



## STRUCTURAL ANALYSIS FOR EQUATION BASED SIMULATORS

Rafael de P. Soares and Argimiro R. Secchi

Grupo de Integração, Modelagem, Simulação, Controle e Otimização de Processos (GIMSCOP)  
Departamento de Engenharia Química, Universidade Federal do Rio Grande do Sul (UFRGS)  
R. Eng. Luis Englert, s/n. Campus Central. CEP: 90040-040 - Porto Alegre - RS - BRASIL,  
E-MAIL: {rafael, arge}@enq.ufrgs.br

**Palavras Chaves:** Modelagem, Análise estrutural, Orientação a Objetos, Índice diferencial.

**Resumo:** Differential-algebraic equations (DAE) systems arise naturally from modelling many dynamic systems and are more difficult to handle than ordinary differential equation (ODE) systems. For instance, it is well known that difficulty arises when DAE's are solved with inconsistent initial values. Furthermore, the solution of high-index problems requires specially designed integration methods or index reduction, which are usually limited. In this work, alternatives for initialising and solving general high-index DAE systems are studied and a new algorithm for index analysis and reduction is introduced.

### 1 INTRODUCTION

Differential-algebraic equations (DAE) systems arise naturally when modelling many dynamic systems. General implicit DAE system can be represented by

$$F(t, y, y') = 0 \quad (1)$$

where  $t$  is the independent variable (usually the time),  $F \in \mathcal{R}^n$ ,  $y$  and  $y' \in \mathcal{R}^n$  are the dependent variables and its derivatives with respect to  $t$ , respectively. If (1) can be written in the *explicit* form

$$y' = f(t, y) \quad (2)$$

with the same state variables  $y$ , then (1) actually is a system of *implicit* Ordinary differential equations (ODE). In this work we are interested in problems for which such conversion is impossible or not desirable. There are several reasons to consider (1) directly (Brenan *et al.*, 1989). One of our particular interest, is when the system of equations is automatically generated by a simulation program where the models are written in an object-oriented language. In this work the process simulator EMSO (Soares and Secchi, 2003) was used to write

models, develop and test methods, and to obtain numerical solutions.

The basic difference between (1) and (2) is the possibility of a singular Jacobian of  $F$  with respect to  $y'$ , denoted by  $\partial F / \partial y'$  or  $F_{y'}$ . If this is the case, the DAE cannot be rewritten as an ODE with same variables  $y$ . Then, in order to convert such DAE systems to an ODE, state transformations or differentiation of the equations are needed. Usually the property known as *index* is used to *measure the distance* between a particular DAE and its equivalent ODE formulation. The minimum number of times that all or part of a DAE (1) must be differentiated with respect to  $t$  to determine  $y'$  as a continuous function of  $y$  and  $t$  is defined as the *differential index*  $\nu$ <sup>1</sup>.

Obviously, accordingly to the above definition, (2) has a differential index  $\nu = 0$ . DAE systems with  $\nu \leq 1$  and  $\nu > 1$  are known as low- and high-index systems, respectively.

The solution of general low-index DAE systems is, in principle, not much more difficult than the solution of ODE systems but the initialisation of

<sup>1</sup> Other definitions of index can be found in Unger *et al.* (1994).



## OKTOBER FÓRUM 2005 – PPGEQ

such systems still can pose problems (Pantelides, 1988). Furthermore, none of the currently available numerical techniques work for all high-index DAE's (Brenan *et al.*, 1989).

In this work alternatives for initialising and solving general high-index DAE systems in form (1), coming from the equation-oriented general process simulator EMSO, are studied. Moreover, a new structural algorithm for index characterisation and reduction is presented and the alternatives are compared when applied in typical problems.

### 2 ALTERNATIVES

Low-index DAE systems can generally be solved numerically by slightly modified ODE codes. However, for high-index systems these methods may converge poorly, they may converge to wrong solutions, or they may not converge at all (Brenan *et al.*, 1989). For these systems there are basically three general approaches:

1. Manually modify the model to obtain a lower-index equivalent model;
2. Integration by specifically designed high-index solvers;
3. Apply automatic index reduction algorithms in order to obtain a lower-index equivalent model.

#### 2.1 Manual Modifications

The index of DAE systems describing dynamic systems is strongly affected by precise formulation of the problem, i.e., by the choice of independent variables and equations. Since these choices can often be made rather arbitrarily, in the work of Lefkopoulos and Stadtherr (1993) it is presented an algorithm for selecting, whenever possible, the independent equations and variables that lead to the formulation of an index-one DAE. But this algorithm is not concerned with index determination, it only seeks to detect whether the index is one or if it is higher than one. If no index-one formulation can be found, the algorithm does not attempt to formulate a minimum index problem or perform index reduction. In a similar way the work of Gani and Cameron (1992) explores how different assumptions such as equilibrium or incompressibility can affect the index of DAE systems.

These approaches can be quite useful for finding lower-index equivalent models for small problems. But clearly they are not suitable to be embedded in a simulation package, because could be quite difficult or impossible to automate it for large-scale models.

#### 2.2 Specifically Designed High-Index Solvers

As already stated, there are no numerical method capable of handling all classes of high-index DAE. Presently available numerical codes are basically modified ODE solvers (Unger *et al.*, 1995), and can be divided in two main groups:

- Solvers for high-index problems of restricted problem structure (e.g. in Hessenberg form or semi-explicit);
- Solvers for high-index problems limited for problems with index three or less.

Again, the former group of solvers is not suitable for our purposes because could be impossible to convert general DAE systems to the required stricter structure.

The last group of solvers appears to be promising for our application, mainly because problems with index higher than three are quite uncommon to appear in mathematical models of chemical processes. Representing this class of solvers, the codes PSIDE (Lioen *et al.*, 1998) and MEBDFI (Abdulla and Cash, 1999) were considered. Although no restriction in problem structure is imposed when using these codes, it is required to inform as input the *index* of each variable. In order to exemplify how the indices of the variables are determined, consider the following system of equations

$$\begin{aligned}y' &= f(t, y, z) \\ z' &= k(t, y, z, u) \\ 0 &= g(t, y)\end{aligned}\quad (3)$$

where  $f_z$ ,  $k_u$ , and  $g_y$  are full rank in the neighbourhood of the solution. Then, (3) has index  $\nu = 3$  and the variable  $y$ ,  $z$ , and  $u$  are of index 1, 2, and 3, respectively. A more detailed description of this procedure can be found at Lioen *et al.* (1998). Should be stressed that (3) was used for demonstration purposes only and the cited codes can handle general problems as (1) but require a consistent initial condition in order to advance in the solution.

From this discussion, it is clear that, in order to implant these solvers in a general equation-oriented process simulator, two problems arise:

- how to automate the determination of the index of the variables
- how to determine a consistent initial condition

#### 2.3 Index Analysis and Reduction

Historically, several works have addressed the problem of index characterisation and reduction. Most of these works consider only the structure of



the problem disregarding numerical values. From this consideration comes the property known as structural index  $\nu_{str}$  which is analogue to the differential index  $\nu$  but considering structural algebra.

Duff and Gear (1986) suggested a structural analysis to identify index-two semi-explicit systems. Gear (1988) conceptually proposed a symbolical algorithm for index reduction, and based on this idea Bachmann *et al.* (1990) presented an algorithm for index reduction of linear systems. Unger *et al.* (1995) extended this algorithm enabling the characterisation and index reduction of general DAEs. Pantelides (1988) introduced a graph-theoretical algorithm addressing the problem of consistent initialisation of general DAEs which can be used for index reduction. The last two approaches are of similar complexity but Pantelides' algorithm is fairly well suited for an implementation in sparse matrix representation (Unger *et al.*, 1995) and therefore was preferred in this work. The basic idea of the Pantelides' algorithm is to determine minimally structurally singular (MSS) subsets of equations and then differentiate it to form an augmented system aiming at a system where no more MSS subsets can be found. In Costa Jr. *et al.* (2001) the Pantelides' algorithm was applied to index-one reduction using automatic differentiation.

#### Algorithm for index reduction

It is well known that structural analysis of DAEs is limited (see Pantelides, 1988; Unger *et al.*, 1995; and Reissig *et al.*, 2000, for further discussion on this subject). For instance, if a DAE system is structurally singular, Pantelides' approach will keep differentiating the same subset of equations ad infinitum (Pantelides, 1988). Besides, the structural algorithm stops when an index-one equivalent

system is obtained and cannot reduce further the index.

In order to remove these two limitations the following modifications to the Pantelides' algorithm are suggested:

- start the search for MSS subsets with respect to all variable derivatives and not only with respect to the variable derivatives that are part of the original system;
- when a MSS subset with respect to the variable derivatives is detected, check if it is singular with respect to the entire set of variables. If this is the case then the analysis is finished because the system is structurally singular.

The first modification makes the algorithm capable to reduce the index until zero, the second checks if the system is structurally singular, avoiding infinite loop when analysing singular systems.

Although it is not necessary to fully understand the new index analysis algorithm presented in this work, a reading on the original version of Pantelides' algorithm could be useful and can be found at Pantelides (1988).

In order to introduce the algorithm for index analysis, consider the following DAE:

$$\begin{aligned} f_1(x, u_1, u_2) &= 0 \\ f_2(x, x', y_1) &= 0 \\ f_3(x, y_2) &= 0 \end{aligned} \quad (4)$$

In optimal control context, we could imagine a problem where  $y_1$  and  $y_2$  are desired outputs (specified optimal profiles),  $u_1$  and  $u_2$  are the control actions and  $x$  is an unknown (state variable). Using the notation found in Figure 1, this problem can be represented in a bipartite graph as shown in Figure 2.


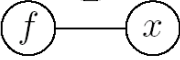



Symbol	Description
	Marked equation $f$ (it will be differentiated in next step)
	Line between equation $f$ and variable $x$
	Matching line between equation $f$ and variable $x$
	Line between equation $f$ and algebraic variable $x$
	Matching line between equation $f$ and algebraic variable $x$

Figure 1. Notation for representing DAE systems as bipartite graphs.



## OKTOBER FÓRUM 2005 – PPGEQ

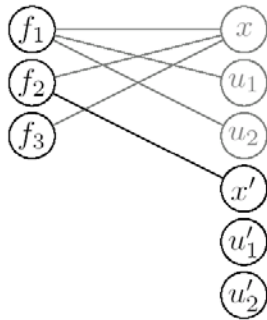


Figure 2. Graph for the DAE system (4).

The graph in Figure 2 is called  $G(E, V, L)$ , where  $E$  is the set of equations, left hand side of the bipartite

Algorithm 1. Pseudocode for the AugmentMatching algorithm.

---

```
Bool AugmentMatching(E, V, L, E_i, flagdiff, M)
1   for each line of L which contains E_i and is not in M:
1.1   if flagdiff is true:
1.2     if the variable of line is diff:
1.3       add line to M and return true
1.4     else: add line to M and return true
2   for each line of L which contains E_i and is in M:
2.1   if the equation of line is not marked:
2.2     set E_j the equation of line and mark E_j
2.3     if AugmentMatching(E, V, L, E_j, flagdiff, M):
2.4       unmark E_j, add line to M and return true
3   return false
```

---

The basic idea behind the Algorithm 1 is to find a matching for the given equation node  $E_i$ , returning *true* if a matching could be found and *false* otherwise. In this algorithm a variable is *diff* if it is the higher order derivative of that variable in  $V$ . The basic difference between this algorithm and the one proposed by Pantelides (1988) are the steps 1.1-1.4 and the additional flag *flagdiff*. With this modification one can select if the wanted matching is with respect only to the higher order derivatives or not. The Pantelides' version is always with respect to the higher order derivatives because the low order ones are removed from the graph during the execution of the procedure.

Using Algorithm 1, Algorithm 2 can analyse and reduce the index of a general DAE system as (1), given its graph  $G(E, V, L)$ .

graph, and  $V$  the set of variables. There is a line  $E_i - V_j$  if the equation  $E_i$  contains the variable  $V_j$ , the set of lines is called  $L$ . A matching  $M$  is a subset of  $L$  for which each line has a unique  $E_i$  and  $V_j$ . The cardinality of a matching is the number of lines that it covers. For more details about graph theory please refer to Diestel (2000).

The graph-theoretical algorithm for index analysis proposed in this work is based on a breadth-first search for finding a maximum cardinality matching, herein after called *AugmentMatching*. Each call to *AugmentMatching*, as stated in Algorithm 1, will increase the cardinality of a current matching  $M$ .

If the given graph  $G(E, V, L)$  is structurally non singular, Algorithm 2 returns *true* and *index* contains the index of the original system of equations. Structural singularities are detected at step 2.3 (which is missing on Pantelides' version). Another improvement of the new version of the algorithm is for a solvable system, when the algorithm stops at step 3, the number of variables uncovered by  $M$  is exactly the number of dynamic degrees of freedom and that set of variables is a structurally feasible set of initial conditions. Moreover, if the Dulmage-Mendelsohn decomposition is applied to the final association, the under-constrained component consists of the entire set of feasible initial conditions.



Algorithm 2. Pseudocode for the index analysis/reduction algorithm.

---

```
Bool IndexAnalysis(E, V, L, index)
1   create an empty matching M and set index to zero
2   for each equation in E not in M:
2.1   if not AugmentMatching(E, V, L, equation, true, M):
2.2   if not AugmentMatching(E, V, L, equation, false, M):
2.3   return false
3   if there is no equation marked: return true
4   else: increase index by one
5   for each equation marked:
5.1   add the derivative of equation to E
5.2   for each variable in equation:
5.3   add the derivative of the variable to V
5.4   add a line to L between the variable derivative and
      the equation derivative
6.   back to 2
```

---

Algorithms 1 and 2 are easy to implement in software packages, where  $G(E, V, L)$  is basically the Jacobian pattern of the system of equations. Should be stressed that the application of the algorithm actually does not require differentiating any equation or variable. Applications of the Algorithm 2 are presented in Section 4.

### 3 INITIALISING GENERAL DAE SYSTEMS

It is well known that difficulties arise when DAE systems are solved with inconsistent initial values and may cause solution failures of many popular DAE solvers (Wu and White, 2001). Therefore, consistent initial values are crucial for obtaining the numerical solution. Actually, most often failures in solving a DAE system occur or have the source in initialisation, for both low- and high-index systems. In Section 2 alternatives for solving (advance in solution) of general high-index DAE systems were studied. But in order to advance in solution, numerical codes require consistent initial values. Popular solvers comes with routines based on approximate methods which can determine the variable derivatives  $y'$  given the values of all variables  $y$  (Brennan, *et. al.*, 1989). But if the user gives inconsistent values for  $y$  the solver will fail, because not all variables  $y$  are independent. Moreover, these routines cannot determine  $y'$  for a given  $y$  for high-index problems even if the given  $y$  is consistent.

In this work an alternative approach that can be used to initialise DAE systems of any index is proposed as follows:

- analyse the system of equations with the index analysis algorithm, Algorithm 2;
- if the system is not structurally singular, the number of dynamic degrees of freedom (number of initial conditions) together with one feasible set of initial conditions are determined;
- considering  $y'$  just as new variables, a system of non-linear algebraic (NLA) equations can be built using *all* equations (original plus the derivatives coming from the algorithm) and *all* variables ( $y$  and  $y'$ );
- the NLA, at this point, will be under-constrained by a number of equations equal to the number of dynamic degrees of freedom of the original DAE system. Appending initial conditions as new equations to this system, a square NLA problem will be obtained, which can be solved with usual Newton's like methods.

It should be noted that with this approach, the initial conditions can be general algebraic or differential equations and not just fixed values of some variables as usual. A comparison of this approach with the ones implemented in popular solvers is shown in Section 4.





#### 4 APPLICATIONS AND DISCUSSION

##### 4.1 Index Analysis algorithm

In this section Algorithm 2 is applied step by step to one structurally singular DAE system to demonstrate one example where the algorithm detects the singularity (the Pantelides' algorithm does not terminate for this problem). Further, the algorithm is applied to the well-known pendulum problem.

##### Uncontrollable DAE

Consider the DAE system (4) and its graph in Figure 1. After the execution of one analysis step of Algorithm 2 (execution from step 1 to 4) we will have the graph in Figure 3.

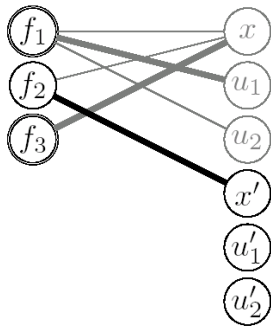


Figure 3. Graph for the DAE system (4) after one analysis step.

In the graph in Figure 3 the bold lines are the lines covered by the current matching  $M$ . The black and

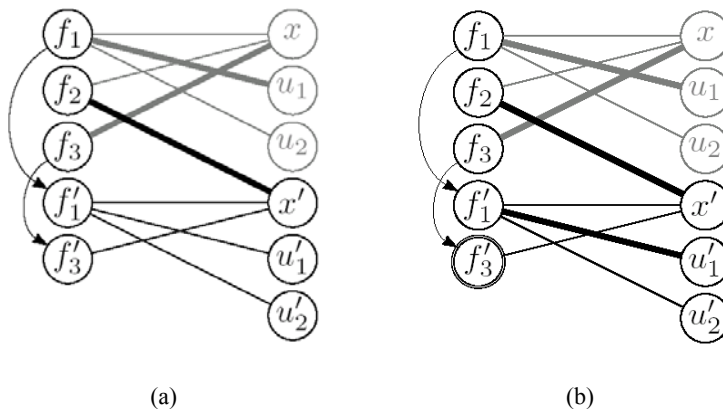


Figure 4. Graph for the DAE system (4): (a) after one analysis step with the derivatives; and (b) when the singularity is detected.

grey bold lines are lines added to  $M$  at steps 2.1 and 2.2 of Algorithm 2, respectively. The grey variables are the *nondiff* variables. From this figure it is easy to conclude that  $f_2$  was connected at step 2.1 and  $f_1$  and  $f_3$  were connected at step 2.2. During the execution of Algorithm 2, Algorithm 1 is called several times and, for this example, it has *marked* the equations  $f_1$  and  $f_3$  (step 2.2 of Algorithm 1). The marked equations need to be differentiated and added to the graph (steps 5 to 6 of Algorithm 2). Then new nodes  $f_1'$  and  $f_3'$  are added to the graph and connected to the derivatives of the variables that are connected with the original equations, resulting in the graph shown in Figure 4(a).



OKTOBER FÓRUM 2005 – PPGEQ

Going to the next analysis step, step 2.1 will successfully add to  $M$  the line  $f_1'-u_1'$  but it will fail to add a line for  $f_3'$ . Once step 2.1 fails, step 2.2 is called, and for equation  $f_3'$  it will fail to add a matching too. Then the Algorithm stops (step 2.3) returning *false*. The graph for this problem after two analysis steps (when the singularity is detected) is shown in Figure 4 (b).

*Pendulum Problem*

Consider the well-known pendulum model in Cartesian co-ordinates:

$$\begin{aligned} x' &= w \\ y' &= z \\ z' &= T.x \end{aligned} \tag{5}$$

$$\begin{aligned} w' &= T.y - g \\ x^2 + y^2 &= L^2 \end{aligned}$$

The graph for this problem can be seen in Figure 5(a). One analysis step of Algorithm 2 for this graph will result in the graph shown in Figure 5(b). As can be seen in Figure 5(b) the algorithm could not find a matching for  $f_5$  with high order derivative variables (step 2.1 of Algorithm 2) but step 2.2 successfully add to  $M$  a connection for it. At this point,  $f_5$  is marked for differentiation and then  $f_5'$  is added to the graph and a new analysis step is done, resulting in Figure 6(a).

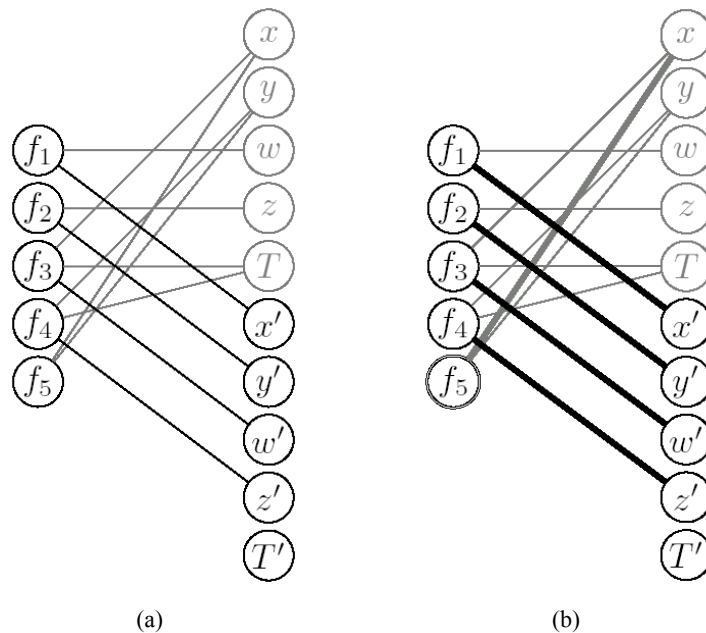


Figure 5. Graph for the pendulum problem: (a) for DAE system (5) and (b) after one analysis step of Algorithm 2.

Again  $f_5'$  could not be directly connected to a high order derivative variable, but at step 2.2 of Algorithm 2,  $f_5'$  could be exchanged with  $f_1$  (step 2.3 of Algorithm 1) and then connected to  $x'$ . During the procedure of finding a connection for  $f_5'$  several equations were marked for differentiation (step 2.2 of Algorithm 1), namely  $f_1$ ,  $f_2$  and  $f_5'$ . After more two steps the algorithm finishes with the graph show in Figure 6(b). As can be seen in this figure, the variables  $y$  and  $z$  are not covered by

the matching  $M$  then this problem has two dynamic degrees of freedom, hence it requires only two arbitrary initial conditions and the uncovered variables are good options for it. Applying the Pantelides algorithm to the DAE (5) a solution is found one step before the algorithm proposed in this work and the final graph has less equations and variables. On the other hand the resulting graph has no information about feasible



initial conditions nor can be used to initialise the

problem with the method presented in Section 3.

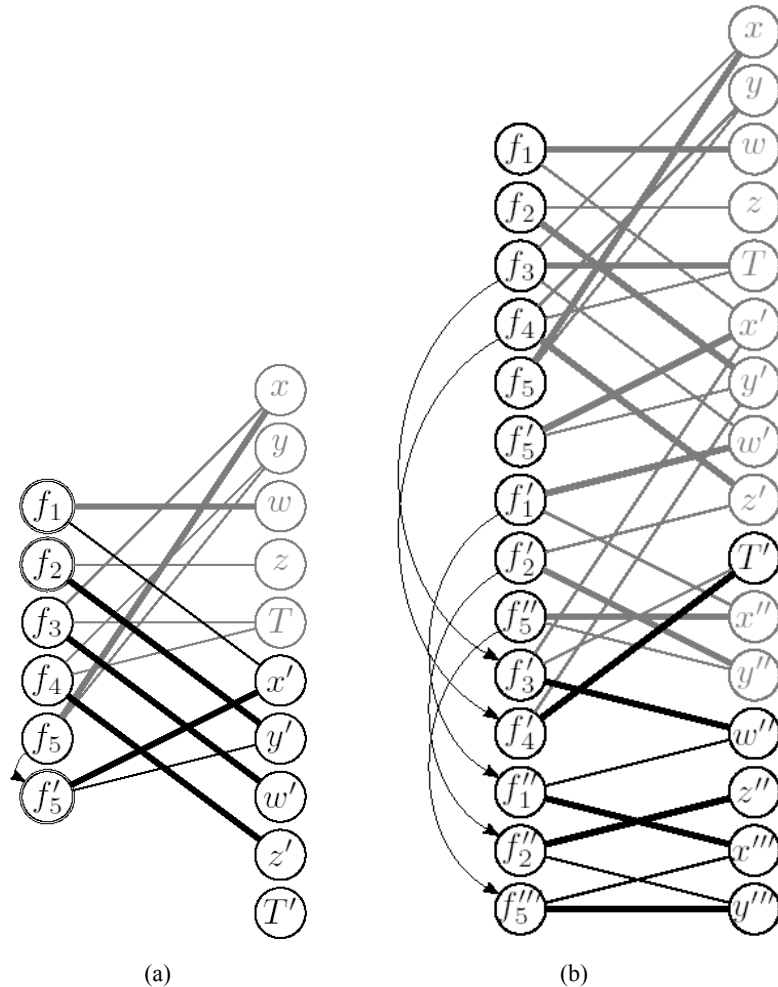


Figure 6. Graph for the pendulum problem: (a) after two analysis steps; and (b) after Algorithm 1 ends.

#### 4.2 Initialisation of general DAE systems

##### Initialisation of low-index DAE systems

In order to compare the approach for initialising general DAE systems proposed in Section 3 with usual techniques, the galvanostatic process of a thin film of nickel hydroxide electrode proposed by Wu and White (2001) was considered. The model due to the authors, is an index-one DAE system with two variables ( $y_1$  and  $y_2$ ) and one dynamic degree of freedom. Then, in order to initialise the system, one of the variables could be arbitrary specified as initial condition and the other three (the other

variable and the derivative of both) must be determined by the initialisation procedure. As most numerical techniques, initialisation codes need initial guesses. Table 1 shows the convergence range of the initial guess of one variable when the other is given as initial condition of popular codes for solving DAE systems.

As can be seen in Table 1 the proposed method is fairly more robust. Moreover, the proposed method can initialise high-index problems depending non-linearly on the derivatives where usual methods fail. On the other hand the method requires the symbolic differentiation of the equations.





Table 1. Convergence range in initialising an index-one problem.

Solver	$y_2$ convergence range, given $y_1=0.05$ as initial condition	$y_1$ convergence range, given $y_2=0.38$ as initial condition
DASSL*	0.321 – 0.370	0.071 – 0.352
LIMEX*	0.318 – 0.377	0.056 – 0.418
RADAU5*	0.348 – 0.352	0.143 – 0.190
DAEIS*	-0.974 – 1.663	0.000 – 1.000
<b>Proposed Method</b>	<b>-2.70 – 2.66</b>	<b><math>-\infty - \infty</math></b>
Consistent value	0.35024	0.15513

\*Data from Wu and White (2001).

#### 4.2.1 Initialisation of high-index DAE systems

The proposed approach was successfully used to initialise high-index problems as the classical index-three pendulum model Eq (5) or the index-three batch distillation column due to Logsdon and Biegler (1993). The classical codes tested in Section 4.2.1 fail to initialise high-index DAE systems, then a comparison was not possible.

#### 4.3 Solving high-index DAE systems

Once the initialisation step terminates successfully one can try to advance in solution. As discussed before, codes designed for general high-index

problems are limited to systems with index at most three and require an index analysis to discover the index of the variables. Another option is to apply an index reduction algorithm, as discussed in Section 2.3, to discover which equations need to be differentiated and then actually differentiate the equations to get a reduced index equivalent system which can be handled by codes designed for low-index problems. Indeed the last option is a more general, because, in principle, it can be applied for any kind of problem. On the other hand, using an index reduced system the solution falls down in the well known “drift-off” effect (see Figure 7).

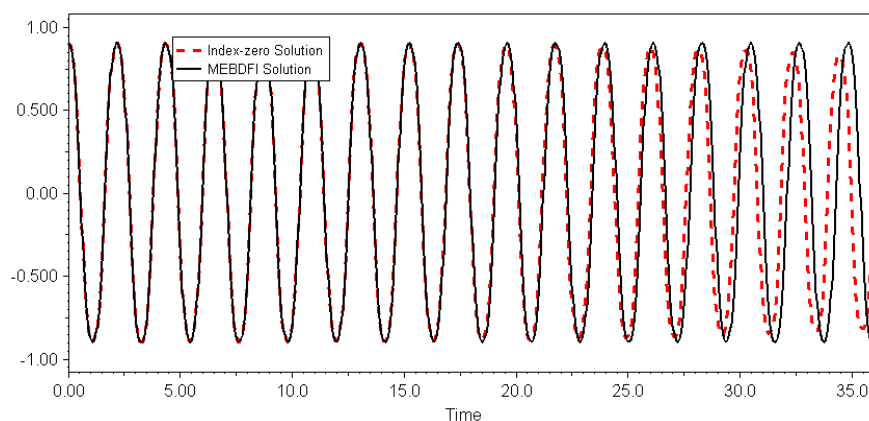


Figure 7. Numerical solution for the position of the index-three pendulum problem.

As can be seen in Figure 7, the numerical solution using an index-reduced system yield acceptable results only for low integration times. For long time the algebraic constraints (cable length in the pendulum problem), which are not directly considered in the index-reduced system, starts to be not respected.

## 5 CONCLUSIONS

Alternatives to initialise and solve general high-index DAE systems coming from the dynamic

process simulator EMSO were studied. A new algorithm for index analysis and reduction was introduced together with an approach for initialising both low- and high-index problems. The proposed method has shown success in initialising high-index problems and proved to be more robust than the classical codes when initialising low-index systems. The algorithm consists in a modification of the well-known Pantelides’ algorithm, improving the detection of singularities where the original



## OKTOBER FÓRUM 2005 – PPGEQ

algorithm does not finishes. Furthermore, when the execution of the algorithm ends, a consistent set of initial conditions is discovered.

For solving high-index DAE systems in software packages it was concluded that, at present, the better alternative is to use codes specially designed for high-index problems as MEBDFI. Unfortunately, these codes can only handle systems with index at most three and still requires a prior index characterisation. On the other hand, the index reduction approach can be applied to systems of any index but the solution can presents degradation. How to reduce this degradation (the “drift-off” effect) is the subject of ongoing research.

Both algorithms, Pantelides’ and the modified version proposed in this work, are implemented in the process simulator EMSO, in such simulation package the derivatives of the equations are obtained by a built-in symbolic differentiation code.

### REFERENCES

- Abdulla T. J. and J. R. Cash, 1999, A Package for the Solution of Initial Value Problem for system of Implicit Differential Algebraic Equations, [www.netlib.org/ode/mebdfi.f](http://www.netlib.org/ode/mebdfi.f).
- Bachmann, R., L Brüll, T. Mrziglod, and U. Pallaske, 1990, On Methods for Reducing the Index of Differential Algebraic Equations, *Comp. Chem. Engng*, 14, 1271-1273.
- Brenan K. E., S. L. Campbell and L. R. Petzold, 1989, Numerical Solution of Initial Value Problems in Differential-Algebraic Equations. North-Holland, New York.
- Costa Jr., E.F., R.C. Vieira, A.R. Secchi, E.C. Biscaia (2001), Automatic Structural Characterization of DAE Systems, ESCAPE-11, Kolding, Denmark, pp. 123-128.
- Diestel R. (2000), Graph Theory – Eletronic Edition 2000, Graduate Texts in Mathematics, vol. 173, Springer-Verlag.
- Duff I. S. and C. W. Gear, 1986, Computing the structural index, *SIAM J. Alg. Disc. Meth.* 7, 594-603.
- Lefkopoulos A. and M. A. Stadtherr, 1993, Index analysis of unsteady-state chemical process systems – I, *Comp. Chem. Engng* 17, 399-413.
- Lioen W. M., J.J.B. Swart and W. A van der Veen, PSIDE Users' Guide, 1998, <http://www.cwi.nl/archive/projects/PSIDE/>.
- Logsdon J. S and L. T. Biegler, 1993, Accurate Determination of Optimal Reflux Policies for the Maximum Distillate Problem in Batch Distillation, *Ind. Eng. Chem. Res.*, 32, No 4, 692-700.
- Gani R. and T. Cameron, 1992, Modelling for dynamic simulation of Chemical Processes – the index problem, *Chem. Engng Sci.* 47, 1311-1315.
- Gear C. W., 1988, Differential-algebraic equation index transformation. *SIAM J. Sci. Stat. Comput.*, 9, 39-47.
- Pantelides C. C., 1988, The consistent initialization of differential-algebraic systems, *SIAM J. Sci. Stat. Comput.* 9, 213-231.
- Reissig G., W. S. Martinson and P. Barton, 2000, Differential-Algebraic Equations of Index 1 may have an Arbitrarily High Structural Index, *SIAM J. Sci. Comput.*, 21, No 6, 1987-1990.
- Soares R. P. and A. R. Secchi, 2003, EMSO: A new Environment for Modelling, Simulation and Optimisation, ESCAPE-13, 947-952.
- Unger J., A. Kröner and W. Marquardt, 1994, Structural Analysis of Differential-Algebraic Equation Systems – Theory and Application, *Computers Chem. Engng*, 19, No 8, pp 867-882.
- Wu B., R.E. White (2001), An initialization subroutine for DAEs solvers: DAEIS, *Computers Chem. Engng*, 25, pp 301-311.