Journal of Applied Mathematics and Physics, 2016, 4, 39-47
Published Online January 2016 in SciRes. http://www.scirp.org/journal/jamp
http://dx.doi.org/10.4236/jamp.2016.41007

Scientific Research Publishing

Efficient Numerical Methods for Solving Differential Algebraic Equations

Ampon Dhamacharoen

Department of Mathematics, Burapha University, Chonburi, Thailand
Email: ampon@buu.ac.th

Received 19 November 2015; accepted 8 January 2016; published 11 January 2016

Abstract

This research aims to solve Differential Algebraic Equation (DAE) problems in their original form, wherein both the differential and algebraic equations remain. The Newton or Newton-Broyden technique along with some integrators such as the Runge-Kutta method is coupled together to solve the problems. Experiments show that the method developed in this paper is efficient, as it demonstrates that implementation of the method is not difficult, and such method is able to provide approximate solutions with ease within some desired accuracy standards.

Keywords

Differential-Algebraic Equations, Newton-Broyden Method, Index-2 Hessenberg DAE

1. Introduction

A differential algebraic Equation (DAE) is an equation involving an unknown function and its derivatives. A DAE in its most general form is given by the following:

$$\boldsymbol{F}\left(t,\boldsymbol{x}(t),\boldsymbol{y}(t),\boldsymbol{y}'(t)\right)=\boldsymbol{0},\quad t\in[a,b] \tag{1}$$

where R is the set of real number, $\boldsymbol{F}:\mathrm{R}\times\mathrm{R}^m\times\mathrm{R}^n\times\mathrm{R}^n\to\mathrm{R}^{m+n}$, $\boldsymbol{x}(t)\in\mathrm{R}^m$, $\boldsymbol{y}(t)\in\mathrm{R}^n$ and $\boldsymbol{y}'(t)\in\mathrm{R}^n$. In this form, the relationship between the variables and derivatives may be implicit. In some systems, the equations may be written in the explicit form of derivatives, as follows:

$$\boldsymbol{y}'(t)=\boldsymbol{f}\left(t,\boldsymbol{x}(t),\boldsymbol{y}(t)\right) \tag{2a}$$

$$\boldsymbol{g}\left(t,\boldsymbol{x}(t),\boldsymbol{y}(t)\right)=\boldsymbol{0} \tag{2b}$$

How to cite this paper: Dhamacharoen, A. (2016) Efficient Numerical Methods for Solving Differential Algebraic Equations. *Journal of Applied Mathematics and Physics*, **4**, 39-47. http://dx.doi.org/10.4236/jamp.2016.41007

where $\boldsymbol{f}: \mathrm{R} \times \mathrm{R}^m \times \mathrm{R}^n \to \mathrm{R}^n$, This set of equations is called a semi-explicit DAE system.

Differential algebraic equations arise in the mathematical modeling of a wide variety of problems found in engineering and science, such as multi-body and flexible body mechanics, electrical circuit design, optimal control, incompressible fluids, molecular dynamics, chemical kinetics and chemical process control [1]-[5].

DAEs can be transformed into ODE problems via differentiation. The number of differentiations needed in the transforming process is called the differentiation index. This number can describe some characteristics of the problem. In general, the higher the index of a DAE, the more difficulties one can expect in its numerical solution [1]-[3] [6].

Although DAEs can be transformed into an explicit ODE so that it can be solved using the methods of ODE, there are still many numerical methods that can solve DAE directly. In the DAE solvers software, the numerical approaches for the solution of DAEs can be divided roughly into two classes: a) direct discretizations of the given system; and b) methods which involve a reformulation (e.g. index reduction), combined with a discretization. Direct discretizations are easier to use, but are limited in their utilitisation to essentially index-1, index-2 Hessenberg, and index -3 Hessenberg DAE systems, while a reformulation may be costly, and it may also require more input from the user and involve more user intervention [3] [6].

In this research, we will place emphasis on constructing an algorithm for solving a semi-explicit DAE. The approach employed is to firstly solve the system of algebraic equations, and subsequently solve the differential equations using the derived information. The Newton-Broyden method plays a key role in solving algebraic equations, since it performs almost as well as the Newton method, but requires less energy, and in addition, it also outperforms other methods of the same order of convergence [7]-[9].

2. The Newton-Broyden Method

The method, which was first proposed by Dhamacharoen in 2011 [7], [8], aims to solve the equation of the form:

$$\boldsymbol{F}\left(\boldsymbol{u}\left(\boldsymbol{x}\right), \boldsymbol{x}\right) = \boldsymbol{0} \tag{3}$$

The Newton scheme of this problem is:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \left[\boldsymbol{F}_{\boldsymbol{u}}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right)\boldsymbol{u}'\left(\boldsymbol{x}_i\right) + \boldsymbol{F}_{\boldsymbol{x}}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right)\right]^{-1} \boldsymbol{F}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right), \quad i = 0,1,2,\cdots. \tag{4}$$

Replacing $\boldsymbol{u}'\left(\boldsymbol{x}_i\right)$ by $\boldsymbol{D}_i$, gives

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \left[\boldsymbol{F}_{\boldsymbol{u}}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right)\boldsymbol{D}_i + \boldsymbol{F}_{\boldsymbol{x}}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right)\right]^{-1} \boldsymbol{F}\left(\boldsymbol{u}\left(\boldsymbol{x}_i\right), \boldsymbol{x}_i\right), \quad i = 0,1,2,\cdots. \tag{5}$$

with updating:

$$\boldsymbol{D}_{i+1} = \boldsymbol{D}_i - \frac{1}{\boldsymbol{b}_i^{\mathrm{T}}\boldsymbol{b}_i}\left(\boldsymbol{u}\left(\boldsymbol{x}_{i+1}\right) - \boldsymbol{u}\left(\boldsymbol{x}_i\right) - \boldsymbol{D}_i\boldsymbol{b}_i\right)\boldsymbol{b}_i^{\mathrm{T}} \tag{6}$$

where $\boldsymbol{b}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i$. Equation (6) is called the Broyden rank-1 update, and (5) with the update (6) is called the Newton-Broyden method. This method retains the good part of the Newton method, and replaces the difficult part of Newton's with Broyden's. With good initial guesses for $\boldsymbol{z}_0$ and $\boldsymbol{D}_0$, the Newton-Broyden Scheme (5) will produce a sequence that converges to a solution of (3), with q-super linear order of convergence.

Although the order of convergence of the Newton-Broyden method is equal to that of the Broyden method, and less than that of the Newton method, in practice, the Newton-Broyden performs well in the sense that a good initial guess is easily found, and it reaches the solution in a reasonable number of ierations. As shown in [7] and [8], for the same problem and using the same initial guess, the sequences from the Newton method and from the Newton-Broyden method reach the solution, while that from the Broyden does not. In addition, the Newton-Broyden method requires less amount of work in comparison with the Newton method.

Note that if function $\boldsymbol{F}$ is linear, then Broyden's update and Newton-Broyden's update coincide.

3. Constructing the Method

The initial value problems:

Assume that the DAE is expressed in the form (2a), (2b) (renumbering)

$$\boldsymbol{y}'(t) = \boldsymbol{f}\left(t, \boldsymbol{x}(t), \boldsymbol{y}(t)\right) \tag{7a}$$

$$\boldsymbol{g}\left(t, \boldsymbol{x}(t), \boldsymbol{y}(t)\right) = \boldsymbol{0} \tag{7b}$$

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0 \tag{7c}$$

in which the system of equations is to be solved for $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$, where $t \in [a, b]$. In solving the system numerically, the conditions on the initial values of $\boldsymbol{y}$ must be imposed sufficiently for the system to have a unique solution. If (7b) can express $\boldsymbol{x}(t)$ in terms of t and $\boldsymbol{y}(t)$ explicitly, then the system becomes a pure ODE. Therefore, we consider the case when (7b) expresses $\boldsymbol{x}(t)$ implicitly.

Let the interval $[a, b]$ be divided into n subintervals $\{a = t_0, t_1, \cdots, t_n = b\}$. In each interval $[t_{i-1}, t_i]$, we will solve (7a) numerically for $\boldsymbol{y}(t_i)$. In an initial value problem, the value $\boldsymbol{y}(t_0)$ is specified, and the value $\boldsymbol{x}(t_0)$ can subsequently be solved from (7b). In order to use the Runge-Kutta method, in each interval $[t_{i-1}, t_i]$ the value $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$, which are needed for computing $\boldsymbol{F}_2$, $\boldsymbol{F}_3$ and $\boldsymbol{F}_4$ respectively, can each be solved from (7b) using the values of $\boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_1/2$ in $\boldsymbol{F}_2$, $\boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_2/2$ in $\boldsymbol{F}_3$ and $\boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_3/2$ in $\boldsymbol{F}_4$. Once the value $\boldsymbol{y}(t_i)$ is computed, $\boldsymbol{x}(t_i)$ can then be solved from (7b). Advance to the next interval, and repeat this procedure until we reach the last subinterval.

Suppose the term $\boldsymbol{x}(t)$ is missing from the Equation (7b). Therefore, the system becomes:

$$\boldsymbol{y}'(t) = \boldsymbol{f}\left(t, \boldsymbol{x}(t), \boldsymbol{y}(t)\right) \tag{8a}$$

$$\boldsymbol{g}\left(t, \boldsymbol{y}(t)\right) = \boldsymbol{0} \tag{8b}$$

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0 \tag{8c}$$

in which the system is called index-2 Hessenberg. In solving the differential Equation (8a), the variable $\boldsymbol{x}(t)$ are treated as unknowns. In each interval $[t_{i-1}, t_i]$, the value of $\boldsymbol{x}(t_{i-1})$ is given as a guess, and (8a) is then solved numerically for $\boldsymbol{y}(t_i)$. Check the condition (8b) $\boldsymbol{g}\left(t_i, \boldsymbol{y}(t_i)\right) = 0$. Update the value of $\boldsymbol{x}(t_0)$ and iterate the process until the condition (8b) is met. Subsequently, advance to the next interval using the last value of $\boldsymbol{x}(t_{i-1})$ as the guessing value for $\boldsymbol{x}(t_i)$. Repeat this procedure until we complete the last subinterval.

In solving the differential equations, we may use the 4th order Runge-Kutta method, or the 4th order Taylor method, since their local error is $\mathrm{O}(h^5)$ and the total error is $\mathrm{O}(h^4)$, which is acceptable and also easy to implement. In solving the algebraic equations associated in the problem, the Newton method is used in (7b), and the Newton-Broyden method is used in (8b).

As per the procedure described above, the formulation will be as follows:

Partition the interval $[a, b]$ into n subintervals. $h = \dfrac{b-a}{n}$, and $t_0 = a,\ t_i = t_{i-1} + h,\ i = 1, 2, \cdots, n.$

Problem (7a), (7b):

Define the function $\boldsymbol{F}$: $\boldsymbol{F}\left(\boldsymbol{x}(t_{i-1})\right) = \boldsymbol{g}\left(t_i, \boldsymbol{x}(t_i), \boldsymbol{y}(t_i)\right)$, $i = 1, 2, \cdots, n$.

The problem become (3)

$$\boldsymbol{F}\left(\boldsymbol{x}(t_{i-1})\right) = \boldsymbol{0}$$

which can be solved using the Newton method.

P1: Newton Method.
Prescribe a small positive real number ε.
Guess the value $\boldsymbol{z}$.
A: Compute $\boldsymbol{F}(\boldsymbol{z})$.
Check if $\|\boldsymbol{F}(\boldsymbol{z})\| < \varepsilon$
If so, proceed to B. If not, carry out the next step.
Compute $\boldsymbol{F}'(\boldsymbol{z})$.

Solve the system $\boldsymbol{F}'(\boldsymbol{z})\boldsymbol{b} = -\boldsymbol{F}(\boldsymbol{z})$, for $\boldsymbol{b}$.
Set $\boldsymbol{z} = \boldsymbol{z} + \boldsymbol{b}$.
Go to A.
B: End.
The process for solving the DAE will be as follows:
Algorithm A:
Initial step:
1) Set the given initial condition $\boldsymbol{y}(t_0) = \boldsymbol{y}_0$.
2) Solve for $\boldsymbol{x}(t_0)$, using P1: Set $\boldsymbol{x}(t_0) = \boldsymbol{z}$.
Main step:
For $i = 1, 2, \cdots, n$; process the following steps:
Step 1. Solve the initial value problem (7a) 1 step to obtain $\boldsymbol{y}(t_i)$, by proceeding using the following steps:
1) Compute: $\boldsymbol{F}_1 = h\boldsymbol{f}\left(t_{i-1}, \boldsymbol{x}(t_{i-1}), \boldsymbol{y}(t_{i-1})\right)$, $\boldsymbol{y}_1 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_1/2$, Solve $\boldsymbol{g}(t_{i-1} + h/2, \boldsymbol{x}_1, \boldsymbol{y}_1) = \boldsymbol{0}$ for $\boldsymbol{x}_1$. (Using P1).
2) Compute: $\boldsymbol{F}_2 = h\boldsymbol{f}\left(t_{i-1} + h/2, \boldsymbol{x}_1, \boldsymbol{y}_1\right)$
$\boldsymbol{y}_2 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_2/2$, Solve $\boldsymbol{g}(t_{i-1} + h/2, \boldsymbol{x}_2, \boldsymbol{y}_2) = \boldsymbol{0}$ for $\boldsymbol{x}_2$. (Using P1).
3) Compute: $\boldsymbol{F}_3 = h\boldsymbol{f}\left(t_{i-1} + h/2, \boldsymbol{x}_2, \boldsymbol{y}_2\right)$
$\boldsymbol{y}_3 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_3/2$, Solve $\boldsymbol{g}(t_{i-1} + h, \boldsymbol{x}_3, \boldsymbol{y}_3) = \boldsymbol{0}$ for $\boldsymbol{x}_3$. (Using P1).
4) Compute: $\boldsymbol{F}_4 = h\boldsymbol{f}\left(t_{i-1} + h, \boldsymbol{x}_3, \boldsymbol{y}_3\right)$
5) $\boldsymbol{y}(t_i) = \boldsymbol{y}(t_{i-1}) + \left(\boldsymbol{F}_1 + 2\left(\boldsymbol{F}_2 + \boldsymbol{F}_3\right) + \boldsymbol{F}_4\right)/6$
Step 2. Solve (7b) for $\boldsymbol{x}(t_i)$, using P1: Set $\boldsymbol{x}(t_i) = \boldsymbol{z}$.
Proceed to the next interval.
Problem (8a), (8b):
Define the function $\boldsymbol{F}$: $\boldsymbol{F}\left(\boldsymbol{x}(t_{i-1}), \boldsymbol{y}(t_{i-1})\right) = \boldsymbol{g}\left(t_i, \boldsymbol{y}(t_i)\right)$, $i = 1, 2, \cdots, n$.
The problem become (3)

$$\boldsymbol{F}\left(\boldsymbol{x}(t_{i-1}), \boldsymbol{y}(t_{i-1})\right) = \boldsymbol{0}$$

which can be solved using the Newton-Broyden method.
P2: One Step Runge-Kutta Method
Given the value $t_{i-1}, \boldsymbol{x}(t_{i-1}), \boldsymbol{y}(t_{i-1})$ and $\boldsymbol{b}$.
Compute: $\boldsymbol{F}_1 = h\boldsymbol{f}\left(t_{i-1}, \boldsymbol{x}(t_{i-1}), \boldsymbol{y}(t_{i-1})\right)$, $\boldsymbol{y}_1 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_1/2$, $\boldsymbol{x}_1 = \boldsymbol{x}(t_{i-1}) + \boldsymbol{b}/2$.
Compute: $\boldsymbol{F}_2 = h\boldsymbol{f}\left(t_{i-1} + h/2, \boldsymbol{x}_1, \boldsymbol{y}_1\right)$, $\boldsymbol{y}_2 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_2/2$, $\boldsymbol{x}_2 = \boldsymbol{x}(t_{i-1}) + \boldsymbol{b}/2$.
Compute: $\boldsymbol{F}_3 = h\boldsymbol{f}\left(t_{i-1} + h/2, \boldsymbol{x}_2, \boldsymbol{y}_2\right)$, $\boldsymbol{y}_3 = \boldsymbol{y}(t_{i-1}) + \boldsymbol{F}_3$, $\boldsymbol{x}_3 = \boldsymbol{x}(t_{i-1}) + \boldsymbol{b}$.
Compute: $\boldsymbol{F}_4 = h\boldsymbol{f}\left(t_{i-1} + h, \boldsymbol{x}_3, \boldsymbol{y}_3\right)$.
Then $\boldsymbol{y}(t_i) = \boldsymbol{y}(t_{i-1}) + \left(\boldsymbol{F}_1 + 2\left(\boldsymbol{F}_2 + \boldsymbol{F}_3\right) + \boldsymbol{F}_4\right)/6$.
The process for solving the DAE will be as follows:
Algorithm B:
Initial step:
1) Set the given initial condition $\boldsymbol{y}(t_0) = \boldsymbol{y}_0$.
2) Guess the initial condition $\boldsymbol{x}(t_0) = \boldsymbol{z}$. $\boldsymbol{x}(t_0) = \boldsymbol{b}$.
3) Guess the initial matrix $\boldsymbol{D}$ (may be = $\boldsymbol{I}$), Guess the value $\boldsymbol{b}$.
4) Solve the initial value problem (8a) 1 step, to obtain $\boldsymbol{y}(t_1)$, using P2.
5) Compute the value $\boldsymbol{F}\left(\boldsymbol{x}(t_0), \boldsymbol{y}(t_0)\right) = \boldsymbol{g}\left(t_1, \boldsymbol{y}(t_1)\right)$.
Main step:
For $i = 1, 2, \cdots, n$; process the following steps:

Step 1. Check the condition $\left\|\boldsymbol{g}\left(t_i, \boldsymbol{y}\left(t_i\right)\right)\right\| < \varepsilon$ where ε is a prescribed small number.

If it passes, proceed to the next interval (next i).
If it fails, carry out step 2.
Step 2.
1) Set $\boldsymbol{u} = \boldsymbol{y}\left(t_i\right)$.
2) Compute the matrix $\boldsymbol{A} = \boldsymbol{F}_x\left(\boldsymbol{x}\left(t_{i-1}\right), \boldsymbol{y}\left(t_i\right)\right) + \boldsymbol{F}_y\left(\boldsymbol{x}\left(t_{i-1}\right), \boldsymbol{y}\left(t_i\right)\right)\boldsymbol{D}$.
3) Solve $\boldsymbol{A}\boldsymbol{b} = -\boldsymbol{g}\left(t_i, \boldsymbol{y}\left(t_i\right)\right)$ for $\boldsymbol{b}$.
4) Set $\boldsymbol{x}\left(t_{i-1}\right) = \boldsymbol{x}\left(t_{i-1}\right) + \boldsymbol{b}$.
5) Solve the initial value problem (8a) 1 step (using P2) to obtain $\boldsymbol{y}\left(t_i\right)$.
6) Compute the new value $\boldsymbol{g}\left(t_i, \boldsymbol{y}\left(t_i\right)\right)$.
7) Update the matrix $\boldsymbol{D} = \boldsymbol{D} + \frac{1}{\boldsymbol{b}^{\mathrm{T}}\boldsymbol{b}}\left(\boldsymbol{y}\left(t_i\right) - \boldsymbol{u} - \boldsymbol{D}\boldsymbol{b}\right)\boldsymbol{b}^{\mathrm{T}}$.
8) Go to step 1.
End.

4. Experiments

Three examples are used to illustrate the method. The first problem is an index-1 Hessenberg DAE system, with nonlinear differential equations and a nonlinear algebraic equation, as follows:

$$x''(t) = -(3t+1)\,y(t) - x(t)\left(4z(t)+1\right),\ \ 0 \le t \le 1$$

$$y''(t) = 4\cos\left(z(t)\right) - y(t)\left(4z(t)+1\right)$$

initial conditions,

$$x(0) = 0,\ \ y(0) = 0,$$

$$x'(0) = 1,\ \ y'(0) = 2$$

algebraic equation

$$4x(t)\cos\left(z(t)\right) + ty^2(t) = 4\left(z(t) - t^2\right)$$

Using Algorithm A, this problem is solved and has a nice result with a small error as compared to the exact solution

$$z(t) = t(t+1)$$

$$x(t) = t\cos\left(z(t)\right),$$

$$y(t) = \sin\left(z(t)\right).$$

Some results are illustrated in **Table 1**. (*xs*, *ys* and *zs* are values from the exact solution).

The second problem is a classical example of the DAE problem which is "The pendulum problem", as expressed in the *xy* co-ordinate plane as follows:

$$\left.\begin{aligned} x''(t) &= -l(t)\,x(t), \quad t \ge 0 \\ y''(t) &= -l(t)\,y(t) - g \\ x(t)^2 + y(t)^2 &= L^2 \end{aligned}\right\} \tag{8}$$

where g = 9.8. This problem is an index 2 semi-explicit DAE problem. [3], [10]. Proceeding to solve the problem numerically, we let L = 1, and impose the initial conditions $x(0)$ = 1, $y(0)$ = 0. Using the new variables $y_1 = x$, $y_2 = x'$, $y_3 = y$ and $y_4 = y'$, we have a system of differential algebraic equations as follows:

Table 1. Resulting values for $x(t)$, $y(t)$ and $z(t)$ from Algorithm A, and the errors.

i	t	x	y	z	$x - xs$	$y - ys$	$z - zs$
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.083333	0.082994	0.180310	0.090278	0.000000	0.000000	0.000000
10	0.1666667	0.1635259	0.3864430	0.1944444	−0.000000	0.0000000	−0.000000
15	0.2500000	0.2378920	0.6148770	0.3125000	−0.000000	0.0000000	−0.000000
20	0.3333333	0.3009499	0.8599127	0.4444444	−0.000000	0.0000000	−0.000000
25	0.4166667	0.3461609	1.1131836	0.5902778	−0.000000	0.0000000	−0.000000
30	0.5000000	0.3658444	1.3632775	0.7500000	−0.000000	0.0000000	−0.000000
35	0.5833333	0.3517169	1.5955682	0.9236111	−0.000000	0.0000000	−0.000000
40	0.6666667	0.2957773	1.7923844	1.1111111	−0.000000	0.0000000	−0.000000
45	0.7500000	0.1915753	1.9336531	1.3125000	−0.000000	0.0000000	0.0000000
50	0.8333333	0.0358377	1.9981497	1.5277778	0.0000000	0.0000000	0.0000000
55	0.9166667	−0.169652	1.9654489	1.7569445	0.0000001	0.0000001	0.0000001
60	1.0000000	−0.416147	1.8185950	2.0000002	0.0000002	0.0000003	0.0000002

$$\left.\begin{aligned} y_1' &= y_2 \\ y_2' &= -\lambda y_1 \\ y_3' &= y_4 \\ y_4' &= -\lambda y_3 - g \end{aligned}\right\}$$

initial conditions:

$$\begin{aligned} y_1(0) &= 1 \\ y_2(0) &= 0 \\ y_3(0) &= 0 \\ y_4(0) &= 0 \end{aligned} \tag{9a}$$

algebraic equation:

$$y_1^2 + y_3^2 - 1 = 0 \tag{9b}$$

If we let $x = \sin\theta$, $y = -\cos\theta$, substitute in the problem (8) and manipulate some derivatives and algebra, we will obtain the equivalent initial value problem of an ordinary differential equation:

$$\left.\begin{aligned} \theta''(t) &= -g\sin\left(\theta(t)\right),\ t \geq 0 \\ \theta(0) &= \frac{\pi}{2} \end{aligned}\right\} \tag{10}$$

From (9b) if we differentiate the equation $y_1(t)^2 + y_3(t)^2 = 1$ twice, we will obtain the equation:

$$\lambda(t) = y_2(t)^2 + y_4(t)^2 - g y_2(t). \tag{11}$$

The system (9a) with (11) becomes a pure ODE, since the function $\lambda(t)$ is expressed in the terms $y_2(t)$ and $y_4(t)$ explicitly.

In this experiment, we solve (9a), (9b) using Algorithm B, and provide some of the results in **Table 2**. System (10) and (9a) with (11) are also solved, and they provide the same results, wherein their values are used to com-

Table 2. Result values for $x(t)$, $y(t)$ and $\lambda(t)$ from Algorithm B, and the errors.

i	t	x	y	$\lambda(t)$	$x - xs$	$y - ys$	$\lambda - \lambda s$
0	0.000000	1.000000	0.000000	0.0066694	0.000000	0.000000	−0.01334
5	0.083333	0.999421	−0.034020	1.2067764	0.000000	0.000000	−0.01327
10	0.166667	0.990763	−0.135608	4.3889120	0.000000	0.000000	−0.01245
15	0.250000	0.953758	−0.300577	9.4089741	−0.000000	−0.000000	−0.00924
20	0.333333	0.858149	−0.513401	15.762372	−0.000000	−0.000000	−0.001840
25	0.416667	0.674223	−0.738528	22.336944	0.000000	0.000000	0.009746
30	0.500000	0.392046	−0.919946	27.443990	0.000000	0.000000	0.021502
35	0.583333	0.039516	−0.999219	29.406564	0.000001	0.000000	0.026667
40	0.666667	−0.320679	−0.947188	27.497515	0.000001	−0.000000	0.021650
45	0.750000	−0.621722	−0.783238	22.424766	−0.000001	0.000001	0.009949
50	0.833333	−0.826959	−0.562262	15.858262	0.000001	−0.000000	−0.001681
55	0.916667	−0.939329	−0.343017	9.4922857	0.000000	−0.000000	−0.009152
60	1.0000000	−0.986140	−0.165918	4.4483962	0.000000	−0.000000	−0.012414

pare to the results from Algorithm B, as shown in **Table 2**. The value xs, ys and λs are from problem (10). Note that the value of $\lambda(t)$'s are at the mid-points $t + h/2$.

The third problem is an index-2 Hessenberg DAE system with nonlinear differential equations and a nonlinear algebraic equation, as follows:

$$x''(t) = x(t)\left(4z(t) - 1\right) + 2(1 - 3t)\, y(t), \quad 0 \le t \le 1$$

$$y''(t) = 2\sin\left(z(t)\right) + y(t)\left(4z(t) - 1\right)$$

initial conditions,

$$x(0) = 0, \quad y(0) = 1,$$

$$x'(0) = 0, \quad y'(0) = 0$$

algebraic equation

$$x^2(t) + t^2 y^2(t) = t^2$$

Note that this problem is similar to the first problem, but there is no variable $z(t)$ in the algebraic equation.

Using Algorithm B, this problem is solved, and provides a nice result as shown in **Table 3**. The exact solution is as follows:

$$z(t) = t(1 - t)$$

$$x(t) = t\sin\left(z(t)\right),$$

$$y(t) = \cos\left(z(t)\right).$$

Algorithm B gives the results for $x(t)$ and $y(t)$, with a small error. However, the solution for $z(t)$ has some errors less than 0.005. Some of the results are illustrated in **Table 3**. Note that the value of $z(t)$'*s* are at the mid-points $t + h/2$.

Table 3. Result values for $x(t)$, $y(t)$ and $z(t)$ from Algorithm A, and the errors.

i	t	x	y	z	$x - xs$	$y - ys$	$z - zs$
0	0.0000000	0.0000000	1.0000000	.0055058	0.0000000	0.0000000	−0.0027581
5	0.0833333	0.0063659	0.9970838	0.0860924	0.0000000	−0.0000000	0.0028285
10	0.1666667	0.0230738	0.9903704	0.1416003	0.0000000	−0.0000000	−0.0027746
15	0.2500000	0.0466008	0.9824733	0.1944127	0.0000000	−0.0000000	0.0028155
20	0.3333333	0.0734659	0.9754101	0.2221708	0.0000000	−0.0000000	−0.0027597
25	0.4166667	0.1002790	0.9706071	0.2471710	0.0000000	−0.0000000	0.0027960
30	0.5000000	0.1237020	0.9689124	0.2471938	0.0000000	−0.0000000	−0.0027367
35	0.5833333	0.1403905	0.9706071	0.2443677	0.0000000	−0.0000000	0.0027705
40	0.6666667	0.1469318	0.9754101	0.2166649	0.0000000	−0.0000000	−0.0027101
45	0.7500000	0.1398025	0.9824733	0.1860087	0.0000000	−0.0000000	0.0027448
50	0.8333333	0.1153690	0.9903704	0.1305767	0.0000000	−0.0000000	−0.0026872
55	0.9166667	0.0699551	0.9970838	0.0721021	0.0000000	−0.0000000	0.0027271
60	1.0000000	0.0000001	1.0000000	−0.0110809	0.0000000	0.0000000	−0.0026781

5. Conclusion

The methods are constructed with the objective of solving DAE systems in their original forms. Both algorithms use the Runge-Kutta method as the integrator, and couple this with a method to solve the algebraic systems associated in the problem. The Newton method is used in Algorithm A for index-1 DAE, while in Algorithm B the Newton-Broyden method is needed for an index-2 DAE system. The methods can give approximate solutions for the problem very well, with only small errors. Experiments have also shown that high index DAEs are harder to solve than lower index DAEs.

Acknowledgements

The author would like to thank the referees for their comments. This work was financially supported by the Research Grant of Burapha University through National Research Council of Thailand (Grant No. 35/2556).

References

[1] Ascher, U.M. and Petzold, L.R. (1998) Computer Method for Ordinary Differential Equations and Differential-Algebraic Equations. Society for Industrial and Applied Mathematics (SIAM), Philadelphia. http://dx.doi.org/10.1137/1.9781611971392

[2] Brenan, K.E., Campbell, S.L. and Petzold, L.R. (1996) Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, Revised and Corrected Reprint of 1989 Original, with an Additional Chapter and Additional References. Classic in Applied Mathematics, 14. SIAM, Philadelphia.

[3] Campbell, S.L., *et al.* (2008) Differential-Algebraic Equations. *Scholarpedia*, **3**, 2849. http://dx.doi.org/10.4249/scholarpedia.2849

[4] Rabier, P.J. and Rheinboldt, W.C. (2002) Theoretical and Numerical Analysis of Differential-Algebraic Equations. *Hand Book of Numerical Analysis*, **8**, 183-540. http://dx.doi.org/10.1016/S1570-8659(02)08004-3

[5] Riaza, R. (2008) Differential-Algebraic System: Analytical Espects and Circuit Applications. World Scientific, Singapore City.

[6] Hairer, E. and Wanner, G. (1998) Solving Ordinary Differential Equation. II. Stiff and Differential-Algebraic Problems, 2nd Edition, Springer Series in Computational Mathematics, 14. Springer-Verlag, Berlin.

[7] Chompuvised, K. and Dhamacharoen, A. (2011) Solving Boundary Value Problems of Ordinary Differential Equations with Non-Separated Boundary Conditions. *Applied Mathematics and Computation*, **217**, 10355-10360. http://dx.doi.org/10.1016/j.amc.2011.05.044

[8] Dhamacharoen, A. (2014) An Efficient Hybrid Method for Solving Systems of Nonlinear Equations. *Journal of Computational and Applied Mathematics*, **263**, 59-68. http://dx.doi.org/10.1016/j.cam.2013.12.006

[9] Dhamacharoen, A. and Chompuvised, K. (2013) An Efficient Method For Solving Multipoint Equation Boundary Value Problems. *World Academy of Science, Engineering and Technology*, **7**, 329-333.

[10] Differential-Algebraic Equation, Wikipedia, the Free Encyclopedia.