Improved Structural Methods for Nonlinear Differential-Algebraic Equations via Combinatorial Relaxation

Taihei Oki
taihei_oki@mist.i.u-tokyo.ac.jp
Department of Mathematical Informatics, Graduate School of Information Science and Technology
University of Tokyo
Bunkyo-ku, Tokyo, Japan

ABSTRACT
Differential-algebraic equations (DAEs) are widely used for modeling of dynamical systems. In numerical analysis of DAEs, consistent initialization and index reduction are important preprocessing prior to numerical integration. Existing DAE solvers commonly adopt structural preprocessing methods, which are based on combinatorial optimization. Unfortunately, the structural methods fail if the DAE has numerical or symbolic cancellations. For such DAEs, methods have been proposed to modify them to other DAEs to which the structural methods are applicable. The modification methods are based on the combinatorial relaxation technique. Existing modification methods, however, work only for a class of DAEs that are linear or close to linear.

This paper proposes two modification methods for nonlinear DAEs: the substitution method and the augmentation method. Both methods are based on the combinatorial relaxation approach and are applicable to a large class of nonlinear DAEs. The substitution method symbolically solves equations for some derivatives based on the implicit function theorem and substitutes the solution back into the system. Instead of solving equations, the augmentation method modifies DAEs by appending new variables and equations. The augmentation method has advantages that the equation solving is not needed and the sparsity of DAEs is retained. It is shown in numerical experiments that both methods successfully modify high-index DAEs that the DAE solver in MATLAB cannot handle.

CCS CONCEPTS
• Mathematics of computing → Differential algebraic equations; • Computing methodologies → Combinatorial algorithms; Algebraic algorithms; Symbolic calculus algorithms.

KEYWORDS
differential-algebraic equations, consistent initialization, index reduction, implicit function theorem, combinatorial relaxation, combinatorial scientific computing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org. ISSAC ’19, July 15–18, 2019, Beijing, China © 2019 Association for Computing Machinery. ACM ISBN 978-1-4503-6084-5/19/07...$15.00 https://doi.org/10.1145/3326229.3326236

1 INTRODUCTION
Let $\mathbb{T} \subseteq \mathbb{R}$ be a nonempty open interval and $\Omega \subseteq \mathbb{R}^{(l+1)n}$ a nonempty open set. An $l$-th order differential-algebraic equation (DAE) of size $n$ for $x: \mathbb{T} \rightarrow \mathbb{R}^n$ is a differential equation in the form

$$F(t, x(t), \dot{x}(t), \ldots, x^{(l)}(t)) = 0,$$

where $F: \mathbb{T} \times \Omega \rightarrow \mathbb{R}^n$ is a sufficiently smooth function. DAEs have aspects of both ordinary differential equations (ODEs) $\ddot{x}(t) = \phi(t, x(t))$ and algebraic equations $G(t, x(t)) = 0$. DAEs are widely used for modeling dynamical systems such as mechanical systems, electrical circuits, and chemical reaction plants.

A fundamental and important problem in the study of DAEs is an initial value problem, which is to find a smooth trajectory $x: \mathbb{T} \rightarrow \mathbb{R}^n$ satisfying $l$ with the initial value condition

$$x(t^0) = x^*(0), \quad \dot{x}(t^0) = x^*_1, \quad \ldots, \quad x^{(l-1)}(t^0) = x^*_{l-1},$$

where $t^0 \in \mathbb{T}$ and $x^*_0, x^*_1, \ldots, x^*_{l-1} \in \mathbb{R}^n$. Unlike ODEs, an initial value problem for a DAE may not have a solution because the DAE can involve algebraic constraints, and the solution must satisfy not only the constraints but also their differentiations, called hidden constraints. While giving a consistent initial value of a DAE is an important process prior to numerical integration, this is known to be a non-trivial task [1, 14, 19].

Another important preprocessing of the numerical simulation of DAEs is an index reduction, which is a process of reducing the differentiation index [2] of a DAE. The differentiation index of a first-order DAE

$$F(t, x(t), \dot{x}(t)) = 0$$

is the minimum integer $\nu \geq 0$ such that the system of equations

$$F(t, x(t), \dot{x}(t)) = 0, \quad \frac{d}{dt} F(t, x(t), \dot{x}(t)) = 0, \ldots, \frac{d^\nu}{dt^\nu} F(t, x(t), \dot{x}(t)) = 0$$

can determine $\ddot{x}$ as a continuous function of $t$ and $x$. In other words, $\nu$ is the number of times one has to differentiate the DAE (3) to get an ODE. Intuitively, the differentiation index represents how far the DAE is from ODEs. The differentiation index of an $l$-th order DAE (1) is defined as that of the first order DAE obtained by replacing higher order derivatives of $x$ with newly introduced variables. It is commonly said to be difficult to numerically solve...
high \((\geq 2)\) index DAEs \([1, 5, 19]\). Therefore, it is important for accurate simulation of dynamical systems to convert a given DAE into a low \((\leq 1)\) index DAE.

Today, most simulation software packages for dynamical systems, such as Dymola, OpenModelica, MapleSim, and Simulink, are equipped with graph-based preprocessing methods, which we call \textit{structural methods}. These methods were first presented by Pantelides \([14]\) for the consistent initialization of DAEs. Pantelides’ method constructs a bipartite graph from DAEs’ structural information and solves an assignment problem on the bipartite graph. This method was subsequently applied to an index reduction method by dummy derivative approach of Mattsson–Söderlind \([11]\) (MSmethod). Pryce \([15]\) proposed a structural analysis method for DAEs, called the \(\Sigma\)-method, based on a variant of Pantelides’ method.

These structural methods, however, do not work even for the following simple DAE
\[
\begin{align*}
x_1 + x_2 + x_3 &= 0, \\
x_1 + x_2 &= 0, \\
x_2 + x_3 &= 0.
\end{align*}
\]
The \(\Sigma\)-method reports that the index is zero whereas it is indeed two. This is because the method cannot detect the singularity of the coefficient matrix
\[
\begin{pmatrix}
1 & 1 & 0 \\
1 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
structural methods, which ignore numerical information, may fail on some DAEs due to numerical cancellations. In general, the structural methods work only if an associated Jacobian matrix, called the \textit{system Jacobian}, is nonsingular.

In order to overcome this issue for a first-order linear DAE
\[
A_1 \dot{x}(t) + A_0 x(t) = f(t)
\]
with constant matrices \(A_0, A_1 \in \mathbb{R}^{n \times n}\) and a smooth function \(f: \mathbb{R} \to \mathbb{R}^n\), Wu et al. \([22]\) presented a method to modify \((4)\) into an equivalent DAE for which the structural methods work, using the \textit{combinatorial relaxation} algorithm by Iwata \([6]\). The combinatorial relaxation is a framework devised by Murota \([13]\) to solve linear algebraic problems by iteratively relaxing them into combinatorial optimization problems. Another modification method for linear DAEs whose coefficient matrices are \textit{mixed matrices} is given in \([7]\), where mixed matrices are matrices consisting of accurate constants and inaccurate parameters. Independently, Tan et al. \([21]\) presented modification methods, called the \textit{LC}-method and the \textit{ES}-method, for nonlinear DAEs based on the same principle as the combinatorial relaxation. All the above methods iteratively replace an equation of DAEs by a linear combination of other equations or their derivatives. These methods can deal only with DAEs close to linear DAEs; see Section 3.2 for detail. In fact, it is able to make DAEs intractable just by changing the coordinate nonlinearly.

In this paper, we present two modification methods for nonlinear DAEs, which we call the \textit{substitution method} and the \textit{augmentation method}. While the previous combinatorial relaxation methods \([6–8, 13, 21, 22]\) are designed only for a class of DAEs that is linear or close to linear, our methods are applicable to a much larger class of nonlinear DAEs. The substitution method explicitly solves equations for some derivatives based on the implicit function theorem (IFT) and then substitutes the solution back into the system. This can be seen as a generalization of linear solving in the \textit{LC}-method. To implement the substitution method, a routine to solve algebraic equations symbolically, which is expensive, is needed. The augmentation method is presented as a remedy for this drawback. In order to avoid solving equations, the augmentation method introduces new variables and equations, which are copies of existing ones. Whereas the size of the modified DAE is increased, the augmentation method does not destroy the sparsity of DAEs. We show in numerical experiments that both methods successfully modify high-index DAEs which cannot be dealt with by the standard DAE-solving library in MATLAB.

\textbf{Related work.} The substitution method repeatedly eliminates some derivatives in the DAE system. In theory of DAEs and partial differential equations (PDEs), this approach is known as \textit{“differential elimination”} or \textit{“projection”} \([3, 16, 17]\), especially for polynomial DAEs and PDEs. Maple provides \texttt{rifsimp} function that simplifies polynomial PDEs based on the differential algebra and the Gröbner basis \([10]\). From practical dynamical systems, however, non-polynomial DAEs often appear. Gear \([3]\) described a naïve index reduction method for nonlinear DAEs using a similar approach to the substitution method that iteratively eliminates derivatives using the IFT. Gear’s method appends differentiations of some equations in the DAE and thus the resultant DAE is overdetermined. Our method is advantageous in this point since it returns DAEs having the same number of equations and variables.

Takamatsu–Iwata \([20]\) proposed an index reduction method which is also named as \textit{“substitution method.”} Our substitution method is different from their substitution method in that their method deals with the first-order linear DAEs with constant coefficients based on combinatorial matrix theory, whereas our method is designed for fully nonlinear DAEs.

\textbf{Organization.} This paper is organized as follows. Section 2 summarizes structural methods for DAEs. Section 3 explains previous modification methods for DAEs based on the combinatorial relaxation. Section 4 describe the proposed methods. Section 5 illustrates an example. Section 6 shows the results of numerical experiments. Finally, Section 7 concludes this paper.

\section{Structural Methods for DAEs}

\subsection{Preliminaries}
Structural methods for DAEs utilize information on which variable each equation depends. We introduce notations to describe them.

Let \(T \subseteq \mathbb{R}\) be a nonempty open interval and \(\Omega \subseteq \mathbb{R}^{(l+1)n}\) a nonempty open set having coordinates \(x, x, \ldots, x(l)\), where \(x^{(k)} = \{x_j^{(k)}\}_{j \in C} \in \mathbb{R}^n\) for \(k \in \{0, 1, \ldots, l\}\). Here \(C\) is a set of indices with \(|C| = n\). Note that each \(x_j^{(k)}\) is regarded not as the \(k\)-th order derivative of some trajectory but as an independent variable here. Let \(f: T \times \Omega \to \mathbb{R}\) be a smooth function. For \(j \in C\) and \(k \in \{0, 1, \ldots, l\}\), the function \(f\) is said to \textit{depend on} \(x_j^{(k)}\) if the partial derivative \(\partial f/\partial x_j^{(k)}\) is not identically zero on the domain \(T \times \Omega\) of \(f\). We denote the maximum nonnegative integer \(k\) such that \(f\) depends on \(x_j^{(k)}\) by \(\sigma(f, x_j)\). If \(f\) does not depend on \(x_j^{(k)}\) for any \(k\), we assign \(\sigma(f, x_j) = -\infty\) for convenience.
The derivative $\dot{f}$ of $f$ with respect to $t$ is defined by

$$\dot{f}(t, x, \dot{x}, \ldots, x^{(l+1)}) := \frac{\partial f}{\partial t}(t, x, \dot{x}, \ldots, x^{(l+1)}) + \sum_{k=0}^{l} \frac{\partial f}{\partial x^{(k)}}(t, x, \dot{x}, \ldots, x^{(l)}) x^{(k+1)}.$$  

For $d \in \mathbb{Z}_{\geq 0}$, the $d$-th order derivative $f^{(d)}$ of $f$ is recursively defined by $f^{(0)} := f$ and $f^{(d)} := f^{(d-1)}$ for $d \geq 1$. It should be noted that the domain of $f$ is not $\mathbb{T} \times \Omega$ but $\mathbb{T} \times \Omega \times \mathbb{R}^n$ because $\dot{f}$ linearly depends on $x^{(l+1)}$. Similarly, for a nonnegative integer $d$, we regard the domain of $f^{(d)}$ as $\mathbb{T} \times \Omega^{(d)}$, where $\Omega^{(d)} := \Omega \times \mathbb{R}^d$.

The following simple proposition plays an important role in structural methods for DAEs.

**Proposition 2.1 (Griewank’s lemma [4, Section 2.2]).** Let $f: \mathbb{T} \times \Omega \rightarrow \mathbb{R}$ be a smooth function. For $j \in C$ and a nonnegative integer $d$, if $\sigma(f, x_j) \leq c$, then

$$\frac{\partial f}{\partial x_j^{(c+d)}}(t, x, \dot{x}, \ldots, x^{(l+d)}) = \frac{\partial f^{(d)}}{\partial x_j^{(c+d)}}(t, x, \dot{x}, \ldots, x^{(l+d)})$$

holds for all $(t, x, \dot{x}, \ldots, x^{(l+d)}) \in \mathbb{T} \times \Omega^{(d)}$.

We sometimes regard the domain of $\partial f^{(d)}/\partial x_j^{(c+d)}$ not as $\mathbb{T} \times \Omega^{(d)}$ but as $\mathbb{T} \times \Omega$ to simply write the equality (5) as $\partial f/\partial x_j^{(c)} = \partial f^{(d)}/\partial x_j^{(c+d)}$. In addition, it follows from Proposition 2.1 that $\sigma(f^{(d)}, x_j) = \sigma(f, x_j) + d$ holds for $j \in C$ and $d \in \mathbb{Z}_{\geq 0}$.

### 2.2 Assignment Problem

Prycz [15] introduced an assignment problem for a reinterpretation of Pantelides’ algorithm [14] as follows.

Consider a DAE (1) of size $n$ with equation index set $R$ and variable index set $C$. Let $G(F)$ denote the bipartite graph with vertex set $R \cup C$ and edge set $E(F) := \{(i, j) \in R \times C \mid \sigma(F, x_j) > -\infty\}$. An edge subset $M \subseteq E(F)$ is called a matching if the ends of edges in $M$ are disjoint. A perfect matching is a matching of size $n$. We set the weight $c_e$ of an edge $e = (i, j) \in E(F)$ by $c_e = c_{ij} := \sigma(F, x_j).

The assignment problem on $G(F)$ is the following problem $P(F)$:

$$P(F) \begin{array}{l}
\text{maximize} & \sum_{e \in M} c_e \\
\text{subject to} & M \subseteq E(F) \text{ is a perfect matching on } G(F).
\end{array}$$

The dual problem $D(F)$ of $P(F)$ is expressed as follows:

$$D(F) \begin{array}{l}
\text{minimize} & \sum_{j \in C} q_j - \sum_{i \in R} p_i \\
\text{subject to} & q_j - p_i \geq c_{ij}, \quad ((i, j) \in E(F)), \\
& p_i \in \mathbb{Z}, \quad (i \in R), \\
& q_j \in \mathbb{Z}, \quad (j \in C).
\end{array}$$

It can be shown from the duality theorem that $D(F)$ has an optimal solution if and only if $G(F)$ has a perfect matching. Let

$$\hat{\delta}(F) := \text{the optimal value of the problem } D(F).$$

This equals the optimal value of $P(F)$ due to the strong duality. If $D(F)$ has no optimal solution, we assign $\hat{\delta}(F) := -\infty$. The Hungarian method [9] efficiently solves $P(F)$ and $D(F)$.

A system Jacobian $D = (D_{i,j})_{(i \in R, j \in C)} : \mathbb{T} \times \Omega \rightarrow \mathbb{R}^{n \times n}$ of $F$ with respect to a feasible solution $(p, q)$ of $D(F)$ is a matrix defined by

$$D_{i,j} := \frac{\partial F_i}{\partial x_j^{(q_j - p_i)}}$$

for $i \in R$ and $j \in C$. The last equality in (6) for $(i, j)$ with $q_j - p_i \geq 0$ is due to Proposition 2.1. The equality also holds for $(i, j)$ with $q_j - p_i < 0$ by regarding $\partial F_i/\partial x_j^{(q_j - p_i)}$ as an identically zero function.

### 2.3 Validity Condition for Structural Methods

Prycz’s $\Sigma$-method [15] uses the assignment problem to determine an equation system for obtaining a consistent initial value. Mattsson–Söderlind’s method [11] (MS-method) reduces the index of DAEs in a structural way based on the dummy derivative approach. The validity of these structural methods are established as follows.

**Theorem 2.2 ([11, Section 3.2],[15, Theorems 4.2, 5.2]).** For a DAE (1), suppose that $D(F)$ has an optimal solution $(p, q)$ and let $D$ be the system Jacobian of (1) with respect to $(p, q)$. If there exists a consistent point $(t^*, X^*)$ of (1) at which $D$ is nonsingular, then $(t^*, X^*)$ can be found by the $\Sigma$-method. In addition, the MS-method returns an equivalent DAE whose index is at most one around $(t^*, X^*)$.

Theorem 2.2 states that the structural methods fail if the system Jacobian $D$ is identically singular. Indeed, two out of nine DAEs in the test set for IVP (initial value problem) solvers [12] have identically singular system Jacobians. It is also known that the nonsingularity of the system Jacobian is easily destroyed by a simple linear transformation of DAEs as follows.

**Example 2.3.** Let $F = 0$ be a DAE and $D$ the system Jacobian with respect to a optimal solution $(p, q)$. Suppose $p_i \neq p_j$ for some distinct $i, j \in R$. Take a “generic” matrix $A \in \mathbb{R}^{n \times n}$, that is, each entry in $A$ is chosen at random. Then $F = 0$ and $AF = 0$ are equivalent DAEs since $A$ is nonsingular (with probability 1), whereas $AF = 0$ does not have a nonsingular system Jacobian.

This failure is attributed to the fact that the structural methods use only combinatorial information and ignore numeric-symbolic information of DAEs assuming that nonzero entries in Jacobian matrices are generic. Then numeric-symbolic cancellations inherent in the DAEs make the system Jacobian identically singular.

### 3 DAE MODIFICATION VIA COMBINATORIAL RELAXATION

#### 3.1 Combinatorial Relaxation

The method of Wu et al. [22] modifies a first-order linear DAE (4) with constant coefficients into an equivalent linear DAE whose system Jacobian is not identically singular. This method relies on the combinatorial relaxation algorithm of Iwata [6], and all other modification methods are based on the combinatorial relaxation approach as well. The combinatorial relaxation method consists of the following three phases [6, 13, 21, 22].

**Phase 1.** Compute an optimal solution $(p, q)$ of $D(F)$. If $D(F)$ has no optimal solution, the algorithm ends with failure.

**Phase 2.** If the system Jacobian $D$ with respect to $(p, q)$ is not identically singular, return the DAE $F = 0$ and halt.
Phase 3. Modify the DAE $F = 0$ into an equivalent DAE $\hat{F} = 0$ such that $\hat{\delta}(\hat{F}) \leq \delta(F) - 1$. Go back to Phase 1.

Since $D(F)$ has an optimal solution if and only if $\hat{\delta}(\hat{F}) \geq 0$, the above process ends in at most $\ln(\delta(F))$ iterations.

A non-trivial part of the combinatorial relaxation method is only Phase 3, which modifies DAEs to decrease the value of $\hat{\delta}$. Iwata’s combinatorial relaxation algorithm modifies first-order linear DAEs with constant coefficients using strict equivalence transformations, which multiply nonsingular constant matrix to equations and variables. A combinatorial relaxation method in [7] for linear DAEs with mixed matrices employs unimodular transformations here. The unimodular transformation is a sequence of trivial equivalent transformations of DAEs that add an equation (or its derivative) to another equation. Iwata–Takamatsu’s index reduction algorithm [8] for first-order linear DAEs with constant coefficients is also based on the combinatorial relaxation and modifies DAEs using unimodular transformations.

3.2 The LC-method

The LC-method of Tan et al. [21] can be regarded as a nonlinear generalization of the method of Wu et al [22], where the difference is only the modification method in Phase 3. The modification method of the LC-method is summarized as follows.

Suppose that we have a DAE (1) and its dual optimal solution $(p, q)$ such that the system Jacobian $D$ with respect to $(p, q)$ is identically singular. First, we find a nonzero vector $(u_i)_{i \in R}$ in the cokernel of $D$, namely, $u$ is a row vector such that $uD$ is identically zero. Let $\mu := \{i \in R \mid u_i$ is not identically zero$\}$. Take $r \in \mu$ such that $p_r \leq p_1$ for all $i \in \mu$ and put $l := \sup \mu \setminus \{r\}$. Then we replace the $r$-th equation $F_r$ of the DAE by $F_{\ell r} = 0$, where

$$F_{\ell r} := u_r F_r + \sum_{i \in \mu} u_i F^{(p_i - p_r)}_i.$$

It is shown that this modification decreases the value of $\hat{\delta}$ if

$$\sigma(u_i, x_j) < q_j - p_r$$

for all $i \in \mu$ and $j \in C$ [21, Theorem 4.1]. Intuitively, (7) means that the highest order derivatives appear linearly in DAEs. For linear DAEs, (7) trivially holds since $\sigma(u_i, x_j) = -\infty$ for all $i, j$.

However, there still exist DAEs to which the LC-method cannot be applied. For example, the following DAE

$$\begin{align*}
    x_1 \dot{x}_2 - 2 \cos^2 t &= 0, \\
    \dot{x}_2^2 + x_1 + x_2 - 4 \cos^4 t - 3 \sin t - 2 &= 0
\end{align*}$$

(8)

given in [21, Section 5.3] cannot be dealt with by the LC-method. While [21] also presents another modification method called the ES-method, it is also inapplicable to (8). Indeed, the following example, which is a nonlinear generalization of Example 2.3, demonstrates that many DAEs are converted to other DAEs not satisfying (7) by nonlinearly changing the coordinate of the codomain of $F$.

Example 3.1. Let $F = 0$ be a DAE and $D$ the system Jacobian with respect to a dual optimal solution $(p, q)$. Suppose $p_i \neq p_i$ for some distinct $i_1, i_2 \in R$ as in Example 2.3. Let $\psi : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be a "generic" nonlinear diffeomorphism such that $\psi(w) = 0$ if and only if $w = 0$. Then $F(\cdot) = 0$ is equivalent to $\psi(F(\cdot)) = 0$, whereas the latter DAE have identically singular system Jacobian but cannot be handled by the LC-method.

The claim in Example 3.1 implies that (7) holds only if we have a special coordinate of the codomain space of $F$. Therefore, from a geometrical point of view, it is natural and important to devise a modification method for such "heavily nonlinear" DAEs.

4 PROPOSED METHODS

In this section, we describe proposed methods for nonlinear DAEs, called the substitution method and the augmentation method. Both methods are used in Phase 3 of the combinatorial relaxation framework. We omit all the proofs due to space limitations.

4.1 Substitution Method

Let $T \subseteq \mathbb{R}$ be a nonempty open interval and $\Omega \subseteq \mathbb{R}^{(l + 1)n}$ a nonempty open set. For a matrix $A$, we denote by $A[I, J]$ the submatrix of $A$ indexed by a row subset $I$ and a column subset $J$. The input of the substitution method is a DAE (1) with real analytic function $F : T \times \Omega \rightarrow \mathbb{R}^n$ such that

1. $G(F)$ has a perfect matching,
2. for any square submatrix $D[I, J]$ of the system Jacobian $D$ with respect to a dual optimal solution, if $D[I, J]$ is not identically singular on $T \times \Omega$, then there exists a consistent point of (1) at which $D[I, J]$ is nonsingular, and
3. $D$ is identically singular.

The smoothness assumption on $F$ is needed to avoid technical difficulties. We remark that (I2) is just a part of a sufficient condition for which the substitution method works, and it suffices in practice to check the condition only for a few submatrices of $D$ that are needed in the method.

The substitution method modifies the DAE (1) into another DAE

$$F_{\text{sub}}(t, x, \dot{x}, \ldots, x^{(l + n)}) = 0$$

of size $n$ such that

(1) $F_{\text{sub}}$ is a real analytic function defined on a nonempty open subset $T^{\text{sub}} \times \Omega_{\text{sub}} \subseteq T \times \Omega^{(l + n)}$ with $r \leq l n$,

(2) the resulting DAE (9) is locally equivalent to the input DAE (1), and

(3) $\hat{\delta}(F_{\text{sub}}) \leq \hat{\delta}(F) - 1$.

See Lemma 4.1 for the precise meaning of "locally equivalent."

We first introduce notations needed to describe the method. Let $R$ and $C$ be the equation index set and the variable index set of the DAE (1), respectively. For $I \subseteq R$, let $F_I$ denote a subvector $(F_i)_{i \in I}$ of $F$ indexed by $I$. Similarly, for $J \subseteq C$, let $x_J$ denote a subvector $(x_i)_{i \in J}$ of $x$ indexed by $J$. Let $p$ and $q$ be the vectors of variables in $D(F)$. In addition, we use the following notations

$$F^{(p_i)} := \left( F^{(p_i)}_i \right)_{i \in I}, \quad x^{(q_j)} := \left( x^{(q_j)}_j \right)_{j \in J}, \quad \frac{\partial F^{(p_i)}}{\partial x^{(q_j)}} := \left( \frac{\partial F^{(p_i)}}{\partial x^{(q_j)}} \right)_{i, j \in I, J}$$

for $I \subseteq R$ and $J \subseteq C$.

Here we start to describe the method. Let $D$ be the system Jacobian of (1) with respect to an optimal solution $(p, q)$ of $D(F)$ and suppose that $D$ is identically singular. We regard $D$ as a matrix over the quotient field $F$ of the ring of real analytic functions on $T \times \Omega$. 

318

Contributed Paper

ISSAC ’19, July 15–18, 2019, Beijing, China
The substitution method first finds \( r \in R, I \subseteq R \setminus \{ r \} \) and \( J \subseteq C \) with \(|I| = |J| = m\) such that

(C1) \( D[I, J] \) is nonsingular,
(C2) \( \text{rank} D[I \cup \{ r \}, C] = m \), and
(C3) \( p_i \leq p_j \) for \( i \in I \).

Here, both the nonsingularity in (C1) and the rank in (C2) are in the sense of those of matrices over \( \mathbb{F} \). Namely, these can be rewritten as

(C1') \( D[I, J] \) is not identically singular, and
(C2') the maximum size of a submatrix in \( D[I \cup \{ r \}, C] \) that is not identically singular is \( m \).

The algorithm for finding such \((r, I, J)\) is described in Section 4.3.

Let \((r, I, J)\) be a triple satisfying (C1)–(C3). Define \( S := R(I \cup \{ r \}) \) and \( T := C \setminus J \). Then (1) is divided into three subsystems as follows:

\[
\begin{align*}
F_r(t, x, \ldots, x^{(l)}) &= 0, \\
F_l(t, x, \ldots, x^{(l)}) &= 0, \\
F_S(t, x, \ldots, x^{(l)}) &= 0, \\
\end{align*}
\]

where \( \partial \) is the vector of ones with appropriate dimension. In addition, the submatrix \( D[I, J] = \partial F_I / \partial x_J^{(1)} = \partial F_I / \partial x_J^{(1)} \) is not identically singular on \( T \setminus \Omega \) from (C1). Therefore, by the assumption (12), there exists a point \((\tilde{t}, \tilde{X}) \in T \times \Omega^k\) such that

\[
F_I^{(p)}(\tilde{t}, \tilde{X}) = 0 \quad \text{and} \quad \partial F_I^{(p-1)} / \partial x_J^{(q-p-1)}(\tilde{t}, \tilde{X}) \text{ is nonsingular, where } \kappa := \max_{l \in I} p_l - p_r. \]

Then via the IFT, we can solve an equation

\[
F_I^{(p-1)}(t, x, \ldots, x^{(l+k)}) = 0
\]

for \( x_J^{(q-p-1)} \) as

\[
x_J^{(q-p-1)} = \varphi(t, x, \ldots, x^{(l+k)}),
\]

where \( \varphi \) is a function that does not depend on \( F_I^{(q-p-1)} \).

Finally, we substitute the right-hand side of (12) into \( F_I^{(q-p-1)} \) in the first equation \( F_r = 0 \) of (10). The modified DAE (9) is

\[
\begin{align*}
F_r^{\text{sub}}(t, x, \ldots, x^{(l+k)}) &= 0, \\
F_l(t, x, \ldots, x^{(l)}) &= 0, \\
F_S(t, x, \ldots, x^{(l)}) &= 0,
\end{align*}
\]

where \( F_r^{\text{sub}} \) is a function obtained from \( F_r \) by substituting (12). The domain \( \Omega_{\text{sub}} \times \Omega_{\text{aug}} \) of (13) is determined via the IFT.

Here we give the precise description of (S2) as follows.

**Lemma 4.1.** Consider a DAE (1) satisfying (I1)–(I3). Let \( x : \Omega_{\text{sub}} \to \mathbb{R}^n \) be a sufficiently smooth trajectory satisfying the initial value condition (2) for \((t', X') \in \Omega_{\text{sub}} \times \Omega_{\text{aug}}\). Then there exists an open subinterval \( I \subseteq \Omega_{\text{sub}} \) containing \( t' \) such that \( x \) is a solution of (1) on \( I \) if and only if \( x \) is a solution of (13) on \( \Omega_{\text{sub}} \).

**Theorem 4.2.** For a DAE (1) satisfying (I1)–(I3), the substitution method outputs a DAE (9) satisfying (S1)–(S3).

### 4.2 Augmentation Method

Next we describe another proposed modification method for nonlinear DAEs, which we call the augmentation method. The input of the augmentation method is a nonlinear DAE (1) satisfying the conditions (I1)–(I3), where \( F : T \times \Omega \to \mathbb{R}^n \) is a real analytic function again. Instead of solving equations symbolically, the augmentation method augments the size of the DAE by introducing a new variable vector \( y \) and attaching new equations. Formally, the augmentation method modifies (1) into a DAE

\[
F_\text{aug}(t, x, \ldots, x^{(l+k)}, y) = 0
\]

of size \( n + m \) such that

(A1) \( F_\text{aug} \) is a real analytic function defined on a nonempty open subset \( \Omega_{\text{aug}} \times Y \subseteq T \times \Omega^k \times \mathbb{R}^m \) with \( \kappa \leq \text{in} \) and \( m \leq n - 1 \),

(A2) the resulting DAE (14) is locally equivalent to (1), and

(A3) \( \delta(F_\text{aug}) \leq \delta(F) - 1 \),

where \( \delta(F) \) is the precise meaning of "locally equivalent."

The substitution method and the augmentation method are the same except for the last modification process. The overlapping part is described here briefly. Let \( R \) and \( C \) be the equation index set and the variable index set of the input DAE (1), respectively. Let \((p, q)\) be an optimal solution of \( D(F) \) and \( D \) denote the system Jacobian with respect to \((p, q)\). We first find \( r \in R, I \subseteq R \setminus \{ r \} \) and \( J \subseteq C \) satisfying (C1)–(C3). Define \( \kappa := \max_{l \in I} p_l - p_r \). Then via the IFT, we can solve an equation

\[
F_\text{aug}(t, x, \ldots, x^{(l+k)}, y) = 0
\]

for \( (\tilde{t}, \tilde{X}) \in T \times \Omega^k \) such that

\[
\begin{align*}
(y_{j}^\ast & \in \Omega_{\text{aug}} & \text{if } j \in J, \ k = q_j - p_r), \\
(x_{j}^\ast & \in \Omega_{\text{aug}} & \text{if } j \in K, \ k = q_j - p_r), \\
(y_{j} & \in \Omega_{\text{aug}} & \text{otherwise})
\end{align*}
\]

where \( \Omega_{\text{aug}} \) is a subset of \( \Omega_{\text{aug}} \) hereafter. For \( x \in \Omega \) and a vector \( y = (y_j)_{j \in J'} \) with index set \( J' \), let \( \psi_{\text{aug}}(X, y) \) be a vector of \( \mathbb{R}^n \) such that

\[
\begin{align*}
(\psi_{\text{aug}}(X, y))_{(j, k)} & := \begin{cases}
 y_j & \text{if } j \in J, k = q_j - p_r, \\
 x_{j, k} & \text{if } j \in K, k = q_j - p_r, \\
 0 & \text{otherwise}
\end{cases},
\end{align*}
\]

where \( (\cdot, \cdot) \) is a projection function. For each \( i \in I \), we define a function

\[
F_\text{aug}^\ast(t, x, \ldots, x^{(l+k)}, y) := F_i^{(p_i - p_r)}(t, \psi_{\text{aug}}(X, y)),
\]

where \( X = (x, \ldots, x^{(l+k)}) \). Namely, \( F_\text{aug}^\ast \) is obtained by replacing \( x_j^{(q_j - p_r)} \) in \( F_i^{(p_i - p_r)} \) with a variable \( y_j \) for \( j \in J \) and with a constant \( x_j^{(q_j - p_r)} \) for \( j \in J \). Put \( F_\text{aug}^\ast := \{ F_i^{\text{aug} } \}_{i \in I'} \). We also define

\[
F_\text{aug}(t, x, \ldots, x^{(l+k)}, y) := F_i(t, \psi_{\text{aug}}(X, y))
\]

in the same way.

The output (14) of the augmentation method is

\[
\begin{align*}
F_r^{\text{aug}}(t, x, \ldots, x^{(l+k)}, y) &= 0, \\
F_l(t, x, \ldots, x^{(l)}) &= 0, \\
F_S(t, x, \ldots, x^{(l)}) &= 0, \\
F_\text{aug}(t, x, \ldots, x^{(l+k)}, y) &= 0
\end{align*}
\]

(15)
with unknown function \((x(t), y(t))\) of \(t\). The domain \(\mathcal{D}^{\text{aug}} \times \Omega^{\text{aug}}\) of (15) is given by \(\mathcal{D}^{\text{aug}} := \mathcal{D}^{\text{sub}}\) and \(\Omega^{\text{aug}} := \{(X, y) \in \mathcal{D}^{\text{sub}} \times \mathbb{R}^m\} \setminus \psi_{\subseteq}(X, y) \in \mathcal{D}^{\text{sub}}\}. The DAE (15) is obtained by copying some equations (or their derivatives), relabelling variables and substituting constants. Hence if the original DAE contains only a few variables in each equation, so does (15). Thus the augmentation method retains the sparsity of DAEs.

The following is the precise description of (A2).

**Lemma 4.3.** Consider a DAE (1) satisfying (I1)–(I3). Let \(x: \mathcal{D}^{\text{aug}} \rightarrow \mathbb{R}^n\) be a sufficiently smooth trajectory satisfying the initial value condition (2) for \((t^*, X^*) \in \mathcal{D}^{\text{aug}} \times \Omega^{\text{aug}}\). Then there exists an open subinterval \(I \subseteq \mathcal{D}^{\text{aug}}\) containing \(t^*\) such that the following two statements are equivalent:

1. \((x, y)\) is a solution of (1) on \(I\), and
2. there uniquely exists a trajectory \(y: I \rightarrow \mathbb{R}^m\) such that \((x, y)\) is a solution of (15) on \(I\).

**Theorem 4.4.** For a DAE (1) satisfying (I1)–(I3), the augmentation method returns a DAE (14) satisfying (A1)–(A3).

### 4.3 Algorithm for Finding \((r, I, J)\)

Let \(D\) be a singular \(n \times n\) matrix over a field \(\mathbb{F}\) with row index set \(R\) and column index set \(C\), and \(p = (p_i)_{i \in R}\) be an integer vector indexed by \(R\). By the setting in Sections 4.1 and 4.2, \(\mathbb{F}\) is the quotient field of the ring of analytic functions on \(T \times \Omega\). We give an algorithm, which uses arithmetic operations over \(\mathbb{F}\), to find \(r \in R, I \subseteq R \setminus \{r\}\) and \(J \subseteq C\) satisfying the conditions (C1)–(C3). First, by column operations, transform \(D\) into \(D' = (D'_{i,j})_{i \in R, j \in C}\) in the form

\[
D' := H \begin{bmatrix} B & C \setminus B \\ R \setminus H & O \end{bmatrix},
\]

where \(H \subseteq R\) and \(B \subseteq C\) with \(|H| = |B| = \text{rank}\ D\). Here \(I\) and \(O\) are the identity and zero matrices of appropriate size, respectively, and \(*\) indicates an arbitrary matrix. Let \(h: B \rightarrow H\) denote the bijection defined by \(h(j) = i\) if and only if \(D'_{i,j} \neq 0\) for \(j \in B\).

Next choose \(\ell \in R \setminus H\) arbitrarily. Note that \(R \setminus H\) is nonempty since \(D'\) is singular. Put \(Z := \{(i) \cup \{h(j)\} \mid j \in B, D'_{i,j} \neq 0\} \subseteq R\). Then we take \(r \in Z\) such that \(p_r \leq p_i\) for all \(i \in Z\). Put \(I := Z \setminus \{r\}\) and choose \(J \subseteq C\) such that \(D[I, J]\) is nonsingular. The existence of \(J\) is guaranteed by the following lemma.

**Lemma 4.5.** The submatrix \(D[Z, C]\) is not of full-row rank and \(D[I, C]\) is of full-row rank for any proper subset \(I \subsetneq Z\).

The following theorem holds from the construction of \((r, I, J)\) together with Lemma 4.5.

**Theorem 4.6.** For a singular matrix \(D \in \mathbb{F}^{n \times n}\), the above algorithm returns \((r, I, J)\) satisfying the conditions (C1)–(C3).

This algorithm uses \(O(n^3)\) arithmetic operations over \(\mathbb{F}\).

### 5 EXAMPLE

We demonstrate our modification methods to the index-1 DAE (8). In Phase 1 of the combinatorial relaxation, we find a dual optimal solution \(p = (0, 0)\) and \(q = (1, 1)\). The system Jacobian with respect to \((p, q)\) is 

\[
D = \begin{pmatrix} 
\frac{\partial x_2}{\partial \bar{x}_1} & \frac{\partial x_1}{\partial \bar{x}_2} \\
\frac{\partial x_2}{\partial \bar{x}_2} & \frac{\partial x_1}{\partial \bar{x}_1} 
\end{pmatrix},
\]

which is identically singular. In Phase 3, we find \(r = 2, I = \{1\}\) and \(J = \{1\}\). We first demonstrate the substitution method applied to (8). By solving \(F_1 = 0\) in (8) for \(x_1\), we get

\[
\bar{x}_1 = -\frac{2\cos^2 t}{x_2}
\]

unless \(x_2 = 0\). Then (16) is substituted into the second equation in (8). The resulting DAE of the substitution method is

\[
\begin{align*}
F_1 : & \quad \bar{x}_1 \bar{x}_2 - 2\cos^2 t = 0, \\
F_2 : & \quad \bar{x}_1 + x_2 - 3\sin t - 2 = 0.
\end{align*}
\]

The system Jacobian of (17) corresponding to a dual optimal solution \(p' = (0, 1), q' = (1, 1)\) is

\[
D' = \begin{pmatrix} 
\frac{\partial x_2}{\partial \bar{x}_1} & \frac{\partial x_1}{\partial \bar{x}_2} \\
1 & 1
\end{pmatrix},
\]

which is identically singular. Thus we are done.

Next, we show the modification of (8) by the augmentation method. Let \(y'\) be a new variable corresponding to \(x_1\) and \(\xi \in \mathbb{R}\) an arbitrary nonzero constant corresponding to \(x_2\). The augmentation method modifies the DAE (8) into the following DAE

\[
\begin{align*}
F_1 : & \quad \bar{x}_1 \bar{x}_2 - 2\cos^2 t = 0, \\
F_2 : & \quad \bar{y}'_1 \xi^2 + x_1 + x_2 - 4\cos^4 t - 3\sin t - 2 = 0, \\
F_3 : & \quad \bar{y}'_1 \xi - 2\cos^2 t = 0
\end{align*}
\]

with unknown function \((x_1, x_2, y')\). A pair of \((p', q') = (0, 1, 1)\) is a new dual optimal solution, and the system Jacobian is

\[
D' = \begin{pmatrix} 
\frac{\partial x_2}{\partial \bar{x}_1} & \frac{\partial x_1}{\partial \bar{x}_2} \\
1 & 1
\end{pmatrix},
\]

with unknown function \((x_1, x_2, y')\). The DAEs have identically singular system Jacobians, and thus the MS-method cannot be applied to them.

### 6 NUMERICAL EXPERIMENTS

We applied our methods in practice to the following two DAEs. The DAEs have identically singular system Jacobians, and thus the MS-method cannot be applied to them.

(a) Nonlinearly modified pendulum (index-3):

\[
\begin{align*}
x_4 & = x_2 \cos x_3 = 0, \\
x_3 & = -2\cos^2 x_3 \sin x_3 + g = 0, \\
x_2 & = x_1^2 + x_3^2 - 1 = 0, \\
x_1 & = \tanh(x_1 - x_4) = 0, \\
x_2 \sin x_3 & + x_2 x_3 \cos x_3 - x_3 = 0
\end{align*}
\]

with \(g = 9.8\). This DAE is obtained by nonlinearly changing the variable \((y, z, \lambda, v_y, v_z)\) of a simple pendulum DAE

\[
\begin{align*}
v_y - y\lambda &= 0, \\
v_z - z\lambda + g &= 0, \\
2y^2 + z^2 - 1 &= 0, \\
y - v_y &= 0, \\
z - v_z &= 0
\end{align*}
\]
by \((y, z, \lambda, y_q, u_2) = (x_1, x_2 \sin x_3, x_2 \cos x_3, x_4, x_5)\). In addition, we equivalently changed the fourth equation \(x_1 - x_4 = 0\) to \(\tan h(x_1 - x_4) = 0\).

(b) Ring modulator (index-2):
\[
\begin{align*}
\dot{x}_1 &= (x_1/R - x_8 + 0.5x_{10} - 0.5x_{11} - x_{14})/C = 0, \\
\dot{x}_2 &= (x_2/R - x_9 + 0.5x_{12} - 0.5x_{13} - x_{15})/C = 0, \\
\dot{x}_{10} &= q(U_{D1}) + q(U_{D4}) = 0, \\
\dot{x}_{11} &= q(U_{D2}) + q(U_{D3}) = 0, \\
\dot{x}_{12} &= q(U_{D1}) - q(U_{D4}) = 0, \\
\dot{x}_{13} &= q(U_{D2}) - q(U_{D4}) = 0, \\
\dot{x}_7 &= (x_7/R_p - q(U_{D1}) - q(U_{D2}) + q(U_{D3}))/(C_p) = 0, \\
\dot{x}_8 &= x_8/L_h = 0, \\
\dot{x}_9 &= x_9/L_h = 0, \\
\dot{x}_{10} &= (-0.5x_1 + x_3 + R_p x_7)/L_s = 0, \\
\dot{x}_{11} &= (0.5x_1 - x_4 + R_p x_{11})/L_s = 0, \\
\dot{x}_{12} &= (-0.5x_2 + x_5 + R_p x_{12})/L_s = 0, \\
\dot{x}_{13} &= (0.5x_2 - x_6 + R_p x_{13})/L_s = 0, \\
\dot{x}_{14} &= (x_1 + (R_g1 + R_1)x_{14} - U_{in1}(t))/L_s = 0, \\
\dot{x}_{15} &= (x_2 + (R_e + R_g1)x_{15})/L_s = 0,
\end{align*}
\]

where
\[
\begin{align*}
U_{D1} &= x_3 - x_5 - x_7 - U_{in2}(t), \\
U_{D2} &= -x_4 + x_6 - x_7 - U_{in2}(t), \\
U_{D3} &= x_4 + x_5 + x_7 + U_{in2}(t), \\
U_{D4} &= -x_3 - x_6 + x_7 + U_{in2}(t),
\end{align*}
\]
\[
q(U) = y(\exp^{\delta U} - 1), U_{in1}(t) = 0.5 \sin 2000\pi t, U_{in2}(t) = 2 \sin 20000\pi t
\]

with positive parameters \(C, C_p, L_h, L_s, \lambda_{s1}, \lambda_{s2}, \lambda_{s3}, \gamma, \delta, R, R_p, R_g1, R_g2, R_g3, R_1\), and \(R_e\). The DAE represents an electrical network describing the behavior of a ring modulator [12].

The above formulation is obtained by setting \(C_q = 0\) in the original problem. Other parameters are set to the same values as in [12].

We computed numerical solutions of DAEs (a) and (b) after performing the following three kinds of preprocessing: (i) no index reduction, (ii) reduce the index by the MS-method after applying the substitution method, and (iii) reduce the index after using the augmentation method; what the default DAE solver in MATLAB can do for the DAEs is only (i). For the rank computation of system Jacobian and the process of finding \((r, I, J)\), we adopted the fast symbolic Gaussian elimination algorithm by Sasaki–Murao [18]. The numerical solutions were computed by ode15i in MATLAB, which is a variable-step variable-order (VSVO) solver for index-0 or 1 DAEs based on the backward differentiation formulas (BDFs) [19]. The parameters of ode15i were set to the default values: \(\text{AbsTol} = 10^{-6}\) and \(\text{RelTol} = 10^{-3}\). We explicitly provided Jacobian matrices to ode15i through the Jacobian option. All the computation were performed on MATLAB R2019a.

**Numerical Results.** Figure 1 shows numerical solutions obtained in the experiments. Without index reduction, ode15i could not yield a numerical solution of the index-3 DAE (a). As for the index-2 DAE (b), ode15i first output a numerical solution but stopped at \(t \approx 0.00045\). In this way, we can see that it is difficult for ode15i to solve high-index DAEs stably.

Both the substitution method and the augmentation method successfully modified the DAEs (a) and (b). With the preprocessing by substitution method, ode15i had stopped at \(t \approx 1.278810\) and \(t \approx 0.000501\) for (a) and (b), respectively. This is because our methods ensure the equivalence of DAEs only in an area in which \(D[I, J]\) is identically nonsingular. Nevertheless, by re-modifying the DAE at that time, our program could go on the computation. This process, called the *dynamic pivoting*, can be applied by re-choosing \((r, I, J)\) such that \(D[I, J]\) is far from singular at the present point. The dynamic pivoting is also known to be needed in the MS-method [11].

In the first application of the substitution method to the DAE (b), we used the following values of \((p, q, r, I, J)\):
\[
p = (0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0), \quad q = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
\]
\[
r = 11, \quad I = \{3, 4, 5, 6, 10, 12, 13\}, \quad J = \{3, 5, 6, 10, 11, 12, 13\}.
\]

There exists another possible values of \((p, q, r, I, J)\) as follows:
\[
p = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), \quad q = (1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1),
\]
\[
r = 5, \quad I = \{3, 4, 6\}, \quad J = \{3, 4, 5\}.
\]

The values in (19) seem to be superior to (18) in that the substitution method need to differentiate equations for (18) but not for (19), and the size \(|l|\) of the equation system (11) for (19) is smaller than that of (18). However, the equation-solving routine in MATLAB could not return a solution of (11) for (19); the solution cannot be represented by a combination of the elementary functions indeed. On the other hand, the equation system (11) for (18) is linear because \(x_3, x_5\), and \(x_6\) appear linearly in the differentiation of the 3–6th equations. Thus the substitution method works for (18).

**7 CONCLUSION**

In this paper, we have presented two modification methods for nonlinear DAEs, called the substitution method and the augmentation method. Based on the combinatorial relaxation approach, both methods modify DAEs into other DAEs for which the structural preprocessing methods work. The substitution method modifies DAEs using the IFT and has a merit that it retains the size of DAEs. The augmentation method modifies DAEs by appending new variables and equations, and is advantageous in that it does not require an equation-solving routine and keeps DAEs’ sparsity. In numerical experiments, both methods successfully modified DAEs that the standard DAE solver in MATLAB cannot handle. The experiments, however, have also shown that the success of the substitution method depends on the selection of the values \((p, q, r, I, J)\). A method to select the best \((p, q, r, I, J)\) is left for further investigation.

**ACKNOWLEDGMENTS**

The author thanks Satoru Iwata for careful reading and helpful comments, Mizuyo Takamatsu for discussions, and Yuichi Yoshida for recommending me to submit the paper to ISSAC. This work was supported by JST CREST Grant Number JPMJCR14D2, Japan, Grant-in-Aid for JSPS Research Fellow Grant Number JP18J22141, Japan and JST ACT-i Grant Number JPMJPR18U9, Japan.
Figure 1: Numerical solutions of the experiments. In the $2 \times 3$ array of graphs, each row corresponds to a DAE and each column corresponds to a modification method. Dotted lines in (a-i) and (b-i) indicate that the DAEs were re-modified at those times.

REFERENCES

[1] K. E. Brenan, S. L. Campbell, and L. R. Petzold. 1996. *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*. SIAM, Philadelphia.

[2] S. L. Campbell and C. W. Gear. 1995. The index of general nonlinear DAEs. *Numer. Math.* 72, 2 (1995), 173–196.

[3] C. W. Gear. 1988. Differential-algebraic equation index transformations. *SIAM J. Sci. Statist. Comput.* 9, 1 (1988), 39–47.

[4] A. Griewank. 1989. On automatic differentiation. In *Mathematical Programming: Recent Developments and Applications*. Kluwer Academic Publishers, Dordrecht, 83–108.

[5] E. Hairer and G. Wanner. 1996. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer-Verlag, Berlin, Heidelberg.

[6] S. Iwata. 2003. Computing the maximum degree of minors in matrix pencils via combinatorial relaxation. *Algorithmica* 36, 4 (2003), 331–341.

[7] S. Iwata, T. Oki, and M. Takamatsu. 2018. Index reduction for differential-algebraic equations with mixed matrices. In *Proc. of the 8th SIAM Workshop on Combinatorial Scientific Computing (CSC’18)*. Bergen, Norway, 45–55.

[8] S. Iwata and M. Takamatsu. 2018. Index reduction via uninomial transformations. *SIAM J. Matrix Anal. Appl.* 39, 3 (2018), 1135–1151.

[9] H. W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2 (1955), 83–97.

[10] Maplesoft, a division of Waterloo Maple, Inc. 2019. Overview of the Rif Command Set Version 1.1 - Maple Programming Help (2019). Retrieved April 11, 2019 from https://www.maplesoft.com/support/help/maple/view.aspx?path=RifTools/Rif

[11] S. E. Mattsson and G. Söderlind. 1993. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM J. Sci. Comput.* 14, 3 (1993), 677–692.

[12] F. Mazzia and C. Magherini. 2008. Test set for initial value problem solvers. Technical Report. Department of Mathematics, University of Bari. URL: https://archimede.dm.uniba.it/~testset/.

[13] K. Murota. 1995. Computing the degree of determinants via combinatorial relaxation. *SIAM J. Comput.* 24, 4 (1995), 765–796.

[14] C. C. Pantelides. 1988. The consistent initialization of differential-algebraic systems. *SIAM J. Sci. Statist. Comput.* 9, 2 (1988), 213–231.

[15] J. D. Pryce. 2001. A simple structural analysis method for DAEs. *BIT* 41, 2 (2001), 384–394.

[16] X. Qin, L. Yang, Y. Feng, B. Bachmann, and P. Fritzson. 2018. Index reduction of differential algebraic equations by differential Dixon resultant. *Appl. Math. Comput.* 328 (2018), 189–202.

[17] G. J. Reid, P. Lin, and A. D. Wittkopf. 2001. Differential elimination - Completion algorithms for DAE and PDAE. *Studies in Applied Mathematics* 106, 1 (2001), 1–45.

[18] T. Sasaki and H. Murao. 1982. Efficient Gaussian elimination method for symbolic determinants and linear systems. *ACM Trans. Math. Software* 8, 3 (1982), 277–289.

[19] L. F. Shampine. 2002. Solving $0 = F(t, y(t), y'(t))$ in Matlab. *J. Numer. Math.* 10, 4 (2002), 291–310.

[20] M. Takamatsu and S. Iwata. 2008. Index reduction for differential-algebraic equations by substitution method. *Linear Algebra and Its Applications* 429, 8–9 (2008), 2268–2277.

[21] G. Tan, N. S. Nedialkov, and J. D. Pryce. 2017. Conversion methods for improving structural analysis of differential-algebraic equation systems. *BIT* 57, 3 (2017), 845–865. arXiv:1505.03445.

[22] X. Wu, Y. Zeng, and J. Cao. 2013. The application of the combinatorial relaxation theory on the structural index reduction of DAE. In *Proc. of the 12th International Symposium on Distributed Computing and Applications to Business, Engineering & Science* IEEE, London, UK, 162–166.