A Mixed Causality Approach to Residual Generation Utilizing Equation System Solvers and Differential-Algebraic Equation Theory*

Carl Svärd and Mattias Nyberg
Department of Electrical Engineering
Linköping University, SE-58183 Linköping, Sweden
{carl, matny}@isy.liu.se

Abstract

The FDI approach to model-based diagnosis is considered. We present a method for residual generation that combines integral and derivative causality, and also utilizes equation system solvers and theory of differential-algebraic equation systems. To achieve this, a framework for computation of variables from sets of dependent differential and/or algebraic equations is introduced. The proposed method is applied to a model of the gas flow in an automotive diesel engine. The application clearly shows the benefit of using a mixed causality approach for residual generation compared with solely integral or derivative causality.

1 Introduction

With the rising demand for reliability and safety of technical systems, fault diagnosis has become increasingly important. In the FDI approach to model-based fault diagnosis, a mathematical model of the system, together with measurements, is utilized to generate residuals, used to detect and isolate faults present in the system. One residual generation approach [Staroswiecki and Declerck, 1989] is to, by means of structural analysis, use a part of the model, i.e. a subset of equations, to compute a subset of the unknown variables and then use a redundant equation as residual. The generation of a residual will thus consist of a finite sequence of variable computations ending with an evaluation of an unused equation, where the computation in each step only require variables that are known, i.e. measured, or have been computed in some previous step. Similar methods have been used in [Blanke *et al.*, 2003], [Cassar and M., 1997], [Ploix *et al.*, 2005], [Travé-Massuyès *et al.*, 2006], and [Pulido and Alonso-Gonzlez, 2004].

In these previous methods, the most common approach has been to use one equation at a time to compute one single unknown variable in each step. This has been done by using scalar equation solvers for algebraic equations, and by applying either integral or derivative causality for differential equations.

However, complex models contain dependencies between equations. This fact gives rise to differential and algebraic loops or cycles, see [Blanke *et al.*, 2003], [Katsillis and Chantler, 1997], which corresponds to systems of dependent differential and/or algebraic equations. Thus, it is important that a method for residual generation is able to handle such systems of equations. Furthermore, as illustrated in this paper, it may be unnecessarily limiting to consider solely integral or derivative causality.

The main contribution in this paper is a method for residual generation that utilizes equation system solvers and combines integral and derivative causality into a mixed causality approach. To achieve this, we present a unifying framework for computation of variables from sets of dependent differential and algebraic equations which utilizes theory for solving and analyzing general differential-algebraic equations. In the proposed method, the causality of differential equations is defused and the way a differential equation is handled depends on the context in which the variables appear, the available tools for equation solving, the available tools for approximate differentiation of measurements, and knowledge about initial conditions.

The paper is organized as follows. Section 2 presents preliminaries and some basic theory and references for differential-algebraic equations and structural analysis. In Section 3, a framework for computation of variables from sets of dependent differential and algebraic equations is presented. Sections 4 to 6 presents the proposed method. In Section 7, an application example clearly shows the benefits of using a mixed causality approach compared with either integral or derivative causality. Section 8 concludes the paper. Due to the limitation of space, proofs are omitted but can be found in [Svärd and Nyberg, 2008].

2 Preliminaries

Consider a *model* $M(E, X, Z)$ or M for short, consisting of a set of equations $E = \{e_1, \ldots, e_m\}$ relating a set of unknown variables $X = \{x_1, \ldots, x_n\}$, and a set of known variables $Z = \{z_1, \ldots, z_p\}$. Introduce a third set, $D = \{\dot{x}_1, \ldots, \dot{x}_n\}$, containing the derivatives of the variables in X. Without loss of generality, we assume that the equations in the set E are in the form

$$e_i: \quad f_i(\dot{x}, x, z) = 0, \quad 1 \leq i \leq m \tag{1}$$

where $\dot{x}$, x, and z are vectors of the elements in the sets D, X, and Z, respectively.

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

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

The set $\mathcal{O}(M)$ is referred to as the *observation set* of the model M. A residual generator is here formally defined as follows.

Definition 1 (Residual Generator for $M(E, X, Z)$)**.** *A system with input z and output r is a residual generator for the model $M(E, X, Z)$ and r is a residual if $z \in \mathcal{O}(M) \Rightarrow r = 0$.*

2.1 Differential-Algebraic Equation Systems

It is assumed that the model (1) contains both differential and algebraic equations, that is, it is a differential-algebraic equation (DAE) system, or descriptor system. 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].

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

A common approach when analyzing and solving general DAE-systems, is to first seek a reformulation of the original DAE into a simpler and well-structured description with the same set of solutions, see [Kunkel and Mehrmann, 2006], and [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 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 ordinary differential equation (ODE), see for example [Brenan *et al.*, 1989]. The reformulation thus aims to write the original DAE as an ODE, i.e. a system in state-space form.

2.2 Structure of the Model

Let $C \subseteq E$ and introduce the notations

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

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

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

$$A = \{(e_i, x_j) : x_j \in \mathrm{var}_X(\{e_i\}), e_i \in E, x_j \in X\}, \quad (3)$$

the set of arcs. We will call the bi-partite graph $G = (E, X, A)$ the *structure* of the model $M(E, X, Z)$. 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 bi-adjacency matrix defined as

$$B = \{b_{ij} : b_{ij} = 1 \text{ if } (e_i, x_j) \in A, 0 \text{ otherwise}\}. \quad (4)$$

A *matching* Γ on the bi-partite graph G 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. 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 is called its *strongly connected components* (SCC), see for example [Asratian *et al.*, 1998].

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 $G^+ = (E^+, X^+, A^+)$, $G_i^0 = (E_i^0, X_i^0, A_i^0), 1 \leq i \leq s$, and $G^- = (E^-, X^-, A^-)$, referred to as 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 $G_i^0 = (E_i^0, X_i^0, A_i^0)$ correspond to the SCC of the directed graph induced by any complete matching on the bi-partite graph G^0, [Murota, 1987].

3 Computability of Variables

Introduce the notation X_I for the subset of X defined as $X_I = \{x_i : i \in I\}$, where $I \subseteq \{1, \ldots, n\}$. A similar convention will be used to denote subsets of D, Z, and E. Also, $\bar{I}$ will be used to denote the complement of the set I in $\{1, \ldots, n\}$, i.e. $\bar{I} = \{1, \ldots, n\} \setminus I$. To retrieve the indices of a set of variables (or equations), the operator $\mathrm{ind}(\cdot)$ is introduced, i.e. $\mathrm{ind}(X_I) = I$. Now, let $I \subseteq \{1, 2, \ldots, n\}$ and $J \subseteq \{1, 2, \ldots, m\}$, and consider the sets X_I and E_J.

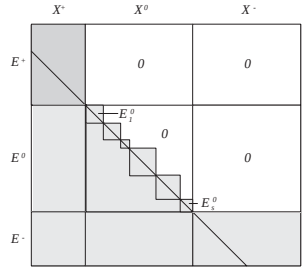


Figure 1: The bi-adjacency matrix showing the DM-decomposition of G. The line along the diagonal in corresponds to a maximum matching on G.

Definition 2 (Computability)**.** *The variables X_I are* computable *from the equations E_J if, given trajectories of the variables $X_{\bar{I}} \cup Z$, trajectories of X_I can be computed with the available tools.*

3.1 Tools for Computation of Variables

Computability of a set of variables from a set of equations generally depends not only on the analytical properties of the equations in the set, but also on the set of tools that are available for use. In this paper, three types of tools are considered:

DE Solving Tools: Tools for solving explicit ordinary differential equations;

AE Solving Tools: Tools for solving algebraic (not differential) equation systems;

Differentiating Tools: Tools for approximate differentiation of measured (known) variables.

An AE solving tool is typically some software package for symbolic or numerical equation solving. A differentiating tool can for example be an implementation of a low-pass filter or a smoothing-spline approximate differentiator, [Wei and Li, 2006]. In this paper, we assume that AE solving tools are available through existing standard software packages like e.g. Maple or Mathematica, and design and implementation of AE solving tools will not be considered. We also assume that DE solving tools are always available, i.e. that the states of an explicit ordinary differential equation (a DAE of differential index 0) can be computed if the initial conditions of the states are known and consistent. This can be motivated by the fact that there exist several efficient methods for solving ODEs, see for example [Brenan *et al.*, 1989]. Implementations are available in for example MATLAB and SIMULINK. Of course the 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.

Proposition 1 (Computability)**.** *The variables X_I are computable from the equations E_J if*

1. *the available AE solving tools admits a transformation of E_J into*

$$\dot{x}_{I^d} = g_d\left(\dot{x}_{\bar{I}}, x_{\bar{I}}, x_{I^d}, x_{I^a}, z\right) \quad (5a)$$

$$x_{I^a} = g_a\left(\dot{x}_{\bar{I}}, x_{\bar{I}}, x_{I^d}, z\right), \quad (5b)$$

   where $I^d = \mathrm{ind}(\mathrm{var}_D(E_J)) \cap I$, and $I^a = I \setminus I^d$,

2. *the initial conditions of the variables in X_{I^d} are known and consistent, and*
3. *the derivatives in $\mathrm{var}_D(E_J) \cap D_{\bar{I}}$ can be obtained with the available differentiating tools.*

Remark 1. *If all equilibrium points of the system* (5a) *are, or with for example state-feedback can be made, (globally) asymptotically stable, the effect of the initial conditions are neglectable and condition 2 can be removed, see for example [Khalil, 2002].*

Remark 2. *One alternative to differentiate unknown variables directly, is to propagate known variables through a set of equations so that derivatives of unknown variables can be expressed as derivatives of known, i.e. measured, variables. Assume for example that we want to compute the derivative $\dot{x}_1$ and we also have that $x_1 = z_1$. To compute $\dot{x}_1$, we use a differentiating tool to compute $\dot{z}_1$ and then use $\dot{x}_1 = \dot{z}_1$.*

There are two important special cases of computability. If the variables X_I are computable from E_J and $I^a = I$, $I^d = \varnothing$, i.e.

$$x_I = g\left(\dot{x}_{\bar{I}}, x_{\bar{I}}, z\right), \tag{6}$$

the variables X_I are said to be *algebraically computable* from E_J. Conversely, if $I^d = I$ and $I^a = \varnothing$, i.e.

$$\dot{x}_I = g\left(\dot{x}_{\bar{I}}, x_{\bar{I}}, x_I, z\right), \tag{7}$$

the variables X_I are said to be *differentially computable* from E_J. If a set of variables is algebraically computable, so called *derivative causality* is used, and if a set of variables is differentially computable *integral causality* is used, see [Blanke *et al.*, 2003]. Thus, if a set of variables is computed according to (5), or if a subset of variables in a model is algebraically computable and another subset of variables is differentially computable, both integral and derivative causality is used, i.e. *mixed causality*.

Remark 3. *If the variables $X_{\bar{I}}$ are regarded as known variables and the sets I^d and I^a are both non-empty,* (5) *is equivalent to a semi-explicit DAE of differential index 1. Furthermore,* (6) *corresponds to an algebraic equation or equivalently an explicit DAE of differential index 1, and* (7) *to an explicit ODE or an explicit DAE of differential index 0, see [Brenan* et al.*, 1989].*

3.2 Initial Conditions and Estimation of Derivatives

The availability of initial conditions in general 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 differentiated variables in the models correspond to physical quantities such as pressures and temperatures, which makes initial conditions known.

If the derivatives of a set of variables can be computed or not, depends both on the available set of differentiating tools and the quality of the measurements of the known variables. There are several approaches for approximate differentiating, e.g. smoothing spline approximation [Wei and Li, 2006]. An extensive survey of methods can be found in [Barford *et al.*, 1999]. Derivative estimation is not in the scope of this paper, and will not be further considered.

4 A Method for Residual Generation

One approach to residual generation for a model is to sequentially compute subsets of the unknown variables from subsets of the equations, and then use an unused equation as residual. The generation of a residual will then consist of a finite sequence of variable computations, ending with an evaluation of a residual equation. The computation of variables in each step can thus only use variables that has been computed in some previous step, and known variables. To describe which variables that should be computed from which set of equations and in which order the variables should be computed, we introduce the concept *variable set matching*.

4.1 Variable Set Matching

Assume that $\mathcal{I} = \{I_1, \ldots, I_s\}$ and $\mathcal{J} = \{J_1, \ldots, J_t\}$ are partitions of $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$ respectively, and let the corresponding induced partitions of X and E be denoted $\mathcal{X} = \{X_{I_1}, \ldots, X_{I_s}\}$ and $\mathcal{E} = \{E_{J_1}, \ldots, E_{J_t}\}$.

Let Λ be a function from $\mathcal{X}$ to $\mathcal{E}$ and assume that $(X_{I_i}, E_{J_i}) \in \Lambda$ and $(X_{I_j}, E_{J_j}) \in \Lambda$. Define the binary relation $\prec$ on $\mathcal{X} \times \mathcal{E}$ such that $(X_{I_i}, E_{J_i}) \prec (X_{I_j}, E_{J_j})$ iff $X_{I_i} \cap \mathrm{var}_X(E_{J_j}) \neq \varnothing$

Definition 3 (Variable Set Matching)**.** *The function Λ is a* variable set matching *for X on E if*

1. *Λ is injective,*
2. *for every $(X_{I_i}, E_{J_i}) \in \Lambda$ it holds that the variables X_{I_i} are computable from E_{J_i}, and*
3. *the directed graph defined by $\prec$ on Λ contains no directed cycles.*

Remark 4. *The first property ensures that Λ is complete with respect to the variable set X. The third property prevents that computation of the variables in X_{I_i} requires the variables in X_{I_j}, which in turn requires the variables in X_{I_i}.*

Proposition 2. *The variables X are computable from the equations E if there exist partitions of X and E such that there exists a variable set matching Λ for X on E.*

The binary relation $\prec$ on the variable set matching Λ defines a *computation order* for X on E. A computation order can thus be represented as a directed acyclic graph.

4.2 Computation Sequence

If the variables X are computable from E, the variable set matching Λ specifies which variables that should be computed from which equations. The order in which the variables in X must be computed is specified by the computation order $\prec$. From a computation order, a computation sequence can be obtained.

Definition 4 (Computation Sequence for X on E)**.** *A linear order obtained by topological ordering of the directed (acyclic) graph defined by $\prec$ on the variable set matching Λ is a* computation sequence *for X on E.*

In general, a computation sequence obtained from a computation order is not unique.

Assume that the variables X are computable from E, and that $\mathcal{X} = \{X_{I_1}, \ldots, X_{I_s}\}$ and $\mathcal{E} = \{E_{J_1}, \ldots, E_{J_t}\}$ are the partitions of X and E for which a variable set matching Λ exists. If we define $R = \{1, \ldots, m\} \setminus \{\bigcup_{i=1}^{s} J_i\}$, the set E_R will contain those equations that are not used in the computation of the variables in X, and will be referred to as the *redundant equation set* associated with the variable set matching Λ.

By using trajectories of the known variables in Z and the equations in $E \setminus E_R$, trajectories of all variables in X can be computed according to the computation sequence. As the trajectories of all variables in X are computed, we can compute a residual from a redundant equation $e_i \in E_R$ as $r = f_i(\dot{x}, x, z)$. The equation e_i will be referred to as the *residual equation*. We have motivated the following proposition.

Proposition 3. *A computation sequence for X on E together with an equation $e_i \in E_R$ is a residual generator for $M(E, X, Z)$.*

To illustrate the concepts presented above, we study a small academic example.

Example 1. Consider the following set of equations

$$\begin{aligned}
e_1: &\quad \dot{x}_1 + x_1 x_2 + x_5 + z_1 = 0 \\
e_2: &\quad \dot{x}_2 + x_1 + x_2 + x_3 + z_2 = 0 \\
e_3: &\quad \dot{x}_3 + x_3 - x_4 = 0 \\
e_4: &\quad x_3 + x_4 + x_5 + z_3 = 0 \\
e_5: &\quad x_5 + z_4 = 0 \\
e_6: &\quad h(x_1, x_4, z_5) = 0,
\end{aligned}$$

where it is assumed that neither x_1 nor x_4 can be computed from e_6. Let $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$, $X =$

$\{x_1, x_2, x_3, x_4, x_5\}$, $Z = \{z_1, z_2, z_3, z_4, z_5\}$, and $D = \{\dot{x}_1, \dot{x}_2, \dot{x}_3, \dot{x}_4, \dot{x}_5\}$.

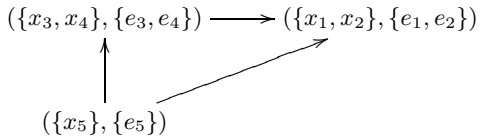
By first studying the equations e_1 and e_2, we see that $\{x_1, x_2\}$ can be (differentially) computed from $\{e_1, e_2\}$, if the initial conditions of x_1 and x_2 are known and consistent, and the available AE solving tools admit that e_1 can be solved for $\dot{x}_1$ and e_2 for $\dot{x}_2$. We also see that if e_3 can, with the available AE solving tools, be solved for $\dot{x}_3$ and e_4 for x_4, the equation set $\{e_3, e_4\}$ becomes

$$\dot{x}_3 = -x_3 + x_4$$
$$x_4 = -x_3 - x_5 - z_3,$$

which is on the form (5). Thus, if also the initial condition of x_3 is known, $\{x_3, x_4\}$ are computable from $\{e_3, e_4\}$. If we assume that our AE solving tools admits that e_5 is solved for x_5, $\{x_5\}$ is (algebraically) computable in $\{e_5\}$. With X and E partitioned as $\mathcal{X} = \{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5\}\}$ and $\mathcal{E} = \{\{e_1, e_2\}, \{e_3, e_4\}, \{e_5\}, \{e_6\}\}$, we now define the function

$$\Lambda = \{(\{x_1, x_2\}, \{e_1, e_2\}), (\{x_3, x_4\}, \{e_3, e_4\}), (\{x_5\}, \{e_5\})\}$$

from $\mathcal{X}$ to $\mathcal{E}$. Since $\{x_5\} \cap \mathrm{var}_X(\{e_3, e_4\}) = \{x_5\} \cap \{x_3, x_4, x_5\} = \{x_5\}$, it holds that $(\{x_5\}, \{e_5\}) \prec (\{x_3, x_4\}, \{e_3, e_4\})$, and by similar calculations, we conclude that $(\{x_5\}, \{e_5\}) \prec (\{x_1, x_2\}, \{e_1, e_2\})$, and $(\{x_3, x_4\}, \{e_3, e_4\}) \prec (\{x_1, x_2\}, \{e_1, e_2\})$. The directed graph defined by $\prec$ on Λ is pictured below.



Since the directed graph contains no directed cycles, the function Λ is injective, and all variables are computable in respective equations for each element of Λ, we conclude that Λ is a variable set matching for X on E. From the directed graph, we obtain the computation sequence

$$(\{x_5\}, \{e_5\}), (\{x_3, x_4\}, \{e_3, e_4\}), (\{x_1, x_2\}, \{e_1, e_2\}). \quad (8)$$

The variables in X can then be computed in the order specified in (8). The only redundant equation in E is thus e_5, and hence the residual is computed as $r = h(x_1, x_4, z_5)$.

5 Finding Computation Sequences

The problem of designing a residual generator for the model $M(E, X, Z)$ can be divided into the following steps

1. Find a variable set matching;
2. Obtain a computation sequence from the computation order associated with the variable set matching;
3. Use a redundant equation as residual equation.

Step 2 is trivial, there are many efficient algorithms for topological ordering, see for example [Cormen *et al.*, 2001]. Since also step 3 is trivial, the key point is to find a variable set matching.

5.1 Finding Variable Set Matchings

A variable set matching Λ for X on E is a function from a partition $\mathcal{X} = \{X_{I_1}, \ldots, X_{I_s}\}$ of X to a partition $\mathcal{E} = \{E_{J_1}, \ldots, E_{J_t}\}$ of E, that fulfills the properties specified in Definition 3. To be more specific, it must hold that for every $X_{I_i} \in \mathcal{X}$ there exists $E_{J_j} \in \mathcal{E}$ such that X_{I_i} are computable from E_{J_j}, and that the directed graph defined by the relation $\prec$ on Λ contains no cycles.

As said in Section 3, computability of variables from a set of equations depends on both the analytical properties of the equations in the set and the set of tools available for use. Naturally, a necessary condition for X_{I_i} to be computable from E_{J_j} is that $X_{I_i} \subseteq \mathrm{var}_X(E_{J_j})$. Regarding the tools, we assume the following.

Assumption 1. *AE solving tools require that* $|E_J| = |X_I|$.

Assumption 2. *AE solving tools prefer, for e.g. numerical reasons, equation sets of small cardinality before equation sets with large cardinality.*

An implication of Assumption 2 is that if the variables X_I are computable from E_J, but there exists a variable set matching $\Lambda = \{(X_{I_1}, E_{J_1}), \ldots, (X_{I_s}, E_{J_s})\}$ for X_I on E_J, it is preferable to compute the variables X_I from the smaller equation sets E_{J_i}.

Finding Equation Sets with Minimum Cardinality

Due to Assumption 2, we should find partitions of X and E with maximum cardinality. Thus, variable set matchings should contain equation (and variable) sets of minimum cardinality. However, equation sets of cardinality one can not always be used due to dependencies between equations. The dependencies will naturally induce cycles in the intended variable set matching.

Consider the bi-partite graph $G = (E, X, A)$, representing the structure of the model $M(E, X, Z)$ according to Section 2.2. Let I and J be subsets of $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$ respectively, such that the submodel $\bar{M}(E_J, X_I, Z)$ of M is just-determined. Let $\bar{G} = (E_J, X_I, \bar{A})$ denote the corresponding bi-partite graph representing the structure of $\bar{M}$. Motivated by the fact that the DM-components are irreducible bi-partite subgraphs, we apply the DM-decomposition to the graph $\bar{G}$ to obtain the DM-components $\bar{G}_i = (E_{J_i}, X_{I_i}, \bar{A}_i)$. Since $\bar{G}$ is just-determined, the DM-components $\bar{G}_i$ are exactly the SCCs of the directed graph induced by any maximum matching on $\bar{G}$, see for example [Murota, 1987]. The following proposition holds.

Proposition 4. *Let* $\bar{G} = (E_J, X_I, \bar{A})$ *be a just-determined part of* $G = (E, X, A)$ *and* $\bar{G}_i = (E_{J_i}, X_{I_i}, \bar{A}_i), 1 \leq i \leq s$ *its strongly-connected components. The set*

$$\Lambda = \{(X_{I_1}, E_{J_1}), \ldots, (X_{I_s}, E_{J_s})\} \quad (9)$$

is a variable set matching for X_I *on* E_J *if for every* $(X_{I_i}, E_{J_i}) \in \Lambda$, *the variables* X_{I_i} *are computable from* E_{J_i}.

A justified question is then if there exists a variable set matching for X_I on E_J, whose equation sets have less cardinality than the equation sets originating from the SCCs according to Proposition 4.

Proposition 5. *Let* $\bar{G}_i = (E_{J_i}, X_{I_i}, \bar{A}_i)$ *be a SCC of* $\bar{G}$, *then there exist no variable set matching with elements of cardinality larger than one for* X_{I_i} *on* E_{J_i}.

Proposition 5 implies that it is impossible to partition E_J into blocks with less cardinality than the SCC, without ending up with a cycle that prohibits a variable set matching.

Remark 5. *SCCs are utilized in [Porté* et al., *1988] and [Katsillis and Chantler, 1997] to determine the causal order [Iwasaki and Simon, 1986] of the variables in a model consisting of algebraic and differential equations. However, the causal order depends only on the occurrences of variables in the equations and does not consider computability, i.e. analytical properties of the involved equations, initial conditions, and available tools. SCCs are also used to partition sparse systems of equations into the so called BLT-form in tools for non-causal simulation, see for example [Fritzon, 2004].*

5.2 An Algorithm for Finding Variable Set Matchings

Proposition 4 states a sufficient condition for finding a variable set matching. Motivated by this and the implication of Proposition 5, we propose Algorithm 1 for finding a variable set matching for X_I on E_J.

The function `findAllSCC` in Algorithm 1 returns equation and variable sets corresponding to the SCCs of the specified just-determined equation set, with respect to the specified set of variables. There are efficient algorithms for finding

Algorithm 1: findVariableSetMatching

Input: A just determined set of equations E_J, a set of variables X_I, a set of AE solving tools T_{AES}, and a set of differentiating tools T_D
Output: A variable set matching Λ for X_I on E_J
$\Lambda := \varnothing$;
$S :=$ `findAllSCC(`E_J,X_I`)`;
foreach $(E_{J_i}, X_{I_i}) \in S$ **do**
  if `isComputable(`E_{J_i},X_{I_i},T_{AES},T_D`)` **then**
    $\Lambda := \Lambda \cup (X_{I_i}, E_{J_i})$;
  else
    return $\varnothing$;
  end
end
return Λ;

SCCs in directed graphs, see for example [Tarjan, 1972]. The function `isComputable` determines if the specified set of variables can be computed from the specified set of equations. This function is described in Algorithm 2, and will be further considered in Section 6.

5.3 Connection to MSO Sets

The problem of residual generation for a given model can, as said in beginning of this section, be divided into the three parts: 1) find a variable set matching, 2) obtain a computation sequence from the variable set matching, 3) use a redundant equation as residual equation. From the discussion above, it is clear that to find a variable set matching it is sufficient to consider a just-determined part of the given model. Hence, to design a residual generator it is sufficient to consider a part of the model that consist of a just-determined part and one redundant equation, that is, a minimal over-determined set of equations or in the structural case, a minimal structurally over-determined (MSO) set. The method for residual generation outlined in the beginning of this section, can thus be refined:

1. Find a MSO set in the model;
2. Find a variable set matching in the MSO set;
3. Obtain a computation sequence from the computation order associated with the variable set matching;
4. Use the redundant equation as residual equation.

There exist several efficient algorithms for finding all MSO sets in a model, see for example [Krysander *et al.*, 2008].

6 Analyzing Computability of Variables

Consider the variable set X_I and equation set E_J. From the development in Section 5.1, it is clear that we can limit our analysis to the case when $|X_I| = |E_J|$, and E_J corresponds to a strongly-connected component.

The decomposition into strongly-connected components is based on the structural representation of the model adopted in Section 2. With this representation, there is no difference between a variable x_j and the corresponding differentiated variable $\dot{x}_j$. The strongly-connected components therefore contains both differentiated and non-differentiated variables and thus both differential and algebraic equations. This means that the equation sets corresponding to SCCs are differential-algebraic equations. One approach for further analysis of computability of X_I from E_J is then to apply methods for analyzing and solving differential-algebraic equations (DAEs).

6.1 Analyzing Computability by Utilizing Differential-Algebraic Equation Theory

Motivated by theories for analyzing and solving differential-algebraic equation systems, we seek a reformulation, or transformation, of E_J into the form (5) according to Proposition 1. The ability to perform such a transformation, depends on the analytical properties of the equations in E_J, as well as the available AE solving tools. Having obtained a transformation of E_J into one differential part (5a), and one algebraic part (5b), it is also desirable that both the differential and algebraic part are further decomposed into smaller just-determined parts, due to Assumption 1 and 2. We illustrate our approach for analyzing computability with an example.

Example 2. Consider the set of equations

$$\begin{aligned}
e_1: &\quad \dot{x}_1 - \dot{x}_2 + x_1x_4 + x_2 + x_5^2 + x_7x_9 = 0\\
e_2: &\quad \dot{x}_1 + \dot{x}_2 + x_2x_3 + 2x_5x_8^2 = 0\\
e_3: &\quad \dot{x}_3 - x_2^2x_3x_6 + \dot{x}_7 = 0\\
e_4: &\quad x_1x_9 + x_4 + x_5 = 0\\
e_5: &\quad x_2 + x_4 - x_5 + \dot{x}_8 = 0\\
e_6: &\quad x_2x_3x_5 + x_6 + \dot{x}_9 = 0,
\end{aligned}$$

which for simplicity contains no known variables. The bi-adjacency matrix representing the structure of the equation set $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ with respect to $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ is shown in (10). It is clear that E corresponds to a SCC of size 6.

Equation	Unknown					
	x_1	x_2	x_3	x_4	x_5	x_6
e_1	1	1		1	1	
e_2	1	1	1		1	
e_3		1	1			1
e_4	1			1	1	
e_5		1		1	1	
e_6		1	1		1	1

(10)

First consider the equation set $\{e_1, e_2, e_3\}$, which contains the differentiated variables $\{\dot{x}_1, \dot{x}_2, \dot{x}_3\}$. If we consider the structure of $\{e_1, e_2, e_3\}$ with respect to $\{\dot{x}_1, \dot{x}_2, \dot{x}_3\}$, we obtain the bi-adjacency matrix shown in (11). We can now partition $\{e_1, e_2, e_3\}$ into the equation sets $\{e_1, e_2\}$, and $\{e_3\}$, corresponding to SCCs of size two and one, with respect to the structure in (11).

Equation	Unknown		
	$\dot{x}_1$	$\dot{x}_2$	$\dot{x}_3$
e_1	1	1	
e_2	1	1	
e_3			1

(11)

If our AE solving tools admits that $\{e_1, e_2\}$ is transformed into

$$\dot{x}_1 = \frac{1}{2}\left(-x_2 - x_2x_3 - x_1x_4 - x_5^2 - 2x_5x_8^2 - x_7x_9\right) \tag{12a}$$

$$\dot{x}_2 = \frac{1}{2}\left(x_2 - x_2x_3 + x_1x_4 + x_5^2 - 2x_5x_8^2 + x_7x_9\right), \tag{12b}$$

and $\{e_3\}$ into

$$\dot{x}_3 = x_2^2x_3x_6 - \dot{x}_7, \tag{13}$$

we have that $\{x_1, x_2\}$ are differentially computable from $\{e_1, e_2\}$, and $\{x_3\}$ is differentially computable from $\{e_3\}$ if the initial conditions of $\{x_1, x_2\}$ and $\{x_3\}$ are known and consistent and the derivative $\{\dot{x}_7\}$ can be computed with the available differentiating tools.

Now instead turn to the equation set $\{e_4, e_5, e_6\}$. From the bi-adjacency matrix in (10), we then see that $\{e_4, e_5, e_6\}$ can be partitioned into the equation sets $\{e_4, e_5\}$ and $\{e_6\}$, which corresponds to SCCs of size two and one respectively. Under the assumption that our AE solving tools admits a transformation of $\{e_4, e_5\}$ into

$$x_4 = \frac{1}{2}\left(-x_2 - x_1x_9 - \dot{x}_8\right) \tag{14a}$$

$$x_5 = \frac{1}{2}\left(x_2 - x_1x_9 + \dot{x}_8\right), \tag{14b}$$

and of $\{e_6\}$ into

$$x_6 = -x_2 x_3 x_5 - \dot{x}_9, \tag{15}$$

we see that $\{x_4, x_4\}$ are algebraically computable from $\{e_4, e_5\}$ and $\{x_6\}$ is algebraically computable from $\{e_6\}$, if the derivatives $\{\dot{x}_8\}$ and $\{\dot{x}_9\}$ can be computed with the available differentiating tools.

We have then transformed the original set of equations E into the form (5), with (12) and (13) corresponding to (5a), and (14) and (15) to (5b). Thus, we have $I^d = \{1, 2, 3\}$, $I^a = \{4, 5, 6\}$, and $\bar{I} = \{7, 8, 9\}$. Hence, if the initial conditions of $\{x_1, x_2, x_3\}$ are known and consistent, and the derivatives $\{\dot{x}_7, \dot{x}_8, \dot{x}_9\}$ can be computed with the available differentiating tools, the variables X are computable from E.

6.2 An Algorithm for Analyzing Computability

Motivated by Example 2 and the three conditions in Proposition 1, we propose the following procedure for analyzing if X_I are computable from E_J

1. Partition I into $\{I^a, I^d\}$, according to $I^d = \mathrm{ind}(\mathrm{var}_D(E_J)) \cap I$, $I^a = I \setminus I^d$;
2. Determine if the initial conditions of the variables X_{I^d} are known and consistent;
3. Determine if the derivatives $\mathrm{var}_D(E_J) \cap D_{\bar{I}}$ can be computed with the available differentiating tools;
4. Partition J into J^a and J^d, such that $\mathrm{var}_D(E_{J^a}) \cap D_{I^d} = \varnothing$;
5. Find the SCCs of $G^a = (E_{J^a}, X_{I^a}, A^a)$. For each SCC $G_i^a = (E_{J_i^a}, X_{I_i^a}, A_i^a)$, determine if $X_{I_i^a}$ can be computed from $E_{J_i^a}$ with the available AE solving tools;
6. Find the SCCs of $G^d = (E_{J^d}, D_{I^d}, A^d)$. For each SCC $G_i^d = (E_{J_i^d}, D_{I_i^d}, A_i^d)$, determine if $D_{I_i^d}$ can be computed from $E_{J_i^d}$ with the available AE solving tools.

A complete, fully automated, algorithm can be found in Algorithm 2. The function `isInitCondKnown` determines if the initial conditions of the specified variables are available and consistent. The function `isAESolvable` determines if the specified variables are computable from the specified set of equations with the available set of AE solving tools. The function `isDifferentiable` determines if the derivatives of the specified variables can be computed with the available set of differentiating tools. The function regards propagation of derivatives as described in Remark 2.

Given ideal AE solving tools, ideal differentiating tools, and consistent initial conditions for the variables X_{I^d}, where $I^d = \mathrm{ind}(\mathrm{var}_D(E_J) \cap D_I)$, Algorithm 2 returns `true` iff the equation set E_J can be transformed into the form (5), possibly with either of the sets I^d or I^a empty. From this and Remark 3 it follows that with ideal AE solving tools, ideal differentiating tools, and consistent initial conditions for the variables $X_{\mathrm{ind}(\mathrm{var}_D(E))}$, a variable set matching for X on E can be found with Algorithm 1 iff E is just-determined and its SCCs can be transformed to semi-explicit DAEs of differential index 1 or explicit DAEs of differential index 1 or 0, i.e. SCCs in the form (5), (6), or (7).

Remark 6. *Although only SCCs corresponding to semi-explicit DAEs of differential index one can be handled with Algorithm 2, equation sets that are of higher differential index as a whole can often be handled with the proposed method. Consider the equation set*

$$\begin{aligned} e_1: &\quad \dot{x}_1 - x_2 = 0 \\ e_2: &\quad \dot{x}_2 - x_3 = 0 \\ e_3: &\quad x_1 - z_1 = 0, \end{aligned}$$

which is a DAE of differential index 3, taken from [Mattson and Söderlind, 1993]. If we assume that our AE and differentiating tools are ideal, Algorithm 1 returns the variable set matching $\Lambda = \{(x_1, e_3), (x_3, e_2), (x_2, e_1)\}$, where each element corresponds to a SCC of size 1. The associated computation sequence is $(x_1, e_3), (x_2, e_1), (x_3, e_2)$ and the variables $\{x_1, x_2, x_3\}$ can be computed from $\{e_1, e_2, e_3\}$ as $x_1 = z_1, x_2 = \dot{x}_1$, and $x_3 = \dot{x}_2$. Thus, even though the original system is a DAE of differential index 3, the proposed method can be used to find a variable set matching since the SCCs of $\{e_1, e_2, e_3\}$ are DAEs of differential index 1.

Algorithm 2: isComputable

Input: A just-determined set of equations E_J, a set of variables X_I, a set of AE solving tools T_{AES}, and a set of differentiating tools T_D
Output: True if the variables X_I are computable from E_J, else false.

```
I^d := ind(var_D(E_J) ∩ D_I);
if not isInitCondKnown(X_{I^d}) then
    return false;
end
if not isDifferentiable(var_D(E_J) ∩ D_Ī, T_D) then
    return false;
end
I^a := I \ I^d;
J^d := ∅;
J^a := ∅;
foreach i ∈ J do
    if var_D({e_i}) ∩ D_I = ∅ then
        J^a := J^a ∪ {i};
    else
        J^d := J^d ∪ {i};
    end
end
S^a := findAllSCC(E_{J^a}, X_{I^a});
foreach (E_{J_i^a}, X_{I_i^a}) ∈ S^a do
    if not isAESolvable(E_{J_i}, X_{I_i}, T_AES) then
        return false;
    end
end
S^d := findAllSCC(E_{J^d}, D_{I^d});
foreach (E_{J_i^d}, D_{I_i^d}) ∈ S^d do
    if not isAESolvable(E_{J_i^d}, D_{I_i^d}, T_AES) then
        return false;
    end
end
return true;
```

7 Application Example

In this section, the proposed method for residual generation is applied to a complex model of the gas flow in an automotive diesel engine.

7.1 The Engine Model

The modeled engine is a six cylinder Scania diesel engine equipped with exhaust gas recirculation (EGR) and a variable geometry turbocharger (VGT). The model focuses on the gas flow in the engine and is described in [Wahlström, 2006]. To be better suited for residual generation, it was modified in [Kingstedt and Johansson, 2008]. The modified model contains in total 50 equations, 47 unknown variables, and 11 known variables. The variables represent physical quantities such as pressures, temperatures, and rotational speeds. The model consists of 8 differential equations and 42 algebraic equations, i.e. the model is a differential-algebraic equation.

7.2 Configurations of the Algorithm

For comparison, the algorithm was applied to the engine model with three different configurations. The following parameters were used for configuration

- Availability of initial conditions;

- Characteristics of AE solving tools;
- Characteristics of differentiating tools.

These parameters naturally influences the possibility to compute variables in different ways, and thus also the possibility to find variable set matchings.

The configurations used are shown in Table 1. With configuration $\mathcal{C}_1$ the only way a set of variables can be computed from a set of differential equations is algebraically since no initial conditions are available, cf. (6). This is often referred to as *derivative causality*, see [Blanke *et al.*, 2003]. This approach for handling differential equations has been used in for example [Izadi-Zamanabadi, 2002] and [Dustegor *et al.*, 2004]. With configuration $\mathcal{C}_2$ on the other hand, the only way to compute a set of variables from a set of differential equations is according to (7), with the additional condition that $D_{\bar{I}} = \varnothing$, since no derivatives are available. This is in the literature referred to as *integral causality*, which is the way differential equations are handled in for example [Pulido and Alonso-Gonzlez, 2004] (still their framework supports the use of both integal and derivative causality). Configuration $\mathcal{C}_3$ thus handles both integral and derivative causality and if a set of variables is computable from a set of equations depends on the analytical properties of the equations in the set and the available AE solving tools, according to Proposition 1. In all three configurations it is assumed that the AE solving tools only can handle equation sets with one element, i.e. SCCs of size one.

	Initial Conditions	AE Sol. Tools	Diff. Tools
$\mathcal{C}_1$	no	scalar equations	yes
$\mathcal{C}_2$	yes	scalar equations	no
$\mathcal{C}_3$	yes	scalar equations	yes

Table 1: Configurations of the algorithm

7.3 Results

By using an implementation of the engine model in MATLAB/SIMULINK, and a MATLAB implementation of Algorithm 1, the proposed method was applied to the engine model. The implementation utilizes the fact discussed in Section 5.3 and hence as a first step, all MSO sets are computed. This step was achieved with the toolbox described in [Frisk *et al.*, 2006].

In total, 90 MSO sets were found in the engine model. In Table 2 it is shown in which of the MSO sets a variable set matching could be found with the different configurations of the algorithm. With configuration $\mathcal{C}_1$, a variable set matching could only be found in one of the MSO sets, with configuration $\mathcal{C}_2$ in four of the MSO sets, and with configuration $\mathcal{C}_3$ a variable set matching could be found in 35 of the 90 MSO sets.

Detailed Study of a Specific MSO Set

We will now consider one of the MSO sets where a variable set matching could be found with configuration $\mathcal{C}_3$, but not with the configurations $\mathcal{C}_1$ and $\mathcal{C}_2$. The MSO set, referred to as MSO set 4 in Table 2, contains 36 equations and 35 unknown variables. Of the 36 equations, only five are differential equations. By using an equation named e_{36} as residual equation, a variable set matching could be found. The structure of the corresponding just-determined part of MSO 4 is shown in Figure 2.

The found variable set matching contains variable sets corresponding to 32 SCCs of size one, and one SCC of size three. The SCCs are marked with a square in Figure 2. The SCC of size three contains the variable set $\{T_1, T_e, x_r\}$ and equation set $\{e_{11}, e_{12}, e_{13}\}$, which are on the form

$$
\begin{aligned}
e_{11}: &\quad f_{11}\left(\dot{x}_r, x_r, T_1, W_{ei}, p_{im}, W_t, p_{em}\right) = 0 \\
e_{12}: &\quad f_{12}\left(\dot{T}_1, T_1, T_e, x_r, T_{im}\right) = 0 \\
e_{13}: &\quad f_{13}\left(T_1, T_e, x_r, W_{eq}, p_{im}, W_t, p_{em}\right) = 0.
\end{aligned}
$$

To compute $\{T_1, T_e, x_r\}$ from $\{e_{11}, e_{12}, e_{13}\}$, a scalar equation solver implemented in MATLAB was used to compute $\dot{x}_r$ from e_{11}, $\dot{T}_1$ from e_{12}, and T_e from e_{13}. The equations $\{e_{11}, e_{12}, e_{13}\}$ could then be written on the form (5) and since the initial conditions of x_r and T_1 were known, $\{T_1, T_e, x_r\}$ were computable from $\{e_{11}, e_{12}, e_{13}\}$.

In the SCC of size one corresponding to the equation set $\{e_1\}$, the variable W_{egr} was algebraically computed by using the differentiated variable $\dot{p}_{im}$. The derivative $\dot{p}_{im}$ was computed with a smoothing spline approximate differentiator implemented in MATLAB, and propagation of measured variables. In a similar way, the variables W_t and P_c were algebraically computed from $\{e_2\}$ and $\{e_{21}\}$, respectively. This was done by using the derivatives $\dot{p}_{em}$ and $\dot{W}_t$.

Since integral causality were used to compute variables from the equation set $\{e_{11}, e_{12}, e_{13}\}$, and derivative causality to compute variables from the equation sets $\{e_1\}$, $\{e_2\}$, and $\{e_{21}\}$, it is clear that no residual generator could have been created from MSO set 4, with e_{36} as residual equation, if either integral or derivative causality had been used.

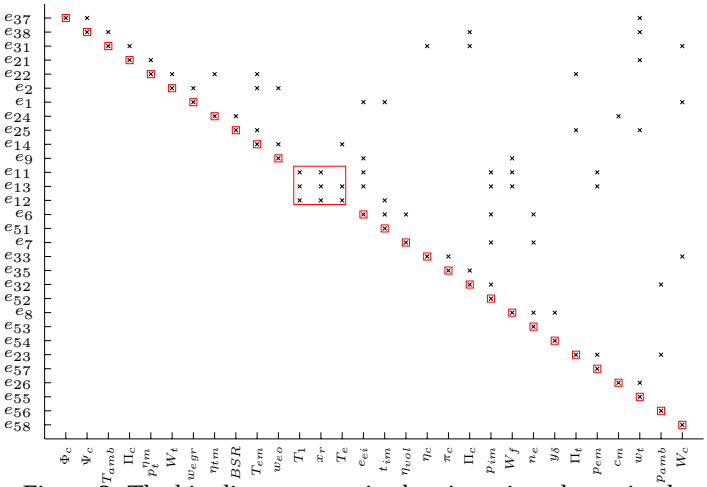


Figure 2: The bi-adjacency matrix showing a just-determined part of MSO set 4. The SCCs are marked with a square.

8 Conclusions

We have presented a mixed causality approach to residual generation, that combines integral and derivative causality and also utilizes equation system solvers and theory of differential-algebraic equations. An important part of the proposed method is a framework for computation of variables from sets of dependent differential and/or algebraic equations. In the mixed causality approach, the way a differential equation is handled depends on the context in which variables appear, the available tools for equation solving and approximate differentiating of measurements, and knowledge about initial conditions.

Complete algorithms for finding residual generators with the proposed method, as well as analysis of computability of variables from dependent differential-algebraic equation systems, have been presented. The algorithms have been applied to a model of the gas flow in an automotive diesel engine. By applying three different configurations of the algorithm, corresponding to integral and derivative causality alone and mixed causality, it has been shown that considerably more residual generators can be found in the engine model with the mixed causality approach.

	MSO set																																				
	1	2	4	5	7	8	10	11	12	20	21	23	24	25	39	40	41	43	44	45	46	51	53	57	58	60	61	62	63	74	76	85	86	88	90
$\mathcal{C}_1$	x																																		
$\mathcal{C}_2$	x																	x													x			x	
$\mathcal{C}_3$	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Table 2: A table showing in which of the MSO sets a variable set matching could be found with the different configurations of Algorithm 1.

References

[Asratian *et al.*, 1998] Armen S. Asratian, Tristan M. J. Denley, and Roland Häggkvist. *Bipartite Graphs and their Applications*. Cambridge University Press, 1998. ISBN 0-521-59345-X.

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

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

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

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

[Cormen *et al.*, 2001] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001. ISBN 0262032937.

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

[Dustegor *et al.*, 2004] D. Dustegor, V. Cocquempot, and M. Staroswiecki. Structural analysis for residual generation: Towards implementation. *Proc. of the 2004 IEEE Inter.Conf. on Control App.*, 2:1217–1222, 2004.

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

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

[Iwasaki and Simon, 1986] S. Iwasaki and H. Simon. Causality in device behavior. *Artificial Intelligence*, 29(1–3):3–32, 1986.

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

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

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

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

[Krysander *et al.*, 2008] M. Krysander, J. Åslund, and M. Nyberg. 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), 2008.

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

[Mattson and Söderlind, 1993] S. E. Mattson and G. Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM Journal of Scientific Computation*, 14(3):677–692, May 1993.

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

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

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

[Porté *et al.*, 1988] N. Porté, S. Boucheron, S. Sallantin, and F. Arlabosse. An algorithmic view at causal ordering. In *Proc. of the 2nd Inter. Workshop on Qualitative Physics QR'88*, Paris, France, 1988.

[Pulido and Alonso-Gonzlez, 2004] B. Pulido and C. Alonso-Gonzlez. 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, 2004.

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

[Svärd and Nyberg, 2008] C. Svärd and M. Nyberg. 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, Sweden, 2008.

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

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

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

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