CWI Quarterly

Volume 11 (1) 1998, pp. 83 - 100

Collecting Real-life Problems to Test Solvers for Implicit Differential Equations

Jacques J. B. de Swart
Paragon Decision Technology B.V.
P.O.Box 3277
2001 DG Haarlem, The Netherlands
and
CWI, P.O.Box 94079,
1090 GB Amsterdam, The Netherlands

Walter M. Lioen
CWI, P.O. Box 94079,
1090 GB Amsterdam, The Netherlands

To test solvers for implicit differential equations, it is important to have a representative collection of test problems. Such a test set can both decrease the effort for the code developer to test his software in a reliable way, and cross the bridge between the application field and numerical mathematics. In this paper we describe the CWI Test Set for IVP solvers and look at two applications in detail, one from car industry and one from biochemistry.

Acknowledgements: This work is supported financially by the 'Technologiestichting STW' (Dutch Foundation for Technical Sciences).

1. Introduction

For many processes in industry it is indispensable to solve differential equations. Examples are the simulation of electrical circuits, the behavior of a train on a rail track, the steering of robots and chemical reactions. Normally, the differential equations are far too complicated to solve analytically, and one has to resort to numerical integration techniques implemented on a computer to obtain an approximation to the solution. Numerical analysts are challenged to come up with faster algorithms, because problems become ever complexer, (for example chip design) and in several cases a decrease in computer time is required, for example it is undesirable to steer the motion of a robot by software that needs several seconds to compute a new move.

There is a wide-spread attitude to take the testing of these algorithms not very seriously. What one typically sees in scientific talks, is that the speaker addresses 99% of his time to describe his beautiful new method, and then, when the chairman has already coughed a couple of times, he quickly puts up a slide with two curves, which are supposed to explain that his new method works better than some existing one, without explaining how he has performed his tests. Quite often, it turns out that the testing involved only a very simple, academic problem. In scientific literature, the attitude is basically the same.

It is easy to justify the need for a precisely defined test protocol that tests on large, real-life problems. Although reliable software for solving differential equations has been around for the past few decades, new codes are necessary because the existing software often performs too slow on large problems, or can not handle nasty complications arising in practice, such as discontinuities in the differential equations, or a phenomenon called *higher index character*, which means that some components of a problem are more sensitive for perturbations than other. Of course it does not make sense to promote a new algorithm on the basis of small, academic test problems that could be solved satisfactorily by existing codes. Besides, a new important issue in code development is to make codes suitable for implementation on parallel computers. Especially for large problems it is sometimes necessary to resort to such computers in order to reduce the computer time to reasonable values. Again it is clear that one should not test these parallel algorithms on small problems.

Another justification of performing well-defined real-life tests is that poor testing may lead to not considering methods that perform badly in such a test, even if they might work well for a certain class of problems, which is not represented by the test problems used. On the other hand, it may also happen that one trusts methods that perform well in a test, although they suffer from all sorts of disadvantages, not revealed by the test. Probably, the latter case occurs more often than the former, since most tests in articles and talks favor the author's method.

Good test problems and a good test protocol should contain the following ingredients.

- A complete specification of the problem should be given, including initial values, parameter settings, integration interval, reference solution, etc. In order to ensure that everyone really uses the same test problem, also an actual implementation should be given.

- It should be clear why one is interested in a problem. What is the quantity of interest and how can we measure it? How do we interpret the problem variables and the results? For example, it does not make sense to test the method by computing the behavior of a car on a smooth road, because that is not the 'hard' part. What matters is a test on a bumpy road.

- A test set should be classified, so that a developer of a code that is meant for a certain application area, can easily select the problems that are of interest to him.
- Along with a test problem, recommendations should be given, for example, 'notice that a solver might have difficulty in solving this problem, if it does not pay proper attention to its discontinuities'.
- A test set should be representative for many areas of applications. Small test sets create inbreeding, in the sense that if codes are developed such that they can solve problems of a non-representative test set, then it may happen that too much attention is paid to only a few features in problems, whereas features that are not present in the test problems, are not addressed at all.
- Guidelines should be given for people who want to use for some reason their own test problems.
- Test problems should be of interest to people from industrial application fields. If so, the problems can help to bridge the gap between industrial engineering and the numerical analysis community. Using industrial real-life problems as test problem assures that solvers developed can deal with practical issues.

The 'CWI test set for IVP solvers' [5] tries to fulfill these demands. It can also save a lot of time and effort for the developer of software for differential equations, because using the test set, he does not have to program the test problems and describe them in his article anymore, nor does he need to produce the test results for reference solvers. The test set is available via the WWW page `http://www.cwi.nl/cwi/projects/IVPtestset/`, or via anonymous ftp at site `ftp.cwi.nl` in the directory `pub/IVPtestset` . Since the first release in August 1996, the test set has grown to fifteen serious test problems, and is used by many researchers all over the world. This success would not have been possible without the many cooperative contributors from different application areas.

In the following two sections we describe the structure of the CWI test set and its use as test platform. We conclude by treating two test problems in some detail. We selected these examples because their relative simplicity allows for self-contained descriptions. In the test set there are more complicated problems: larger systems, much more CPU time and more difficult to solve.

2. The structure of the CWI Test Set

The CWI test set consists of a descriptive part and a software part. The first part describes test problems and reports on the behavior of a few state-of-the-art solvers on these problems. The software serves as a platform on which one can test the performance of a solver on a particular test problem oneself.

2.1. The descriptive part

The test set has been divided into chapters, each describing one test problem. Every chapter contains the following 4 sections:

1. General information, listing e.g. the problem identification, dimension, index, contributor, etc.

2. Mathematical description of the problem, with all ingredients that are necessary for implementation given in mathematical formulas.

3. Origin of the problem, with a description of the origin, which gives a physical interpretation of the problem. References to the literature are given for further details. This section is important for bridging the gap between the application field and numerical mathematics. It describes the modeling process and gives a feeling for the important characteristics of the problem.

4. Numerical solution of the problem, which adreses the following items:

   (a) Solution in the endpoint. The values of the solution components in the endpoint are listed.

   (b) Run characteristics. A table with integration statistics of runs with some well-known codes are given. For example, for problem HIRES, which will be described in §4, these characteristics are in Table 2.

   The symbols in this table have the following meaning:

   solver The name of the numerical solver with which the run was performed, currently one of DASSL [6], RADAU5 [3], PSIDE [10] and VODE [1].

   rtol, atol and h0 The user supplied relative and absolute error tolerance and the initial stepsize, if requested by the solver.

   scd The scd values are an indication of the quality of the numerical solution. They denote the minimum number of significant correct digits in the endpoint, i.e.

   $$\text{scd} := -\log_{10}(\text{max. norm of the relative error in the endpoint}).$$

   steps Total number of steps taken by the solver (including rejected steps).

   accept The number of accepted steps.

   # f and # Jac The number of evaluations of the function f and its Jacobian, respectively, where f is the problem defining function.

   # LU The number of LU-decompositions.

CPU The CPU time in seconds to perform the run on an SGI workstation, an *Indy* with a 100 MHz R4000SC processor, using the Fortran 77 compiler with optimization: `f77 -O`.

For all solvers, the integration characteristics mentioned previously and presented in the tables, refer to execution on a one-processor computer. With respect to the run characteristics of PSIDE - Parallel Software for Implicit Differential Equations - we remark that this solver aims at execution on a parallel shared memory computer with four processors. On such a computer, one may divide the number of evaluations, decompositions and solves by four to obtain the *effective* characteristics.

(c) Behavior of the numerical solution. Plots of (some of) the components over (part of) the integration interval are presented. For example, Figure 7 shows the behavior of the concentrations in problem HIRES.

(d) Work-precision diagram. For every relevant solver, a range of input tolerances and, if necessary, a range of initial stepsizes, were used to produce a plot of the resulting scd values against the number of CPU seconds needed for the run on the aforementioned computer. One thus gets an impression of the work needed to obtain a certain precision. To give an impression of the performance of PSIDE on a parallel computer we plotted two PSIDE curves in the work-precision diagrams, PSIDE-1 and PSIDE-4. The first curve refers to PSIDE on one processor. The latter curve was obtained by dividing the CPU timings of the runs on one processor by the speed-up factor for one single run as obtained on four processors of a Cray C90. Figure 6 is an example of such a diagram for problem HIRES.

2.2. The test set as test platform

In order to perform a test run, the test set offers a number of Fortran 77 files, which can be divided into four categories:

Problem definition Such files contain six subroutines. The first gives some general characteristics of the problem, such as the problem identification, the dimension and the time points where the problem has a discontinuity in time. The second routine gives the initial values y_0 and y'_0 (the latter is not necessary for ordinary differential equations). The problem defining function is in the third routine. The fourth and fifth routine contain the derivatives of the problem defining function. In order to be able to determine the accuracy of the numerical solution produced by the test run, it has to be compared with a reference solution, which is available in the sixth routine.

Driver These drivers contain global declarations and form an interface between the problem defining routines and a specific solver. All the solver dependent settings are done here as well. The availability of these drivers decreases the effort of performing runs considerably, because the design of them is a cumbersome and error-prone job.

Solver Currently, the following solvers are available in the test set: RADAU5 [3], VODE [1], DASSL [6] and PSIDE [10].

I/O This file asks for the input parameters in a user-friendly fashion and prints the integration characteristics and numerical solution in a surveyable manner.

Auxiliary linear algebra Every solver requires linear algebra routines which are supplied separately from the solver.

As an example, we perform a test run, in which we solve problem HIRES, which will be described in §4 and whose subroutines are in the file `hires.f`, with RADAU5, of which the source code, driver and auxiliary routines are in the files `radau5.f`, `radaud.f` and `radaua.f`, respectively.

Figure 1 shows what one has to do.

3. Example: The car axis problem

3.1. The model

The car axis problem was taken from [8]. It is an example of a rather simple multibody system, in which the behavior of a car axis on a bumpy road is modeled by a set of differential-algebraic equations. We selected this example because its simplicity allows for a self-contained description, which includes the main aspects of the modeling process of more complicated multibody systems in the test set. Moreover, it gives us the opportunity to focus briefly on what is meant by *the index of a variable* and *consistent initial values.*

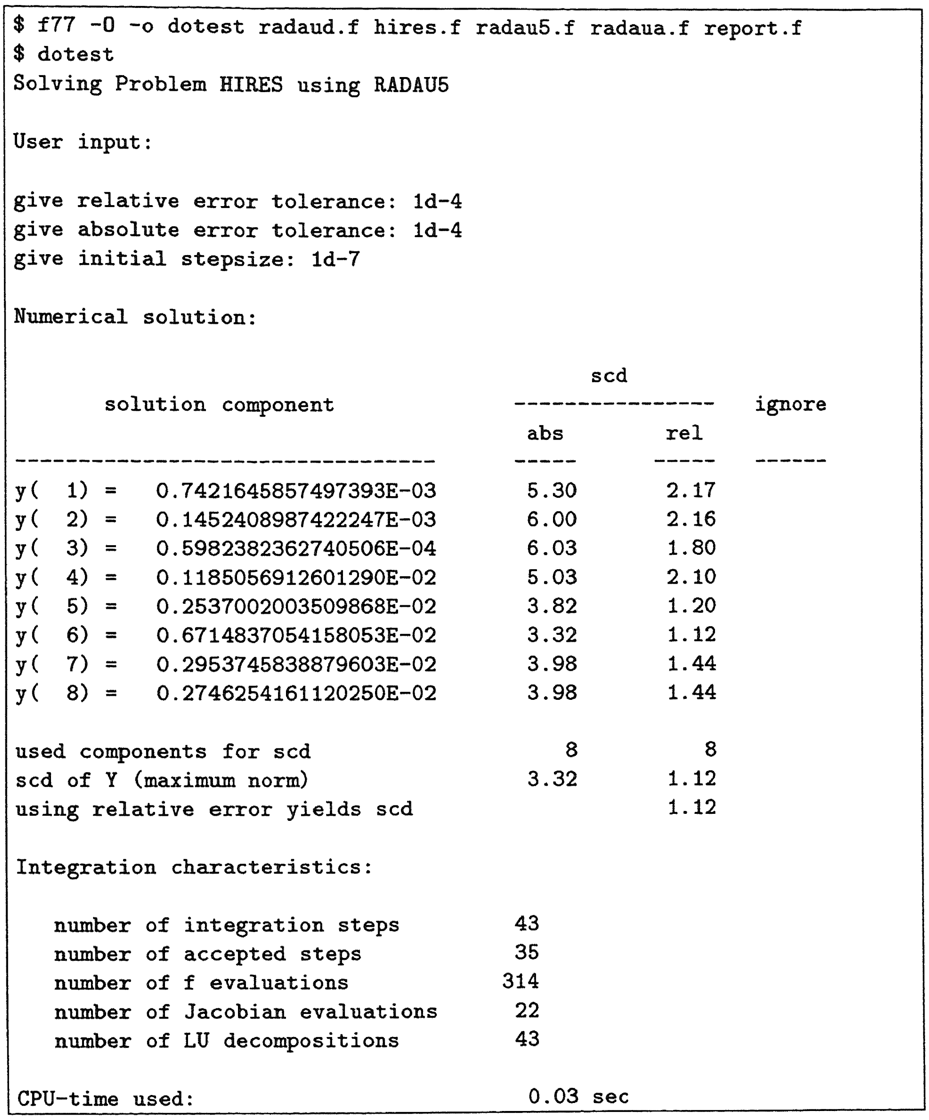
A simplification of the car is depicted in Figure 2. We model the situation that the left wheel at the origin $(0,0)$ rolls on a flat surface and the right wheel at coordinates (x_b, y_b) rolls over a hill of height h every τ seconds. This means that y_b varies over time according to

$$y_b = \frac{1}{2} h \sin\left(\frac{2\pi}{\tau} t\right). \tag{1}$$

The length of the axis, denoted by l, remains constant over time. Consequently, the equation for x_b is given by

$$x_b = \sqrt{l^2 - y_b^2(t)}. \tag{2}$$

Two springs carry over the movement of the axis between the wheels to the chassis of the car, which is represented by the bar (x_l, y_l)–(x_r, y_r) of mass M. The two springs are assumed to be massless and have Hooke's constant $1/\epsilon^2$ and length l_0 at rest.



```
$ f77 -O -o dotest radaud.f hires.f radau5.f radaua.f report.f
$ dotest
Solving Problem HIRES using RADAU5

User input:

give relative error tolerance: 1d-4
give absolute error tolerance: 1d-4
give initial stepsize: 1d-7

Numerical solution:

                                                    scd
       solution component                    ----------------   ignore
                                              abs         rel
 ----------------------------------          -----       -----   ------
y(  1) =   0.7421645857497393E-03            5.30        2.17
y(  2) =   0.1452408987422247E-03            6.00        2.16
y(  3) =   0.5982382362740506E-04            6.03        1.80
y(  4) =   0.1185056912601290E-02            5.03        2.10
y(  5) =   0.2537002003509868E-02            3.82        1.20
y(  6) =   0.6714837054158053E-02            3.32        1.12
y(  7) =   0.2953745838879603E-02            3.98        1.44
y(  8) =   0.2746254161120250E-02            3.98        1.44

used components for scd                         8           8
scd of Y (maximum norm)                      3.32        1.12
using relative error yields scd                          1.12

Integration characteristics:

   number of integration steps          43
   number of accepted steps             35
   number of f evaluations             314
   number of Jacobian evaluations       22
   number of LU decompositions          43

CPU-time used:                           0.03 sec
```

FIGURE 1. Example of performing run, in which we solve problem HIRES with RADAU5.

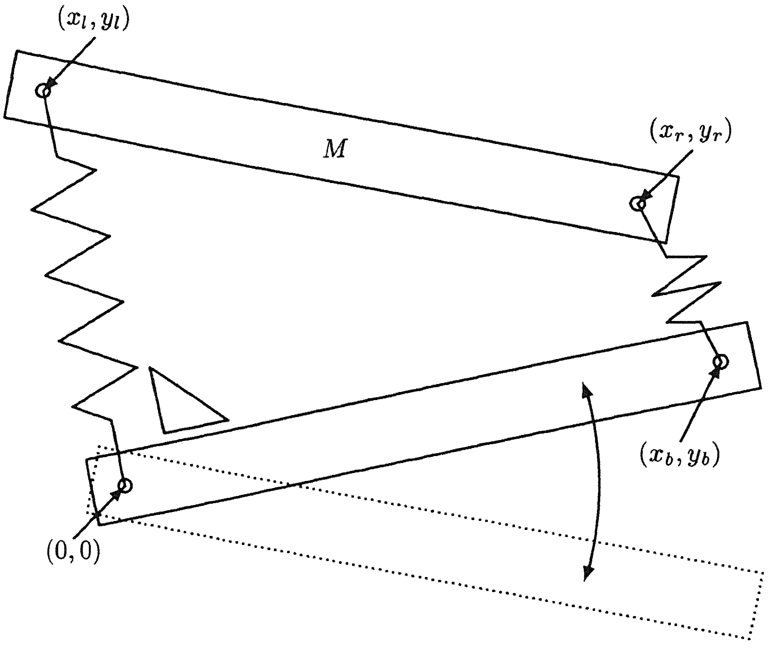


FIGURE 2. Model of the car axis.

There are two position constraints. Firstly, the distance between (x_l, y_l) and (x_r, y_r) must remain constantly l:

$$(x_l - x_r)^2 + (y_l - y_r)^2 - l^2 = 0. \tag{3}$$

Secondly, for simplicity of the model, we assume that the left spring remains orthogonal to the axis:

$$x_b x_l + y_b y_l = 0. \tag{4}$$

For later reference, we introduce the vector $p := (x_l, y_l, x_r, y_r)^{\mathrm{T}}$ and use the shorthand notation

$$\phi(t,p) = 0, \qquad \phi : \mathrm{I\!R}^5 \to \mathrm{I\!R}^2 \tag{5}$$

for the equations (3) and (4).

The setting of the parameters is as follows:

$l = 1$	$\epsilon = 10^{-2}$	$h = 1/5$
$l_0 = 1/2$	$M = 10$	$\tau = \pi/5$

3.2. Lagrangian mechanics

Using Lagrangian mechanics, the equations of motions for p are now given by

$$\frac{M}{2}\frac{\mathrm{d}^2p}{\mathrm{d}t^2} = F_{\mathrm{H}} + G^{\mathrm{T}}\lambda + F_{\mathrm{g}}. \tag{6}$$

Here, G is the 2×4 Jacobian matrix of the function ϕ with respect to p and λ is the 2-dimensional vector containing the so-called Lagrange multipliers. The factor $M/2$ is explained by the fact that the mass M is divided equally over (x_l, y_l) and (x_r, y_r). The force F_{H} represents the spring forces:

$$F_{\mathrm{H}} = -(\cos(\alpha_l)F_l, \sin(\alpha_l)F_l, \cos(\alpha_r)F_r, \sin(\alpha_r)F_r)^{\mathrm{T}},$$

where F_l and F_r are the forces induced by the left and right spring, respectively, according to Hooke's law:

$$\begin{aligned} F_l &= (l_l - l_0)/\epsilon^2, \\ F_r &= (l_r - l_0)/\epsilon^2. \end{aligned}$$

Here, l_l and l_r are the actual lengths of the left and right spring, respectively:

$$\begin{aligned} l_l &= \sqrt{x_l^2 + y_l^2}, \\ l_r &= \sqrt{(x_r - x_b)^2 + (y_r - y_b)^2}. \end{aligned}$$

Furthermore, α_l and α_r are the angles of the left and right spring with respect to the horizontal axis of the coordinate system:

$$\begin{aligned} \alpha_l &= \arctan(y_l/x_l), \\ \alpha_r &= \arctan((y_r - y_b)/(x_r - x_b)). \end{aligned}$$

Finally, F_g represents the gravitational force

$$F_g = -(0, 1, 0, 1)^{\mathrm{T}}\frac{M}{2}g.$$

The original formulation [8] sets $g = 1$.

We rewrite (6) as a system of first order differential equations by introducing the velocity vector q, so that we obtain the first order differential equations

$$\begin{aligned} \frac{\mathrm{d}p}{\mathrm{d}t} &= q, \\ \frac{M}{2}\frac{\mathrm{d}q}{\mathrm{d}t} &= F_{\mathrm{H}} + G^{\mathrm{T}}\lambda + F_{\mathrm{g}}. \end{aligned}$$

We thus modeled the system by eight differential and 2 two algebraic equations. The integration is started at $t_0 = 0$ and continued during 3 seconds.

3.3. Index determination

In order to be able to integrate the system of differential-algebraic equations numerically with a variable stepsize, one usually estimates a local error, in which the index of the variables has to be taken into account.

For the car axis it is easy to determine these indexes, because the problem falls in the class of multibody systems with position constraints. For this class, the indexes of the points p, velocities q, and Lagrange multipliers λ is 1, 2, and 3, respectively (see e.g. [9, p. 176] or [2]). The index information belongs to the test problem definition and is user-supplied input for the solvers.

3.4. Consistent initial conditions

The numerical integration needs a set of initial values $(t_0, p_0, q_0, \lambda_0)$ that is *consistent*, which means that not only the constraint

$$\phi(t_0, p_0) = 0 \tag{7}$$

has to be satisfied, but also the 1 up to $k-1$ times differentiated constraint (7), where k is the highest variable index. To facilitate notation, we introduce $\tilde{p} := (t, p^{\mathrm{T}})^{\mathrm{T}}$ and its derivative $\tilde{q} := \frac{\mathrm{d}\tilde{p}}{\mathrm{d}t} = (1, q^{\mathrm{T}})^{\mathrm{T}}$. The Jacobian of ϕ with respect to $\tilde{p}$ will be denoted by $\tilde{G}$. For the car axis problem $k = 3$, yielding the additional conditions

$$\tilde{G}(\tilde{p}_0)\tilde{q}_0 = 0 \tag{8}$$

and

$$\phi_{\tilde{p}\tilde{p}}(\tilde{p}_0)(\tilde{q}_0, \tilde{q}_0) + \tilde{G}(\tilde{p}_0)\tilde{q}_0' = 0,$$

where $\phi_{\tilde{p}\tilde{p}}$ denotes the second derivative of ϕ with respect to $\tilde{p}$. Using (6) and the fact that the first component of $\tilde{q}_0'$ vanishes, the latter condition equals

$$\phi_{\tilde{p}\tilde{p}}(\tilde{p}_0)(\tilde{q}_0, \tilde{q}_0) + \frac{2}{M} G(p_0) \left(F_{\mathrm{H}}(p_0) + G^{\mathrm{T}}(p_0)\lambda_0 + F_{\mathrm{g}}(p_0) \right) = 0, \tag{9}$$

The equations (7)–(9) are solved for

$$\begin{aligned}
x_r &= l \\
x_l &= 0 \\
y_r &= y_l = l_0 \\
x_r' &= x_l' = -\frac{l_0}{l}\frac{\pi}{\tau}h \\
y_r' &= \frac{l^2\tau}{M\pi h}(2\lambda_1 - \lambda_2) \\
y_l' &= \frac{l^2\tau}{M\pi h}(2\lambda_1 - \lambda_2) \pm l\sqrt{\frac{-8\lambda_1 + 2\lambda_2}{M}}
\end{aligned}$$

where the initial conditions for x_r, x_l, y_r, and y_l are given. We choose $\lambda_1 = \lambda_2 = 0$, so that $y_r' = y_l' = 0$.

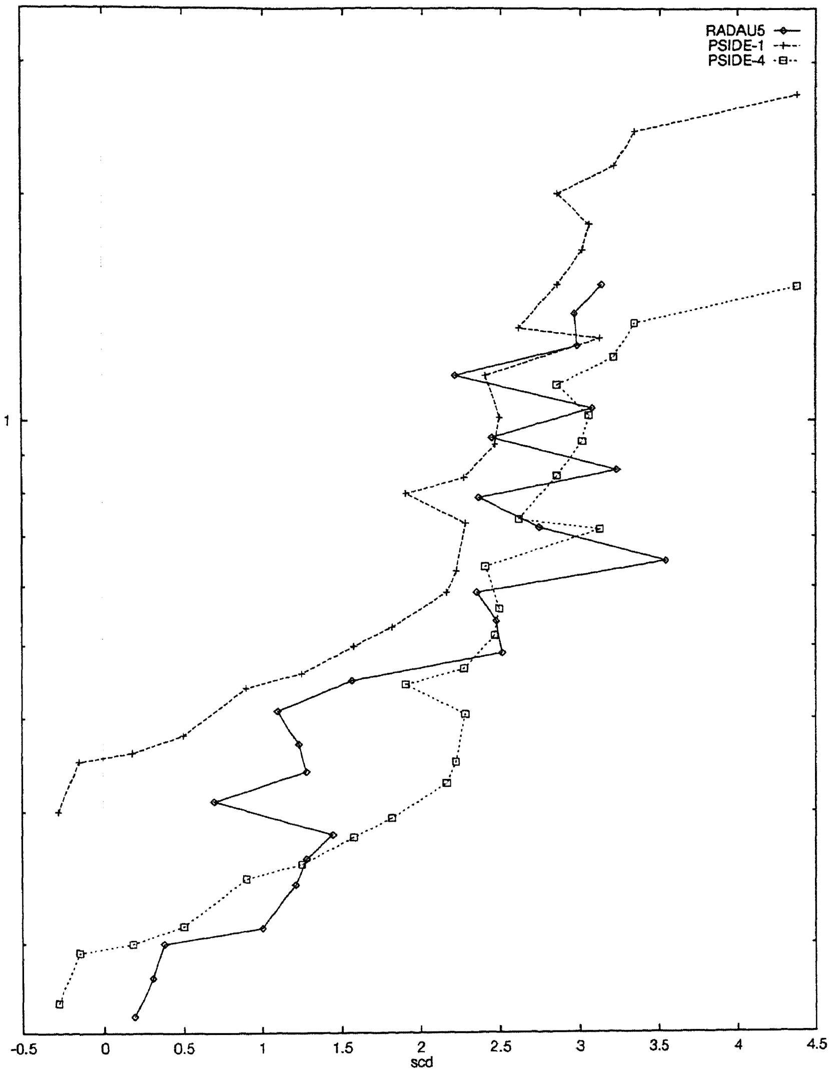


FIGURE 3. Work-precision diagram for the car axis problem.

3.5. The solution

Solving the car axis problem numerically yielded the following results: Table 1 shows the run characteristics; Figure 3 and Figure 4 show the work-precision diagram and the behavior of the solution over the integration interval, respectively.

From Figure 4 we see that connecting the chassis to the axis with only springs and no shock absorbers does not give a satisfactory behavior: bumps in the road are slightly magnified and extra vibrations of the chassis are introduced.

solver	rtol	scd	steps	accept	# f	# Jac	# LU	CPU
RADAU5	10^{-4}	0.19	98	97	850	95	98	0.16
	10^{-7}	2.51	289	288	2559	282	288	0.49
	10^{-10}	3.13	884	883	8101	861	883	1.53
PSIDE	10^{-4}	−0.28	55	54	1403	42	220	0.30
	10^{-7}	2.27	179	172	4103	83	464	0.84
	10^{-10}	4.38	621	608	14117	114	980	2.72

TABLE 1. Run characteristics for the car axis problem (atol = rtol and for RADAU5: h0 = rtol).

4. EXAMPLE: PROBLEM HIRES

4.1. The reaction scheme

The HIRES problem originates from plant physiology and describes how light is involved in morphogenesis. To be precise, it explains the "High Irradiance Responses" (HIRES) of photomorphogenesis on the basis of phytochrome, by means of a chemical reaction involving eight reactants. The reaction scheme is given in Figure 5.

P_r and P_{fr} refer to the red and far-red absorbing form of phytochrome, respectively. They can be bound by two receptors X and X′, partially influenced by the enzyme E. For more details, we refer to [7].

The values of the reaction parameters are

$k_1 = 1.71$	$k_3 = 8.32$	$k_5 = 0.035$	$k_+ = 280$	$k^* = 0.69$
$k_2 = 0.43$	$k_4 = 0.69$	$k_6 = 8.32$	$k_- = 0.69$	$o_{k_s} = 0.0007$

The chemical process is started at $t_0 = 0$ by mixing one mol of P_r with 0.0057 mol of E. The integration process is continued during 321.8122 seconds, this value was chosen by HAIRER & WANNER [4].

The reason that we take this problem as an example in this paper, is not that the problem is so large, but that it can serve as an example for how to model every chemical reaction scheme. In the following, we give a general recipe for this modeling procedure and then illustrate every step of it by carrying it out for problem HIRES.

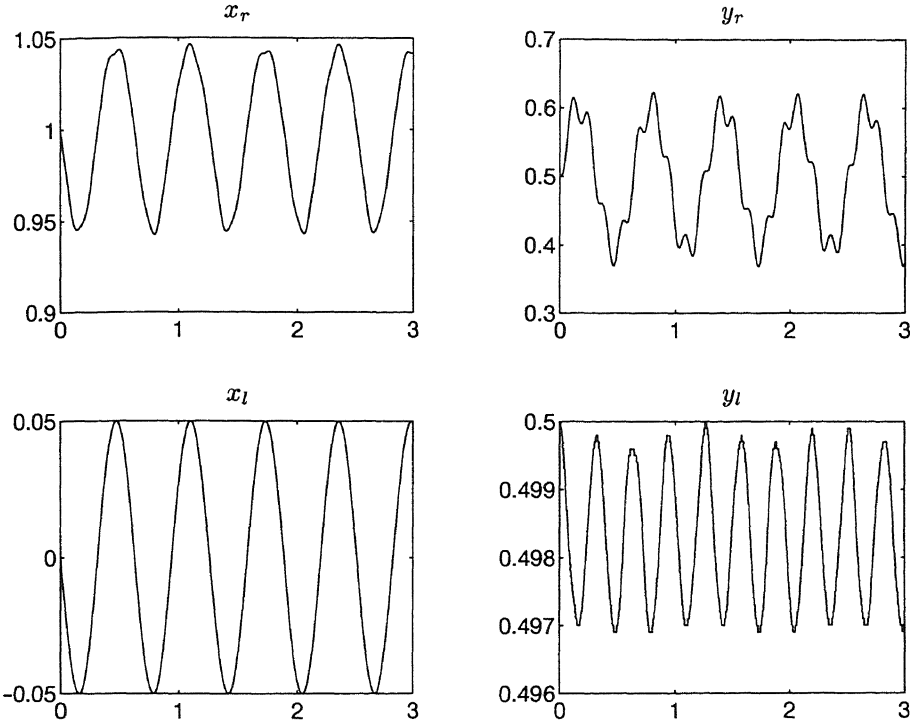


FIGURE 4. The coordinates (x_r, y_r) and (x_l, y_l) for the car axis problem plotted as function of time in seconds.

4.2. The modeling of chemical reactions

Every reaction scheme of the type shown in Figure 5 can be described by a set of species $\mathcal{S}$, a set of nodes $\mathcal{N}$, a transition matrix $\mathcal{M}$ and a source term vector τ. The set $\mathcal{S}$ contains all species that influence the chemical process. For problem HIRES, $\mathcal{S}$ is given by

$$\mathcal{S} = \{\mathrm{P_r}, \mathrm{P_{fr}}, \mathrm{P_rX}, \mathrm{P_{fr}X}, \mathrm{P_rX'}, \mathrm{P_{fr}X'}, \mathrm{P_{fr}X'E}, \mathrm{E}\}.$$

The species X, X′ and $\mathrm{P_{fr}}'$ are not in $\mathcal{S}$, the receptors, because they are assumed to abound, and $\mathrm{P_{fr}}'$ because its amount can be derived from the conservation of the total amount of phytochrome.

The set $\mathcal{N}$ contains all the possible mixtures that react or are formed by a reaction. If we view a reaction scheme as a graph, then the nodes of this graph are in $\mathcal{N}$. For problem HIRES, $\mathcal{N}$ reads

$$\mathcal{N} = \{\{\mathrm{P_r}\}, \{\mathrm{P_{fr}}\}, \{\mathrm{P_rX}\}, \{\mathrm{P_{fr}X}\}, \{\mathrm{P_rX'}\}, \{\mathrm{P_{fr}X'}\}, \{\mathrm{E}, \mathrm{P_rX'}\}, \{\mathrm{P_{fr}X'E}\},$$
$$\{\mathrm{P_{fr}X'}, \mathrm{E}\}, \{\mathrm{E}\}\}.$$

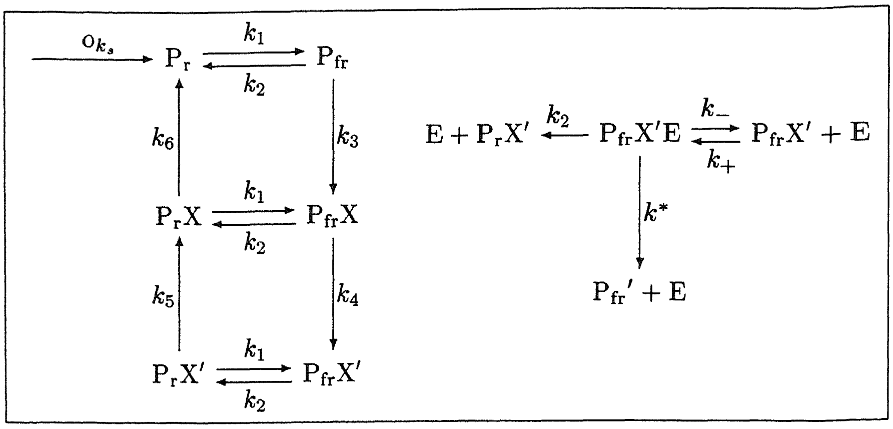


FIGURE 5. Reaction scheme for problem HIRES.

For the specific HIRES reaction scheme, it does not occur that a node contains more than one entity of a species. We remark that if such a situation would arise, then one would have to list the species two times in the definition of the node. For example, for the reaction

$$2A \xrightarrow{k} B$$

the left node would read $\{A, A\}$.

In the graph interpretation, the matrix $\mathcal{M}$ contains the arcs of the graph. If entry $\mathcal{M}_{ij}$ is non-zero, then there is a reaction from the ith node in $\mathcal{N}$ to the jth node in $\mathcal{N}$ with reaction constant $\mathcal{M}_{ij}$. For problem HIRES, the matrix $\mathcal{M}$ is given by

$$\mathcal{M} = \begin{bmatrix} 0 & k_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ k_2 & 0 & 0 & k_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ k_6 & 0 & 0 & k_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_2 & 0 & 0 & k_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_5 & 0 & 0 & k_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & k_2 & 0 & k_- & k^* \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_+ & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_+ & 0 & 0 \end{bmatrix}.$$

Notice that the reaction scheme of problem HIRES has two disjunct parts, which is reflected in the block diagonal form of $\mathcal{M}$.

It may happen that certain species are continuously added to the chemical process. These added quantities are called source terms and are the entries of the vector r, whose dimension equals the cardinality of $\mathcal{S}$. For problem HIRES, there is only a source term for P_r, which is the first element of $\mathcal{S}$, so that the vector τ takes the form

$$\tau = (o_{k_s}, 0, 0, 0, 0, 0, 0, 0)^T.$$

Once $\mathcal{S}$, $\mathcal{N}$, $\mathcal{M}$ and τ are available, it is possible to write down the differential equations that describe the change of the concentrations of the species in time. Every non-zero element $\mathcal{M}_{ij}$ describes a chemical reaction, whose velocity, denoted by v_{ij} satisfies

$$v_{ij} = \mathcal{M}_{ij} \prod_{\{k | S_k \in \mathcal{N}_i\}} [S_k], \tag{10}$$

where $[S_k]$ denotes concentration of S_k, the kth species in $\mathcal{S}$. Formula 10 states that the velocity is proportional to the concentrations of all species in the node, from which the reaction evolves, and the proportionality constant is $\mathcal{M}_{ij}$.

The concentration $[S_k]$ decreases as a result of every reaction that evolved from a node that contains S_k. Vice versa, it increases whenever S_k is in a node, to which a reaction leads. The decrease and increase take place at a rate that equals the velocities of the corresponding reaction. Concentration $[S_k]$ has an extra increase if the kth element of τ is non-zero. This leads to the following differential equation for $[S_k]$:

$$\mathrm{d}[S_k]/\mathrm{d}t = \sum_i \sum_{\{j | \mathcal{M}_{ij} \neq 0 \ \wedge S_k \in \mathcal{N}_j\}} v_{ij} - \sum_j \sum_{\{i | \mathcal{M}_{ij} \neq 0 \ \wedge S_k \in \mathcal{N}_i\}} v_{ij} + \tau_k.$$

For problem HIRES, this results in the following differential equations:

$$\begin{aligned}
\mathrm{d}[P_r]/\mathrm{d}t &= -k_1[P_r] + k_2[P_{fr}] + k_6[P_rX] + o_{k_s}, \\
\mathrm{d}[P_{fr}]/\mathrm{d}t &= k_1[P_r] - (k_2 + k_3)[P_{fr}], \\
\mathrm{d}[P_rX]/\mathrm{d}t &= -(k_1 + k_6)[P_rX] + k_2[P_{fr}X] + k_5[P_rX'], \\
\mathrm{d}[P_{fr}X]/\mathrm{d}t &= k_3[P_{fr}] + k_1[P_rX] - (k_2 + k_4)[P_{fr}X], \\
\mathrm{d}[P_rX']/\mathrm{d}t &= -(k_1 + k_5)[P_rX'] + k_2[P_{fr}X'] + k_2[P_{fr}X'E], \\
\mathrm{d}[P_{fr}X']/\mathrm{d}t &= k_4[P_{fr}X] + k_1[P_rX'] - k_2[P_{fr}X'] + k_-[P_{fr}X'E] - k_+[P_{fr}X'][E], \\
\mathrm{d}[P_{fr}X'E]/\mathrm{d}t &= -(k_2 + k_- + k_*)[P_{fr}X'E] + k_+[P_{fr}X'][E], \\
\mathrm{d}[E]/\mathrm{d}t &= (k_2 + k_- + k_*)[P_{fr}X'E] - k_+[P_{fr}X'][E].
\end{aligned}$$

4.3. The solution

Solving the HIRES problem numerically yielded the following results: Table 2 shows the run characteristics; Figure 6 and Figure 7 show the work-precision diagram and the behavior of the solution over the integration interval, respectively.

From Figure 7 we see that after about 320 seconds, the concentrations of the species in $\mathcal{S}$ containing P are very small, which indicates that almost all phytochrome has been transformed to P_{fr}'. (Remember that the amount of P_{fr}' can be derived from the conservation of the total amount of phytochrome.)

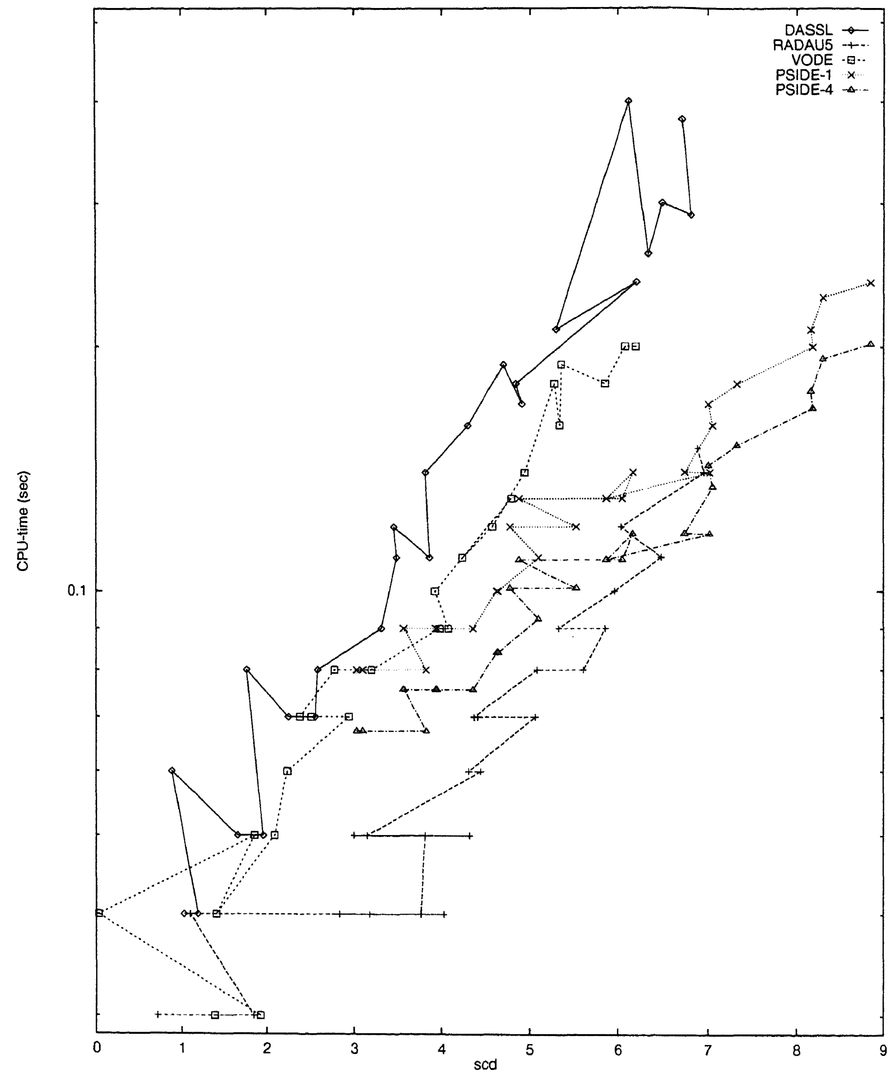


FIGURE 6. Work-precision diagram for problem HIRES.

solver	rtol	scd	steps	accept	# f	# Jac	# LU	CPU
DASSL	10^{-4}	1.03	108	99	173	31		0.04
	10^{-7}	3.87	320	309	473	40		0.11
	10^{-10}	6.70	1150	1134	1588	55		0.38
RADAU5	10^{-4}	0.72	42	33	333	21	41	0.03
	10^{-7}	4.31	79	72	684	31	61	0.06
	10^{-10}	6.88	203	202	1684	61	100	0.15
VODE	10^{-4}	1.39	133	131	191	10	25	0.03
	10^{-7}	3.98	415	390	608	9	70	0.09
	10^{-10}	6.20	933	880	1224	15	134	0.20
PSIDE	10^{-4}	3.03	43	37	665	20	168	0.08
	10^{-7}	4.88	68	60	1208	25	252	0.13
	10^{-10}	8.85	152	151	2528	35	344	0.24

TABLE 2. Run characteristics for problem HIRES (atol = rtol and for RADAU5: h0 = rtol/100).

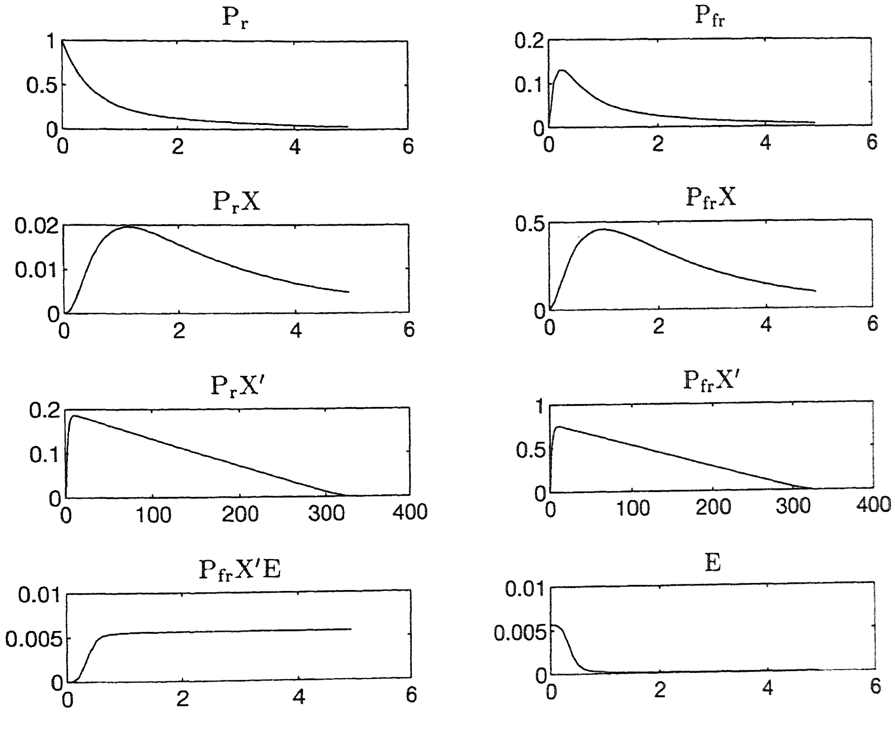


FIGURE 7. Concentration in mols for problem HIRES plotted as function of time in seconds.

References

1. Peter N. Brown, Alan C. Hindmarsh, and George D. Byrne (1992). *VODE: A variable coefficient ODE solver.* Available at `http://www.netlib.org/ode/vode.f`.
2. E. Hairer, C. Lubich, and M. Roche (1989). *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods.* Lecture Notes in Mathematics 1409. Springer-Verlag.
3. E. Hairer and G. Wanner (1996). *RADAU5.* Available at `ftp://ftp.unige.ch/pub/doc/math/stiff/radau5.f`.
4. E. Hairer and G. Wanner (1996). *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition.
5. W.M. Lioen, J.J.B. de Swart, and W.A. van der Veen (1998). *Test Set for IVP Solvers.* Available at `http://www.cwi.nl/cwi/projects/IVPtestset/`.
6. L.R. Petzold (1991). *DASSL: A Differential/Algebraic System Solver.* Available at `http://www.netlib.org/ode/ddassl.f`.
7. E. Schäfer (1975). A new approach to explain the 'high irradiance responses' of photomorphogenesis on the basis of phytochrome. *J. of Math. Biology*, 2:41–56.
8. S. Schneider (1994). *Intégration de systèmes d'équations différentielles raides et différentielles-algébriques par des méthodes de collocations et méthodes générales linéaires.* PhD thesis, Université de Genève.
9. J.J.B. de Swart (1997). *Parallel Software for Implicit Differential Equations.* PhD thesis, University of Amsterdam.
10. J.J.B. de Swart, W.M. Lioen, and W.A. van der Veen (1997). *PSIDE.* Available at `http://www.cwi.nl/cwi/projects/PSIDE/`.