solutions [Kunkel and Mehrmann, 2006], [Brenan et al., 1989]. To classify how difficult such a reformulation is, the concept of index has been introduced. There are different index concepts depending on what kind of reformulation that is sought. In this paper we will use the *differential index*, which is defined as the minimum number of times that all or parts of the DAE must be differentiated with respect to time in order to write the DAE as an *explicit ordinary differential equation* (ODE), $\dot{\mathbf{x}} = \mathbf{g}(\mathbf{x}, \mathbf{y})$, see for example [Brenan et al., 1989].

Semi-Explicit DAEs

An important class of DAEs are *semi-explicit* DAEs

$$\dot{\mathbf{z}} = \mathbf{g}(\mathbf{z}, \mathbf{w}, \mathbf{y}) \tag{6a}$$

$$0 = \mathbf{h}(\mathbf{z}, \mathbf{w}, \mathbf{y}), \tag{6b}$$

where $\mathbf{z}$ and $\mathbf{w}$ are vectors of unknown variables, and $\mathbf{y}$ a vector of known variables. A semi-explicit DAE is of index one if and only if (6b) can be (locally) solved for $\mathbf{w}$ so that $\mathbf{w} = \tilde{\mathbf{h}}(\mathbf{z}, \mathbf{y})$, see e.g. [Brenan et al., 1989]. An explicit ODE can easily be obtained from a semi-explicit DAE of index one by substituting $\mathbf{w} = \tilde{\mathbf{h}}(\mathbf{z}, \mathbf{y})$ into (6a).

3 Sequential Computation of Variables

In this section, a framework for sequential computation of variables is presented. The framework is built upon the concepts BLT semi-explicit DAE form, tools, and computation sequence. The small model (1) introduced in Section 1, will be used as a running example to illustrate and exemplify the theory.

Large sets of equations often has a sparse structure, i.e. only a few unknown variables in each equation. This makes it possible to partition the set of equations into subsets that can be solved, in a sequence, for only a subset of the unknowns. The main argument for computing variables in this way is efficiency and in some cases this may be the only feasible way to compute the unknowns. This approach has been used in the context of equation solving, see [Steward, 1962], [Kron, 1963] [Steward, 1965], and is also utilized in methods for non-causal simulation [Fritzon, 2004].

3.1 BLT Semi-Explicit DAE form

One property that the partitioning must fulfill, is that computation of variables from a certain subset of equations must only use variables that are known, that is, measured or have been computed from another subset in a previous step of the sequence.

Furthermore, with the efficiency argument in mind, it is most desirable to partition the set of equations into as small blocks, i.e. subsets, as possible.

However, even if the equation set has a sparse structure, there could be algebraic or differential loops, that makes it impossible to consider subsets of solely one equation.

In addition, it is desirable that the equations are partitioned into blocks or subsets from which variables can be computed in a straightforward manner. Since the considered set of equations (3) contains both differential and algebraic equations, subsets will correspond to DAEs. Computation of variables from semi-explicit DAEs of index one, referred to as simulation of the DAE, is a well studied problem and several methods exist, see e.g. [Hairer and Wanner, 2002], [Ascher and Petzold, 1998]. Furthermore, as said in Section 2.4, a semi-explicit DAE of index one can trivially be transformed to an explicit ODE. Explicit ODEs are suitable for real-time simulation in embedded systems, for example Engine Control Units (ECUs), because real-time simulation often require use of an explicit integration method, e.g. forward Euler [Ascher and Petzold, 1998], which assumes an explicit ODE. For a detailed discussion regarding real-time simulation, see [Cellier and Kofman, 2006].

Motivated by these arguments, we consider a partitioning of the equation set so that a block-lower triangular form is achieved, where each block corresponds to a semi-explicit DAE of index one.

Definition 2 (BLT semi-explicit DAE form)**.** *The system*

$$\begin{aligned}
\dot{\mathbf{z}}_1 &= \mathbf{g}_1\left(\mathbf{z}_1, \mathbf{w}_1, \mathbf{y}\right) \\
\dot{\mathbf{z}}_2 &= \mathbf{g}_2\left(\mathbf{z}_1, \mathbf{z}_2, \mathbf{w}_1, \mathbf{w}_2, \mathbf{y}\right) \\
&\vdots \\
\dot{\mathbf{z}}_s &= \mathbf{g}_s\left(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_s, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_s, \mathbf{y}\right)
\end{aligned} \tag{7}$$

where $\mathbf{w}_i = \left(\mathbf{w}_i^1, \mathbf{w}_i^2, \ldots, \mathbf{w}_i^{p_i}\right)$ *and*

$$\begin{aligned}
\mathbf{w}_i^1 &= \mathbf{h}_i^1\left(\Psi_i, \mathbf{y}\right) \\
\mathbf{w}_i^2 &= \mathbf{h}_i^2\left(\Psi_i, \mathbf{w}_i^1, \mathbf{y}\right) \\
&\vdots \\
\mathbf{w}_i^{p_i} &= \mathbf{h}_i^{p_i}\left(\Psi_i, \mathbf{w}_i^2, \ldots, \mathbf{w}_i^{p_i-1}, \mathbf{y}\right),
\end{aligned}$$

where

$$\Psi_i = \left(\dot{\mathbf{w}}_1, \dot{\mathbf{w}}_2, \ldots, \dot{\mathbf{w}}_{i-1}, \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_i, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{i-1}\right),$$

for $i = 1, 2, \ldots, s$*, and where* $\mathbf{z}_i$ *and* $\mathbf{w}_i$ *are vectors of unknown variables, all pairwise disjoint, and* $\mathbf{y}$ *a vector of known variables, is in* Block-Lower Triangular semi-explicit DAE form *(BLT semi-explicit DAE form).*

Note that it is not necessary that both $\mathbf{z}_i$ and $\mathbf{w}_i$ are present in (7) for every $i = 1, 2, \ldots, s$. In particular, the system

$$\begin{aligned}
\mathbf{w}_1 &= \mathbf{h}_1\left(\mathbf{y}\right) \\
\mathbf{w}_2 &= \mathbf{h}_2\left(\mathbf{w}_1, \mathbf{y}\right) \\
&\vdots \\
\mathbf{w}_s &= \mathbf{h}_s\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{s-1}, \mathbf{y}\right),
\end{aligned}$$

containing no differentiated variables at all, also is in BLT semi-explicit DAE form.

Some Properties of the BLT Semi-Explicit DAE Form

Consider the system

$$\dot{\mathbf{z}}_1 = \mathbf{g}_1\left(\mathbf{z}_1, \mathbf{w}_1, \mathbf{y}\right) \tag{8a}$$
$$\mathbf{w}_1^1 = \mathbf{h}_1^1\left(\mathbf{z}_1, \mathbf{y}\right) \tag{8b}$$
$$\mathbf{w}_1^2 = \mathbf{h}_1^2\left(\mathbf{z}_1, \mathbf{w}_1^1, \mathbf{y}\right) \tag{8c}$$
$$\dot{\mathbf{z}}_2 = \mathbf{g}_2\left(\mathbf{z}_1, \mathbf{z}_2, \mathbf{w}_1, \mathbf{w}_2, \mathbf{y}\right) \tag{8d}$$
$$\mathbf{w}_2^1 = \mathbf{h}_2^1\left(\dot{\mathbf{w}}_1, \mathbf{z}_1, \mathbf{z}_2, \mathbf{w}_1, \mathbf{y}\right) \tag{8e}$$
$$\mathbf{w}_2^2 = \mathbf{h}_2^2\left(\dot{\mathbf{w}}_1, \mathbf{z}_1, \mathbf{z}_2, \mathbf{w}_1, \mathbf{w}_2^1, \mathbf{y}\right), \tag{8f}$$

where $\mathbf{w}_1 = \left(\mathbf{w}_1^1, \mathbf{w}_1^2\right)$ and $\mathbf{w}_2 = \left(\mathbf{w}_2^1, \mathbf{w}_2^2\right)$, is in BLT semi-explicit DAE form with $s = 2$ and $p_1 = p_2 = 2$. By studying the system (8), we can deduce some properties of the BLT semi-explicit DAE form;

Mixed Causality The form generalizes the use of integral and derivative causality, since for example integral causality is used in (8a) and derivative causality in (8e).

Blocks are DAEs of Index One or Zero Each block, e.g. (8a)-(8c), corresponds to a semi-explicit DAE of, at most, index one with respect to the unknown variables in each block, i.e. $\mathbf{z}_1$ and $\mathbf{w}_1$ in the first block and $\mathbf{z}_2$ and $\mathbf{w}_2$ in the second block. Note that in accordance with the note above, vectors $\mathbf{z}_1$, $\mathbf{z}_2$, $\mathbf{w}_1$, and $\mathbf{w}_2$ must not all be present in (8). If, for instance, $\mathbf{w}_1$ is missing and hence also (8b) and (8c), the first block is an explicit ODE, i.e. a DAE of index zero. If both $\mathbf{z}_1$ and $\mathbf{w}_1$ are present, the first block corresponds to a semi-explicit DAE of index one.

Transformation to ODE Due to the previous property, a system in BLT semi-explicit DAE form can trivially be transformed to a variant of an explicit ODE. In (8), we may substitute (8b) into (8c) and then substitute the result along with (8b) into (8a) so that we obtain

$$\begin{aligned}\dot{\mathbf{z}}_1 &= \mathbf{g}_1\left(\mathbf{z}_1, \mathbf{w}_1, \mathbf{y}\right) \\ &= \mathbf{g}_1\left(\mathbf{z}_1, \left[\mathbf{h}_1^1\left(\mathbf{z}_1, \mathbf{y}\right), \mathbf{h}_1^2\left(\mathbf{z}_1, \mathbf{h}_1^1\left(\mathbf{z}_1, \mathbf{y}\right)\right)\right], \mathbf{y}\right) \\ &= \tilde{\mathbf{g}}_1\left(\mathbf{z}_1, \mathbf{y}\right),\end{aligned}$$

and then repeat the procedure for the second block to obtain

$$\begin{aligned}\dot{\mathbf{z}}_1 &= \tilde{\mathbf{g}}_1\left(\mathbf{z}_1, \mathbf{y}\right) \\ \dot{\mathbf{z}}_2 &= \tilde{\mathbf{g}}_2\left(\mathbf{z}_1, \mathbf{z}_2, \dot{\mathbf{y}}, \mathbf{y}\right).\end{aligned}$$

As said above, ODEs may be preferable in real-time applications.

Blocks are SCCs Each block in the BLT semi-explicit DAE form is a SCC of the structure of the corresponding equations with respect to the unknown variables in that block. This can be seen by studying the structure[2] of the equations in (8) with respect to the variables $\left\{\mathbf{z}_1, \mathbf{w}_1^1, \mathbf{w}_1^2, \mathbf{z}_2, \mathbf{w}_2^1, \mathbf{w}_2^2\right\}$, which is shown in (9). In this structure, the equation in (8a) has been named e_1, the equation in (8b) has been named e_2, and so forth.

	$\mathbf{z}_1$	$\mathbf{w}_1^1$	$\mathbf{w}_1^2$	$\mathbf{z}_2$	$\mathbf{w}_2^1$	$\mathbf{w}_2^2$
e_1	**1**	**1**	**1**			
e_2	**1**	**1**				
e_3	**1**	**1**	**1**			
e_4	1	1	1	**1**	**1**	**1**
e_5	1	1	1	**1**	**1**	
e_6	1	1	1	**1**	**1**	**1**

(9)

Efficiency Recall the discussion regarding efficiency in the beginning of Section 3.1. As a consequence of the previous property, the original set of equations is partitioned in as small blocks as possible, in the sense that there are no dependencies between blocks, i.e. no loops occur.

Sequential Computation of Variables The block-lower triangular structure makes it possible to compute variables sequentially by considering the blocks one at the time, starting from the first block. Since the structure guarantees that a certain block only contains unknown variables from the present and previous blocks.

[2]It is here assumed that $f(x)$ implies $\frac{\partial f}{\partial x} \not\equiv 0$.

3.2 Tools

If a system in BLT semi-explicit DAE form can be obtained from a given set of equations and if trajectories of the unknown variables can be computed from the resulting system, depends naturally on the properties of the equations in the model. Equally important is also the set of *tools* that are available for use.

Consider the BLT semi-explicit DAE form (8). To obtain for example the function $\mathbf{h}_1^1$ in (8b) from a subset of equations given in the model, some kind of tool for algebraic equation solving is needed. To compute a trajectory of the variable $\mathbf{z}_1$ from (8a), a differential equation must be solved and hence a tool for this is needed. Furthermore, to obtain the derivative $\dot{\mathbf{w}}_1$, present in (8e), from the trajectory of $\mathbf{w}_1$ computed in (8b) and (8c), a tool for differentiation is needed.

Motivated by this discussion, we consider three types of tools; algebraic equation solving tools, differential equation solving tools, and differentiation tools.

Algebraic Equation Solving Tools

A tool for algebraic equation solving is typically some software package for symbolic or numeric solving of linear or non-linear algebraic equations. Algebraic equation solving tools are essential for handling models containing algebraic loops. If, for example, the available tool for algebraic equation solving tool only can solve scalar equations, loops can not be handled.

More precisely, an algebraic equation solving tool (AE tool) is a function taking a set of variables $V_i \subseteq X \cup D$ and a set of equations $E_i \subseteq E$ as arguments, and returning a function $\mathbf{g}_i$, which can be a symbolic expression or numeric algorithm, taking variables from $\{X \cup D\} \setminus V_i$ and Y as arguments and returning a vector corresponding to the elements in V_i. Now assume that $\mathbf{g}_i$ is the function returned by an AE tool when E_i and V_i are used as arguments, and that the equation set $\bar{E}_i$ corresponds to $\mathbf{v}_i = \mathbf{g}_i(\mathbf{u}_i, \mathbf{y})$, where $\mathbf{v}_i$ is a vector of the elements in V_i, $\mathbf{u}_i$ a vector of the elements in $U_i \subseteq \{X \cup D\} \setminus V_i$, and $\mathbf{y}$ a vector of known variables. A natural assumption regarding an AE tool, whatever algorithm or method it corresponds to, is that the AE tool should not introduce new solutions. That is, a solution to $\bar{E}_i$ should also be a solution to the original equation set E_i. Moreover, an AE tool should neither remove solutions, i.e. solutions to E_i must also be solutions to $\bar{E}_i$. Furthermore, motivated by the idea of using sequential computation of variables for residual equation, we are interested in unique solutions. This discussion justifies the following assumption.

Assumption 1. *Given U_i and $\mathbf{y}$, the solution sets of $\bar{E}_i$, obtained from the AE tool, and E_i, with respect to V_i, are equal and unique.*

AE tools giving unique solutions generally assumes that the given set of equations contains as many equations as unknown variables. For example,

Newton iteration which is a common numerical method for solving non-linear equations require a just-determined equation set, see e.g. [Ortega and Rheinboldt, 2000]. In addition, under- and over-determined sets of equations for which an unique analytical solution exists are rare. This motivates the following assumption.

Assumption 2. *An AE tool requires that its arguments E_i and V_i correspond to a just-determined equation set.*

In this work, we assume that tools for algebraic equation solving are available through existing standard software packages like e.g. Maple or Mathematica, and design and implementation of such tools will not be considered. For solving algebraic loops, also tearing can be a successful approach, [Steward, 1965]. In the following, we also assume that AE tools fulfill the properties stated in Assumptions 1 and 2.

Differential Equation Solving Tools

A differential equation solving tool is typically a method or software for numerical integration of an (explicit or implicit) ODE, i.e. a DAE of index zero. Numerical integration is a well studied area and there are several efficient approaches and methods, see e.g. [Brenan et al., 1989], [Ascher and Petzold, 1998]. Implementations are available in for example MATLAB and SIMULINK.

Recall from Section 3.1 that each block in a BLT semi-explicit DAE system can be transformed to an explicit ODE. In the following, we assume that differential equation solving tools are always available and that an explicit ODE can be solved, i.e. that trajectories of the state variables in the ODE can be computed, if the initial conditions of the state variables are known and consistent. Of course, this assumption is not always valid and numerical solving of ODEs involves difficulties and problems such as stability and stiffness, but this is not in the scope of this paper.

The availability of initial conditions depends on the knowledge about the underlying system represented by the model. For complex physical systems, object-oriented modeling tools, e.g. Modelica [Mattson et al., 1998], are frequently used to build models. Often, this leads to that state variables in the models correspond to physical quantities such as pressures and temperatures, which makes initial conditions known.

If all equilibrium points of the considered ODE are, or with for example state-feedback can be made (globally) asymptotically stable, see for example [Khalil, 2002], the effect of the initial conditions are neglectable. However, the computed trajectory will in this case differ from the true trajectory for some time due to transients.

Differentiation Tools

A differentiation tool is for example an implementation of a method for approximate differentiating of known variables. There are several approaches, e.g. low-pass filtering or smoothing spline approximation [Wei and Li, 2006]. An extensive survey of methods can be found in [Barford et al., 1999]. Methods for approximate differentiation is not in the scope of this paper, and will not be further considered.

In the following, we assume that differentiation of a set of known variables either is possible or not possible. That is, if a tool for approximate differentiation is available, we assume that the quality of the measurements of the involved variables are good enough to support the tool.

One alternative to differentiate variables directly, is to propagate unknown differentiated variables through a set of equations so that these can be expressed as derivatives of measured variables only. Assume for example that we want to compute the derivative $\dot{x}_1$ and we also have that $x_1 = y_1$. To compute $\dot{x}_1$, we use a differentiation tool to compute $\dot{y}_1$ and then use $\dot{x}_1 = \dot{y}_1$.

3.3 Computation Sequence

To describe the way and order in which a set of variables is computed from a set of equations, we will introduce the concept *computation sequence*. Before going into details, we need some additional notation. Let $V \subseteq X \cup D$ and define

$$\text{Diff}\,(V) = \left\{\dot{x}_j \in D : \dot{x}_j \in V \vee x_j \in V\right\}, \tag{10}$$

$$\text{unDiff}\,(V) = \left\{x_j \in X : x_j \in V \vee \dot{x}_j \in V\right\}. \tag{11}$$

For instance, we have that $\text{Diff}\,(\{\dot{x}_1, x_2\}) = \{\dot{x}_1, \dot{x}_2\}$ and $\text{unDiff}\,(\{\dot{x}_1, x_2\}) = \{x_1, x_2\}$.

Now consider the model $M(E, X, Y)$, where E is the set of equations specified in (3), X the set of unknown variables, and Y the set of known variables.

Definition 3 (Computation Sequence)**.** *Given are a set of variables $X' \subseteq X$, an AE tool $\mathcal{T}$, and an ordered set*

$$\mathcal{C} = \left((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)\right),$$

where $V_i \subseteq \text{var}_X(E_i) \cup \text{var}_D(E_i)$ and $\{E_i\}$ is pairwise disjoint. The ordered set $\mathcal{C}$ is a computation sequence for X' with $\mathcal{T}$*, if*

1. $X' \subseteq \text{unDiff}\,(V_1 \cup V_2 \cup \ldots \cup V_k)$*, and*
2. *a system in BLT semi-explicit DAE form is obtained by sequentially calling the tool $\mathcal{T}$, with arguments V_i and E_i, for each element $(V_i, E_i) \in \mathcal{C}$.*

For an example, recall the model (1), where $E = \{e_1, e_2, e_3, e_4, e_5\}$, $X = \{x_1, x_2, x_3, x_4\}$ and $Y = \{y_1, y_2, y_3\}$. Assume that the given AE tool $\mathcal{T}$ is ideal,

in the sense that it can solve all solvable linear and non-linear equations. Then the ordered set

$$\mathcal{C} = (\left(\{x_3\}, \{e_4\}\right), \left(\{x_4\}, \{e_2\}\right), \left(\{\dot{x}_1\}, \{e_1\}\right), \left(\{x_2\}, \{e_3\}\right)) \tag{12}$$

is a computation sequence for $\{x_1, x_2, x_3, x_4\}$ with $\mathcal{T}$ according to Definition 3, since

$$\text{unDiff}\left(\{x_3\} \cup \{x_4\} \cup \{\dot{x}_1\} \cup \{x_2\}\right) = \{x_1, x_2, x_3, x_4\},$$

and the BLT semi-explicit DAE system

$$x_3 = y_3 \tag{13a}$$
$$x_4 = \dot{x}_3 \tag{13b}$$
$$\dot{x}_1 = x_2 \tag{13c}$$
$$x_2 = \frac{-\dot{x}_4 x_1 + y_1}{2x_4}, \tag{13d}$$

is obtained by sequentially calling $\mathcal{T}$ with elements from $\mathcal{C}$ as arguments.

Note that the obtained BLT semi-explicit DAE system (13) has three blocks; the first block corresponds to (13a), the second to (13b), and the third to (13c) and (13d). Also note that the equation set $\{e_3, e_1\}$, containing a differential loop, corresponds to a semi-explicit DAE of index one given by (13c) and (13d), and that derivative causality is used in (13b) and integral causality in (13c).

4 Sequential Residual Generation

In this section it is shown how a computation sequence can be utilized for residual generation. A residual generator based on a computation sequence will be defined as a sequential residual generator. In a sequential residual generator, the generation of a residual will consist of finite sequence of variable computations ending with evaluation of an unused equation. The concepts of minimal and irreducible computation sequence, as well as proper sequential residual generator will then be introduced. A necessary condition for the existence of a proper sequential residual generator is given. The section ends with an algorithm able to find proper sequential residual generators, given a model and an AE tool.

An important property of a computation sequence is given by the following lemma.

Lemma 3. *Let the ordered set*

$$\mathcal{C} = (\left(V_1, E_1\right), \left(V_2, E_2\right), \ldots, \left(V_k, E_k\right))$$

be a computation sequence for the variables $X' \subseteq X$ with the AE tool $\mathcal{T}$, and let $\bar{E}'$ be the set of equations in BLT semi-explicit DAE form obtained from $\mathcal{C}$ with the AE tool $\mathcal{T}$. Then the solution sets of $\bar{E}'$ and $E_1 \cup E_2 \cup \ldots \cup E_k$, with respect to $V_1 \cup V_2 \cup \ldots \cup V_k$, are equal and unique.

With this lemma, the following important result can be proved.

Theorem 1. *Let $M(E, X, Y)$ be a model, $\mathcal{T}$ an AE tool, and*

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)),$$

a computation sequence for $X' \subseteq X$ with $\mathcal{T}$, where $E_i \subseteq E$. Also, let $e_i \in E \setminus E_1 \cup E_2 \cup \ldots \cup E_k$ where $\mathrm{var}_X(e_i) \subseteq X'$ and it is assumed that e_i is written as $f_i(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}) = 0$. Then the BLT semi-explicit DAE system obtained from $\mathcal{C}$ with $\mathcal{T}$ and $r = f_i(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y})$, is a residual generator for $M(E, X, Y)$ if

1. *consistent initial conditions of all states are available, and*
2. *all needed derivatives can be computed with the available differentiation tools.*

Motivated by this theorem, we define a sequential residual generator as follows.

Definition 4 (Sequential Residual Generator)**.** *A residual generator for $M(E, X, Y)$ obtained from a computation sequence $\mathcal{C}$ and an equation $e_i \in E$, in accordance with the description in Theorem 1, is a* sequential residual generator *for $M(E, X, Y)$, denoted $\mathcal{S} = (\mathcal{T}(\mathcal{C}), e_i)$, and e_i is a* residual equation*.*

4.1 Proper Sequential Residual Generator

Regarding implementation aspects, e.g. complexity or numerical issues, smaller computation sequences are generally better. In particular, it is unnecessary to compute variables that are not contained in the residual equation, or not used to compute any of the variables contained in the residual equation. Motivated by this discussion, we make the following definition.

Definition 5 (Minimal Computation Sequence)**.** *Given a set of variables $X' \subseteq X$ and an AE tool $\mathcal{T}$, a computation sequence $\mathcal{C}$ for X' with $\mathcal{T}$ is* minimal*, if there is no other computation sequence $\mathcal{C}'$ for X' with $\mathcal{T}$ such that $\mathcal{C}' \subset \mathcal{C}$.*

Return to the model (1) in Section 1. Consider the last two equations in the model,

$$\begin{aligned} e_4: &\quad x_3 - y_3 = 0 \\ e_5: &\quad x_2 - y_2 = 0, \end{aligned}$$

and let $\mathcal{T}$ be an ideal AE tool. The computation sequence

$$\mathcal{C}_1 = ((\{x_3\}, \{e_4\}), (\{x_2\}, \{e_5\})) \tag{14}$$

for $\{x_2, x_3\}$ with $\mathcal{T}$ is minimal. The resulting BLT semi-explicit DAE form is given by

$$x_3 = y_3 \tag{15a}$$
$$x_2 = y_2. \tag{15b}$$

However, $\mathcal{C}_1$ is not minimal for $\{x_3\}$ since $\mathcal{C}_2 = (\{x_3\}, \{e_4\})$ is a (minimal) computation sequence for $\{x_3\}$ with $\mathcal{T}$, and $\mathcal{C}_2 \subset \mathcal{C}_1$.

Computation of variables according to a minimal computation sequence thus implies that no unnecessary variables are computed. However, with the complexity and numerical aspects in mind, it is also most desirable that computation of variables in each step is performed from as small equation sets as possible. This leads to the following definition.

Definition 6 (Irreducible Computation Sequence)**.** *Given a set of variables $X' \subseteq X$ and an AE tool $\mathcal{T}$, a computation sequence*

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)),$$

for X' with $\mathcal{T}$ is irreducible*, if no element $(V_i, E_i) \in \mathcal{C}$ can be partitioned as $V_i = V_{i1} \cup V_{i2}$ and $E_i = E_{i1} \cup E_{i2}$, such that*

$$\mathcal{C}' = ((V_1, E_1), \ldots, (V_{i1}, E_{i1}), (V_{i2}, E_{i2}), \ldots, (V_k, E_k))$$

is a computation sequence for X' with $\mathcal{T}$.

Return to the equation set $\{e_4, e_5\}$ considered above. Clearly, the ordered set $\mathcal{C}_3 = (\{x_2, x_3\}, \{e_4, e_5\})$ is a minimal computation sequence for $\{x_2, x_3\}$ with the ideal AE tool $\mathcal{T}$. The corresponding BLT semi-explicit DAE system is given by (15). However, $\mathcal{C}_3$ is not irreducible since $\mathcal{C}_1$ given by (14) is also a computation sequence for $\{x_2, x_3\}$.

From now on, we will only consider AE tools fulfilling the following, quite non-limiting, property.

Assumption 3. *Let $E_i = E_{i1} \cup E_{i2}$ and $V_i = V_{i1} \cup V_{i2}$, in accordance with Definition 6. If an AE tool can solve E_i for V_i, it can also solve E_{i1} for V_{i1} and E_{i2} for V_{i2}.*

Sequential residual generators based on minimal and irreducible computation sequences are of particular interest.

Definition 7 (Proper Sequential Residual Generator)**.** *Given an equation $e_i \in E$, an AE tool $\mathcal{T}$, and a computation sequence $\mathcal{C}$ for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$, a sequential residual generator $\mathcal{S} = (\mathcal{T}(\mathcal{C}), e_i)$ is* proper*, if $\mathcal{C}$ is a minimal and irreducible computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$.*

For construction of a sequential residual generator, a computation sequence and a residual equation is needed. Due to Assumption 2, the equation set contained in a computation sequence is a just-determined set of equations. Since the residual equation is redundant, see Theorem 1, it follows that the equations in a computation sequence and the residual equation constitute an over-determined equation set. Hence, an over-determined set of equations is needed to construct a sequential residual generator. For construction of a proper sequential residual generator, a *minimal structurally over-determined* (MSO) set [Krysander et al., 2008], is needed.

Theorem 2. *Let $\mathcal{S} = (\mathcal{T}(\mathcal{C}), e_i)$ be a proper sequential residual generator, where*

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)),$$

then the equation set $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ is an MSO set with respect to $\mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i)$

Note that Theorem 2 establishes a link between structural and analytical methods. This is done without the use of any assumptions of generic equations as in e.g. [Krysander et al., 2008], instead assumptions have been placed on the tools.

Recall again the model (1) and consider the computation sequence $\mathcal{C}$, given by (12), with the corresponding BLT semi-explicit DAE form (13). The computation sequence $\mathcal{C}$ together with the equation e_5 is a sequential residual generator for the model (1), if we assume that the initial condition of x_1 is known and consistent and the derivatives $\dot{x}_3$ and $\dot{x}_4$ can be computed with the available differentiation tools. As a matter of fact, the residual generator is a *proper* sequential residual generator since the computation sequence $\mathcal{C}$ for $\mathrm{var}_X(e_5) = \{x_2\}$ with the ideal AE tool $\mathcal{T}$ is minimal and irreducible. Hence, we can by Theorem 2 conclude that the equation set $E = \{e_1, e_2, e_3, e_4, e_5\}$ is an MSO set.

4.2 Finding Proper Sequential Residual Generators

Theorem 2 states a necessary condition for the existence of a proper sequential residual generator. Hence, a first step when searching for all proper sequential residual generators may be to find all MSO sets. There are efficient algorithms for finding all MSO sets in large equation sets, see e.g. [Krysander et al., 2008].

Motivated by this, we propose the following algorithm for finding proper sequential residual generators, given a model $M(E, X, Y)$ and an AE tool $\mathcal{T}$. The function FINDALLMSOS is assumed to find all MSO sets in the equation set E. The function FINDCOMPUTATIONSEQUENCE, taking an equation set E', a variable set X' and an AE tool $\mathcal{T}$, is assumed to return a minimal and proper computation sequence for X' with $\mathcal{T}$.

```
1:  function FINDRESIDUALGENERATORS(E, X, T)
2:      R := ∅
3:      MSOs := FINDALLMSOS(E, X)
4:      for all Ē ∈ MSOs do
5:          X' := var_X(Ē)
6:          for all e_i ∈ Ē do
7:              E' := Ē \ e_i
8:              C := FINDCOMPUTATIONSEQUENCE(E', X', T)
9:              if C ≠ ∅ then
10:                 R = R ∪ {(T(C), e_i)}
11:             end if
```

```
12:         end for
13:     end for
14:     return R
15: end function
```

The algorithm is justified by the following theorem.

Theorem 3. *Let $M(E, X, Y)$ be a model and $\mathcal{T}$ an AE tool. Also, let R be the set returned by* FINDRESIDUALGENERATORS *when E, X, and $\mathcal{T}$ are used as input. Then all elements $(\mathcal{T}(\mathcal{C}), e_i) \in R$ are proper sequential residual generators for $M(E, X, Y)$ if, in accordance with Theorem 1, consistent initial conditions of all states are available, and all needed derivatives can be computed with the available differentiation tools.*

The most important step in FINDRESIDUALGENERATORS is thus to find a minimal and irreducible computation sequence, i.e. the function FINDCOMPUTATIONSEQUENCE. This is the topic of next section.

5 Method for Finding a Computation Sequence

A proper sequential residual generator consists of a BLT semi-explicit DAE system, obtained from a minimal and irreducible computation sequence, and a residual equation. Essential for construction of a proper sequential residual generator is thus to find a minimal and irreducible computation sequence. The method that we propose for finding a computation sequence is presented in this section. First, the different steps of the method are illustrated by studying an example.

5.1 Illustrative Example

Consider the following set of equations,

$$\begin{aligned}
e_1: &\quad \dot{x}_1 + x_1 x_6 - x_3 - x_5^2 x_7 = 0 \\
e_2: &\quad \dot{x}_2 + x_2 x_3 + y_1 = 0 \\
e_3: &\quad \dot{x}_3 + x_3 - x_2 x_4 + y_2 = 0 \\
e_4: &\quad \dot{x}_4 + x_2 - x_5 - y_3 = 0 \\
e_5: &\quad x_1 - x_2 x_3 - x_4 + x_6 - 2x_7 - y_4 = 0 \\
e_6: &\quad x_3^2 - x_6 - x_7 + y_5 = 0 \\
e_7: &\quad x_4 - y_6 = 0,
\end{aligned}$$

where $X = \{x_1, x_2, \ldots, x_7\}$ are unknown variables and $Z = \{y_1, y_2 \ldots, y_6\}$ known variables. Assume that we want to find a computation sequence for X with a given AE tool.

First identify the SCCs, recall Section 2.3, of the structure of $E = \{e_1, e_2, \ldots, e_7\}$ with respect to X, and order the corresponding partitions of the equation and

variable sets accordingly

	x_4	x_3	x_2	x_5	x_6	x_7	x_1
e_7	**1**						
e_3	1	**1**	**1**				
e_2		**1**	**1**				
e_4	1		1	**1**			
e_6		1			**1**	**1**	
e_1		1		1	**1**	**1**	**1**
e_5	1	1	1		**1**	**1**	**1**

(16)

The ordered partitions are

$$\mathcal{E} = (\{e_7\}, \{e_2, e_3\}, \{e_4\}, \{e_1, e_5, e_6\})$$

and

$$\mathcal{X} = (\{x_4\}, \{x_2, x_3\}, \{x_5\}, \{x_1, x_6, x_7\}),$$

where each element in $\mathcal{E}$ is a SCC with respect to the corresponding element in $\mathcal{X}$, e.g. $(\{e_2, e_3\}, \{x_2, x_3\})$. The SCCs are marked with bold in (16).

The first SCC, $(\{e_4\}, \{x_7\})$, contains one linear algebraic equation. Under assumption that our AE tool can handle such equations, e_7 is solved for x_4 and we obtain

$$x_4 = y_6. \tag{17}$$

Then consider the next SCC, $(\{e_2, e_3\}, \{x_2, x_3\})$ which contains two differential equations. The permuted structure of $\{e_2, e_3\}$ with respect to the differentiated variables $\{\dot{x}_2, \dot{x}_3\}$ is

	$\dot{x}_3$	$\dot{x}_2$
e_3	1	
e_2		1

(18)

As seen, the structure (18) contains two SCCs of size one, $(\{e_3\}, \{\dot{x}_3\})$ and $(\{e_2\}, \{\dot{x}_2\})$. Assuming our AE tool admits it, we then solve e_3 for $\dot{x}_3$ and e_2 for $\dot{x}_2$ and obtain

$$\begin{aligned} \dot{x}_3 &= -x_3 + x_2 x_4 - y_2 \\ \dot{x}_2 &= -x_2 x_3 - y_1. \end{aligned} \tag{19}$$

The next SCC, $(\{e_4\}, \{x_5\})$, contains a differential equation. However, since x_5 is the variable intended to compute from the equation, we can handle e_6 as an algebraic equation and solve it for x_5,

$$x_5 = x_2 + \dot{x}_4 - y_3. \tag{20}$$

The SCC $(\{e_1, e_5, e_6\}, \{x_1, x_6, x_7\})$ contains the differential equation e_1 and the two algebraic equations e_6 and e_5. By analyzing the equations we see that x_6 and x_7 are algebraic variables contained in both e_6 and e_5 and that x_1 is a differentiated variable present in e_1. We then solve e_1 for $\dot{x}_1$ and obtain

$$\dot{x}_1 = -x_1 x_6 + x_5^2 x_7 + x_3. \tag{21}$$

The structure of $\{e_5, e_6\}$ with respect to $\{x_6, x_7\}$ reveals a SCC of size two, see (22).

	x_6	x_7
e_5	1	1
e_6	1	1

(22)

Under the assumption that our AE tool can handle it, we solve the equation system $\{e_5, e_6\}$ for $\{x_6, x_7\}$ and obtain

$$\begin{aligned} x_6 &= 2x_3^2 + x_1 - x_2 x_3 - x_4 + 2y_5 - y_4 \\ x_7 &= x_1 - x_2 x_3 - x_4 + x_3^2 + y_5 - y_4. \end{aligned} \tag{23}$$

Collecting the equations (17), (19), (20), (21), and (23) gives

$$x_4 = y_6 \tag{24a}$$
$$\dot{x}_3 = -x_3 + x_2 x_4 - y_2 \tag{24b}$$
$$\dot{x}_2 = -x_2 x_3 - y_1 \tag{24c}$$
$$x_5 = x_2 + \dot{x}_4 - y_3 \tag{24d}$$
$$\dot{x}_1 = -x_1 x_6 + x_5^2 x_7 + x_3 \tag{24e}$$
$$x_6 = 2x_3^2 + x_1 - x_2 x_3 - x_4 + 2y_5 - y_4 \tag{24f}$$
$$x_7 = x_1 - x_2 x_3 - x_4 + x_3^2 + y_5 - y_4, \tag{24g}$$

which is a system in BLT semi-explicit DAE form with four blocks. The equation (24a) correspond to the first block, which only contains an algebraic equation. The second block is given by (24b) and (24c), and correspond to an explicit ODE with respect to the variables $\{x_2, x_3\}$. Hence, integral causality is used in this block. The third block contains (24d), which is a differential equation in which derivative causality is used. The equations (24e)- (24g) constitute the fourth and last block. This block corresponds to a semi-explicit DAE of index one, with respect to the variables $\{x_1, x_6, x_7\}$.

The resulting computation sequence for $\{x_1, x_2, \ldots, x_7\}$ with the given AE tool is,

$$\begin{aligned} \mathcal{C} = (&(\{x_4\}, \{e_7\}), (\{\dot{x}_3\}, \{e_3\}), (\{\dot{x}_2\}, \{e_2\}), (\{x_5\}, \{e_4\}), \\ &(\{\dot{x}_1\}, \{e_1\}), (\{x_6, x_7\}, \{e_6, e_5\})). \end{aligned}$$

5.2 Summary of the Method

Given an AE tool and a just-determined set of equations, the proposed method for finding a computation sequence can be outlined as follows.

1. Find the SCCs of the structure of the equation set with respect to the unknown variables. No distinction is made between a variable and its derivative.
2. For each SCC, split the equations into one set of differential equations and one set of algebraic equations, and the variables into one set of differential variables and one set of algebraic variables.
3. For the differential equations, find the SCCs of the structure of the differential equations with respect to the differentiated variables. For each SCC, try to solve the differential equations for the intended differential variables with the AE tool. Note that due to the assumption that each differential equation only contains one differentiated variable, all SCCs are of size one.
4. For the algebraic equations, find the SCCs of the structure of the algebraic equations with respect to the algebraic variables. For each SCC, try to solve the algebraic equations for the intended algebraic variables with the AE tool.

5.3 Algorithm

The method is formally described in the function FINDCOMPUTATIONSEQUENCE below. The function takes a just-determined equation set $E' \subseteq E$, a set of unknown variables $X' \subseteq X$, and an AE tool $\mathcal{T}$ as input, and returns an ordered set $\mathcal{C}$ as output. The function FINDALLSCCS is assumed to return an ordered set of equation and variable pairs, where each pair corresponds to a SCC of the structure of the equation set with respect to the variable set. The order of the SCCs returned by FINDALLSCCS is assumed to be the one depicted in Figure 1, for more information regarding ordering of SCCs please refer to [Murota, 1987]. There are efficient algorithms for finding SCCs in directed graphs, see for example [Tarjan, 1972]. The DM-decomposition [Dulmage and Mendelsohn, 1958] can also be utilized. In MATLAB, the DM-decomposition is implemented in the function `dmperm`, from which also the order of the SCCs, according to Figure 1, easily can be obtained. Other functions used in FINDCOMPUTATIONSEQUENCE are:

- DIFF and UNDIFF, takes a variable set as input and returns its differentiated and un-differentiated correspondence, see (10) and (11).
- ISINITCONDKNOWN determines if the initial conditions of the given variables are known and consistent, and the function ISDIFFERENTIABLE determines if the given variables can be differentiated with the available differentiation tool.

- ISJUSTDETERMINED is used to determine if the structure of the given equation set, with respect to the given variable set, is just-determined. This is essential, since otherwise the computation of SCCs makes no sense.
- GETDIFFERENTIALEQUATIONS takes a set of equations and a set of differentiated variables as input, and returns the differential equations in which the given differentiated variables are contained.
- ISTOOLSOLVABLE determines if the given AE tool can solve the given equations for the given set of variables.
- APPEND, taking an ordered set and an element as input, simply appends the element to the end of the set.
- The operator $|\cdot|$, taking a set as input, is assumed to return the number of elements in the set and the notion $A(i)$ is used to refer to the i:th element of the ordered set A.

1: **function** FINDCOMPUTATIONSEQUENCE($E', X', \mathcal{T}$)
2: $\mathcal{C} := \emptyset$
3: $S :=$ FINDALLSCCS(E', X')
4: **for** $i = 1, 2, \ldots, |S|$ **do**
5: $(E_i, X_i) := S(i)$
6: $D_i :=$ DIFF(X_i)
7: $Z_i := \text{var}_D(E_i) \cap D_i$
8: $W_i := X_i \setminus$ UNDIFF(Z_i)
9: **if** not ISINITCONDKNOWN(Z_i) **then**
10: **return** $\emptyset$
11: **end if**
12: $E_{Z_i} :=$ GETDIFFERENTIALEQUATIONS(E_i, Z_i)
13: $E_{W_i} := E_i \setminus E_{Z_i}$
14: $S_{Z_i} :=$ FINDALLSCCS(E_{Z_i}, Z_i)
15: **for** $j = 1, 2, \ldots, \left|S_{Z_i}\right|$ **do**
16: $\left(E^j_{Z_i}, Z^j_i\right) := S_{Z_i}(j)$
17: **if** ISTOOLSOLVABLE($Z^j_i, E^j_{Z_i}, \mathcal{T}$) **then**
18: APPEND($\mathcal{C}, \left(Z^j_i, E^j_{Z_i}\right)$)
19: **else**
20: **return** $\emptyset$
21: **end if**
22: **end for**
23: **if** ISJUSTDETERMINED(E_{W_i}, W_i) **then**
24: $S_{W_i} :=$ FINDALLSCCS(E_{W_i}, W_i)
25: **for** $j = 1, 2, \ldots, \left|S_{W_i}\right|$ **do**
26: $\left(E^j_{W_i}, W^j_i\right) := S_{W_i}(j)$
27: **if** ISTOOLSOLVABLE($W^j_i, E^j_{W_i}, \mathcal{T}$) **then**

```
28:                 APPEND(C, (W_i^j, E_{W_i}^j))
29:             else
30:                 return ∅
31:             end if
32:         end for
33:     else
34:         return ∅
35:     end if
36: end for
37: return C
38: end function
```

That the ordered set $\mathcal{C}$ returned by FINDCOMPUTATIONSEQUENCE, indeed, is a minimal and irreducible computation sequence is verified in the following theorem.

Theorem 4. *Let $E' \subseteq E$ be a just-determined set of equations with respect to the variables $X' \subseteq X$, and $\mathcal{T}$ an AE tool. If E', X', and $\mathcal{T}$ are used as arguments to* FINDCOMPUTATIONSEQUENCE *and a non-empty $\mathcal{C}$ is returned, then $\mathcal{C}$ is a minimal and irreducible computation sequence for X' with $\mathcal{T}$.*

6 Application Studies

The objective of this section is to empirically show the benefits of the method for finding sequential residual generators proposed in Sections 4.2 and 5.3. This is done by applying the method to models of an automotive diesel engine and an auxiliary hydraulic braking system. In addition, we illustrate how a sequential residual generator for the diesel engine, found with the proposed method, can be realized. The realized residual generator is then evaluated using real measurements from a truck.

6.1 Implementation and Configuration of the Method

The analytical models of the two systems were obtained from SIMULINK models by using the toolbox described in [Frisk et al., 2006]. The resulting models are complex DAEs containing non-linearities like min- and max-functions, look-up tables, saturations, and polynomials.

The functions FINDRESIDUALGENERATORS and FINDCOMPUTATIONSEQUENCE, described in Sections 4.2 and 5.3, were implemented in MATLAB. In the implementation of FINDCOMPUTATIONSEQUENCE, the symbolic equation solver in MAPLE was used as AE tool. To find all MSO sets, the algorithm described in [Krysander et al., 2008] was used. The MSO sets were arranged in classes, so that MSOs containing the same set of known variables belongs to the same MSO class.

Configurations

For comparison, different configurations of FINDCOMPUTATIONSEQUENCE were applied to the models. The following parameters, which naturally influences the possibility to find computation sequences, were used for configuration

SCC: The ability to handle equation sets containing algebraic or differential loops, that is, SCCs of larger size than one,

IC: The ability to use integral causality, and

DC: The ability to use derivative causality.

Note that if a configuration uses integral causality, it is assumed that all initial conditions are available. Moreover, it is assumed that all needed derivatives can be computed when a configuration uses derivative causality.

The six possible different configurations are shown in Table 1. For example, configuration SI is able to handle equation sets containing loops and use integral causality, but can not use derivative causality. The configuration corresponding to the novel approach for finding sequential residual generators proposed in this paper is SDI.

	D	I	DI	SD	SI	SDI
SCC				x	x	x
IC		x	x		x	x
DC	x		x	x		x

Table 1: The Six Configurations of the Method used in the Studies.

6.2 Performance Measures

A sequential residual generator is sensitive to those faults that influence its residual equation and the equations contained in its computation sequence. Different MSO sets correspond to different subsets of the equations in the model. Sequential residual generators obtained from computation sequences and residual equations originating from different MSO sets will thus naturally be sensitive to different subsets of faults. To achieve good fault isolation, it is hence important that residual generators can be constructed from as many MSO sets as possible.

In the automotive applications studied here, it is especially important to detect and isolate faults present in sensors and actuators, that is, faults affecting measurements of known variables. Hence, it also important that residual generators can be constructed from as many MSO classes as possible.

Additionally, different residual generators constructed from the same MSO set or MSO class may have different properties regarding for example numerical aspects, sensitivity to faults, and sensitivity for disturbances such as measurement noise or modeling errors. Hence it is most desirable to be able to

evaluate as many residual generators as possible, with real measurement data, to decide which set of residual generators to use in the final diagnosis system. Motivated by this discussion, we will use the following performance measures to compare the different configurations of the method:

MSO Sets: In how many of the total number of MSO sets at least one residual generator could be found.

MSO Classes: In how many of the total number of MSO classes at least one residual generator could be found.

Residual Generators: The total number of residual generators found.

6.3 Automotive Diesel Engine

The studied engine is a 13-liter, 6-cylinder Scania diesel engine equipped with exhaust gas recirculation (EGR) and a variable geometry turbocharger (VGT). A cutaway of the engine can be found in Figure 2.

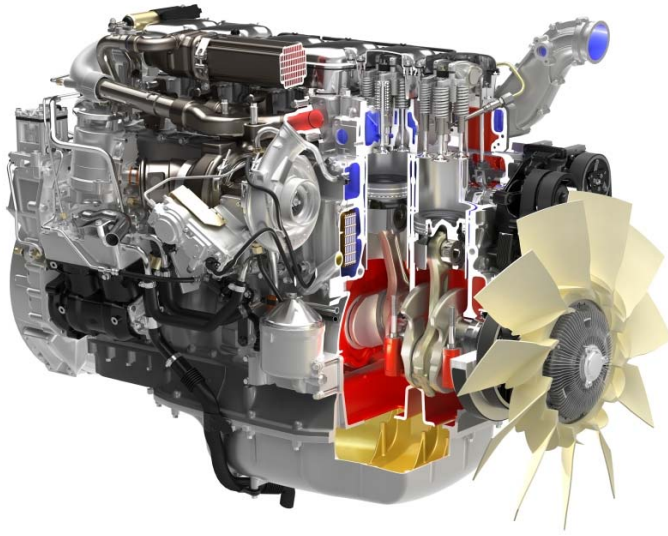


Figure 2: Cutaway of a Scania 13-liter, 6-cylinder diesel engine equipped with EGR and VGT.

The model describes the gas-flow in the engine, see [Wahlström, 2006] for more details. The analytical model extracted from the SIMULINK model is a non-linear DAE system and contains 282 equations, 272 unknown variables, and 11 known variables. Of the equations, 8 are differential and the rest are algebraic. The differentiated variables represent physical quantities such as pressures, temperatures, and rotational speeds.

	MSO Sets	MSO Classes	Residual Generators
D	4	4	46
I	1	1	5
DI	4	4	46
SD	4	4	46
SI	23	20	58
SDI	120	72	1636
Potential	598	210	135772

Table 2: Results for Diesel Engine.

In total, 598 MSO sets could be found in the engine model. The MSO sets could be arranged into 210 MSO classes. Theoretically, the total number of potential residual generators that can be constructed from an MSO set is equal to the total number of equations in the MSO set. In this case, 135772 different residual generators could be theoretically constructed from the 598 MSO sets.

Results

The total number of residual generators found and how many of the MSO sets and MSO classes that could be used, for each configuration of the method, is shown in Table 2 and Figure 3. The columns to the left and in the middle of Table 2 shows in how many of the MSO sets and MSO classes at least one residual generator could be found. The column to the right shows the total number of residual generators that could be found for each configuration of the method. It is obvious that a very small fraction of the potential residual generators were found, about 1.2 %, and that only a small fraction of the MSO sets and MSO classes could be used, independent of configuration. The main reason for this is the complexity of the engine model. The model contains large algebraic and differential loops, including complex non-linear equations, which are impossible to solve analytically. Nevertheless, many more residual generators were found and more MSO sets could be used with configuration SDI, in comparison with any other configuration.

6.4 Hydraulic Braking System

The Scania auxiliary hydraulic braking system, called *retarder*, is used on heavy duty trucks for long continuous braking, for example to maintain constant speed down a slope. By using the retarder, braking discs can be saved for short time braking.

The model of the hydraulic braking system contains 49 equations, 44 unknown variables, and 9 known variables. It is a non-linear DAE system and contains 4 differential equations and 45 algebraic equations. The model contains 125 MSO sets, which can be arranged into 83 MSO classes. The total

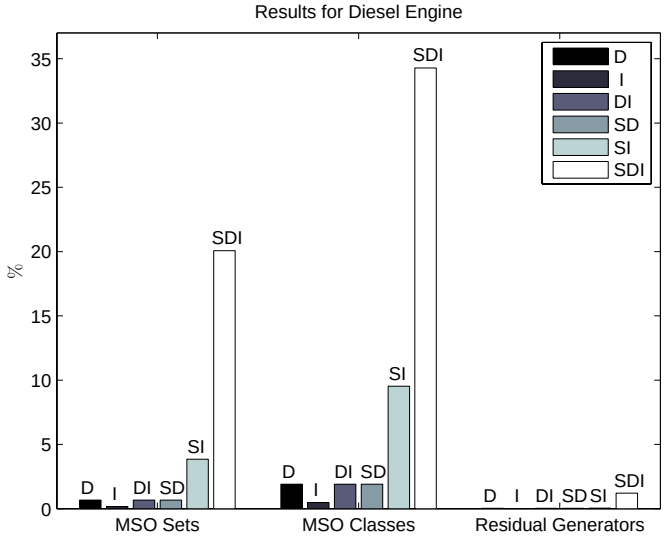


Figure 3: The bars to the left and in the middle shows the fractions of the total number of MSO sets and MSO classes in which a residual generator could be found with each configuration of the method. The bars to the right shows the fractions of the number of potential residual generators that could be found with each configuration of the method.

number of possible residual generators for the model of the hydraulic braking system is, theoretically, 4607.

Results

Table 3 and Figure 4 shows, for each configuration of the method, how many of the MSO sets and MSO classes that could be used and the total number of residual generators found for the model of the hydraulic braking system. As seen, a significantly larger fraction of the MSO sets and MSO classes could be used and more residual generators could be found with configuration SDI, in comparison with any other configuration.

6.5 Realization of a Residual Generator for the Diesel Engine

The purpose of this section is to briefly show how a residual generator for the diesel engine is constructed from a computation sequence obtained with the proposed method.

	MSO Sets	MSO Classes	Residual Generators
D	21	14	145
I	6	6	18
DI	21	14	147
SD	33	22	288
SI	29	29	71
SDI	65	44	1293
Potential	125	83	4607

Table 3: Results for Hydraulic Braking System

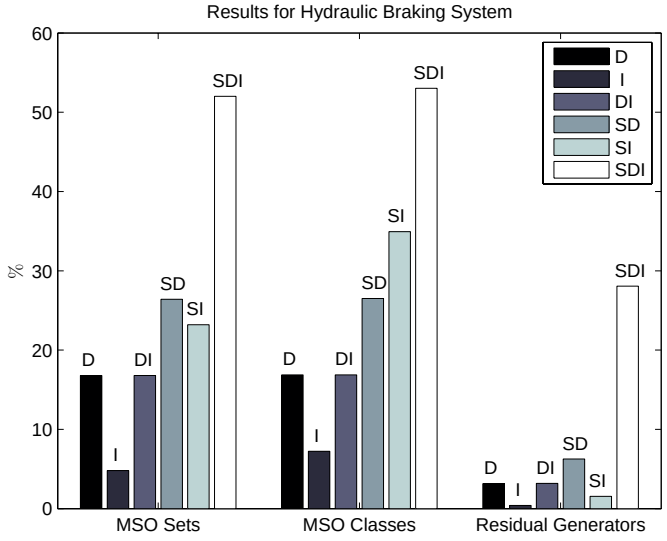


Figure 4: The bars to the left and in the middle shows the fractions of the total number of MSO sets and MSO classes in which a residual generator could be found with each configuration of the method. The bars to the right shows the fractions of the number of potential residual generators that could be found with each configuration of the method.

Properties of the Computation Sequence

The considered computation sequence originates from an MSO set containing in total 204 equations, 203 unknown variables, and 8 known variables. Thus, the computation sequence contains 203 equations and 203 unknown variables. In total 33 residual generators were found in the MSO class to which the MSO set belongs. All 33 residual generators were found with configuration SDI of FINDCOMPUTATIONSEQUENCE.

The computation sequence contains 102 elements. All elements but the last one contains one equation and one variable. The last element contains 102 equations and 102 variables and corresponds to a SCC of size 102. The structure of the 203 equations contained in the computation sequence, with respect to the 203 unknown variables, is shown in Figure 5. The SCCs of the structure, corresponding to the elements in the computation sequence, marked with squares in Figure 5. The residual equation used in the residual generator, i.e. the equation removed from the MSO set when the corresponding computation sequence was found, compares the measured and computed pressure in the intake manifold of the diesel engine.

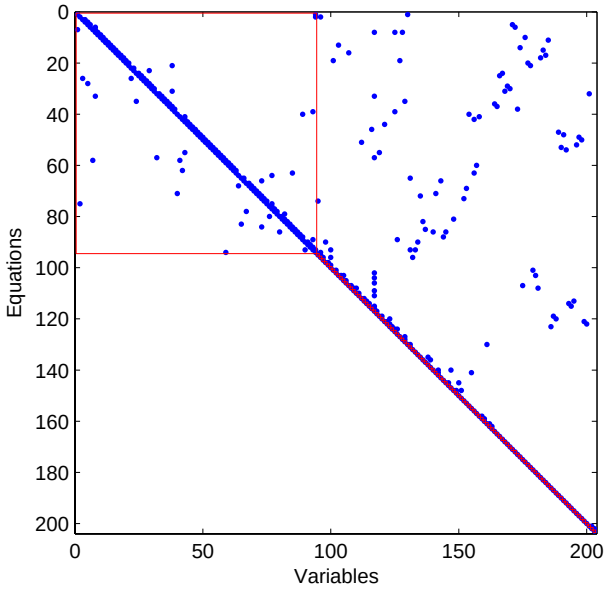


Figure 5: Structure of the 203 equations in the considered computation sequence, with respect to the 203 unknown variables. The SCCs of the structure, corresponding to the elements in the computation sequence, are marked with squares. The large SCC contains 102 equations.

Properties of the BLT semi-explicit DAE System

Since the computation sequence contains 102 elements, the BLT semi-explicit DAE system obtained from the computation sequence contains 102 blocks. The BLT semi-explicit DAE system has the following form

$$
\begin{aligned}
\mathbf{w}_1 &= \mathbf{h}_1\left(\mathbf{y}\right)\\
\mathbf{w}_2 &= \mathbf{h}_2\left(\mathbf{w}_1, \mathbf{y}\right)\\
&\vdots\\
\mathbf{w}_{64} &= \mathbf{h}_{64}\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{63}, \mathbf{y}\right)\\
\mathbf{w}_{65} &= \mathbf{h}_{65}\left(\dot{\mathbf{w}}_{64}, \mathbf{w}_1, \ldots, \mathbf{w}_{64}, \mathbf{y}\right)\\
\mathbf{w}_{66} &= \mathbf{h}_{66}\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{65}, \mathbf{y}\right)\\
&\vdots\\
\mathbf{w}_{76} &= \mathbf{h}_{76}\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{75}, \mathbf{y}\right)\\
\mathbf{w}_{77} &= \mathbf{h}_{77}\left(\dot{\mathbf{w}}_{76}, \mathbf{w}_1, \ldots, \mathbf{w}_{76}, \mathbf{y}\right)\\
\mathbf{w}_{78} &= \mathbf{h}_{78}\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{77}, \mathbf{y}\right)\\
&\vdots\\
\mathbf{w}_{100} &= \mathbf{h}_{100}\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{99}, \mathbf{y}\right)\\
\dot{\mathbf{z}}_{101} &= \mathbf{g}_{101}\left(\mathbf{w}_1, \ldots, \mathbf{w}_{101}, \mathbf{y}\right)\\
\mathbf{w}^1_{101} &= \mathbf{h}^1_{101}\left(\mathbf{z}_{101}, \mathbf{w}_1, \ldots, \mathbf{w}_{100}, \mathbf{y}\right)\\
\mathbf{w}^2_{101} &= \mathbf{h}^2_{101}\left(\mathbf{z}_{101}, \mathbf{w}_1, \ldots, \mathbf{w}_{100}, \mathbf{w}^1_{101}, \mathbf{y}\right)\\
&\vdots\\
\mathbf{w}^{99}_{101} &= \mathbf{h}^{99}_{101}\left(\mathbf{z}_{101}, \mathbf{w}_1, \ldots, \mathbf{w}_{100}, \mathbf{w}^1_{101}, \ldots, \mathbf{w}^{98}_{101}, \mathbf{y}\right),
\end{aligned}
\tag{25}
$$

where $\mathbf{w}_{101} = \left(\mathbf{w}^1_{101}, \mathbf{w}^2_{101}, \ldots, \mathbf{w}^{99}_{101}\right)$ and $\mathbf{z}_{101}$ is of dimension three and all $\mathbf{w}_i$, $\mathbf{w}^j_i$ of dimension one. The largest block, 101 in (25), is a semi-explicit DAE of index one with three differential equations with variables $\mathbf{z}_{101}$ and 99 algebraic equations with variables $\mathbf{w}^1_{101}, \ldots, \mathbf{w}^{99}_{101}$, corresponding to a differential loop and a SCC of size 102. Since the block is a semi-explicit DAE of index one, integral causality is used in this block. In two of the blocks, 66 and 77 in (25), derivative causality is used. The remaining blocks, 1 - 65, 67 - 76, and 78 - 100 correspond to algebraic equations. In total, the BLT semi-explicit DAE system contains five differential equations and 198 algebraic equations.

Implementation Issues

The residual generator, i.e. the obtained BLT semi-explicit DAE system and the residual equation, was implemented in MATLAB. To compute the values of the

unknown variables, the approach described in Section 3.1 was used. To solve the resulting explicit ODE, Euler forward with fixed step-size was utilized. All state variables in the residual generators represent physical quantities, hence initial conditions were easy to obtain from the available measurements.

Approximate Differentiation In the two blocks where derivative causality is used, 66 and 77 in (25), derivatives of variables computed in previous blocks had to be computed. By propagating the two differentiated variables through equations in earlier blocks of the obtained BLT semi-explicit DAE system, the differentiated variables could be expressed as derivatives of known variables only, see Section 3.2. The known variables that had to be differentiated were measurements of the pressure in the exhaust manifold, and the rotational speed of the turbo turbine.

The differentiation tool, i.e. the method for differentiation of known variables, used in this case study was a sliding-window least square polynomial fit approach. By finding a linear approximation, in a least square sense, to a set of consecutive measurements, referred to as a window, an approximation of the first-order derivative of the measured signal in the window can be obtained as the slope of the linear approximation, see e.g. [Barford et al., 1999]. This approach was used since it is simple and straight-forward to implement, and gave good results. An implementation was done in MATLAB, a window-size of 40 measurements, 20 past and 20 future, was used.

Results

Real measurements of the known variables in the engine model were collected by driving a truck on the road. Two sets of measurements were collected, one with a fault-free engine and one with an implemented fault. The implemented fault was a constant bias in the sensor measuring the pressure in the intake manifold of the diesel engine.

The residual generator was run off-board by using the collected measurements. The residual was then low-pass filtered to remove some measurement noise and finally scaled. In Figure 6, the resulting residual is shown. During the first 100 seconds, the measurements are fault-free. The remaining time, the measurements contain the implemented bias fault. It is obvious that the residual can be used to detect the injected fault.

7 Conclusions

We have in Section 1 concluded that it is important that there is a large selection of different candidate residual generators to choose between when designing diagnosis systems. In this spirit we have in this paper presented a method for deriving residual generators with the key property that it is able to find a large number of different residual generators. This property is firstly

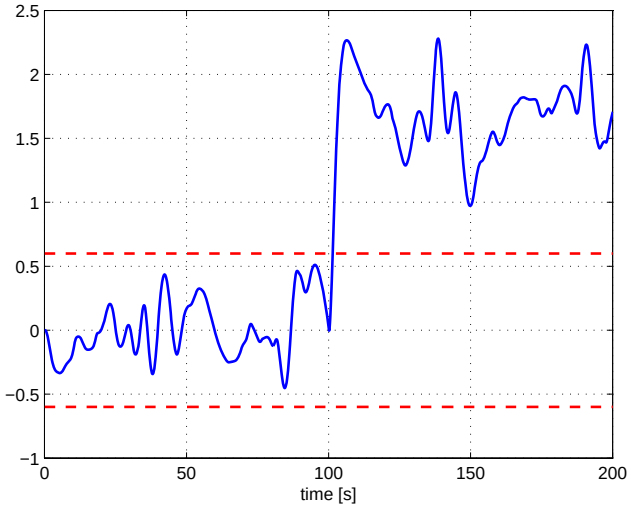


Figure 6: The residual obtained from the constructed residual generator. No fault is present the first 100 seconds. During the remaining 100 seconds, there is a bias fault in the sensor measuring the pressure in the intake manifold. The dashed lines suggests how thresholds could be chosen in order to detect the fault.

due to the fact that the method belongs to a class of methods that we refer to as sequential residual generation. This class of methods has in earlier works been shown to be powerful for real non-linear systems [Dustegor et al., 2004], [Izadi-Zamanabadi, 2002], [Cocquempot et al., 1998], [Svärd and Wassén, 2006], [Hansen and Molin, 2006]. Secondly, which is the key contribution of the paper, we have extended these earlier methods by handling mixed causality and also, in a systematic manner, equation sets containing differential and algebraic loops.

The method has been presented as an algorithm utilizing an assumed given toolbox of e.g. algebraic equation solvers. We have proven, in Theorem 1, that the algorithm really finds residual generators and, in Theorems 3 and 4, that the residual generators, or rather sequential residual generators, found are *proper*. Properness guarantees that the residual generator is not containing unnecessary computations, and that computations are performed from as small equation sets as possible. We have also proven, in Theorem 2, that proper sequential residual generators are always found within MSO sets. This fact has been utilized in the algorithm since there is no need to look for sequential residual generators in other equation sets than MSO sets. Furthermore, this theorem provides a link between structural and analytical methods without the use of any assumptions of generic equations, such as in e.g. [Krysander et al., 2008].

In the empirical study in Section 6, models of real automotive systems have been considered, and we have compared our method handling mixed causality and differential and algebraic loops, with the alternatives using solely differential or integral causality, or only handling scalar equations. It is evident that our method outperforms the other alternatives.

Acknowledgment

This work was sponsored by Scania CV AB and VINNOVA (Swedish Governmental Agency for Innovation Systems).

References

[Ascher and Petzold, 1998] Ascher, U. M. and Petzold, L. M. (1998). *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Siam.

[Asratian et al., 1998] Asratian, A. S., Denley, T. M. J., and Häggkvist, R. (1998). *Bipartite Graphs and their Applications*. Cambridge University Press. ISBN 0-521-59345-X.

[Barford et al., 1999] Barford, L., Manders, E., Biswas, G., Mosterman, P., Ram, V., and Barnett, J. (1999). Derivative estimation for diagnosis. Technical report, HP Labs Technical Reports.

[Blanke et al., 2003] Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2003). *Diagnosis and Fault-Tolerant Control*. Springer.

[Brenan et al., 1989] Brenan, K. E., Campbell, S. L., and Petzold, L. R. (1989). *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. Siam.

[Brockett, 1970] Brockett, R. (1970). *Finite-dimensional linear systems*. Wiley, New York.

[Cassar and Staroswiecki, 1997] Cassar, J. and Staroswiecki, M. (1997). A structural approach for the design of failure detection and identification systems. In *Proceedings of IFAC Control Ind. Syst.*, Belfort, France.

[Cellier and Elmqvist, 1993] Cellier, F. and Elmqvist, H. (1993). Automated formula manipulation supports object-oriented continuous-system modeling. *Control Systems Magazine, IEEE*, 13(2):28–38.

[Cellier and Kofman, 2006] Cellier, F. and Kofman, E. (2006). *Continuous System Simulation*. Springer.

[Cocquempot et al., 1998] Cocquempot, V., Izadi-Zamanabadi, R., Staroswiecki, M., and Blanke, M. (1998). Residual generation for the ship benchmark using structural approach. In *Proceedings of the UKACC International Conference on Control '98*, pages 1480–1485.

[De Persis and Isidori, 2001] De Persis, C. and Isidori, A. (2001). A geometric approach to nonlinear fault detection and isolation. *IEEE Transactions on Automatic Control*, 46:853–865.

[Dulmage and Mendelsohn, 1958] Dulmage, A. and Mendelsohn, N. (1958). Coverings of bi-partite graphs. *Canadian Journal of Mathematics*, 10:517–534.

[Dustegor et al., 2004] Dustegor, D., Cocquempot, V., and Staroswiecki, M. (2004). Structural analysis for residual generation: Towards implementation. In *Proceedings of the 2004 IEEE Inter. Conf. on Control App.*, pages 1217–1222.

[Frisk et al., 2006] Frisk, E., Krysander, M., Nyberg, M., and Åslund, J. (2006). A toolbox for design of diagnosis systems. In *Proceedings of IFAC Safeprocess'06*, Beijing, China.

[Fritzon, 2004] Fritzon, P. (2004). *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. IEEE Press.

[Hairer and Wanner, 2002] Hairer, E. and Wanner, G. (2002). *Solving Ordinary Equations II - Stiff and Differential-Algebraic Problems*. Springer.

[Hansen and Molin, 2006] Hansen, J. and Molin, J. (2006). Design and evaluation of an automatically generated diagnosis system. Master's thesis, Linköpings Universitet, SE-581 83 Linköping.

[Izadi-Zamanabadi, 2002] Izadi-Zamanabadi, R. (2002). Structural analysis approach to fault fiagnosis with application to fixed-wing aircraft motion. In *Proceedings of the 2002 American Control Conference*, volume 5, pages 3949–3954.

[Katsillis and Chantler, 1997] Katsillis, G. and Chantler, M. (1997). Can dependency-based diagnosis cope with simultaneous equations? In *Proceedings of the 8th Inter. Workshop on Princ. of Diagnosis, DX'97*, Le Mont-Saint-Michel, France.

[Khalil, 2002] Khalil, H. K. (2002). *Nonlinear Systems*. Prentice Hall.

[Kron, 1963] Kron, G. (1963). *Diakoptics*. Macdonald, London.

[Krysander et al., 2008] Krysander, M., Åslund, J., and Nyberg, M. (2008). An efficient algorithm for finding minimal over-constrained sub-systems for model-based diagnosis. *IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 38(1).

[Kunkel and Mehrmann, 2006] Kunkel, P. and Mehrmann, V. (2006). *Differential-Algebraic Equations - Analysis and Numerical Solution*. European Mathematical Society.

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

[Murota, 1987] Murota, K. (1987). *System Analysis by Graphs and Matroids*. Springer-Verlag Berlin Heidelberg.

[Nyberg, 1999] Nyberg, M. (1999). Automatic design of diagnosis systems with application to an automotive engine. *Control Engineering Practice*, 87(8):993–1005.

[Nyberg and Krysander, 2008] Nyberg, M. and Krysander, M. (2008). Statistical properties and design criterions for AI-based fault isolation. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea.

[Ortega and Rheinboldt, 2000] Ortega, J. and Rheinboldt, W. (2000). *Iterative Solution of Nonlinear Equations in Several Variables*. SIAM Classics.

[Ploix et al., 2005] Ploix, S., Desinde, M., and Touaf, S. (2005). Automatic design of detection tests in complex dynamic systems. In *Proceedings of 16th IFAC World Congress*, Prague, Czech Republic.

[Pulido et al., 2007] Pulido, B., Alonso, C., Bregón, A., Puig, V., and Escobet, T. (2007). Analyzing the influence of temporal constraints in possible conflicts calculation for model-based diagnosis. In *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX-07)*, Nashville, TN, USA.

[Pulido and Alonso-González, 2004] Pulido, B. and Alonso-González, C. (2004). Possible conflicts: a compilation technique for consistency-based diagnosis. *"IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics", Special Issue on Diagnosis of Complex Systems*, 34(5):2192–2206.

[Pulido et al., 2008] Pulido, B., Bregón, A., and Alonso, C. (2008). Combining state estimation and simulation in consistency-based diagnosis using possible conflicts. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, pages 339–346, Blue Mountains, NSW, Australia.

[Staroswiecki, 2002] Staroswiecki, M. (2002). *Fault Diagnosis and Fault Tolerant Control*, chapter Structural Analysis for Fault Detection and Isolation and for Fault Tolerant Control. Encyclopedia of Life Support Systems, Eolss Publishers, Oxford, UK.

[Staroswiecki and Declerck, 1989] Staroswiecki, M. and Declerck, P. (1989). Analytical redundancy in non-linear interconnected systems by means of structural analysis. In *Proceedings of IFAC AIPAC'89*, pages 51–55, Nancy, France.

[Steward, 1962] Steward, D. V. (1962). On an approach to techniques for the analysis of the structure of large systems of equations. *SIAM Review*, 4(2):321–342.

[Steward, 1965] Steward, D. V. (1965). Partitioning and tearing systems of equations. *SIAM Journal on Numerical Analysis*, 2(2):345–365.

[Svärd and Nyberg, 2008] Svärd, C. and Nyberg, M. (2008). A mixed causality approach to residual generation utilizing equation system solvers and differential-algebraic equation theory. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, Blue Mountains, Australia.

[Svärd and Wassén, 2006] Svärd, C. and Wassén, H. (2006). Development of methods for automatic design of residual generators. Master's thesis, Linköpings Universitet, SE-581 83 Linköping.

[Tarjan, 1972] Tarjan, R. (1972). Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160.

[Travé-Massuyès et al., 2006] Travé-Massuyès, L., Escobet, T., and Olive, X. (2006). Diagnosability analysis based on component-supported analytical redundancy. *IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 36(6):1146–1160.

[United Nations, 2008] United Nations (2008). Regulation no. 49: Uniform provisions concerning the measures to be taken against the emission of gaseous and particulate pollutants from compressionignition engines for use in vehicles, and the emission of gaseous pollutants from positive-ignition engines fuelled with natural gas or liquefied petroleum gas for use in vehicles. ECE-R49.

[Wahlström, 2006] Wahlström, J. (2006). Control of EGR and VGT for emission control and pumping work minimization in diesel engines. Technical report, Linköpings Universitet. LiU-TEK-LIC-2006:52, Thesis No. 1271.

[Wei and Li, 2006] Wei, T. and Li, M. (2006). High order numerical derivatives for one-dimensional scattered noisy data. *Applied Mathematics and Computation*, 175:1744–1759.

A Proofs of Theorems and Lemmas

Proof of Lemma 3. Consider an element $(V_i, E_i) \in \mathcal{C}$, and let $\bar{E}'_i$ denote the set of equations obtained when $\mathcal{T}$ is called with arguments V_i and E_i. It then holds that $\bar{E}' = \bar{E}'_1 \cup \bar{E}'_2 \cup \ldots \cup \bar{E}'_k$. Given $\mathbf{y}$, let $\tilde{\mathbf{x}}$ be an arbitrary solution to $\bar{E}'$, i.e. a trajectory fulfilling every equation $e_i \in \bar{E}'$. Trivially, $\tilde{\mathbf{x}}$ also is a solution to the equations in every $\bar{E}'_i$, since $\bar{E}'_i \subseteq \bar{E}'$. Assumption 1 then implies that $\tilde{\mathbf{x}}$ is a unique solution and also a solution to every E_i, and hence to $E_1 \cup E_2 \cup \ldots \cup E_k$. By taking an arbitrary solution to $E_1 \cup E_2 \cup \ldots \cup E_k$ and applying the same arguments as above, it can be shown that this solution is unique and also satisfies $\bar{E}'$, which completes the proof. □

Proof of Theorem 1. Consider the model $M(E, X, Y)$ and assume that $\tilde{\mathbf{y}} \in \mathcal{O}(M)$. Due to the definition of $\mathcal{O}(M)$ in (4), we know that given $\tilde{\mathbf{y}}$ there exists at least one trajectory of the variables in X that satisfies the equations in E. Since describing $E_1 \cup E_2 \cup \ldots \cup E_k \subseteq E$, it holds that the trajectory $\tilde{\mathbf{y}}$ also belongs to the observation set of the sub-model of $M(E, X, Y)$ given by $E_1 \cup E_2 \cup \ldots \cup E_k$, i.e. the equation set contained in the computation sequence $\mathcal{C}$. Hence, given $\tilde{\mathbf{y}}$, there exists a trajectory $\tilde{\mathbf{x}}$ of the variables in $\mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$ that satisfies $E_1 \cup E_2 \cup \ldots \cup E_k$. By Lemma 3 we know that $\tilde{\mathbf{x}}$ is a unique solution that also satisfies the equations of the BLT semi-explicit DAE system obtained by sequentially applying the tool $\mathcal{T}$ to the computation sequence $\mathcal{C}$.

As said in Section 3.1, a BLT semi-explicit DAE system can be transformed to an explicit ODE, with the exception that the ODE will contains derivatives of known variables. Furthermore, after the discussion in Section 3.2, that an explicit ODE always can be solved if initial conditions are available. From this it follows that given $\tilde{\mathbf{y}}$, consistent initial conditions of the states in the BLT semi-explicit DAE system, i.e. $\mathbf{z}_i$ in (7), and the ability the compute all needed derivatives, the trajectory $\tilde{\mathbf{x}}$ can be computed from the BLT semi-explicit DAE system. Since $e_i \in E \setminus E_1 \cup E_2 \cup \ldots \cup E_k$ and $\mathrm{var}_X(e_i) \subseteq X' \subseteq \mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$, the trajectory $\tilde{\mathbf{x}}$ will also satisfy e_i. We then have that $f_i(\dot{\tilde{\mathbf{x}}}, \tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = 0$. Hence, with $r = f_i(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y})$, $\tilde{\mathbf{y}} \in \mathcal{O}(M)$ implies $r = 0$ and we can use r as residual. Thus the BLT semi-explicit DAE system obtained from the computation sequence $\mathcal{C}$ with $\mathcal{T}$, together with e_i is a residual generator for $M(E, X, Y)$. □

Some important properties of a computation sequence, used in sub-sequential proofs, is given by the following lemma.

Lemma 4. *Let $\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots (V_k, E_k))$ be a computation sequence for the variables X' with the AE tool $\mathcal{T}$, then $\{\mathrm{unDiff}(V_i)\}$ is pairwise disjoint and $\mathrm{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k) = \mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$.*

Proof. From Definition 3, we have that a system in BLT semi-explicit DAE form can be obtained by sequentially calling $\mathcal{T}$ with arguments V_i and E_i for every $(V_i, E_i) \in \mathcal{C}$. From this fact, it follows that each variable $x_j \in \mathrm{unDiff}(V_i)$ is

present in some vector $\mathbf{z}_k$ or $\mathbf{w}_l$ in the obtained BLT semi-explicit DAE system. Since the set of all vectors of known variables in a BLT semi-explicit DAE system by Definition 2 is pairwise disjoint, it follows that $\{\text{unDiff}(V_i)\}$ is pairwise disjoint and we have shown the first claim. For the second claim, we start by noting that $V_i \subseteq \text{var}_X(E_i) \cup \text{var}_D(E_i)$ due to Definition 3. Since a system in BLT semi-explicit DAE form can be obtained from $\mathcal{C}$ and, according to Lemma 3, the solution sets of $E_1 \cup E_2 \cup \ldots \cup E_k$ and the BLT semi-explicit DAE system, with respect to $V_1 \cup V_2 \cup \ldots \cup V_k$, are equal and unique, it holds that each unknown variable in $E_1 \cup E_2 \cup \ldots \cup E_k$, differentiated or undifferentiated, must be present in some V_i. From this fact and by the definitions of the operators unDiff() and $\text{var}_X()$, it must also hold that

$$\text{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k) = \text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k).$$

□

For the next proof, we need some additional graph theoretical concepts, see e.g. [Asratian et al., 1998], [Murota, 1987], therefore consider the bi-partite graph $G = (E, X, A)$ describing the structure of E with respect to X, see Section 2.2. A *path* on the graph G is a sequence of distinct vertices $v_1, v_2, \ldots, v_n$ such that $(v_i, v_{i+1}) \in A$ and $v_i \in E \cup X$. An *alternating path* is a path in which the edges belong alternatively to a matching and not to the matching. A vertex is said to be *free*, if it is not an endpoint of an edge in a matching.

Proof of Theorem 2. In this proof we will use a characterization of an MSO set given in [Krysander et al., 2008], saying that an equation set E is an MSO set if and only if E is a proper structurally over-determined (PSO) set and E contains one redundant equation. Furthermore, an equation set E is a PSO set if $E = E^+$, where E^+ is the structurally over-determined part obtained from the DM-decomposition, recall Section 2.3, or equivalently the equations $e \in E$ such that, for any maximal matching, there exists an alternating path between at least one free equation and e.

Returning to our case, we must show that $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ is a PSO set and contains one redundant equation, with respect to the variables $\text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$. We begin with the second property, i.e. that $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ contains a redundant equation. Since $\mathcal{S} = (\mathcal{T}(\mathcal{C}), e_i)$ is a proper sequential residual generator, it follows from Definition 7 that $\mathcal{C}$ is a minimal and irreducible computation sequence for $\text{var}_X(e_i)$ with $\mathcal{T}$. If we let

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)), \tag{26}$$

we have from Definition 3 that a system in BLT semi-explicit DAE form is obtained by sequentially calling the AE tool $\mathcal{T}$ with arguments V_i and E_i for every $(V_i, E_i) \in \mathcal{C}$. This and Assumption 2, implies that $|V_i| = |E_i|$ for every $(V_i, E_i) \in \mathcal{C}$ and hence $\sum_{i=1}^{k} |V_i| = \sum_{i=1}^{k} |E_i|$. By the definition of the operator unDiff() in (11), we can conclude that $|V_i| = |\text{unDiff}(V_i)|$ and therefore

it also holds that $\sum_{i=1}^{k} |\text{unDiff}(V_i)| = \sum_{i=1}^{k} |E_i|$. By Lemma 4 we have that $\{\text{unDiff}(V_i)\}$ is pairwise disjoint which implies that

$$\begin{aligned}\sum_{i=1}^{k} |\text{unDiff}(V_i)| &= |\text{unDiff}(V_1) \cup \text{unDiff}(V_2) \cup \ldots \cup \text{unDiff}(V_k)| \\ &= |\text{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k)|.\end{aligned}$$

Definition 3 states that also $\{E_i\}$ is pairwise disjoint and therefore

$$|E_1 \cup E_2 \cup \ldots \cup E_k| = \sum_{i=1}^{k} |E_i|.$$

Thus, it holds that

$$|\text{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k)| = |E_1 \cup E_2 \cup \ldots \cup E_k|.$$

By Lemma 4, we have that $\text{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k) = \text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$ and therefore it also holds that

$$|E_1 \cup E_2 \cup \ldots \cup E_k| = |\text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)|,$$

i.e. $E_1 \cup E_2 \cup \ldots \cup E_k$ contains as many equations as unknowns. Since $\mathcal{C}$ is a computation sequence for $\text{var}_X(e_i)$ with $\mathcal{T}$, we have from Definition 3 that $\text{var}_X(e_i) \subseteq \text{unDiff}(V_1 \cup V_2 \cup \ldots \cup V_k) = \text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$, where the last equality follows from Lemma 4, implying that adding e_i to $E_1 \cup E_2 \cup \ldots \cup E_k$ will not introduce any new unknown variables, i.e. e_i is redundant. Hence, the equation set $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ contains one more equation than unknown variables, since

$$\begin{aligned}|E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i| &= |E_1 \cup E_2 \cup \ldots \cup E_k| + |e_i| \\ &= |\text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)| + 1.\end{aligned}$$

We will now show that $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ is a PSO set with respect to $\text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i)$. To show this, we must show that for any maximum matching on the bi-partite graph describing the structure of $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$, with respect to $\text{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i)$, there exists an alternating path between a free equation and every equation in $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$. We start by constructing a maximum matching and finding a free equation. Consider the computation sequence $\mathcal{C}$ described by (26) and recall that $\mathcal{C}$, given by (26), is a minimal and irreducible computation sequence for $\text{var}_X(e_i)$ with $\mathcal{T}$. The irreducibility of $\mathcal{C}$ implies that for each element $(V_i, E_i) \in \mathcal{C}$, it holds that the structure of E_i with respect to $\text{unDiff}(V_i)$ corresponds to a SCC. To see this, assume that (V_i, E_i) not corresponds to a SCC. This implies that it is possible to partition V_i and E_i into $V_i = V_{i1} \cup V_{i2} \cup \ldots \cup V_{is}$ and $E_i = E_{i1} \cup E_{i2} \cup \ldots \cup E_{is}$ so that

$$\mathcal{C}' = ((V_1, E_1), \ldots, (V_{i1}, E_{i1}), \ldots, (V_{is}, E_{is}), \ldots, (V_k, E_k)),$$

is also a computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$, due to Assumption 3. This contradicts the irreducibility of $\mathcal{C}$ and hence (V_i, E_i) must be a SCC. From this property it follows, by the definition of a SCC, that there exists a maximum matching Γ_i on the bi-partite graph the structure of E_i with respect to unDiff (V_i). This implies that a maximum matching, let it be denoted Γ, in the structure of $E_1 \cup E_2 \cup \ldots \cup E_k$ with respect to unDiff $(V_1 \cup V_2 \cup \ldots \cup V_k)$ can be constructed as $\Gamma = \bigcup_i^k \Gamma_i$, see e.g. [Murota, 1987]. By Lemma 4, we have that unDiff $(V_1 \cup V_2 \cup \ldots \cup V_k) = \mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$ and therefore Γ is also a maximum matching in the structure of $E_1 \cup E_2 \cup \ldots \cup E_k$ with respect to $\mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k)$. In the first part of this proof, we concluded that the equation e_i is redundant and therefore Γ is also a maximum matching on the structure of $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ with respect to $\mathrm{var}_X(E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i)$ and e_i is a free equation, since it is not contained in Γ.

Since it trivially exists a path between e_i and e_i, it is sufficient to show that there exists an alternating path between the free equation e_i and every equation in $E_1 \cup E_2 \cup \ldots \cup E_k$. Due to the fact that each $(V_i, E_i) \in \mathcal{C}$ corresponds to a SCC, there exists an alternating path between any two vertices, i.e. equations or variables, in the bi-partite graph describing the structure of E_i with respect to unDiff (V_i), see e.g. [Asratian et al., 1998]. Moreover, the minimality of $\mathcal{C}$ implies that for $(V_k, E_k) \in \mathcal{C}$ there exists at least one variable $x_m \in$ unDiff (V_k) such that $x_m \in \mathrm{var}_X(e_i)$, since otherwise $\mathcal{C}' = \mathcal{C} \setminus (V_k, E_k)$ is a computation sequence for $\mathrm{var}_X(e_i)$ and $\mathcal{C}$ is not minimal. With the same argument, we have that for $(V_i, E_i) \in \mathcal{C}$, $i = 1, 2, \ldots, k-1$, there exists at least one variable $x_m \in$ unDiff (V_i) such that either $x_m \in \mathrm{var}_X(e_i)$, or else $x_m \in \mathrm{var}_X(E_j)$ where $(V_j, E_j) \in \mathcal{C}$ and $j \in \{i+1, i+2, \ldots, k\}$. This means that there exists an alternating path between at least one variable in each $(V_i, E_i) \in \mathcal{C}$ to e_i, either directly or via one or several other $(V_j, E_j) \in \mathcal{C}$. Thus, there exists an alternating path between e_i and every equation in $E_1 \cup E_2 \cup \ldots \cup E_k$. We have by this shown that $E_1 \cup E_2 \cup \ldots \cup E_k \cup e_i$ is a PSO set. □

The proof of Theorem 3 is based on the following lemma.

Lemma 5. *Let $\bar{E} \subseteq E$ be an MSO set, $\mathcal{T}$ an AE tool, $X' = \mathrm{var}_X(\bar{E})$, and $E' = \bar{E} \setminus e_i$, where $e_i \in \bar{E}$. A minimal and irreducible computation sequence*

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)),$$

for X' with $\mathcal{T}$, where $E_i \subseteq \bar{E}$, is also a minimal and irreducible computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$.

Proof. Assume that $\mathcal{C}$ is a minimal and irreducible computation sequence for X' with $\mathcal{T}$. First of all, since $e_i \in \bar{E}$ and $X' = \mathrm{var}_X(\bar{E})$ it trivially holds that $\mathrm{var}_X(e_i) \subseteq X'$ and hence $\mathcal{C}$ is a computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$. As well, it directly follows from Definition 6 that $\mathcal{C}$ is an irreducible computation sequence for any subset of X', in particular $\mathrm{var}_X(e_i)$. To show that $\mathcal{C}$ also is a minimal computation sequence for $\mathrm{var}_X(e_i)$, assume that there exists a computation sequence $\mathcal{C}' \subset \mathcal{C}$ for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$. Let $\bar{E}'$ and $\bar{X}' = \mathrm{var}_X(\bar{E}')$ denote

the equations and variables, contained in the elements of $\mathcal{C}'$ and note that since $\mathcal{C}' \subset \mathcal{C}$, it holds that $\bar{E}' \subset \bar{E}$. By the argumentation in the proof to Theorem 2, we can conclude that $|\bar{E}'| = |\bar{X}'|$, i.e. $\bar{E}$ contains as many equations as unknowns. Since $\mathcal{C}'$ is a computation sequence for $\mathrm{var}_X(e_i)$, it must hold that $\mathrm{var}_X(e_i) \subseteq \bar{X}'$. This means that $\bar{E}' \cup e_i$ is a structurally over-determined set of equations with respect to $\bar{X}'$, which shows that there exists a proper structurally over-determined subset of $\bar{E}$. This contradicts the fact that $\bar{E}$ is an MSO set, and hence there can not exist a computation sequence $\mathcal{C}' \subset \mathcal{C}$ for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$. Thus, $\mathcal{C}$ is a minimal computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$. □

Proof of Theorem 3. Consider the model $M(E, X, Y)$ and let $(\mathcal{T}(\mathcal{C}), e_i) \in R$. Due to line 9 in FINDRESIDUALGENERATORS, we can conclude that $\mathcal{C}$ is non-empty. Let

$$\mathcal{C} = ((V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)),$$

where $E_i \subseteq E'$, be the minimal and irreducible computation sequence for X' with $\mathcal{T}$, returned by the function FINDCOMPUATATIONSEQUENCE on line 8. Due to lines 3-7, we have that $E' = \bar{E} \setminus e_i$ and $X' = \mathrm{var}_X(\bar{E})$, where $\bar{E} \subseteq E$ is an MSO set and $e_i \in \bar{E} \setminus E'$. Lemma 5 then implies that $\mathcal{C}$ also is a minimal and irreducible computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$. Now note that since $e_i \in \bar{E} \setminus E'$ and it holds that $\bar{E} \subseteq E$, we have that $e_i \in E \setminus E'$. Trivially, since $X' = \mathrm{var}_X(\bar{E})$ and $X' \subseteq X$ it also holds that $\mathrm{var}_X(e_i) \subseteq X' \subseteq X$. Thus the computation sequence $\mathcal{C}$ for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$ and the equation e_i fulfills the prerequisites of Theorem 1. Hence, since all initial conditions are known and all needed derivatives can be computed, we can by Theorem 1 conclude that the BLT semi-explicit DAE system obtained from $\mathcal{C}$ with $\mathcal{T}$ and e_i is a residual generator for $M(E, X, Y)$. Thus, $(\mathcal{T}(\mathcal{C}), e_i)$ is a sequential residual generator. Since, in fact, $\mathcal{C}$ is a minimal and irreducible computation sequence for $\mathrm{var}_X(e_i)$ with $\mathcal{T}$, $(\mathcal{T}(\mathcal{C}), e_i)$ is a proper sequential residual generator. □

Proof of Theorem 4. On line 3 in FINDCOMPUTATIONSEQUENCE the SCCs of the structure of E' with respect to X' are computed. If we assume that the structure contains s SCCs, the ordered set returned by the function FINDALLSCC can be written as

$$S = ((E_1, X_1), (E_2, X_2), \ldots, (E_s, X_s)), \tag{27}$$

where each element $(E_i, X_i) \in S$ corresponds to a SCC of the structure of E' with respect to X'. Note that since E' is just-determined with respect X', the SCCs of the structure of E' with respect X' are unique, see Section 2.3. As said in Section 5.3, we assume that the SCCs in S are ordered according to Figure 1. Note that this ordering implies the important property

$$\mathrm{var}_X(E_i) \cap \{X_{i+1} \cup X_{i+2} \cup \ldots \cup X_s\} = \emptyset, \tag{28}$$

for $i = 1, 2, \ldots, s-1$. On lines 6-8, the variables in X_i are partitioned into differentiated variables Z_i and un-differentiated variable W_i, i.e. $X_i = \mathrm{unDiff}(Z_i) \cup$

W_i, where Z_i contains variables that appear as differentiated in some equation in E_i. On lines 12-14, a corresponding partitioning of the equations in E_i into $E_i = E_{Z_i} \cup E_{W_i}$ is done, where E_{Z_i} are equations that contain any of the differentiated variables Z_i, and E_{W_i} are equations that do not contain any of the differentiated variables Z_i, but may contain variables from unDiff (Z_i). Now note that, due to the assumptions regarding the model in Section 2, each equation in E_{Z_i} contains only one differentiated , which furthermore only is present in that equation. This means first of all that E_{Z_i} is just-determined with respect to the variables in Z_i, and second that the structure of E_{Z_i} with respect to Z_i only contains SCCs of size one. On line 14, these SCCs are computed. Assuming that the structure contains s_i SCCs, the ordered set returned by FINDALLSCC on line 14 can be written as

$$S_{Z_i} = \left(\left(Z_i^1, E_{Z_i}^1 \right), \left(Z_i^2, E_{Z_i}^2 \right), \ldots, \left(Z_i^{s_i}, E_{Z_i}^{s_i} \right) \right). \tag{29}$$

Due to line 23, we know that the equation set E_{W_i} is just-determined with respect to W_i, and hence the structure of E_{W_i} with respect to W_i can be uniquely partitioned into SCCs. On line 24 these SCCs are computed and as above, the ordered set of SCCs can be written as

$$S_{W_i} = \left(\left(W_i^1, E_{W_i}^1 \right), \left(W_i^2, E_{W_i}^2 \right), \ldots, \left(W_i^{p_i}, E_{W_i}^{p_i} \right) \right). \tag{30}$$

Furthermore, as in the case with the set S in (27), the ordering of the SCCs in S_{W_i} implies that

$$\mathrm{var}_X(E_{W_i}^j) \cap \left\{ W_i^{j+1} \cup W_i^{j+2} \cup \ldots \cup W_i^{p_i} \right\} = \emptyset, \tag{31}$$

for $j = 1, 2, \ldots, p_i - 1$. From the discussion above, we have that a non-empty $\mathcal{C}$ returned by FINDCOMPUTATIONSEQUENCE have the form

$$\begin{aligned} \mathcal{C} = \Big(& \left(Z_1^1, E_{Z_1}^1 \right), \left(Z_1^2, E_{Z_1}^2 \right), \ldots, \left(Z_1^{s_1}, E_{Z_1}^{s_1} \right), \\ & \left(W_1^1, E_{W_1}^1 \right), \left(W_1^2, E_{W_1}^2 \right), \ldots, \left(W_1^{p_1}, E_{W_1}^{p_1} \right), \ldots, \\ & \left(Z_2^1, E_{Z_2}^1 \right), \left(Z_2^2, E_{Z_2}^2 \right), \ldots, \left(Z_2^{s_2}, E_{Z_2}^{s_2} \right), \\ & \left(W_2^1, E_{W_2}^1 \right), \left(W_2^2, E_{W_2}^2 \right), \ldots, \left(W_2^{p_2}, E_{W_2}^{p_2} \right), \ldots, \\ & \left(Z_s^1, E_{Z_s}^1 \right), \left(Z_s^2, E_{Z_s}^2 \right), \ldots, \left(Z_s^{s_s}, E_{Z_s}^{s_s} \right), \\ & \left(W_s^1, E_{W_s}^1 \right), \left(W_s^2, E_{W_s}^2 \right), \ldots, \left(W_s^{p_s}, E_{W_p}^{p_s} \right) \Big), \end{aligned} \tag{32}$$

where every $\left(Z_i^j, E_{Z_i}^j \right) \in \mathcal{C}$ and $\left(W_i^j, E_{W_i}^j \right) \in \mathcal{C}$ corresponds to a SCC.

We will now utilize Definition 3 to show that the the ordered set $\mathcal{C}$ in (32) is a computation sequence for X' with $\mathcal{T}$. First note that $Z_i^j \subseteq \mathrm{var}_D(E_{Z_i}^j)$ and

$W_i^j \subseteq \mathrm{var}_X(E_{W_i}^j)$. When the structure of a just-determined equation set with respect to a set of variables is decomposed into its SCCs, unique partitions of the equation and variable sets are also obtained, see for example [Dulmage and Mendelsohn, 1958] and Figure 1 for an illustration. From this fact it follows that every equation in E' is present in some E_i in (27) only once. When the equations in E_i are split into differential equations E_{Z_i} and algebraic equations E_{W_i} on line 13, it is guaranteed that $E_{Z_i} \cap E_{W_i} = \emptyset$. Moreover, again due to the fact that a decomposition into SCCs gives an unique partition of the equation and variable set, we have that every equation in E_{Z_i} is present in some equation set $E_{Z_i}^j$ in (29) only once and that every equation in E_{W_i} is present in some $E_{W_i}^j$ in (30) only once. Thus, we can conclude that each equation in E' is contained in only one equation set in $\mathcal{C}$, that is, all equation sets in $\mathcal{C}$ are disjoint. Hence, the ordered set $\mathcal{C}$ fulfills the prerequisites in Definition 3. According to conditions 1) and 2) in Definition 3, $\mathcal{C}$ is a computation sequence for X' with $\mathcal{T}$ if

$$X' \subseteq \bigcup_{i=1}^{s} \left(\bigcup_{j=1}^{s_i} \mathrm{unDiff}\left(Z_i^j\right) \cup \bigcup_{j=1}^{p_i} W_i^j \right) \tag{33}$$

and a system in BLT semi-explicit DAE form is obtained by sequentially calling the tool $\mathcal{T}$, with arguments Z_i^j and $E_{Z_i}^j$ for every element $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$, and with arguments W_i^j and $E_{W_i}^j$ for every element $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$.

We start by showing condition 1), i.e. (33). From the fact mentioned above that a decomposition of a structure into its SCCs also induces a partitioning of the corresponding equation and variable sets, it follows that every variable in X' is present in some X_i in (27). That is, we have that $X' = \bigcup_i^s X_i$. When the variables in X_i are split into differentiated variables Z_i and un-differentiated variables W_i, it holds that $X_i = \mathrm{unDiff}(Z_i) \cup W_i$. In addition, it holds that every variable in Z_i is present in some variable set Z_i^j in (29) and that every variable in W_i is present in some W_i^j in (30), so that $Z_i = \bigcup_{j=1}^{s_i} Z_i^j$ and $W_i = \bigcup_{j=1}^{p_i} W_i^j$. Hence,

$$\begin{aligned} X' &= \bigcup_i^s X_i = \bigcup_i^s \left(\mathrm{unDiff}\left(Z_i\right) \cup W_i\right) \\ &= \bigcup_i^s \left(\mathrm{unDiff}\left(\bigcup_{j=1}^{s_i} Z_i^j \right) \cup \bigcup_{j=1}^{p_i} W_i^j \right) \\ &= \bigcup_i^s \left(\bigcup_{j=1}^{s_i} \mathrm{unDiff}\left(Z_i^j\right) \cup \bigcup_{j=1}^{p_i} W_i^j \right), \end{aligned} \tag{34}$$

where the last equality trivially follows from the definition of unDiff () in (11). The property (33) and thus condition 1) has then been verified.

Condition 2) of Definition 3 will now be verified, that is, that $\mathcal{C}$ can be used to obtain a system in BLT semi-explicit DAE form. Consider an element $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$. Since $E_{Z_i}^j \subseteq E_{Z_i} \subseteq E_i$, and we have that $(X_i, E_i) \in S$, the property (28) implies that

$$\mathrm{var}_X(E_{Z_i}^j) \cap \{X_{i+1} \cup X_{i+2} \cup \ldots \cup X_s\} = \emptyset, \tag{35}$$

for $i = 1, 2, \ldots, s-1$. From lines 17-21 in the algorithm, it follows that the AE tool $\mathcal{T}$ can be used to solve the equations in $E_{Z_i}^j$ for the variables in Z_i^j. Since we have assumed that each differential equation contains at most one differentiated variable and (35) holds, we can use $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$ and the AE tool $\mathcal{T}$ to obtain

$$\dot{\mathbf{z}}_i^j = \mathbf{g}_i^j\left(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \mathbf{y}\right), \tag{36}$$

where $\dot{\mathbf{z}}_i^j$ is a vector of the variables in Z_i^j, $\mathbf{x}_k$ a vector of the variables in X_k, $\mathbf{y}$ a vector of the known variables in E', and $\mathbf{g}_i^j$ a function returned by $\mathcal{T}$ when the arguments are Z_i^j and $E_{Z_i}^j$. From the elements $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$, $j = 1, 2, \ldots, s_i$, we can thus, by using (36) and also that $X_i = \text{unDiff}\,(Z_i) \cup W_i$, obtain

$$\dot{\mathbf{z}}_i = \mathbf{g}_i\left(\mathbf{z}_1, \mathbf{z}_1, \ldots, \mathbf{z}_i, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_i, \mathbf{y}\right), \tag{37}$$

where $\mathbf{z}_i = \left(\mathbf{z}_i^1, \mathbf{z}_i^2, \ldots, \mathbf{z}_i^{s_i}\right)$ and a vector of the variables in Z_i, $\mathbf{w}_i$ a vector of the variables in W_i, $\mathbf{y}$ a vector of the known variables in E', and $\mathbf{g}_i = \left(\mathbf{g}_i^1, \mathbf{g}_i^2, \ldots, \mathbf{g}_i^{s_i}\right)$.

Now instead consider an element $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$. Since also $\left(W_i^j, E_{W_i}^j\right) \in S_{W_i}$, where S_{W_i} is given by (30) the property (31) holds. Since $E_{W_i}^j \subseteq E_{W_i} \subseteq E_i$, and $(X_i, E_i) \in S$ we also have that

$$\mathrm{var}_X(E_{W_i}^j) \cap \{X_{i+1} \cup X_{i+2} \cup \ldots \cup X_s\} = \emptyset, \tag{38}$$

for $i = 1, 2, \ldots, s-1$. By using that the AE tool $\mathcal{T}$ can solve $E_{W_i}^j$ for W_i^j due to lines 27-31, that $X_i = \text{unDiff}\,(Z_i) \cup W_i$ and $\mathrm{var}_D(E_{W_i}) \cap Z_i = \emptyset$ due to lines 6-8 and 12-14, and then utilize (31) and (38), we can obtain

$$\begin{aligned}\mathbf{w}_i^j = \mathbf{h}_i^j\big(&\dot{\mathbf{w}}_1, \ldots, \dot{\mathbf{w}}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i, \mathbf{w}_1, \ldots, \mathbf{w}_{i-1}, \\ &\mathbf{w}_i^1, \ldots, \mathbf{w}_i^{j-1}, \mathbf{y}\Big),\end{aligned} \tag{39}$$

from $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$, where $\mathbf{w}_i^j$ is a vector of the variables in W_i^j, $\mathbf{z}_i$ a vector of the variables in Z_i, and $\mathbf{h}_i^j$ a function returned by $\mathcal{T}$ when the arguments are

W_i^j and $E_{W_i}^j$. Note that the absence of vectors $\dot{\mathbf{z}}_i$ in (39) is a direct implication of the assumption that each differentiated variable is present in only one equation in the original model and therefore also in the BLT semi-explicit DAE system. Since $\dot{\mathbf{z}}_i$, obviously, is present in (37), it can not be present in (39).

By using (39), we can then obtain

$$\begin{aligned}
\mathbf{w}_i^1 &= \mathbf{h}_i^1\left(\dot{\mathbf{w}}_1, \ldots, \dot{\mathbf{w}}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i, \mathbf{w}_1, \ldots, \mathbf{w}_{i-1}, \mathbf{y}\right) \\
\mathbf{w}_i^2 &= \mathbf{h}_i^2\left(\dot{\mathbf{w}}_1, \ldots, \dot{\mathbf{w}}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i, \mathbf{w}_1, \ldots, \mathbf{w}_{i-1}, \mathbf{w}_i^1, \mathbf{y}\right) \\
&\vdots \\
\mathbf{w}_i^{p_i} &= \mathbf{h}_i^{p_i}\left(\dot{\mathbf{w}}_1, \ldots, \dot{\mathbf{w}}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i, \mathbf{w}_1, \ldots, \mathbf{w}_{i-1},\right. \\
&\quad \left.\mathbf{w}_i^1, \ldots, \mathbf{w}_i^{p_i}, \mathbf{y}\right)
\end{aligned} \tag{40}$$

from the elements $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$, $j = 1, 2, \ldots, p_i$. Comparing (37) and (40) with the system in Definition 2, shows that the elements $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$, $j = 1, 2, \ldots, s_i$ and $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$, $j = 1, 2, \ldots, p_i$, corresponds to the i:th block of a BLT semi-explicit DAE form. Applying the above arguments for $i = 1, 2, \ldots, s$ then implies that the ordered set $\mathcal{C}$ in (32) can be used to obtain a system in BLT semi-explicit DAE form with s blocks. Thus, $\mathcal{C}$ is computation sequence for X' with $\mathcal{T}$.

It now remains to show that $\mathcal{C}$ is a minimal and irreducible computation sequence for X' with $\mathcal{T}$. We begin with the irreducibility of $\mathcal{C}$. In the beginning of this proof, we showed that all elements of $\mathcal{C}$, given by (32), correspond to SCCs. We have also concluded that due to the assumptions regarding the model in Section 1, all elements $\left(Z_i^j, E_{Z_i}^j\right) \in \mathcal{C}$ are of size one, i.e. trivially irreducible. Now consider an element $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$ and assume that we partition W_i^j as $W_i^j = W_{i1}^j \cup W_{i2}^j$ and $E_{W_i}^j$ as $E_{W_i}^j = E_{W_i1}^j \cup E_{W_i2}^j$ and form the two new elements $\left(W_{i1}^j, E_{W_i1}^j\right)$ and $\left(W_{i2}^j, E_{W_i2}^j\right)$. Due to the fact that $\left(W_i^j, E_{W_i}^j\right)$ corresponds to a SCC, $E_{W_i}^j$ is a dependent equation set with respect to the variables in W_i^j. This implies that when applying $\mathcal{T}$ to the elements $\left(W_{i1}^j, E_{W_i1}^j\right)$ and $\left(W_{i2}^j, E_{W_i2}^j\right)$, we obtain the two equations

$$\begin{aligned}
\mathbf{w}_{i1}^j &= \mathbf{h}_{i1}^1\left(\ldots, \mathbf{w}_{i2}^j, \ldots\right) \\
\mathbf{w}_{i2}^j &= \mathbf{h}_{i2}^1\left(\ldots, \mathbf{w}_{i1}^j, \ldots\right),
\end{aligned}$$

which clearly not has the structure of equations contained in a BLT semi-explicit DAE system, due to the cyclic dependence between the equations. Hence, a

system in BLT semi-explicit DAE form can not be obtained when the element $\left(W_i^j, E_{W_i}^j\right) \in \mathcal{C}$ is partitioned, which violates condition 2) in Definition 3. We can then conclude that no elements of $\mathcal{C}$ can be further partitioned and hence $\mathcal{C}$ is an irreducible computation sequence for X' with $\mathcal{T}$.

The minimality of $\mathcal{C}$ for X' with $\mathcal{T}$ trivially follows from the fact that (34) holds. Since as (34) is fulfilled, all elements in $\mathcal{C}$ is needed to compute the variables in X'. This implies that any attempt to form a computation sequence for X' with $\mathcal{T}$ by using a subset of $\mathcal{C}$ will violate condition 1) in Definition 3. This completes the proof. □