Test Set for Initial Value Problem Solvers

release 2.4

Francesca Mazzia, Cecilia Magherini

Report 4/2008

Reports

Department of Mathematics

University of Bari

ITALY

via E.Orabona 4

I-70125 Bari (ITALY)

Test Set for Initial Value Problem Solvers

Francesca Mazzia*, Cecilia Magherini**
* *Dipartimento di Matematica, Università degli studi di Bari, Italy*
** *Dipartimento di Matematica Applicata, Università di Pisa*
(mazzia@dm.uniba.it,cecilia.magherini@ing.unipi.it)

Release 2.4 February 2008

Abstract

The test set for IVP solvers presents a collection of Initial Value Problems to test solvers for implicit differential equations. This 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, by helping people working in several branches of scientific disciplines in choosing the code most suitable for their problems. This document contains the descriptive part of the test set. It describes the solvers used in the comparisons, the test problems and their origin, and reports on the behavior of the solvers on these problems. The latest version of this document and the software part of the test set is available via the world wide web at http://www.dm.uniba.it/~testset. The software part serves as a platform on which one can test the performance of a solver on a particular test problem oneself. Instructions how to use this software are in this paper as well. The idea to develop this test set was discussed at the workshop ODE to NODE, held in Geiranger, Norway, 19–22 June 1995 and was developed by the CWI group. After the workshop ANODE01, held in Auckland, New Zealand, 2001, the testset moved to the University of Bari.

1991 Mathematics Subject Classification: Primary: 65Y20, Secondary: 65-04, 65C20, 65L05.
Computing Reviews Classification System: G.1.7, G.4.
*Keywords and Phrases:*test problems, software, IVP, IDE, ODE, DAE.

Acknowledgements: This work has been supported by the project "Metodi numerici e software per equazioni differenziali ordinarie: problemi a valori iniziali ed al contorno", sponsored by the University of Bari (2006-2008) and by the PRIN 2004-2006 project "Metodi numerici e software matematico per le applicazioni". We thanks the GNCS-INDAM that supported financially the project form 2001 to 2005 and made possible the use of supercomputer facilities in 2002. We

thanks the CWI group, that formerly maintained the testset, for providing useful software. We thank all contributors to this test set, without whom it would not be possible to collect problems from such a wide variety of application fields.

Contents

PART I - SOLVERS

Part II - Problems

ODE Test problems

DAE Test problems

IDE Test problems

I Introduction

I.1 The idea behind this test set

Both engineers and computational scientists alike will benefit greatly from having a standard test set for Initial Value Problems (IVPs) which includes documentation of the test problems, experimental results from a number of proven solvers, and Fortran subroutines providing a common interface to the defining problem functions. Engineers will be able to see at a glance which methods will be most effective for their class of problems. Researchers will be able to compare their new methods with the results of existing ones without incurring additional programming workload; they will have a reference with which their colleagues are familiar. This test set tries to fulfill these demands and tries to set a standard for IVP solver testing. We hope that the following features of this test set will enable the achievement of this goal:

- uniform presentation of the problems,
- ample description of the origin of the problems,
- robust interfaces between problem and drivers,
- portability among different platforms,
- contributions by people from several application fields,
- presence of real-life problems,
- being used, tested and debugged by a large, international group of researchers,
- comparisons of the performance of well-known solvers,
- interpretation of the numerical solution in terms of the application field,
- ease of access and use.

There exist other test sets, e.g., NSDTST and STDTST by Enright & Pryce [EP87], PADETEST by Bellen [Bel92], the Geneva test set by Hairer & Wanner [HW] and the Test Frame for Ordinary Differential Equations by Nowak and Gebauer [NG97], which all have their own qualities.

I.2 Structure of this test set

The test set consists of a descriptive part and a software part. The first part describes solvers and test problems and reports on the behavior of the solvers when applied to these problems. Section II explains how this information is presented. The software serves as a platform to test the performance of a solver on a particular test problem by a user of the test set. In Section IV we specify the format of the Fortran subroutines and explain how to run test problems with the help of drivers that make these codes suitable for runs with a number of solvers. Currently, BIMD, DASSL, GAMD, MEBDFDAE, MEBDFI, PSIDE, RADAU, RADAU5 and VODE are supported.

I.3 How to submit new test problems

We invite people to contribute new test problems to this test set. To restrict the amount of time required for the maintainers of the test set to incorporate new problems, it is important that the submissions are in the prescribed format. Firstly, every problem should have a description containing the 4 sections mentioned in Section II, preferably as a LaTeX-file. Secondly, a set of Fortran subroutines that is necessary for the implementation has to be supplied in the format specified in Section IV

Submissions can be sent by e-mail to `testset@dm.uniba.it`

I.4 How to obtain this test set

The latest release of this test set can be obtained via the WWW page with URL

http://www.dm.uniba.it/~testset ,

The first release of this test set appeared in [LSV96], the second release in [LS98], the other releases in [MI03], [MMI06].

I.5 Acknowledgements

We gratefully acknowledge Jacek Kierzenka for his help in defining the interface that that allow the use of the IVP test set problems in the MATLAB environment, the CWI group that set up the first two versions of the testset: P.J. van der Houwen , W. Hoffmann, B.P. Sommeijer, W.M. Lioen, W.A. van der Veen, J.J.B. de Swart, J.E. Frank. In particular we wish to thank P.J. van der Houwen and Walter Lioen, who helped us during the installation procedure.

I.6 People involved

This test set is maintained by the INdAM Bari unit project group Codes and test problems for Differential Equations (coordinator F. Mazzia). The revision 2.4 has been sponsored by the project "Metodi numerici e software per equazioni differenziali ordinarie: problemi a valori iniziali ed al contorno" of the University of Bari (2006). The revision 2.3 has been sponsored by the project PRIN 2004 "Metodi numerici e software matematico per le applicazioni" (coordinator L. Brugnano, local coordinator F. Mazzia) and by the project "Metodi Numerici per equazioni differenziali" (coordinator P. Amodio), sponsored by the University of Bari. In January 2002 a steering committee of A. Bellen (Università di Trieste, Italy) , J. R. Cash (Imperial College, London, U.K.), E. Hairer (Université de Genève, Switzerland), F. Krogh (Math à la Carte, Tujiunga, California, U.S.A), L. Petzold (University of California, Snata Barbara, U.S.A), B. Simeon, G. Soderlind (Lund University,Sweden), D. Trigiante (Università di Firenze, Italy) and P.J. van der Houwen (formerly at CWI, Amsterdam, The Netherlands) has been set up to oversee this project.

References

[Bel92] A. Bellen. PADETEST: a set of real-life test differential equations for parallel computing. Technical Report 103, Dipartimento di Scienze Matematiche, Università di Trieste, 1992.

[EP87] W.H. Enright and J.D. Pryce. Two Fortran packages for assessing initial value methods. *ACM Transactions on Mathematical Software*, 13-I:1–27, 1987.

[HW] E. Hairer and G. Wanner. *Testset of Stiff ODEs*. Geneva. Available at http://www.unige.ch/math/folks/hairer/testset/testset.html.

[LS98] W.M. Lioen and J.J.B. de Swart. *Test Set for Initial Value Problem Solvers*, dec 1998. Available at http://www.dm.uniba.it/~testset.

[LSV96] W.M. Lioen, J.J.B. de Swart, and W.A. van der Veen. Test set for IVP solvers. Technical Report NM-R9615, CWI, Amsterdam, 1996.

[MI03] F. Mazzia and F. Iavernaro. Test set for initial value problem solvers. Technical Report 40, Department of Mathematics, University of Bari, 2003. Available at http://www.dm.uniba.it/~testset.

[MMI06] F. Mazzia, C. Magherini, and F. Iavernaro. Test set for initial value problem solvers. Technical Report 43, Department of Mathematics, University of Bari, 2006. Available at http://www.dm.uniba.it/~testset.

[NG97] Ulrich Nowak and Susanna Gebauer. A new test frame for ordinary differential equations. Technical Report SC 97-68, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, 1997.

II Format of the problem descriptions

Every problem description contains the four sections described below.

II.1 General information

The problem identification is given: the type of problem (IDE, ODE or DAE), its dimension and index. The contributor and any further relevant information are listed too. What is meant here by IDE, ODE, DAE and index, is explained in §IV.

II.2 Mathematical description of the problem

All ingredients that are necessary for implementation are given in mathematical formulas.

II.3 Origin of the problem

A brief description of the origin of the problem, in order to give its physical interpretation. References to the literature are given for further details.

II.4 Numerical solution of the problem

This section contains:

1. **Reference solution at the end of the integration interval.** The values of (some of) the components of a reference solution at the end of the integration interval are listed.

2. **Run characteristics.** Integration statistics, if applicable, of runs with BIMD, DASSL, GAMD, MEBDFDAE, MEBDFI, PSIDE, RADAU, RADAU5, and VODE serve to give insight in the numerical difficulty of the problem.

   The experiments were done on an Alphaserver DS20E, with a 667 MHz EV67 processor. We used the Fortran 90 compiler with optimization: `f90 -O5 <source code>`. If a run did not produce correct results then we report what went wrong.

   The characteristics are in the following format:

   - *solver*
     The name of the numerical solver with which the run was performed.
   - *rtol*
     The user supplied relative error tolerance.
   - *atol*
     The user supplied absolute error tolerance.
   - *h0*
     The user supplied initial step size (if relevant).
   - *scd*
     The scd values denote the minimum number of significant correct digits in the numerical solution at the end of the integration interval, i.e.

     $$\text{scd} := -\log_{10}(\| \text{ relative error at the end of the integration interval } \|_\infty). \qquad (.II.1)$$

     If some components of the solution vector are not taken into account for the computation of the scd value, or if the absolute error is computed instead of the relative error, then this is specified locally.

- *mescd*

$$\text{mescd} := -\log_{10}(\| \text{ absolute error } ./ \text{ (atol./rtol} + |\text{ ytrue }|) \ \|_\infty). \qquad (.\text{II.2})$$

  where the absolute error is computed at the end of the integration interval, atol and rtol are the input tolerances, ytrue is the exact solution at the end of the integration interval and ./ and .∗ are element by element operators. In this case all the components of the solution are taken into account.

- *steps*
  Total number of steps taken by the solver (including rejected steps due to error test failures and/or convergence test failures).

- *accept*
  The number of accepted steps.

- *# f* and *# Jac*
  The number of evaluations of the derivative function and its Jacobians, respectively.

- *# LU*
  The number of LU-decompositions (not for DASSL). The codes, except for RADAU and RADAU5, count the LU-decompositions of systems of dimension d, where d is the dimension of the test problem.

  RADAU and RADAU5 use an s-stage Radau IIA method. For RADAU5, $s = 3$ and for RADAU, $s = 3$, 5 or 7. Every iteration of the inexact Newton process, used for solving systems of non-linear equations, requires the solution of a linear system of dimension sd. By means of transformations, this linear system is reduced to $(s+1)/2$ linear systems of dimension d. Of these systems, one system is real, and $(s-1)/2$ systems are complex. The decompositions of all $(s+1)/2$ linear systems are counted by RADAU and RADAU5 as 1 LU-decomposition.

- *CPU*
  The CPU time in seconds to perform the run on the aforementioned computer. Since timings may depend on other processes (like e.g. daemons), we perform 10 runs, discard the maximum and minimum values and list the medium of the CPU times.

PSIDE – Parallel Software for Implicit Differential Equations – is a Fortran 77 code for solving IDE problems. It is developed for parallel, shared memory computers. The integration characteristics in the tables refer to a one-processor computer. Since PSIDE can do four function evaluations and four linear system solves concurrently on a computer with four processors, one may divide the number of function evaluations, decompositions and solves in the tables by four to obtain the analogous *effective* characteristics for four-processor machines.

3. **Behavior of the numerical solution.** Plots of (some of) the solution components over (part of) the integration interval are presented.

4. **Work-precision diagrams.** For every relevant solver, a range of input tolerances and, if necessary, a range of initial stepsizes, were used to produce plots of the resulting scd or mescd values, defined in Formulas (.II.1) and (.II.2), against the number of CPU seconds needed for the run on the aforementioned computer, with the setting as described before. Here we took again the medium of the CPU times of 10 runs, after discarding the maximum and minimum values. The format of these diagrams is as in Hairer & Wanner [HW96, pp. 166–167, 324–325]. The range of input tolerances and initial stepsizes is problem dependent and specified locally. The input parameters for the runs in the tables with run characteristics are such that these runs appear in the work-precision diagrams as well. The code PSIDE has been performed only on one processor.

We want to emphasize that the reader should be careful with using these diagrams for a mutual comparison of the solvers. The diagrams just show the result of runs with the prescribed input on the specified computer. A more sophisticated setting of the input parameters, another computer or compiler, as well as another range of tolerances might change the diagrams considerably.

References

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems*. Springer-Verlag, second revised edition, 1996.

III Format of the solver descriptions

Every solver description contains the four sections described below.

III.1 General information

The name of the solver, the type of problem it solves (ODE, IDE or DAE), the authors, the date of the first release, the language, the official link where it is possible to retrieve the software and any further relevant information are listed. What is meant here by IDE, ODE, DAE and index, is explained in §IV.

III.2 Numerical method

General details about the numerical method implemented in the code and references to the literature are given.

III.3 Implementation details

A brief description of the implementation choices used in the codes, like the step variation strategy, the numerical solution of linear and non linear systems and any other useful information together with references to the literature are given.

III.4 How to solve test problems with the solver

This section contains a description of the compiling sequence and explains how to solve test problems in the test set format.

IV The software part of the test set

IV.1 Classification of test problems

We have categorized the test problems in three classes: IDEs, ODEs and DAEs.

In this test set, we call a problem an **IDE** (system of Implicit Differential Equations) if it is of the form

$$\begin{aligned} &f(t, y, y') = 0, \qquad t_0 \le t \le t_{\text{end}}, \\ &y, f \in \mathbf{R}^d, \\ &y(t_0) \text{ and } y'(t_0) \text{ are given.} \end{aligned}$$

A problem is named an **ODE** (system of Ordinary Differential Equations), if it has the form

$$\begin{aligned} &y' = f(t, y), \qquad t_0 \le t \le t_{\text{end}}, \\ &y, f \in \mathbf{R}^d, \\ &y(t_0) \text{ is given,} \end{aligned}$$

whereas the label **DAE** is given to problems which can be cast in the form

$$\begin{aligned} &My' = f(t, y), \qquad t_0 \le t \le t_{\text{end}}, \\ &y, f \in \mathbf{R}^d, \qquad M \in \mathbf{R}^{d \times d}, \\ &y(t_0) \text{ and } y'(t_0) \text{ are given,} \end{aligned}$$

where M is a constant, possibly singular matrix. Note that ODEs and DAEs are subclasses of IDEs.

IV.2 How to perform tests

You can perform one of the following types of tests:

- solve test set problems with solvers that are supported in the test set,
- solve test set problems with your own solver,
- solve your own problem with solvers that are supported in the test set,
- solve a test set problem using the web facility,
- solve your own problem using the web facility,
- solve test set problems using a MATLAB solver,
- solve you own problem in the test set format using a MATLAB solver.

For the first five types of tests, four types of codes are involved: a solver, a driver, a problem code and auxiliary routines, for the last two types of tests the matlab interface of the problem is generated using two axiliary routines. The solvers available are described in §I-1-1–I-9-1. Currently, there are 9 solvers available:

1. BIMD for ODEs and DAEs of index less than or equal to 3,
2. DASSL for ODEs and IDEs/DAEs of index less than or equal to 1,
3. GAMD for ODEs and DAEs of index less than or equal to 3,
4. MEBDFDAE for ODEs and DAEs of index less than or equal to 3,
5. MEBDFI for ODEs and IDEs/DAEs of index less than or equal to 3,

6. PSIDE for ODEs and IDEs/DAEs of index upto at least 3,
7. RADAU for ODEs and DAEs of index less than or equal to 3,
8. RADAU5 for ODEs and DAEs of index less than or equal to 3, and
9. VODE for ODEs.

These solvers can be obtained via [MM08] in the files `bimd.f`, `ddassl.f`, `gamd.f90`, `mebdfd.f`, `mebdfi.f`, `pside.f`, `radau.f`, `radau5.f` and `vode.f`. These files contain versions of the solvers with which the numerical experiments were conducted. The official links to the solvers, which possibly direct to more recent versions, can be found at [MM08] too.

The drivers `bimdd.f`, `dassld.f`, `gamdd.f`, `mebdfdaed.f`, `mebdfid.f`, `psided.f`, `radaud.f`, `radau5d.f` and `voded.f`, which are available at [MM08], are such that runs can be performed that solve the problem numerically with the aforementioned solvers.

For every test problem, the file `problem.f` contains a set of nine Fortran 77 subroutines defining the problem. Although the format of the subroutines is the same for all three classes, the meaning of the arguments may depend on the problem class. Section IV.3 describes the format of the problem codes.

The auxiliary linear algebra routines for the solvers are in `bimda.f`, `dassla.f`, `gamda.f90`, `psidea.f`, `radaua.f` (for both RADAU and RADAU5) and `vodea.f`. For MEBDFDAE/MEBFI, the linear algebra routines are included in `mebdfdae.f`/`mebdfi.f`. The auxiliary file `report.f` contains a user interface. All these files are available at [MM08] as well.

IV.2.1 How to solve test problems with available solvers

Compiling

```
f77  -o dotest  solverd.f  problem.f  solvera.f  solver.f  report.f,
```

for the solvers written in Fortran 77, will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`. A complete description of each solver together with some examples are reported in the SOLVERS sections §I-2-1–I-9-1. A `makefile` is also available in the [MM08] to help in the compilation steps.

IV.2.2 How to solve test problems with your own solver

The following guidelines serve to test your own solver with the test set problems.

- Write your own solver in a format similar to existing solvers in the file `own.f`.
- (Optional) You may like to put the linear algebra subroutines in a separate file `owna.f`. In this way you can, for example, use the linear algebra of an existing solver.
- Write driver subroutines in the file `ownd.f`. If the format of your solver is similar to that of a solver that is already available in the test set, then this will only require minor modifications of the driver routines of that solver.
- Adjust the file `report.f` as indicated in the comment lines of this file. This will only be a minor modification.
- Compiling

  ```
  f77  -o dotest  ownd.f  problem.f  own.f  owna.f  report.f,
  ```

  will yield an executable `dotest` that solves the problem, of which the Fortran routines are in the file `problem.f`

IV.2.3 How to solve your own problem with available solvers

The following guidelines serve to solve your own problem with the solvers that are supported in the test set.

- Write your own problem in a format similar to that of the test set problems in the file `newprob.f`. This format is described precisely in Section IV.3.
- Adjust the file `report.f` as indicated in the comment lines of this file. This will only be a minor modification.
- To solve your problem with, for example, DASSL, compiling

  ```
  f77  -o dotest  dassld.f  newprob.f  ddassl.f  dassla.f  report.f,
  ```

  will give you the desired executable `dotest`.

IV.2.4 How to solve a test set problem using the web facility

In [MM08], following the link "compile and run on line" it is possible to solve a test set problem on-line, using the supported solvers. The user input are the relative tolerance *rtol*, the absolute tolerance *atol* and the initial stepsize *h0* for the solvers that need it. As a results the solution computed in the last point, the *scd* and *mescd* and some integration characteristics, as described in §II.4, are displayed. The plots of some component of the solution are also visualized.

IV.2.5 How to solve your own problem using the web facility

In [MM08], following the link "compile and run on line" it is possible to upload a file containing the subroutines describing the problem written using the format described precisely in §IV.3. Then it is possible to choose one of the supported solver for the solution of the problem. The user input are the relative tolerance *rtol*, the absolute tolerance *atol* and the initial stepsize *h0* for the solvers that need it. As a results the solution computed in the last point, the *scd* and *mescd* if the reference solution is available and some integration characteristics, as described in §II.4, are displayed. The plots of the components of the solution defined in the subroutine `setoutput` are also visualized.

IV.2.6 How to solve test set problems using a MATLAB solver

The MATLAB [Mat] function `minterface.m` together with the fortran function `matlab_interface.F`, allow to construct the mex files to run problems in the MATLAB environment. The only restriction is that you need to put the problems and the auxiliary routines in the correct directory. We suggest to download the complete distribution tree of the IVP test set in [MM08] if you want to use the matlab interface.

The MATLAB instruction:

```
MPROB = minterface(problem)
```

returns a function handle to a MEX-Function interface to problem `problem`. If needed, the Fortran MEX-Function interface is automatically generated and compiled, you need a Fortran compiler compatible with the MATLAB environment to complete the compilation steps. Moreover, before using, for the first time, this utilities, at the MATLAB prompt type

```
mex -setup
```

and select the Fortran compiler you want to use.

The interface `mprob` supports the following calling sequences:

```
PROB = MPROB('Prob')
[YO,YPO,CONSIST] = MPROB('Init',NEQ,TO)
[ATOL,RTOL]      =  MPROB('Tolerances',NEQN,ATOL,RTOL)
[F,IERR,RPAR,IPAR] = MPROB('Feval',NEQ,T,Y,YP,RPAR,IPAR)
[J,IERR,RPAR,IPAR] = MPROB('Jeval',LDIM,NEQ,T,Y,YP,RPAR,IPAR)
[M,IERR,RPAR,IPAR] = MPROB('Meval',LDIM,NEQ,T,Y,YP,RPAR,IPAR)
Y = MPROB('Solut',NEQ,TFINAL)
[MESCD,SCD] = MPROB('Report',NEQ,YREF,Y,PROBNM,TOLVEC,ATOL,RTOL)
MPROB('Help')
```

The input parameters are the same defined in IV.3 for the fortran functions defining the problem.

The function `odetest.m` contains a user interface to run and compile the problems in the MATLAB environment. As an example the instruction:

```
>> [sol,stats] = odetest(problem,'ode15s',1e-5,1e-4,1)
```

solves the problem using the matlab solver 'ode15s', with absolute tolerance equal to 1e-5, relative tolerance equal to 1e-4, the first component of the solution is plotted using the MATLAB function 'odeplot'. The output variable sol contains information about the solution.

Use the MATLAB help to have information about the input/output parameters of the functions.

IV.2.7 How to solve you own problem in the test set format using a MATLAB solver

Write your own problem in a format similar to that of the test set problems, as described in Section IV.3 in the file newprob.f. Then put the file in the correct directory in the testset distribution. The instruction:

```
MPROB = minterface(newprob)
```

returns a function handle to a MEX-Function interface to problem `newprob`.

The instruction `odetest`

```
>> [sol,stats] = odetest(newprob,'ode15s',1e-5,1e-4,1)
```

will automatically generate the function handle and solve the problem with the MATLAB solver `ode15s`.

IV.3 Format of the problem codes

The eight subroutines that define the problem are called `PROB`, `INIT`, `SETTOLERANCES`, `SETOUTPUT`, `FEVAL`, `JEVAL`, `MEVAL`, and `SOLUT`. The following subsections describe the format of these subroutines in full detail. An additional function `PIDATE` allows to check the problem interface date, for the current release this function should be equal to:

```
integer function pidate()
pidate = 20060828
return
end
```

In the sequel, the variables listed under `INTENT(IN)`, `INTENT(INOUT)`, and `INTENT(OUT)` are input, update and output variables, respectively.

IV.3.1 Subroutine `PROB`

This routine gives some general information about the test problem.

```
      SUBROUTINE PROB(FULLNM,PROBLM,TYPE,
     +                NEQN,NDISC,T,
     +                NUMJAC,MLJAC,MUJAC,
     +                NUMMAS,MLMAS,MUMAS,
     +                IND)
      CHARACTER*(*) FULLNM, PROBLM, TYPE
      INTEGER NEQN,NDISC,MLJAC,MUJAC,MLMAS,MUMAS,IND(*)
      DOUBLE PRECISION T(0:*)
      LOGICAL NUMJAC, NUMMAS
C     INTENT(OUT) FULLNM,PROBLM,TYPE,NEQN,NDISC,T,NUMJAC,MLJAC,
C    +            MUJAC,NUMMAS,MLMAS,MUMAS,IND
```

Meaning of the arguments:

`FULLNM`
: This character string contains the long name of the problem, e.g. `Chemical Akzo Nobel problem`.

`PROBLM`
: This character string contains the short name of the problem, e.g. `chemakzo`, and corresponds to the name of the Fortran source file.

`TYPE`
: This character string takes the value `IDE`, `ODE` or `DAE`, depending on the type of problem.

`NEQN`
: The dimension d of the problem, which is the number of equations to be solved.

`NDISC`
: The number of discontinuities in time of the function f or its derivative. The solver is restarted at every such discontinuity by the driver.

`T`
: An array containing time points.

  - If `NDISC .EQ. 0`, then `T(0)` contains t_0 and `T(1)` contains t_{end}.
  - If `NDISC .GT. 0`, then `T(0)` contains t_0, `T(NDISC+1)` contains t_{end} and `T(1)` ... `T(NDISC)` are the time points where the function f or its derivative has a discontinuity in time.

`NUMJAC`
: To solve the problem numerically, it is necessary to use the partial derivative $J := \partial f/\partial y$. If J is available analytically, then `NUMJAC = .FALSE.` and J is provided via subroutine `JEVAL`. If J is not available, then `NUMJAC = .TRUE.` and `JEVAL` is a dummy subroutine. In this case, the solvers approximate J by numerical differencing.

`MLJAC` and `MUJAC`
: These integers contain information about the structure of $J := \partial f/\partial y$. If J is a full matrix, then `MLJAC = NEQN`, otherwise `MLJAC` and `MUJAC` equal the number of nonzero lower co-diagonals and the number of nonzero upper co-diagonals of J, respectively.

`NUMMAS`
: Only relevant for IDEs.

- For IDEs, it is necessary to use the partial derivative $M := \partial f/\partial y'$. If M is available analytically, then `NUMMAS = .FALSE.` and M is provided via subroutine `MEVAL`. If M is not available, then `NUMMAS = .TRUE.` and `MEVAL` is a dummy subroutine. In this case, the solvers have to approximate M by numerical differencing.
- For DAEs and ODEs, `NUMMAS` is not referenced.

`MLMAS` and `MUMAS`
: These integers contain information about the structure of the constant matrix M (for DAEs) or the matrix $M := \partial f/\partial y'$ (for IDEs).

- For IDEs and DAEs: If M is a full matrix, then `MLMAS = NEQN`, otherwise `MLMAS` and `MUMAS` equal the number of nonzero lower co-diagonals and the number of nonzero upper co-diagonals of M, respectively.
- For ODEs, `MLMAS` and `MUMAS` are not referenced.

`IND`
: Connected to IDEs and DAEs is the concept of index.

- For ODEs, `IND` is not referenced.
- For IDEs and DAEs, `IND` is an array of length `NEQN` and `IND(I)` specifies the index of variable `I`.

IV.3.2 Subroutine `INIT`

This routine contains the initial values $y(t_0)$ and $y'(t_0)$.

```
      SUBROUTINE INIT(NEQN,T,Y,YPRIME,CONSIS)
      INTEGER NEQN
      DOUBLE PRECISION T,Y(NEQN),YPRIME(NEQN)
      LOGICAL CONSIS
C     INTENT(IN)  NEQN,T
C     INTENT(OUT) Y,YPRIME,CONSIS
```

Meaning of the arguments:

`NEQN`
: The dimension of the problem.

`Y(NEQN)`
: Contains the initial value $y(t_0)$.

`YPRIME(NEQN)`
: Only relevant for IDEs and DAEs.

- For IDEs and DAEs, `YPRIME` contains the initial value $y'(t_0)$.
- For ODEs, `YPRIME` is not set. If needed by the solver, it is computed in the driver as $y'(t_0) = f(t_0, y_0)$.

`CONSIS`
: Only relevant for IDEs and DAEs.

- For IDEs and DAEs, `CONSIS` is a switch for the consistency of the initial values. If `CONSIS .EQ. .TRUE.`, then $y(t_0)$ and $y'(t_0)$ are assumed to be consistent. If `CONSIS .EQ. .FALSE.`, then $y(t_0)$ and $y'(t_0)$ are possibly inconsistent. Solvers with a facility to compute consistent initial values internally, will try to do so in this case. Currently, all problems in the test set have consistent initial values.
- For ODEs, `CONSIS` is not referenced.

IV.3.3 Subroutine `SETTOLERANCES`

This routine defines the input tolerances `RTOL` and `ATOL`.

```
      SUBROUTINE SETTOLERANCES(NEQN,RTOL,ATOL,TOLVEC)
      INTEGER NEQN
      LOGICAL TOLVEC
      DOUBLE PRECISION RTOL(NEQN), ATOL(NEQN)
C     INTENT(IN)    NEQN
C     INTENT(INOUT) RTOL, ATOL
C     INTENT(OUT)   TOLVEC
```

Meaning of the arguments:

`NEQN`
The dimension of the problem.

`RTOL`
Contains the relative tolerances.

- In input contains the value `RTOL(1)`.
- In output could contain a vector valued `RTOL`, with different values for the relative tolerances in each component.

`ATOL`
Contains the absolute tolerances.

- In input contains the value `ATOL(1)`.
- In output could contain a vector valued `ATOL`, with different values for the absolute tolerances in each component.

`TOLVEC`
Logical output variable.

- `TOLVEC = .TRUE.` if all the component of `RTOL` and `ATOL` are initialized.
- `TOLVEC = .FALSE.` if only the first component of `RTOL` and `ATOL` is initialized.

IV.3.4 Subroutine `SETOUTPUT`

This routine contains information about the required output.

```
      SUBROUTINE SETOUTPUT(NEQN,SOLREF,PRINTSOLOUT,
     +                     NINDSOL,INDSOL)

      LOGICAL SOLREF, PRINTSOLOUT
      INTEGER NEQN, NINDSOL
```

```
      INTEGER INDSOL(NEQN)
C     INTENT(IN)    NEQN
C     INTENT(OUT)   NINDSOL, INDSOL(NEQN), PRINTSOLOUT, SOLREF
```

Meaning of the arguments:

`NEQN`
The dimension of the problem.

`SOLREF`
Contains information about the reference solution.

- `SOLREF = .TRUE.` means that the reference solution is available in the function solut.
- `SOLREF = .FALSE.` means that the reference solution is not available, the subroutine `SOLOUT` must be a dummy subroutine.

`PRINTSOLOUT`
Contains information about the required output.

- `PRINTSOLOUT=.TRUE.` means that some components of the intermediate computed values of the solution are printed in the output file called problemSOLVER.txt.
- This option is not activated for the code `pside`. Moreover a MATLAB file called problemSOLVER.m and a SCILAB file called problemSOLVER.sci are generated as utilities to generate the plots of the printed components of the solution.
- `PRINTSOLOUT=.FALSE.` means that no intermediate values are printed.

`NINDSOL`
If `PRINTSOLOUT=.TRUE.`, `NINDSOL` contains the number of components to be printed.

`INDSOL`
If `PRINTSOLOUT=.TRUE.`, `INDSOL(1:NINDSOL)` contains the index of the `NINDSOL` components to be printed.

IV.3.5 Subroutine `FEVAL`

This subroutine evaluates the function f.

```
      SUBROUTINE FEVAL(NEQN,T,Y,YPRIME,F,IERR,RPAR,IPAR)
      INTEGER NEQN,IERR,IPAR(*)
      DOUBLE PRECISION T,Y(NEQN),YPRIME(NEQN),F(NEQN),RPAR(*)
C     INTENT(IN)    NEQN,T,Y,YPRIME
C     INTENT(INOUT) RPAR,IPAR
C     INTENT(OUT)   F,IERR
```

Meaning of the arguments:

`NEQN`
The dimension of the problem.

`T`
The time point where the function is evaluated.

`Y(NEQN)`
The value of y in which the function is evaluated.

`YPRIME(NEQN)`
Only relevant for IDEs.

- For IDEs, this is the value of y' in which the function f is evaluated.
- For ODEs and DAEs, `YPRIME` is not referenced.

`F(NEQN)`
The resulting function value f(`T`, `Y`) (for ODEs and DAEs), or f(`T`, `Y`, `YPRIME`) (for IDEs).

`IERR`
`IERR` is an integer flag which is always equal to zero on input. Subroutine `FEVAL` sets `IERR = -1` if `FEVAL` can not be evaluated for the current values of `T`, `Y` and `YPRIME`. Some solvers have the facility to attempt to prevent the occurrence of `IERR = -1`, or return to the driver in that case.

`IERR` has an analogous meaning in subroutines `JEVAL` and `MEVAL`.

`RPAR` and `IPAR`
`RPAR` and `IPAR` are double precision and integer arrays, respectively, which can be used for communication between the driver and the subroutines `FEVAL`, `JEVAL` and `MEVAL`. If `RPAR` and `IPAR` are not needed, then these parameters are ignored by treating them as dummy arguments.

`RPAR` and `IPAR` have the same meaning in subroutines `JEVAL` and `MEVAL`.

IV.3.6 Subroutine `JEVAL`

This subroutine evaluates the derivative (or Jacobian) of the function f with respect to y.

```
      SUBROUTINE JEVAL(LDIM,NEQN,T,Y,YPRIME,DFDY,IERR,RPAR,IPAR)
      INTEGER LDIM,NEQN,IERR,IPAR(*)
      DOUBLE PRECISION T,Y(NEQN),YPRIME(NEQN),DFDY(LDIM,NEQN),RPAR(*)
C     INTENT(IN)    LDIM,NEQN,T,Y,YPRIME
C     INTENT(INOUT) RPAR,IPAR
C     INTENT(OUT)   DFDY,IERR
```

Meaning of the arguments:

`LDIM`
The leading dimension of the array `DFDY`.

`NEQN`
The dimension of the problem.

`T`
The time point where the derivative is evaluated.

`Y(NEQN)`
The value of y in which the derivative is evaluated.

`YPRIME(NEQN)`
Only relevant for IDEs.

- For IDEs, this is the value of y' in which the derivative $\partial f(t, y, y')/\partial y$ is evaluated.
- For ODEs and DAEs, `YPRIME` is not referenced.

`DFDY(LDIM,NEQN)`
The array with the resulting Jacobian matrix.

- If $\partial f/\partial y$ is a full matrix (`MLJAC` = `NEQN`), then `DFDY(I,J)` contains $\partial f_{\mathrm{I}}/\partial y_{\mathrm{J}}$.
- If $\partial f/\partial y$ is a band matrix (0 $\leq$ `MLJAC` $<$ `NEQN`), then `DFDY(I-J+MUJAC+1,J)` contains $\partial f_{\mathrm{I}}/\partial y_{\mathrm{J}}$ (LAPACK / LINPACK / BLAS storage).

`IERR`, `RPAR` and `IPAR`
See the description of subroutine `FEVAL`.

IV.3.7 Subroutine `MEVAL`

For ODEs, `MEVAL` is not called and a dummy subroutine is supplied. For DAEs, it supplies the constant matrix M. For IDEs, it evaluates the matrix $M := \partial f/\partial y'$.

```
      SUBROUTINE MEVAL(LDIM,NEQN,T,Y,YPRIME,DFDDY,IERR,RPAR,IPAR)
      INTEGER LDIM,NEQN,IERR,IPAR(*)
      DOUBLE PRECISION T,Y(NEQN),YPRIME(NEQN),DFDDY(LDIM,NEQN),RPAR(*)
C     INTENT(IN)    LDIM,NEQN,T,Y,YPRIME
C     INTENT(INOUT) RPAR,IPAR
C     INTENT(OUT)   DFDDY,IERR
```

Meaning of the arguments:

`LDIM`
The leading dimension of the matrix M.

`NEQN`
The dimension of the problem.

`T`
The time point where M is evaluated. (For DAEs, `T` is not referenced.)

`Y(NEQN)`
The value of y in which M is evaluated. (For DAEs, `Y` is not referenced.)

`YPRIME(NEQN)`
The value of y' in which M is evaluated. (For DAEs, `YPRIME` is not referenced.)

`DFDDY(LDIM,NEQN)`
This array contains the constant matrix M (for DAEs) or $M := \partial f/\partial y'$ (for IDEs).

- If M is a full matrix (`MLMAS` = `NEQN`), then `DFDDY(I,J)` contains $M_{\mathrm{I,J}}$ for DAEs and $\partial f_{\mathrm{I}}/\partial y'_{\mathrm{J}}$ for IDEs.
- If M is a band matrix (0 $\leq$ `MLMAS` $<$ `NEQN`), then `DFDDY(I-J+MUMAS+1,J)` contains $M_{\mathrm{I,J}}$ for DAEs and $\partial f_{\mathrm{I}}/\partial y'_{\mathrm{J}}$ for IDEs. (LAPACK / LINPACK / BLAS storage).

`IERR`, `RPAR` and `IPAR`
See the description of subroutine `FEVAL`.

IV.3.8 Subroutine `SOLUT`

This routine contains the reference solution.

```
      SUBROUTINE SOLUT(NEQN,T,Y)
      INTEGER NEQN
      DOUBLE PRECISION T,Y(NEQN)
C     INTENT(IN)  NEQN,T
C     INTENT(OUT) Y
```

Meaning of the arguments:

`NEQN`
: The dimension of the problem.

`T`
: The value of t, in which the reference solution is given (normally t_{end}).

`Y(NEQN)`
: This array contains the reference solution in $t =$ `T`.

IV.4 Format of the solver codes

The following guidelines serve to write a solver that could be easily inserted in the test set.

- Write your own solver in a format similar to existing solvers in the file `own.f`.
- Put the linear algebra subroutines in a separate file `owna.f`.
- Write driver subroutines in the file `ownd.f`. If the format of your solver is similar to that of a solver that is already available in the test set, then this will only require minor modifications of the driver routines of that solver.
- Adjust the file `report.f` as indicated in the comment lines of this file. This will only be a minor modification.

References

[Mat] The Mathworks. Matlab. http://www.mathworks.com/.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

Part I
Solvers

This part contains a brief description of the solvers used in the comparisons. The description is not meant in substitution of the information given by the authors of the solvers, but just to provide the users with some general specifics of the solvers supported and to collect the most useful bibliography.

Also, some suggestions on how to use the codes in combination with the software provided in the test set home page are given.

1 Solver BiMD

1.1 General information

Authors:	C. Magherini and L. Brugnano
first version:	October, 2005
last update:	September, 2006
language:	Fortran 77
availability:	the code BiMD is freely available (in the public domain)
official link:	http://www.math.unifi.it/~brugnano/BiM/index.html
problem type:	ODEs, DAEs up to index 3
IVPtestset files:	solver: bimd.f
	driver: bimdd.f
	auxiliary files: bimda.f (auxiliary routines)

1.2 Numerical method

The code BiMD (written in FORTRAN 77) is based on Blended Implicit Methods of orders 4, 6, 8, 10 and 12. These are a class of L-stable Block Implicit Methods defined as a suitable combination (*blending*) of two equivalent forms of a basic method in order to favorably meet implementation requirements [BT01, BM02, BMM06, Mag04].

1.3 Implementation details

Nonlinear systems are solved by means of an iterative procedure, called *blended iteration*, based on a nonlinear splitting "naturally" associated to the methods. The strategies for the variation of both the stepsize of integration and the order of the method rely on an estimate of the local truncation errors, obtained through a deferred correction-like procedure, and on an estimate of the convergence properties of the blended iteration. Almost all the details concerning the construction of the code are described in [BM04, BM05, BM06]. The style used during the formulation of the code is very similar to the one used in the codes RADAU and GAM, from which the authors imported some subroutines and comments. Moreover, the name and the meaning of a number of input parameters and local variables have been fully inherited from such codes.

1.4 How to solve test problems with BiMD

Compiling

```
f90  -o dotest  bimdd.f  problem.f  bimda.f  bimd.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.1.1 shows what one has to do.

References

[BM02] L. Brugnano and C. Magherini. Blended implementation of block implicit methods for odes. *Appl. Numer. Math.*, 42:19–45, 2002.

[BM04] L. Brugnano and C. Magherini. The bim code for the numerical solution of odes. *J. Comput. Appl. Math.*, 164-165:145–158, 2004.

```
$ f90 -O5 -o dotest bimdd.f hires.f bimda.f  bimd.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using BiMD

 User input:

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

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel    mix - abs,rel
 ---------------------------------       -----      -----      -----   -------------
 y(  1) =  0.7370390869868378E-003        7.04       7.04       3.90
 y(  2) =  0.1442309432867305E-003        7.75       7.75       3.91
 y(  3) =  0.5886726446999230E-004        7.70       7.70       3.47
 y(  4) =  0.1175514405948053E-002        6.86       6.86       3.93
 y(  5) =  0.2382225270095926E-002        5.38       5.38       2.76
 y(  6) =  0.6222129415035646E-002        4.78       4.77       2.57
 y(  7) =  0.2849350956905541E-002        6.19       6.19       3.64
 y(  8) =  0.2850649043094471E-002        6.19       6.19       3.64

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  4.78       4.77       2.57


 using mixed error yields mescd           4.78
 using relative error yields scd                                 2.57

 Integration characteristics:

    number of integration steps          36
    number of accepted steps             33
    number of f evaluations             559
    number of Jacobian evaluations       30
    number of LU decompositions          36

 CPU-time used:                      0.0020 sec
```

FIGURE I.1.1: *Example of performing a test run, in which we solve problem HIRES with BiMD. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[BM05] L. Brugnano and C. Magherini. Some linear algebra issues concerning the implementation of blended implicit methods. *Numer. Linear Alg. Appl.*, 12:305–314, 2005.

[BM06] L. Brugnano and C. Magherini. Economical error estimates for block implicit methods for odes via deferred correction. *Appl. Numer. Math.*, 56:608–617, 2006.

[BMM06] L. Brugnano, C. Magherini, and F. Mugnai. Blended implicit methods for the numerical solution of dae problems. *J. Comput. Appl. Math.*, 189:34–50, 2006.

[BT01] L. Brugnano and D. Trigiante. Block implicit methods for odes. *Recent Trends in Numerical Analysis, D. Trigiante Ed. Nova Science Publ. Inc.*, pages 81–105, 2001.

[Mag04] C. Magherini. *Numerical Solution of Stiff ODE-IVPs via Blended Implicit Methods: Theory and Numerics.* PhD thesis, Dipartimento di Matematica U. Dini, Università degli Studi di Firenze, September 2004.

2 Solver DASSL

2.1 General information

Author:	L. Petzold
first version:	March 15, 1983
last update:	July 11, 2000
language:	Fortran 77
availability:	the code DASSL is freely available (in the public domain)
official link:	http://www.netlib.org/ode/ddassl.f
problem type:	IDEs/DAEs of index less or equal to 1
IVPtestset files:	solver: ddassl.f
	driver: dassld.f
	auxiliary files: dassla.f (auxiliary linear algebra routines)

2.2 Numerical method

This code implements the Backward Differentiation Formulas of orders one through five to solve an IDE for y and y'. Values for y and y' at the initial time must be given as input. These values must be consistent, (that is, if $t_0,\ y_0,\ y_0'$ are the given initial values, they must satisfy $f(t_0, y_0, y_0') = 0$) [BCP96].

2.3 Implementation details

The subroutine solves the system from t_0 to t_{out} (final integration time). It allows to continue the solution to get results at additional t_{out}. This is the interval mode of operation. Intermediate results can also be obtained easily by using the intermediate-output capability. The derivatives are approximated by backward differentiation formulae (BDFs), and the resulting nonlinear system at each time-step is solved by Newton's method. The linear systems are solved using routines from the LINPACK subroutine package. Error handling is accomplished using routines from the SLATEC common mathematical library package. This code is good for stiff ODEs and for DAEs of moderate size, where it is appropriate to treat the Jacobian matrix with dense or banded direct LU decomposition. For large-scale stiff ODE and DAE problems, the user should consider DASPK. For ODE or DAE problems which must stop at the root of a given function of the solution, the user should consider DASKR. The code includes an extensive amount of documentation.

2.4 How to solve test problems with DASSL

Compiling

```
f90  -o dotest  dassld.f  problem.f  ddassl.f  dassla.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

Although DASSL is a code written for problems of index ≤ 1, it can handle some of the higher index problems by adjusting the error control. If possible, this is done in the driver `dassld.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.2.1 shows what one has to do.

```
$ f90 -O5 -o dotest dassld.f hires.f ddassl.f dassla.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using DASSL

 User input:

 give relative error tolerance:
1d-4
 give absolute error tolerance:
1d-4

 Numerical solution:

                                              scd
        solution component                ----------------------------     ignore
                                          mixed       abs        rel    mix - abs,rel
 ---------------------------------        -----      -----      -----   -------------
 y(  1) =  0.7437259735671353E-003         5.18       5.18       2.05
 y(  2) =  0.1455514426118115E-003         5.89       5.89       2.04
 y(  3) =  0.6009984916041035E-004         5.92       5.92       1.69
 y(  4) =  0.1188134706173305E-002         4.90       4.90       1.97
 y(  5) =  0.2577046600086416E-002         3.72       3.72       1.10
 y(  6) =  0.6824947575510993E-002         3.23       3.23       1.03
 y(  7) =  0.2989385921555588E-002         3.86       3.86       1.31
 y(  8) =  0.2710614078444423E-002         3.86       3.86       1.31

 used components for  scd                     8          8          8
 scd of Y (maximum norm)                   3.23       3.23       1.03


 using mixed error yields mescd            3.23
 using relative error yields scd                                 1.03

 Integration characteristics:

    number of integration steps        108
    number of accepted steps            99
    number of f evaluations            173
    number of Jacobian evaluations      31

 CPU-time used:                       0.0010 sec
```

FIGURE I.2.1: *Example of performing a test run, in which we solve problem HIRES with DASSL. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

References

[BCP96] K.E. Brenan, S.L. Campbell, and L.R. Petzold. *Numerical Solution of Initial–Value Problems in Differential–Algebraic Equations.* SIAM, second edition, 1996.

3 Solver GAMD

3.1 General information

Authors:	F. Iavernaro and F. Mazzia
first version:	August 1997 (GAM)
last update:	February, 2006
language:	Fortran 90
availability:	the code GAMD is freely available (in the public domain)
official link:	http://www.dm.uniba.it/~mazzia/ode/readme.html
problem type:	ODEs, DAEs of index less than 3
IVPtestset files:	solver: gamd.f90
	driver: gamdd.f
	auxiliary files: gamda.f90 (auxiliary routines)

3.2 Numerical method

The code GAMD (written in FORTRAN 90) uses the Generalized Adams Methods in block form, of orders 3, 5, 7 and 9. These are A-stable formulae belonging to the class of Boundary Value Methods [BT98, IM99].

3.3 Implementation details

The solution of nonlinear systems is obtained by means of a one-step splitting Newton iteration. The order variation and stepsize selection strategies are based upon an estimation of the local truncation errors for the current, lower and upper order formulae, obtained by means of a deferred correction-like procedure [IM98]. The philosophy and the style used during the formulation of the code are very similar to those characterizing the code RADAU5, from which the authors imported some subroutines, comments and implementation techniques, leaving unchanged the name and the meaning of a number of variables. A preprocessed version of the code GAMD, that allows the user to switch beetwen quadruple and double precision, is also available at the official link http://www.dm.uniba.it/~mazzia/ode/readme.html.

3.4 How to solve test problems with GAMD

Some machines need more virtual memory to compile the subroutine `gamda.f90` ; for example if you are using an ALPHAserver DS20E, with a 667MHz, EV67 processor, execute the following command before the compilation: `ulimit -Sd 241000` . Compiling

```
f90  -o dotest  gamdd.f  problem.f  gamda.f90  gamd.f90  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.3.1 shows what one has to do.

References

[BT98] L. Brugnano and D. Trigiante. *Solving Differential Problems by Multistep Initial and Boundary Value Methods.* Gordon & Breach, Amsterdam, 1998.

```
$ f90 -O5 -o dotest gamdd.f hires.f gamda.f90  gamd.f90 report.f
$ dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using GAMD90

 User input:

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

 Numerical solution:

                                              scd
       solution component              ----------------------------     ignore
                                       mixed       abs        rel    mix - abs,rel
 ---------------------------------     -----      -----      -----   -------------
 y(  1) =  0.7370189658683070E-003      6.95       6.95       3.82
 y(  2) =  0.1442269592313960E-003      7.67       7.67       3.82
 y(  3) =  0.5886363518265143E-004      7.63       7.63       3.40
 y(  4) =  0.1175477661507891E-002      6.76       6.76       3.83
 y(  5) =  0.2381655379215545E-002      5.33       5.33       2.71
 y(  6) =  0.6221249713391935E-002      4.75       4.75       2.55
 y(  7) =  0.2848304918830136E-002      5.77       5.77       3.23
 y(  8) =  0.2851695081169868E-002      5.77       5.77       3.23

 used components for  scd                  8          8          8
 scd of Y (maximum norm)                4.75       4.75       2.55


 using mixed error yields mescd         4.75
 using relative error yields scd                                2.55

 Integration characteristics:

    number of integration steps         29
    number of accepted steps            24
    number of f evaluations            967
    number of Jacobian evaluations      24
    number of LU decompositions         29

 CPU-time used:                        0.0020 sec
```

FIGURE I.3.1: *Example of performing a test run, in which we solve problem HIRES with GAMD. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[IM98] F. Iavernaro and F. Mazzia. Solving ordinary differential equations by generalized adams methods: properties and implementation techniques. *Appl. Num. Math.*, 28:107–126, 1998.

[IM99] F. Iavernaro and F. Mazzia. Block-boundary value methods for the solution of ordinary differential equations. *SIAM J. Sci. Comput.*, 21(1):323–339, 1999.

4 Solver MEBDFDAE

4.1 General information

Author:	J. Cash
first version:	November , 1998
last update:	February, 2006
language:	Fortran 77
availability:	the code MEBDFDAE is freely available (in the public domain)
official link:	http://www.ma.ic.ac.uk/~jcash/IVP_software/mebdftest/mebdfdae.f
problems type:	ODEs and DAEs of index less than or equal to 3
IVPtestset files:	solver: mebdfdae.f driver: mebdfd.f auxiliary files: the linear algebra routines are included in `medbdfdae.f`.

4.2 Numerical method

The code MEBDFDAE uses the Modified Extended Backward Differentiation Formulas of Cash, that increase the absolute stability regions of the classical BDFs [Cas79, Cas83, Cas03, Hin83, HW96]. These methods are A-stable up to the order 4 and stiffly stable for orders up to 9; therefore they are especially suited for the solution of stiff systems of ODEs [CC92]. The orders of the implemented formulae range from 1 to 8.

4.3 Implementation details

The formulae implemented are three-stages general linear methods with the same Jacobian to be used in the Newton iteration for all the stages. Blas and Lapack auxiliary routines are also used. Versions of this solver for the solutions of ODEs are MEBDF and MEBDFSO, the last one is designed to solve stiff Initial Value Problems for very large sparse systems of ODEs, where the linear equation solver is replaced by the sparse solver YSMP [EGSS77]. Extensions of MEBDFDAE for the solution of very large sparse systems of DAEs is given by the solver MEBDFSD, where the sparse solver used is MA28 [I.S77]. A MATLAB translation of MEBDFDAE is available at the official link http://www.ma.ic.ac.uk/~jcash/MATLAB_software/MEBDF.m.

4.4 How to solve test problems with MEBDFDAE

Compiling

```
f90  -o dotest  mebdfdaed.f  problem.f  mebdfdae.f    report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.4.1 shows what one has to do.

References

[Cas79] J. Cash. *Stable Recursions with applications to the numerical solution of stiff systems.* Academic Press, New York, 1979.

[Cas83] J. Cash. The integration of stiff initial value problems in o.d.e.s using modified extended backward differentiation formulae. *Comp. and Maths. with Applics.*, 9:645–657, 1983.

```
$ f90 -O5  -o dotest mebdfdaed.f hires.f mebdfdae.f   report.f
$ dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using MEBDFDAE

 User input:

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

 Numerical solution:

                                              scd
       solution component              ----------------------------     ignore
                                       mixed       abs        rel    mix - abs,rel
 ---------------------------------     -----      -----      -----   -------------
 y(  1) =  0.7324251767207330E-003      5.33       5.33       2.19
 y(  2) =  0.1433221554010029E-003      6.03       6.03       2.19
 y(  3) =  0.5800420518076766E-004      6.05       6.05       1.82
 y(  4) =  0.1166962417102632E-002      5.06       5.06       2.13
 y(  5) =  0.2241753919183594E-002      3.84       3.84       1.22
 y(  6) =  0.5760280012688669E-002      3.32       3.32       1.12
 y(  7) =  0.2767358761415102E-002      4.08       4.08       1.54
 y(  8) =  0.2932641238585708E-002      4.08       4.08       1.54

 used components for  scd                  8          8          8
 scd of Y (maximum norm)                3.32       3.32       1.12


 using mixed error yields mescd         3.32
 using relative error yields scd                                1.12

 Integration characteristics:

    number of integration steps         97
    number of accepted steps            94
    number of f evaluations            168
    number of Jacobian evaluations      21
    number of LU decompositions         21

 CPU-time used:                       0.0020 sec
```

FIGURE I.4.1: *Example of performing a test run, in which we solve problem HIRES with MEBDFDAE. The experiment was done on an ALPHAserver DS20E, with a 667MH EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[Cas03] J. Cash. Efficient numerical methods for the solution of stiff initial-value problems and differential algebraic equations. *Proc. Roy. Soc. London, A*, 459:797–815, 2003.

[CC92] J. Cash and S. Considine. An mebdf code for stiff initial value problems. *Acm Trans Math Software*, pages 142–158, 1992.

[EGSS77] S.C. Eisenstat, M.C. Gursky, M.H. Schultz, and A.H. Sherman. Yale sparse matrix package ii. the nonsymmetric codes. Technical Report 114, Department of Computer Science, Yale University, New Haven, CT, 1977.

[Hin83] Alan C. Hindmarsh. ODEPACK, a systemized collection of ODE solvers. In R. Stepleman et al., editors, *Scientific Computing*, pages 55–64, Amsterdam, 1983. IMACS, North-Holland Publishing Company.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

[I.S77] I.S.Duff. Ma28-a set of fortran subroutines for sparse unsymmetric linear equations. Technical report, Technical Report AERE-R8730, Harwell, 1977.

5 Solver MEBDFI

5.1 General information

Authors:	T.J. Abdulla and J.R. Cash
first version:	October 31, 2003
last update:	February, 2006
language:	Fortran 77
availability:	the code MEBDFI is freely available (in the public domain)
official link:	http://www.ma.ic.ac.uk/~jcash/IVP_software/itest/mebdfi.f
problems type:	ODEs, DAEs and IDEs of index less than or equal to 3
IVPtestset files:	solver: mebdfi.f
	driver: mebdfid.f
	auxiliary files: the linear algebra routines are included in `mebdfi.f`.

5.2 Numerical method

The code MEBDFI is an extension of MEBDFDAE for the solution of implicit differential equations and uses the Modified Extended Backward Differentiation Formulas of Cash, that increase the absolute stability regions of the classical BDFs [Cas79, Cas83, Cas03, Hin83, HW96]. These methods are A-stable up to the order 4 and stiffly stable for orders up to 9; therefore they are especially suited for the solution of stiff systems of ODEs [CC92]. The orders of the implemented formulae range from 1 to 8.

5.3 Implementation details

The formulae implemented are three-stages general linear methods with the same Jacobian to be used in the Newton iteration for all the stages. Blas and Lapack auxiliary routines are also used. A Fortran 95 translation of MEBDFI made by Bill Paxton is available at the official link of MESA (Modules for Experiments in Stellar Astrophysics) http://theory.kitp.ucsb.edu/~paxton/mesa/mesa_doc/index.html.

5.4 How to solve test problems with MEBDFI

Compiling

```
f90  -o dotest  mebdfid.f  problem.f  mebdfi.f    report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.5.1 shows what one has to do.

References

[Cas79] J. Cash. *Stable Recursions with applications to the numerical solution of stiff systems.* Academic Press, New York, 1979.

[Cas83] J. Cash. The integration of stiff initial value problems in o.d.e.s using modified extended backward differentiation formulae. *Comp. and Maths. with Applics.*, 9:645–657, 1983.

```
$ f90 -O5  -o dotest mebdfid.f hires.f mebdfi.f   report.f
$ dotest
  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using MEBDFI

 User input:

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

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel    mix - abs,rel
 ---------------------------------       -----      -----      -----   -------------
 y(  1) =  0.7360756579676240E-003        5.98       5.98       2.84
 y(  2) =  0.1440435009167338E-003        6.69       6.69       2.85
 y(  3) =  0.5867365037055238E-004        6.67       6.67       2.44
 y(  4) =  0.1173828077122226E-002        5.74       5.74       2.81
 y(  5) =  0.2347013337886003E-002        4.41       4.41       1.78
 y(  6) =  0.6023708667056447E-002        3.67       3.67       1.46
 y(  7) =  0.2893696909773767E-002        4.36       4.36       1.81
 y(  8) =  0.2806303090227050E-002        4.36       4.36       1.81

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  3.67       3.67       1.46


 using mixed error yields mescd           3.67
 using relative error yields scd                                 1.46

 Integration characteristics:

    number of integration steps          92
    number of accepted steps             89
    number of f evaluations             311
    number of Jacobian evaluations       18
    number of LU decompositions          18

 CPU-time used:                        0.0010 sec
```

FIGURE I.5.1: *Example of performing a test run, in which we solve problem HIRES with MEBDFI. The experiment was done on an ALPHAserver DS20E, with a 667MH EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[Cas03] J. Cash. Efficient numerical methods for the solution of stiff initial-value problems and differential algebraic equations. *Proc. Roy. Soc. London, A*, 459:797–815, 2003.

[CC92] J. Cash and S. Considine. An mebdf code for stiff initial value problems. *Acm Trans Math Software*, pages 142–158, 1992.

[Hin83] Alan C. Hindmarsh. ODEPACK, a systemized collection of ODE solvers. In R. Stepleman et al., editors, *Scientific Computing*, pages 55–64, Amsterdam, 1983. IMACS, North-Holland Publishing Company.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

6 Solver PSIDE

6.1 General information

Authors:	J.J.B. de Swart, W.M. Lioen, and W.A. van der Veen
first version:	November 28 1997 (version 1.0)
last update:	November 25 1998 (version 1.3)
language:	Fortran 77
availability:	the code PSIDE is freely available (in the public domain)
official link:	http://www.cwi.nl/cwi/projects/PSIDE/
problem type:	IDEs/DAEs of index upto at least 3
IVPtestset files:	solver: pside.f
	driver: psided.f
	auxiliary files: psidea.f (auxiliary linear algebra routines)

6.2 Numerical method

The code uses the four-stage Radau IIA method.

6.3 Implementation details

PSIDE is a Parallel Software for Implicit Differential Equations [SLV97a, SLV97b]. It has been designed for working on shared memory parallel computers, using the OPENMP parallel tools.

The nonlinear systems are solved by a modified Newton process, in which every Newton iterate itself is computed by means of the Parallel Iterative Linear system Solver for Runge-Kutta (PILSRK) proposed in [HS97]. This process is constructed such that the four stage values can be computed simultaneously, thereby making PSIDE suitable for execution on four processors. Full details about the algorithmic choices and the implementation of PSIDE can be found in [SLV97c].

6.4 How to solve test problems with PSIDE

Compiling

```
f90  -o dotest  psided.f  problem.f  pside.f  psidea.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`. In order to have the correct solution, before the compilation, change the auxiliary routine I1MACH and D1MACH, in the file `dassla.f` because they are machine dependent.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.6.1 shows what one has to do.

References

[HS97] P.J. van der Houwen and J.J.B. de Swart. Parallel linear system solvers for Runge–Kutta methods. *Advances in Computational Mathematics*, 7:157–181, 1997.

[SLV97a] J.J.B. de Swart, W.M. Lioen, and W.A. van der Veen. *PSIDE*, December 1997. Available at http://www.cwi.nl/cwi/projects/PSIDE/.

[SLV97b] J.J.B. de Swart, W.M. Lioen, and W.A. van der Veen. *PSIDE Users' Guide*, 1997. Available at http://www.cwi.nl/cwi/projects/PSIDE/.

```
$ f90 -O5 -o dotest psided.f hires.f pside.f psidea.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using PSIDE

 User input:

 give relative error tolerance:
1d-4
 give absolute error tolerance:
1d-4

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel    mix - abs,rel
 ----------------------------------      -----      -----      -----   -------------
 y(  1) =  0.7371770832059414E-003        7.34       7.34       4.21
 y(  2) =  0.1442575715381605E-003        8.05       8.05       4.20
 y(  3) =  0.5889602259243881E-004        8.06       8.06       3.83
 y(  4) =  0.1175734704403569E-002        7.08       7.08       4.15
 y(  5) =  0.2387823243162753E-002        5.83       5.83       3.21
 y(  6) =  0.6244778711349675E-002        5.24       5.24       3.03
 y(  7) =  0.2850043711924880E-002        7.34       7.34       4.80
 y(  8) =  0.2849956288075124E-002        7.34       7.34       4.80

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  5.24       5.24       3.03


 using mixed error yields mescd           5.24
 using relative error yields scd                                 3.03


 Integration characteristics:

    number of integration steps           43
    number of accepted steps              37
    number of f evaluations              665
    number of Jacobian evaluations        20
    number of LU decompositions          168

 CPU-time used:                          0.0029 sec
```

FIGURE I.6.1: *Example of performing a test run, in which we solve problem HIRES with PSIDE. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[SLV97c] J.J.B. de Swart, W.M. Lioen, and W.A. van der Veen. *Specification of PSIDE*. CWI, 1997. Available at http://www.cwi.nl/cwi/projects/PSIDE/.

7 Solver RADAU

7.1 General information

Authors:	E. Hairer and G. Wanner
first version:	April 23, 1998
last update:	January 18, 2002
language:	Fortran 77
availability:	the code RADAU is freely available (in the public domain)
official link:	http://www.unige.ch/~hairer/prog/stiff/radau.f
problem type:	ODEs and DAEs of index less than or equal to 3
IVPtestset files:	solver: radau.f
	driver: radaud.f
	auxiliary files: radaua.f (auxiliary linear algebra routines)

7.2 Numerical method

The code RADAU is based on implicit Runge-Kutta methods (Radau IIa) of orders 5, 9 and 13. These methods are L-stable and were firstly implemented in fixed order mode in the code RADAUP [HW96]. It is written for problems of the form $My' = f(t, y)$ with a possibly singular matrix M. It is therefore also suitable for the solution of differential-algebraic problems.

7.3 Implementation details

All the implementation techniques described for RADAU5 hold here as well. The code has been provided with an order variation strategy. This is based upon the observation that high order methods perform better than low order methods as soon as the convergence of the simplified Newton iteration is sufficiently fast (a measure of the rate of convergence is the so called *contractivity factor*) [HW99].

7.4 How to solve test problems with RADAU

Compiling

```
f90  -o dotest  radaud.f  problem.f  radau.f  radaua.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.7.1 shows what one has to do.

References

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

[HW99] E. Hairer and G. Wanner. Stiff differential equations solved by radau methods. *J. Comput. Appl. Math.*, 111:93–111, 1999.

```
$ f90 -O5 -o dotest radaud.f hires.f radau.f radaua.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using RADAU

 User input:

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

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel     mix - abs,rel
 ----------------------------------      -----      -----      -----    -------------
 y(  1) =  0.7485152484440879E-003        4.94       4.94       1.81
 y(  2) =  0.1464912389469645E-003        5.65       5.65       1.81
 y(  3) =  0.6101426280653334E-004        5.67       5.67       1.44
 y(  4) =  0.1196763210067838E-002        4.68       4.68       1.75
 y(  5) =  0.2731889907948499E-002        3.46       3.46       0.84
 y(  6) =  0.7347017643277632E-002        2.96       2.96       0.75
 y(  7) =  0.3074620885907540E-002        3.65       3.65       1.10
 y(  8) =  0.2625379114092413E-002        3.65       3.65       1.10

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  2.96       2.96       0.75


 using mixed error yields mescd           2.96
 using relative error yields scd                                 0.75


 Integration characteristics:

    number of integration steps           38
    number of accepted steps              31
    number of f evaluations              295
    number of Jacobian evaluations        20
    number of LU decompositions           37

 CPU-time used:                         0.0010 sec
```

FIGURE I.7.1: *Example of performing a test run, in which we solve problem HIRES with RADAU. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

8 Solver RADAU5

8.1 General information

Authors:	E. Hairer and G. Wanner
last update:	January 18, 2002
language:	Fortran 77
availability:	the code RADAU5 is freely available (in the public domain)
official link:	http://www.unige.ch/~hairer/prog/stiff/radau5.f
problem type:	ODEs and DAEs of index less than or equal to 3
IVPtestset files:	solver: radau5.f
	driver: radau5d.f
	auxiliary files: radaua.f (auxiliary linear algebra routines)

8.2 Numerical method

The code RADAU5 uses an implicit Runge-Kutta method (Radau IIa) of order 5 (three stages) with step size control and continuous output. It is written for problems of the form $My' = f(t, y)$ with a possibly singular matrix M. It is therefore also suitable for the solution of differential-algebraic problems.

8.3 Implementation details

Nonlinear systems are solved by a simplified Newton iteration. A similarity transformation on the inverse of the Butcher array is performed in order to reduce the computational cost associated to the solution of linear systems (see [HW96], page 121) so that, each time the Jacobian is updated, a factorization of one real and one complex matrix of the same dimension as that of the continuous problem is needed.

8.4 How to solve test problems with RADAU5

Compiling

```
f90  -o dotest  radau5d.f  problem.f  radau5.f  radaua.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.8.1 shows what one has to do.

References

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

```
$ f90 -O5 -o dotest radau5d.f hires.f radau5.f radaua.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using RADAU5

 User input:

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

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel     mix - abs,rel
 ---------------------------------       -----      -----      -----    -------------
 y(  1) =  0.7485152484440879E-003        4.94       4.94       1.81
 y(  2) =  0.1464912389469645E-003        5.65       5.65       1.81
 y(  3) =  0.6101426280653334E-004        5.67       5.67       1.44
 y(  4) =  0.1196763210067838E-002        4.68       4.68       1.75
 y(  5) =  0.2731889907948499E-002        3.46       3.46       0.84
 y(  6) =  0.7347017643277632E-002        2.96       2.96       0.75
 y(  7) =  0.3074620885907540E-002        3.65       3.65       1.10
 y(  8) =  0.2625379114092413E-002        3.65       3.65       1.10

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  2.96       2.96       0.75


 using mixed error yields mescd           2.96
 using relative error yields scd                                 0.75

 Integration characteristics:

    number of integration steps           38
    number of accepted steps              31
    number of f evaluations              295
    number of Jacobian evaluations        20
    number of LU decompositions           36

 CPU-time used:                         0.0010 sec
```

FIGURE I.8.1: *Example of performing a test run, in which we solve problem HIRES with RADAU5. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

9 Solver VODE

9.1 General information

Authors:	Peter N. Brown, George D. Byrne and Alan C. Hindmarsh
first version:	June 15 1989
last update:	April 30, 2000
language:	Fortran 77
availability:	the code VODE is freely available (in the public domain)
official link:	http://www.netlib.org/ode/vode.f
problem type:	ODE
IVPtestset files:	solver: vode.f
	driver: voded.f
	auxiliary files: vodea.f (auxiliary linear algebra routines)

9.2 Numerical method

The code is based upon linear multistep methods used with variable coefficients (but fixed leading term) to take account for the stepsize change. It allows the use of Adams and BDFs methods to handle both non stiff and stiff problems [Byr75].

9.3 Implementation details

VODE [BBA89] is a package based on the EPISODE and EPISODEB packages [HB77, BH76], and on the ODEPACK user interface standard [Hin83], with minor modifications. The code may switch between two different techniques, namely functional iteration and the modified Newton method, to solve nonlinear systems at each time-step. Recently, a FORTRAN 90 version of this solver has been made available at the URL http://www.radford.edu/~thompson/vodef90web/.

9.4 How to solve test problems with VODE

Compiling

```
f90  -o dotest  voded.f  problem.f  vode.f  vodea.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.9.1 shows what one has to do.

References

[BBA89] P. N. Brown, G. D. Byrne, and Hindmarsh A.C. Vode: A variable coefficient ode solver. *SIAM J. Sci. Stat. Comput.*, 10:1038–1051, 1989. Also, LLNL Report UCRL-98412, June 1988.

[BH76] G. D. Byrne and A. C. Hindmarsh. Episodeb: An experimental package for the integration of systems of ordinary differential equations with banded jacobians. Technical Report UCID-30132, April 1976., LLNL, 1976.

[Byr75] A. C. Byrne, G. D.and Hindmarsh. A polyalgorithm for the numerical solution of ordinary differential equations. *Acm Trans Math Software*, 1:71–96, 1975.

```
$ f90 -O5 -o dotest voded.f hires.f vode.f  vodea.f report.f
$ dotest
  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using VODE

 User input:

 give relative error tolerance:
1d-4
 give absolute error tolerance:
1d-4

 Numerical solution:

                                              scd
       solution component              ---------------------------     ignore
                                       mixed       abs        rel    mix - abs,rel
 ---------------------------------     -----      -----      -----   -------------
 y(  1) =  0.7405428802164954E-003       5.47       5.47       2.33
 y(  2) =  0.1449232356407335E-003       6.17       6.17       2.33
 y(  3) =  0.5951034500912568E-004       6.21       6.21       1.98
 y(  4) =  0.1182096389331148E-002       5.19       5.19       2.26
 y(  5) =  0.2483586047844519E-002       4.01       4.01       1.39
 y(  6) =  0.6494848234786107E-002       3.59       3.59       1.39
 y(  7) =  0.2954272405089350E-002       3.98       3.98       1.44
 y(  8) =  0.2745727594910732E-002       3.98       3.98       1.44

 used components for  scd                   8          8          8
 scd of Y (maximum norm)                 3.59       3.59       1.39


 using mixed error yields mescd          3.59
 using relative error yields scd                                1.39

 Integration characteristics:

    number of integration steps        133
    number of accepted steps           131
    number of f evaluations            191
    number of Jacobian evaluations      10
    number of LU decompositions         25

 CPU-time used:                       0.0010 sec
```

FIGURE I.9.1: *Example of performing a test run, in which we solve problem HIRES with VODE. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.

[HB77] A. C. Hindmarsh and G. D. Byrne. Episode: An effective package for the integration of systems of ordinary differential equations. Technical Report UCID-30112, Rev. 1, April 1977, LLNL, 1977.

[Hin83] Alan C. Hindmarsh. ODEPACK, a systemized collection of ODE solvers. In R. Stepleman et al., editors, *Scientific Computing*, pages 55–64, Amsterdam, 1983. IMACS, North-Holland Publishing Company.

Part II
Problems

This part is the core of the report. All the test problems collected are described. The problems are ordered as ODEs, DAEs and IDEs.

1 Problem HIRES

1.1 General information

This IVP is a stiff system of 8 non-linear Ordinary Differential Equations. It was proposed by Schäfer in 1975 [Sch75]. The name HIRES was given by Hairer & Wanner [HW96]. It refers to 'High Irradiance RESponse', which is described by this ODE. The parallel-IVP-algorithm group of CWI contributed this problem to the test set. The software part of the problem is in the file hires.f available at [MM08].

1.2 Mathematical description of the problem

The problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \quad y(0) = y_0,$$

with

$$y \in \mathbb{R}^8, \quad 0 \le t \le 321.8122.$$

The function f is defined by

$$f(y) = \begin{pmatrix} -1.71y_1 & +0.43y_2 & +8.32y_3 & +0.0007 & \\ 1.71y_1 & -8.75y_2 & & & \\ -10.03y_3 & +0.43y_4 & +0.035y_5 & & \\ 8.32y_2 & +1.71y_3 & -1.12y_4 & & \\ -1.745y_5 & +0.43y_6 & +0.43y_7 & & \\ -280y_6y_8 & +0.69y_4 & +1.71y_5 & -0.43y_6 & +0.69y_7 \\ 280y_6y_8 & -1.81y_7 & & & \\ -280y_6y_8 & +1.81y_7 & & & \end{pmatrix}.$$

The initial vector y_0 is given by $(1, 0, 0, 0, 0, 0, 0, 0.0057)^{\mathrm{T}}$.

1.3 Origin of the problem

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. It has been promoted as a test problem by Gottwald in [Got77]. The reaction scheme is given in Figure II.1.1.

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. The values of the parameters were taken from [HW96]

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

For more details, we refer to [Sch75].

Identifying the concentrations of P_r, P_{fr}, P_rX, $P_{fr}X$, P_rX', $P_{fr}X'$, $P_{fr}X'E$ and E with y_i, $i \in \{1, \ldots, 8\}$, respectively, the differential equations mentioned in §1.2 easily follow. See [SL98] for a more detailed description of this modeling process.

The end point of the integration interval, 321.8122, was chosen arbitrarily[Wan98].

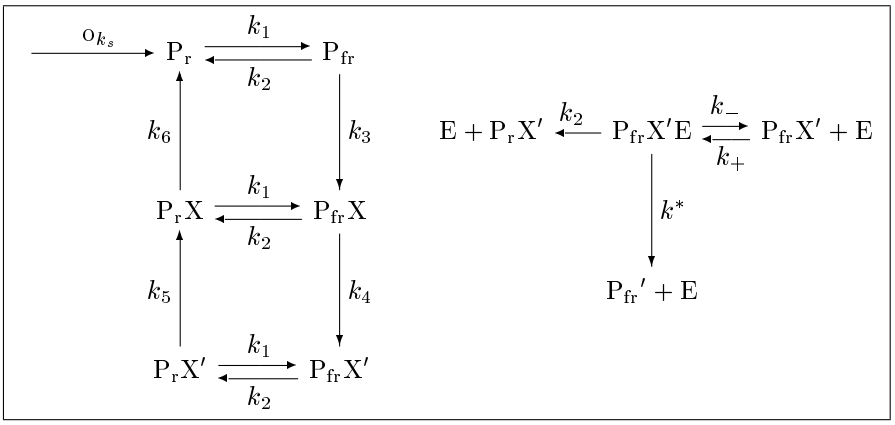

FIGURE II.1.1: *Reaction scheme for problem HIRES.*

1.4 Numerical solution of the problem

Tables II.1.1–II.1.2 and Figures II.1.2–II.1.6 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over (part of) the integration interval and the work-precision diagrams, respectively. The reference solution was computed by RADAU5 on a Cray C90, using double precision, `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol = h0` $= 1.1 \cdot 10^{-18}$. For the work-precision diagrams, we used: rtol $= 10^{-(5+m/4)}$, $m = 0, 1, \ldots, 28$; atol = rtol; h0 $= 10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

TABLE II.1.1: *Reference solution at the end of the integration interval.*

y_1	$0.7371312573325668 \cdot 10^{-3}$	y_5	$0.2386356198831331 \cdot 10^{-2}$
y_2	$0.1442485726316185 \cdot 10^{-3}$	y_6	$0.6238968252742796 \cdot 10^{-2}$
y_3	$0.5888729740967575 \cdot 10^{-4}$	y_7	$0.2849998395185769 \cdot 10^{-2}$
y_4	$0.1175651343283149 \cdot 10^{-2}$	y_8	$0.2850001604814231 \cdot 10^{-2}$

References

[Got77] B.A. Gottwald. MISS - ein einfaches Simulations-System für biologische und chemische Prozesse. *EDV in Medizin und Biologie*, 3:85–90, 1977.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems*. Springer-Verlag, second revised edition, 1996.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Sch75] E. Schäfer. A new approach to explain the 'high irradiance responses' of photomorphogenesis on the basis of phytochrome. *J. of Math. Biology*, 2:41–56, 1975.

[SL98] J.J.B. de Swart and W.M. Lioen. Collecting real-life problems to test solvers for implicit differential equations. *CWI Quarterly*, 11(1):83–100, 1998.

TABLE II.1.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-7}	10^{-7}	10^{-9}	8.42	6.21	48	47	1395	42	48	0.0039
	10^{-10}	10^{-10}	10^{-12}	11.49	9.28	89	89	2854	82	88	0.0088
DDASSL	10^{-7}	10^{-7}		6.02	3.81	380	369	591	32		0.0039
	10^{-10}	10^{-10}		8.99	6.78	1160	1148	1557	45		0.0098
GAMD	10^{-7}	10^{-7}	10^{-9}	8.51	6.00	38	34	2167	33	38	0.0049
	10^{-10}	10^{-10}	10^{-12}	10.26	7.82	55	50	4164	51	55	0.0098
MEBDFI	10^{-7}	10^{-7}	10^{-9}	6.45	4.24	218	214	767	29	29	0.0029
	10^{-10}	10^{-10}	10^{-12}	9.51	7.30	420	416	1492	46	46	0.0068
PSIDE-1	10^{-7}	10^{-7}		7.24	4.88	68	60	1208	25	252	0.0039
	10^{-10}	10^{-10}		11.06	8.85	152	151	2528	35	344	0.0068
RADAU	10^{-7}	10^{-7}	10^{-9}	7.11	4.91	51	40	985	22	51	0.0020
	10^{-10}	10^{-10}	10^{-12}	10.65	8.03	69	58	1511	29	68	0.0039
VODE	10^{-7}	10^{-7}		6.19	3.98	415	390	608	9	70	0.0029
	10^{-10}	10^{-10}		8.75	6.20	933	880	1224	15	134	0.0059

[Wan98] G. Wanner, 1998. Private communication.

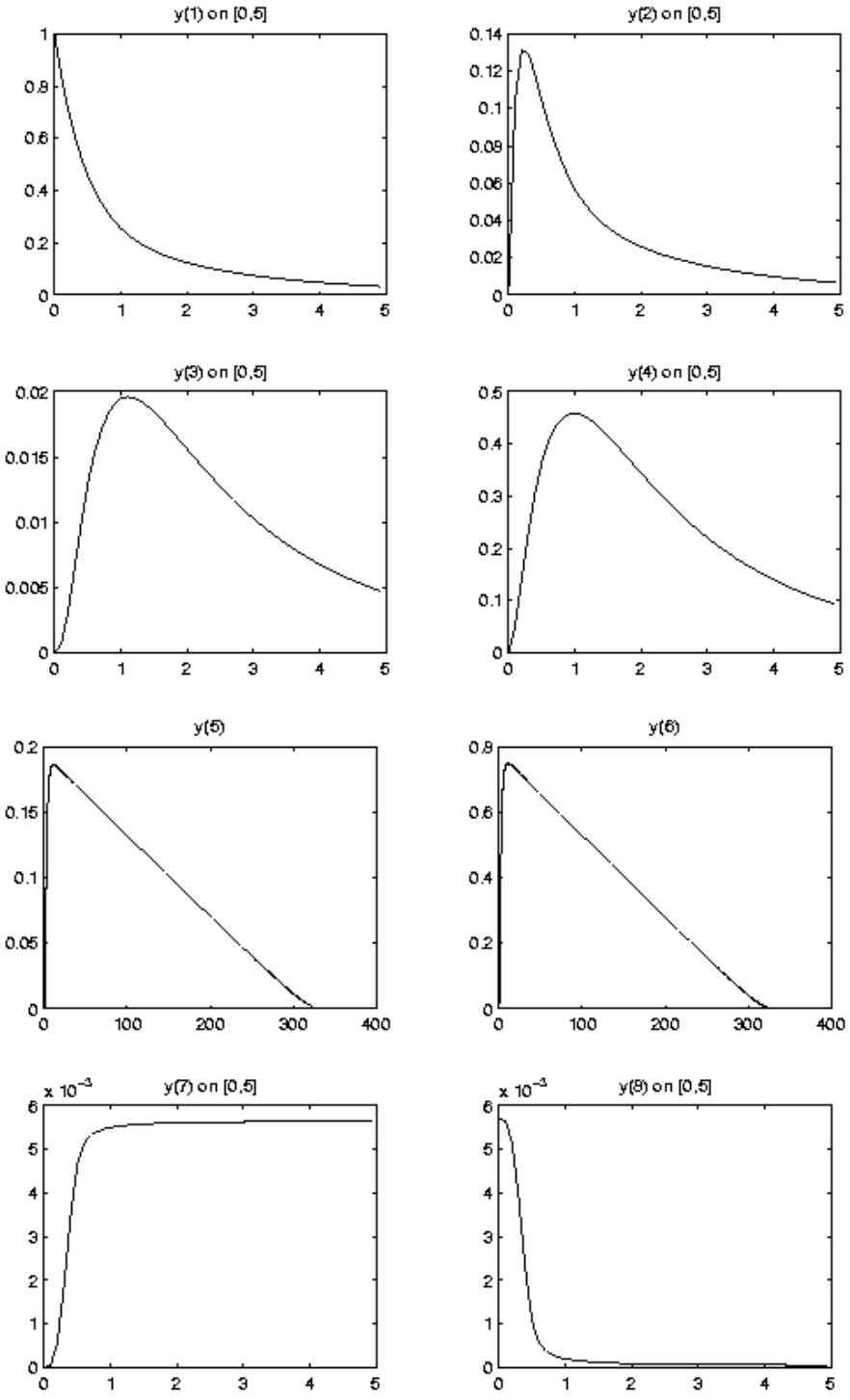

FIGURE II.1.2: *Behavior of the solution over the integration interval.*

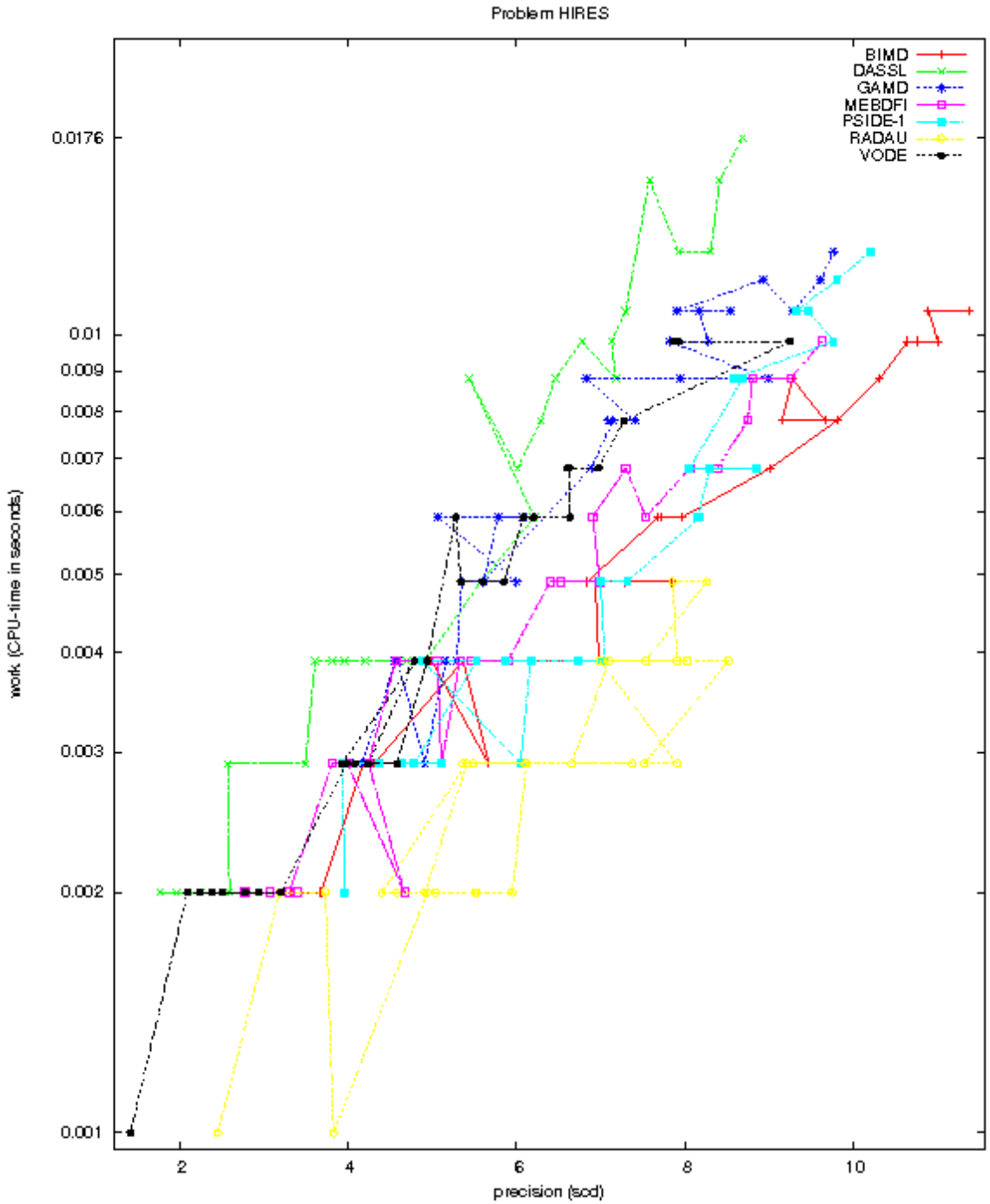

FIGURE II.1.3: *Work-precision diagram (scd versus CPU-time).*

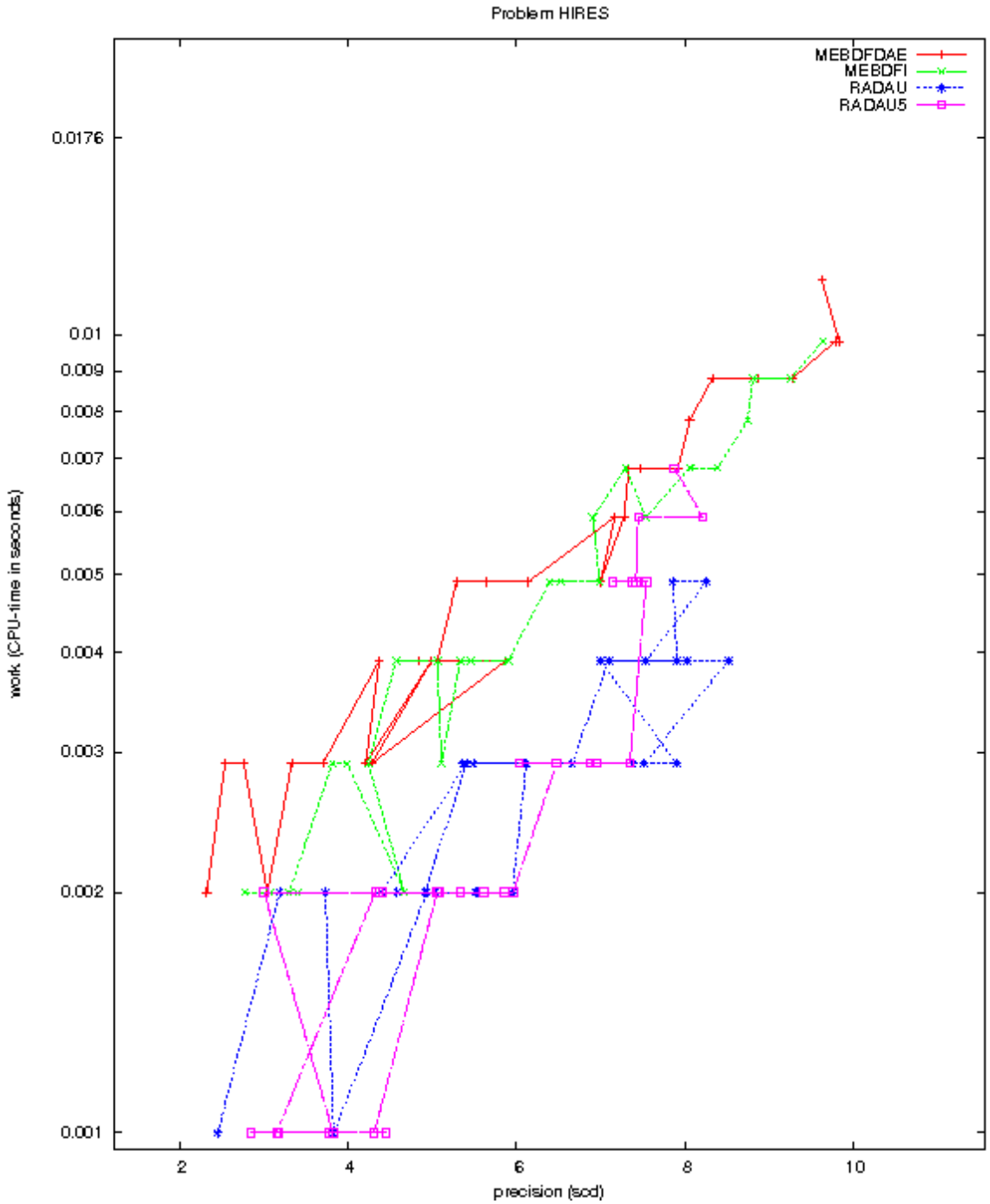

FIGURE II.1.4: *Work-precision diagram (scd versus CPU-time).*

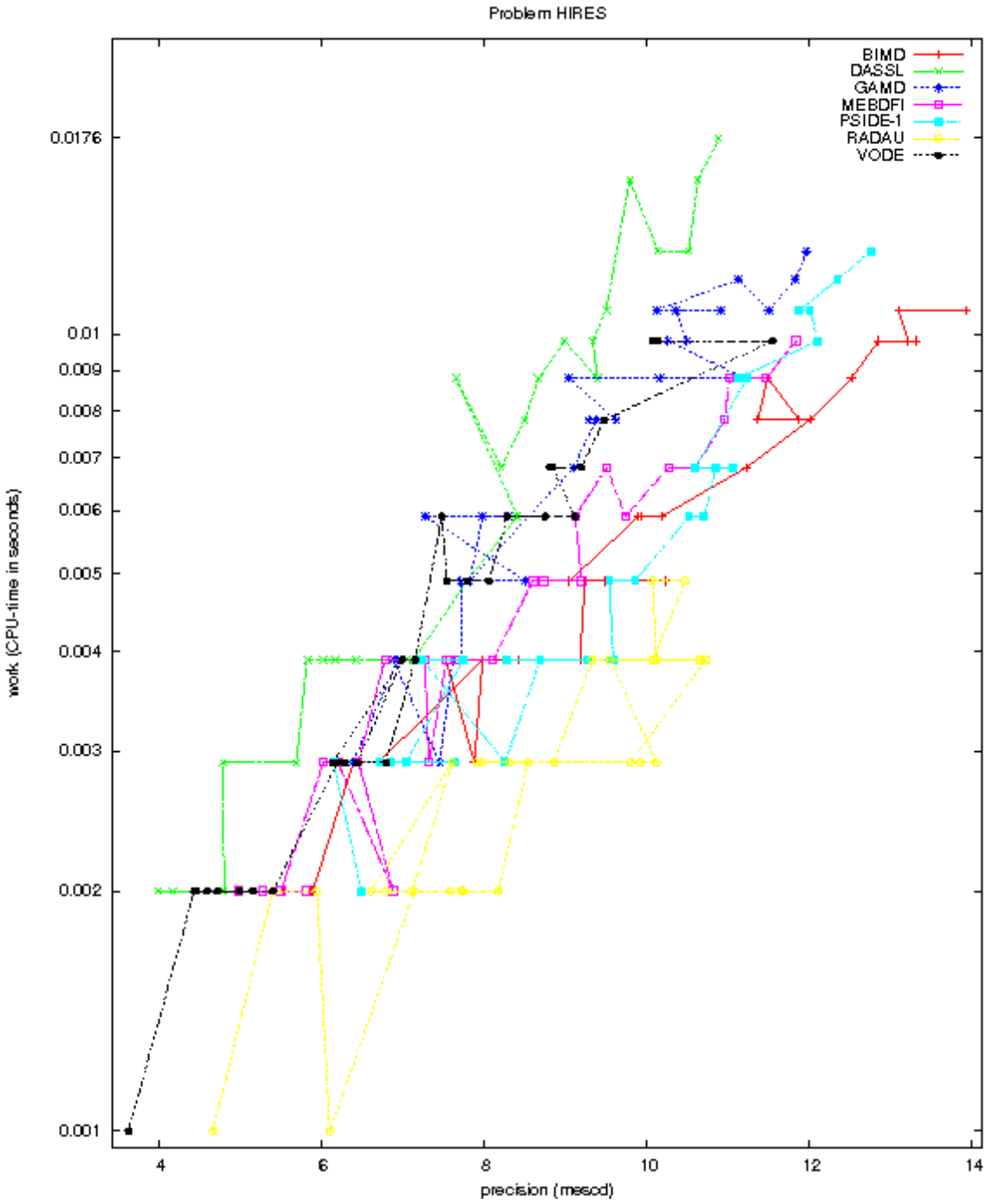

FIGURE II.1.5: *Work-precision diagram (mescd versus CPU-time).*

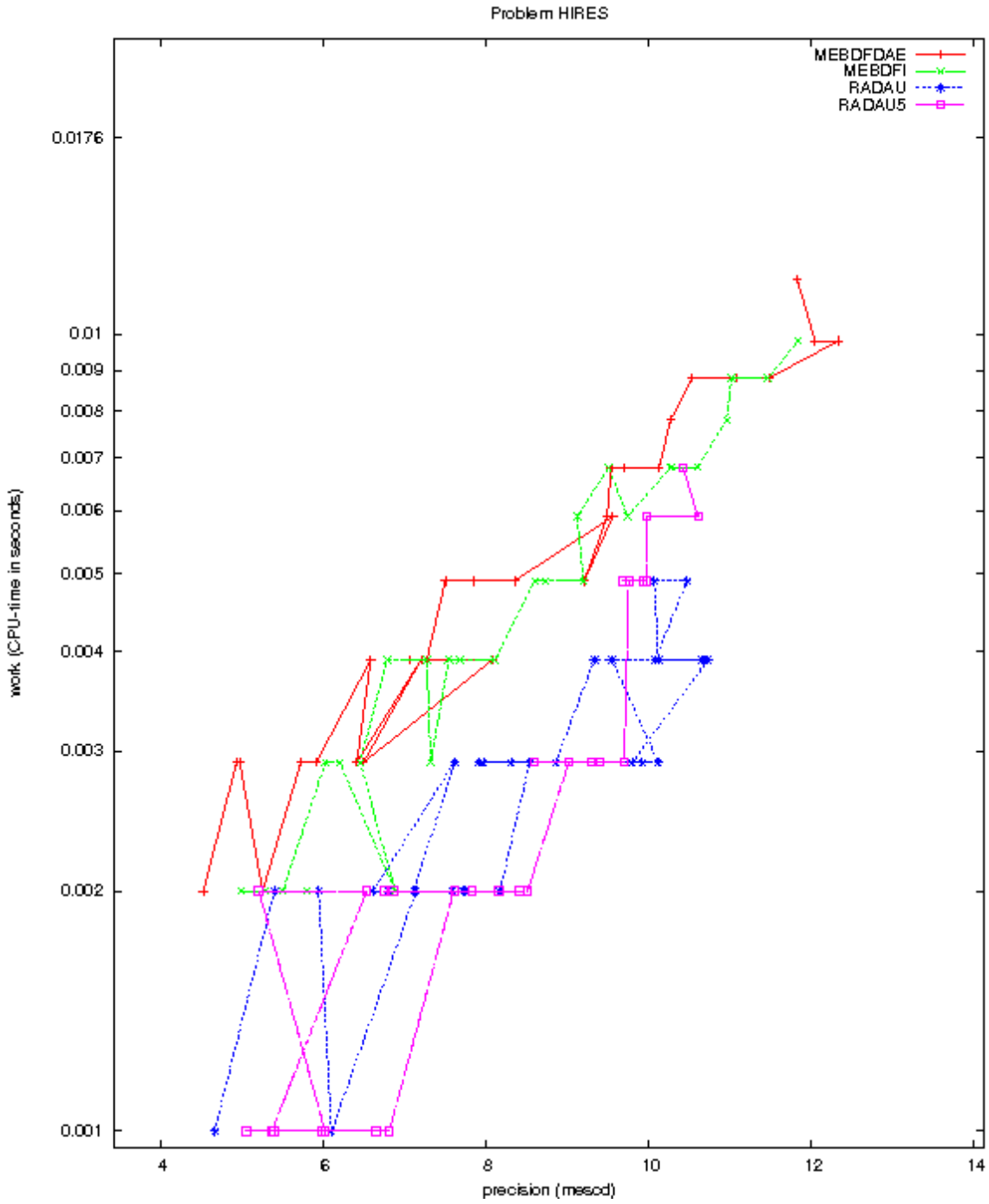

FIGURE II.1.6: *Work-precision diagram (mescd versus CPU-time).*

2 Pollution problem

2.1 General information

This IVP is a stiff system of 20 non-linear Ordinary Differential Equations. It is the chemical reaction part of the air pollution model developed at The Dutch National Institute of Public Health and Environmental Protection (RIVM) and it is described by Verwer in [Ver94]. The parallel-IVP-algorithm group of CWI contributed this problem to the test set.

The software part of the problem is in the file pollu.f available at [MM08].

2.2 Mathematical description of the problem

The problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \quad y(0) = y_0, \tag{II.2.1}$$

with

$$y \in \mathbb{R}^{20}, \quad 0 \le t \le 60.$$

The function f is defined by

$$f = \begin{pmatrix} -\sum_{j\in\{1,10,14,23,24\}} r_j + \sum_{j\in\{2,3,9,11,12,22,25\}} r_j \\ -r_2 - r_3 - r_9 - r_{12} + r_1 + r_{21} \\ -r_{15} + r_1 + r_{17} + r_{19} + r_{22} \\ -r_2 - r_{16} - r_{17} - r_{23} + r_{15} \\ -r_3 + 2r_4 + r_6 + r_7 + r_{13} + r_{20} \\ -r_6 - r_8 - r_{14} - r_{20} + r_3 + 2r_{18} \\ -r_4 - r_5 - r_6 + r_{13} \\ r_4 + r_5 + r_6 + r_7 \\ -r_7 - r_8 \\ -r_{12} + r_7 + r_9 \\ -r_9 - r_{10} + r_8 + r_{11} \\ r_9 \\ -r_{11} + r_{10} \\ -r_{13} + r_{12} \\ r_{14} \\ -r_{18} - r_{19} + r_{16} \\ -r_{20} \\ r_{20} \\ -r_{21} - r_{22} - r_{24} + r_{23} + r_{25} \\ -r_{25} + r_{24} \end{pmatrix},$$

where the r_i are auxiliary variables, given in Table II.2.1. The values of the parameters k_j are in Table II.2.2. Finally, the initial vector y_0 is given by

$$y_0 = (0, 0.2, 0, 0.04, 0, 0, 0.1, 0.3, 0.01, 0, 0, 0, 0, 0, 0, 0, 0.007, 0, 0, 0)^{\mathrm{T}}.$$

2.3 Origin of the problem

The problem is a chemical model consisting of 25 reactions and 20 reacting compounds. Figure II.2.1 shows the reaction scheme. Writing down the reaction velocities r_j for every reaction equation and making the identification in Table II.2.3, which also lists the concentrations at $t = 0$, one arrives at the system of differential equations (II.2.1). The time interval [0,60] represents the behavior of the reactants sufficiently.

TABLE II.2.1: *Auxiliary variables.*

r_1	=	$k_1 \cdot y_1$	r_{10}	=	$k_{10} \cdot y_{11} \cdot y_1$	r_{19}	=	$k_{19} \cdot y_{16}$
r_2	=	$k_2 \cdot y_2 \cdot y_4$	r_{11}	=	$k_{11} \cdot y_{13}$	r_{20}	=	$k_{20} \cdot y_{17} \cdot y_6$
r_3	=	$k_3 \cdot y_5 \cdot y_2$	r_{12}	=	$k_{12} \cdot y_{10} \cdot y_2$	r_{21}	=	$k_{21} \cdot y_{19}$
r_4	=	$k_4 \cdot y_7$	r_{13}	=	$k_{13} \cdot y_{14}$	r_{22}	=	$k_{22} \cdot y_{19}$
r_5	=	$k_5 \cdot y_7$	r_{14}	=	$k_{14} \cdot y_1 \cdot y_6$	r_{23}	=	$k_{23} \cdot y_1 \cdot y_4$
r_6	=	$k_6 \cdot y_7 \cdot y_6$	r_{15}	=	$k_{15} \cdot y_3$	r_{24}	=	$k_{24} \cdot y_{19} \cdot y_1$
r_7	=	$k_7 \cdot y_9$	r_{16}	=	$k_{16} \cdot y_4$	r_{25}	=	$k_{25} \cdot y_{20}$
r_8	=	$k_8 \cdot y_9 \cdot y_6$	r_{17}	=	$k_{17} \cdot y_4$			
r_9	=	$k_9 \cdot y_{11} \cdot y_2$	r_{18}	=	$k_{18} \cdot y_{16}$			

TABLE II.2.2: *Parameter values.*

k_1	$= 0.350$	k_{10}	$= 0.900 \cdot 10^4$	k_{19}	$= 0.444 \cdot 10^{12}$
k_2	$= 0.266 \cdot 10^2$	k_{11}	$= 0.220 \cdot 10^{-1}$	k_{20}	$= 0.124 \cdot 10^4$
k_3[†]	$= 0.123 \cdot 10^5$	k_{12}	$= 0.120 \cdot 10^5$	k_{21}	$= 0.210 \cdot 10$
k_4	$= 0.860 \cdot 10^{-3}$	k_{13}	$= 0.188 \cdot 10$	k_{22}	$= 0.578 \cdot 10$
k_5	$= 0.820 \cdot 10^{-3}$	k_{14}	$= 0.163 \cdot 10^5$	k_{23}	$= 0.474 \cdot 10^{-1}$
k_6	$= 0.150 \cdot 10^5$	k_{15}	$= 0.480 \cdot 10^7$	k_{24}	$= 0.178 \cdot 10^4$
k_7	$= 0.130 \cdot 10^{-3}$	k_{16}	$= 0.350 \cdot 10^{-3}$	k_{25}	$= 0.312 \cdot 10$
k_8	$= 0.240 \cdot 10^5$	k_{17}	$= 0.175 \cdot 10^{-1}$		
k_9	$= 0.165 \cdot 10^5$	k_{18}	$= 0.100 \cdot 10^9$		

[†] Notice that this constant has a typing error in [Ver94].

1.	NO2	→	NO+O3P	14.	NO2+OH	→	HNO3
2.	NO+O3	→	NO2	15.	O3P	→	O3
3.	HO2+NO	→	NO2+OH	16.	O3	→	O1D
4.	HCHO	→	2 HO2+CO	17.	O3	→	O3P
5.	HCHO	→	CO	18.	O1D	→	2 OH
6.	HCHO+OH	→	HO2+CO	19.	O1D	→	O3P
7.	ALD	→	MEO2+HO2+CO	20.	SO2+OH	→	SO4+HO2
8.	ALD+OH	→	C2O3	21.	NO3	→	NO
9.	C2O3+NO	→	NO2+MEO2+CO2	22.	NO3	→	NO2+O3P
10.	C2O3+NO2	→	PAN	23.	NO2+O3	→	NO3
11.	PAN	→	C2O3+NO2	24.	NO3+NO2	→	N2O5
12.	MEO2+NO	→	CH3O+NO2	25.	N2O5	→	NO3+NO2
13.	CH3O	→	HCHO+HO2				

FIGURE II.2.1: *Reaction scheme.*

TABLE II.2.3: *Identification of variables with species. The square brackets '[ ]' denote concentrations.*

variable	species	initial value	variable	species	initial value
y_1	[NO2]	0	y_{11}	[C2O3]	0
y_2	[NO]	0.2	y_{12}	[CO2]	0
y_3	[O3P]	0	y_{13}	[PAN]	0
y_4	[O3]	0.04	y_{14}	[CH3O]	0
y_5	[HO2]	0	y_{15}	[HNO3]	0
y_6	[OH]	0	y_{16}	[O1D]	0
y_7	[HCHO]	0.1	y_{17}	[SO2]	0.007
y_8	[CO]	0.3	y_{18}	[SO4]	0
y_9	[ALD]	0.01	y_{19}	[NO3]	0
y_{10}	[MEO2]	0	y_{20}	[N2O5]	0

TABLE II.2.4: *Reference solution at the end of the integration interval.*

y_1	$0.5646255480022769 \cdot 10^{-1}$	y_{11}	$0.1135863833257075 \cdot 10^{-7}$
y_2	0.1342484130422339	y_{12}	$0.2230505975721359 \cdot 10^{-2}$
y_3	$0.4139734331099427 \cdot 10^{-8}$	y_{13}	$0.2087162882798630 \cdot 10^{-3}$
y_4	$0.5523140207484359 \cdot 10^{-2}$	y_{14}	$0.1396921016840158 \cdot 10^{-4}$
y_5	$0.2018977262302196 \cdot 10^{-6}$	y_{15}	$0.8964884856898295 \cdot 10^{-2}$
y_6	$0.1464541863493966 \cdot 10^{-6}$	y_{16}	$0.4352846369330103 \cdot 10^{-17}$
y_7	$0.7784249118997964 \cdot 10^{-1}$	y_{17}	$0.6899219696263405 \cdot 10^{-2}$
y_8	0.3245075353396018	y_{18}	$0.1007803037365946 \cdot 10^{-3}$
y_9	$0.7494013383880406 \cdot 10^{-2}$	y_{19}	$0.1772146513969984 \cdot 10^{-5}$
y_{10}	$0.1622293157301561 \cdot 10^{-7}$	y_{20}	$0.5682943292316392 \cdot 10^{-4}$

2.4 Numerical solution of the problem

Tables II.2.4–II.2.5 and Figures II.2.2–II.2.6 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the interval [0,12] and the work-precision diagrams, respectively. The reference solution was computed by RADAU5 on a Cray C90, using double precision, `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol = h0` $= 1.1 \cdot 10^{-18}$. For the work-precision diagrams, we used: rtol $= 10^{-(5+m/4)}$, $m = 0, 1, \ldots, 32$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

.

References

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Ver94] J.G. Verwer. Gauss-Seidel iteration for stiff ODEs from chemical kinetics. *SIAM J. Sci. Comput.*, 15(5):1243–1259, 1994.

TABLE II.2.5: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-7}	10^{-7}	10^{-7}	9.25	5.63	25	25	572	22	25	0.0039
	10^{-10}	10^{-10}	10^{-10}	11.73	8.73	41	41	1257	27	41	0.0107
DDASSL	10^{-7}	10^{-7}		5.94	4.13	135	135	188	23		0.0039
	10^{-10}	10^{-10}		9.04	5.91	536	532	669	38		0.0107
GAMD	10^{-7}	10^{-7}	10^{-7}	8.16	6.31	23	23	625	23	23	0.0049
	10^{-10}	10^{-10}	10^{-10}	11.35	5.36	36	36	1401	36	36	0.0098
MEBDFI	10^{-7}	10^{-7}	10^{-7}	8.46	6.46	120	118	391	20	20	0.0039
	10^{-10}	10^{-10}	10^{-10}	11.45	9.32	235	235	763	33	33	0.0078
PSIDE-1	10^{-7}	10^{-7}		7.51	4.84	31	29	465	9	124	0.0049
	10^{-10}	10^{-10}		10.64	8.04	63	62	970	12	188	0.0098
RADAU	10^{-7}	10^{-7}	10^{-7}	5.59	3.78	32	29	227	21	32	0.0029
	10^{-10}	10^{-10}	10^{-10}	10.00	7.75	35	35	449	21	35	0.0049
VODE	10^{-7}	10^{-7}		6.61	3.32	149	149	208	4	27	0.0029
	10^{-10}	10^{-10}		8.79	4.78	393	375	528	7	61	0.0059

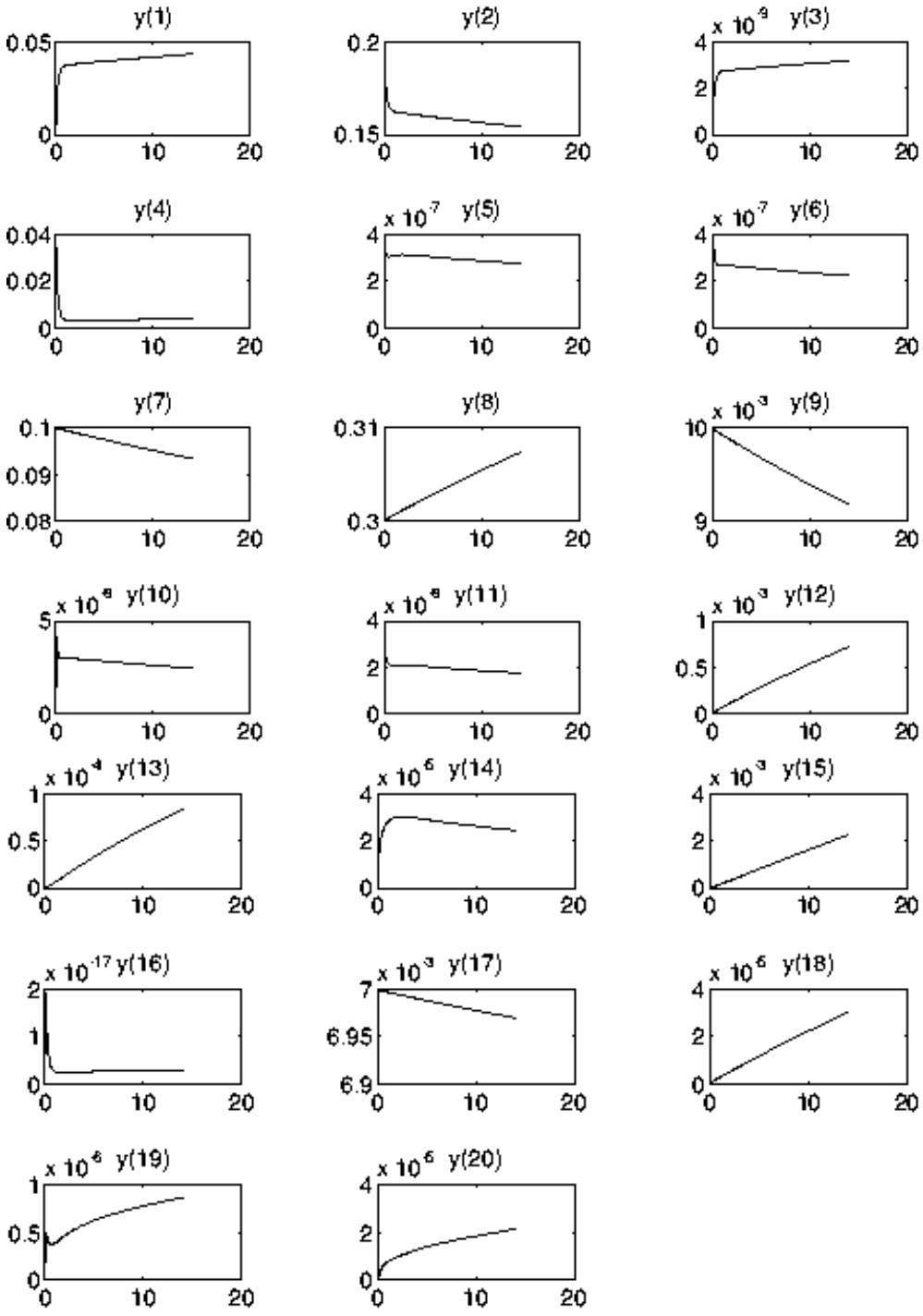

FIGURE II.2.2: *Behavior of the solution over the interval [0,12].*

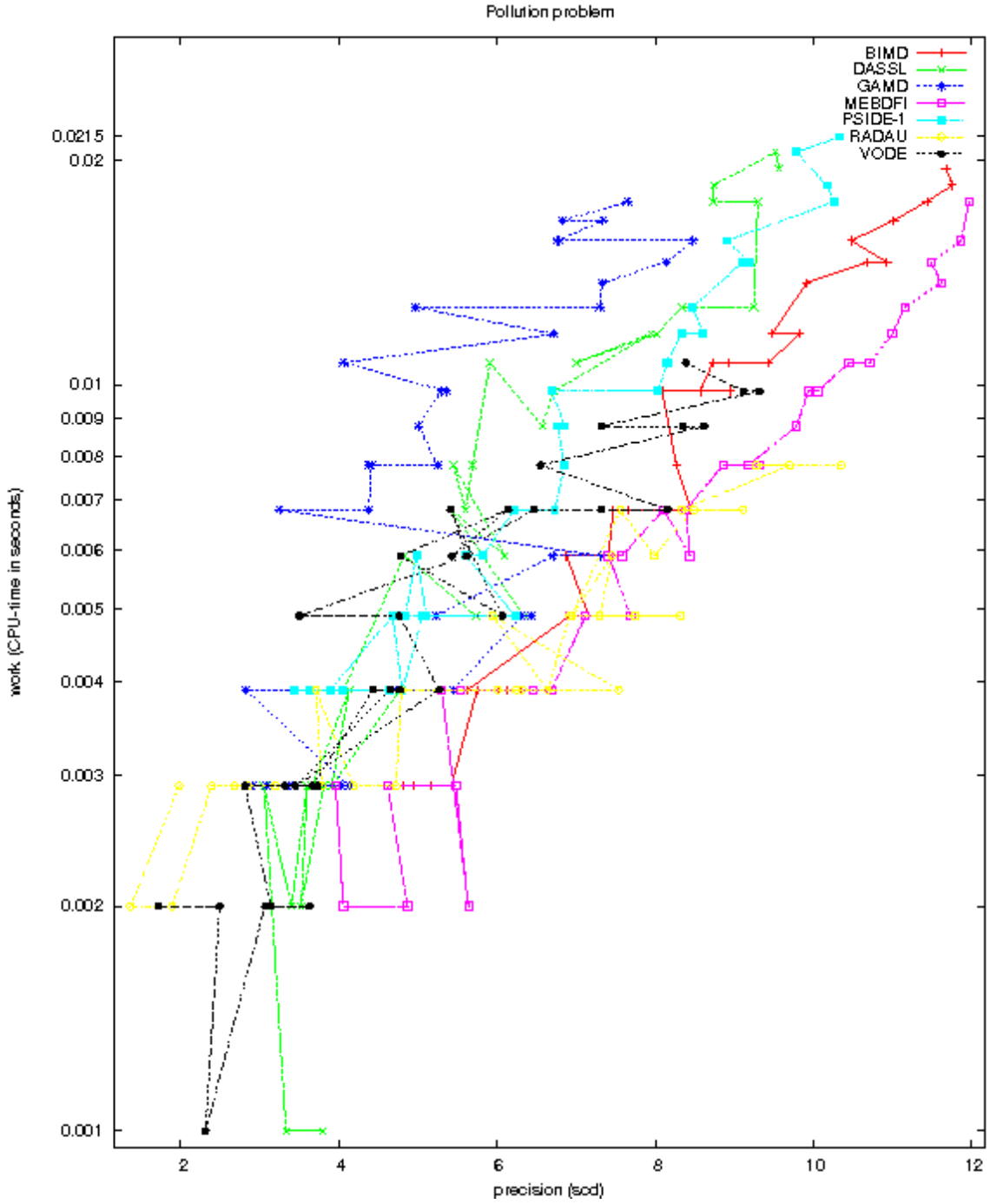

FIGURE II.2.3: *Work-precision diagram (scd versus CPU-time).*

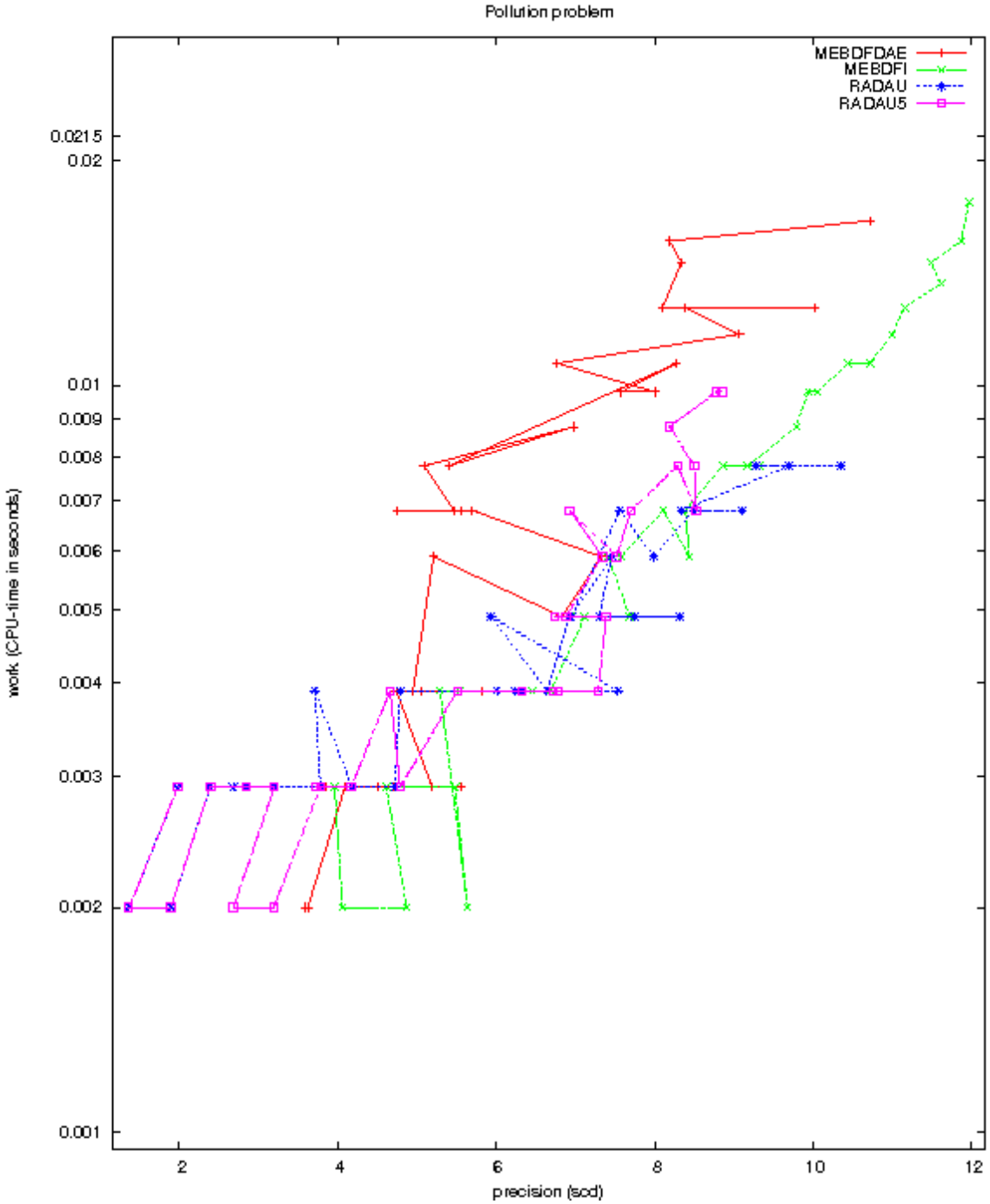

FIGURE II.2.4: *Work-precision diagram (scd versus CPU-time).*

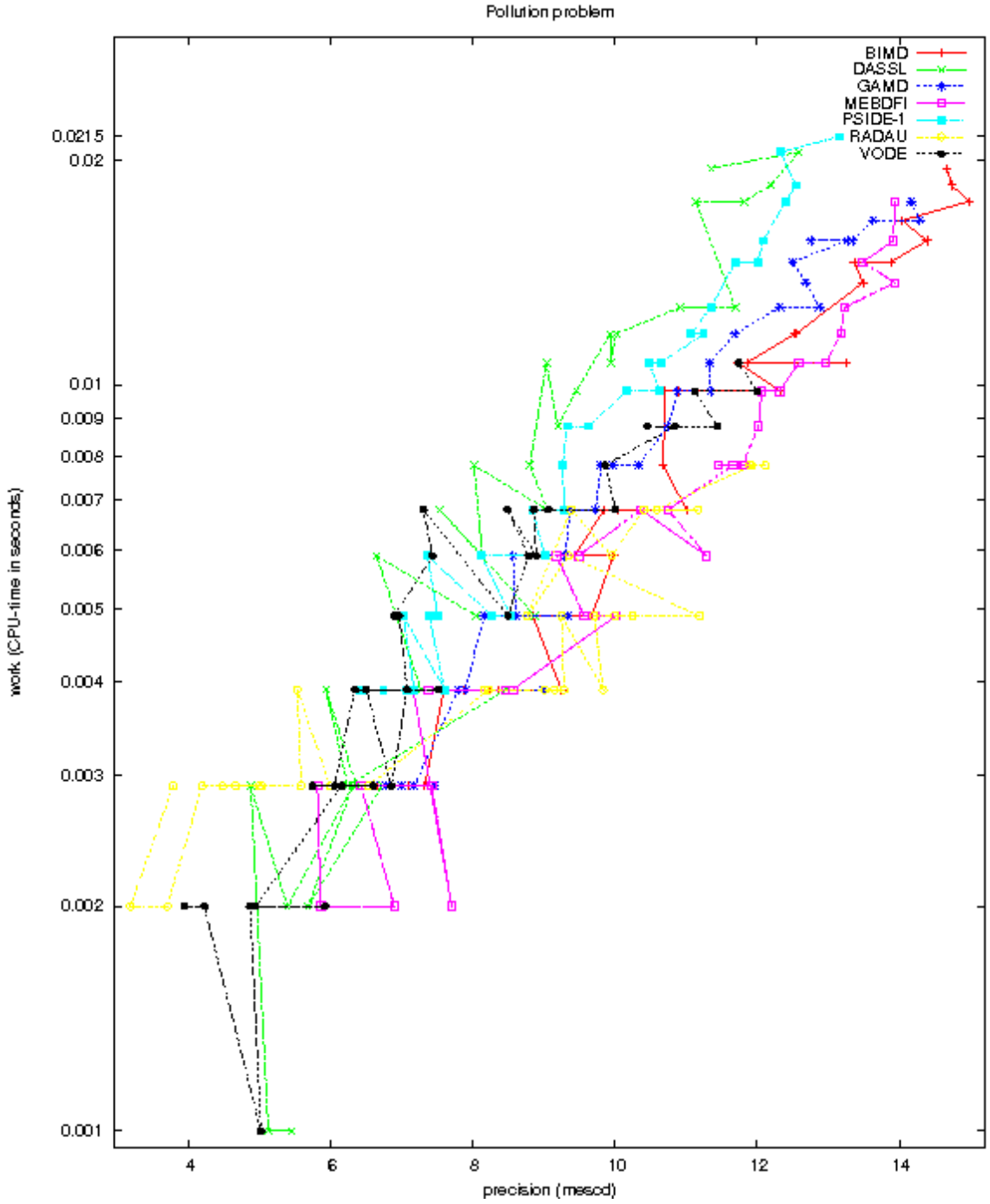

FIGURE II.2.5: *Work-precision diagram (mescd versus CPU-time).*

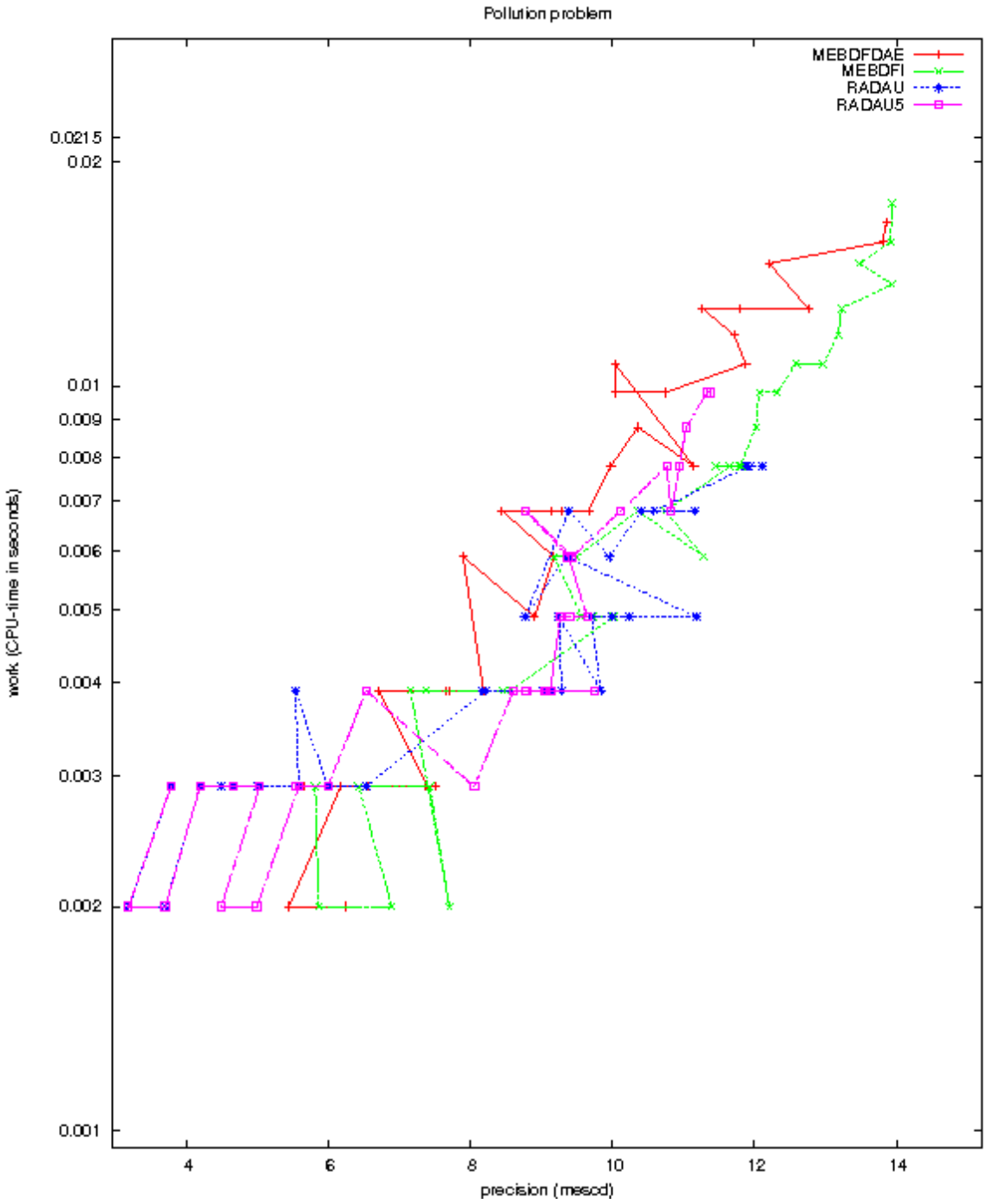

FIGURE II.2.6: *Work-precision diagram (mescd versus CPU-time).*

3 Ring modulator

3.1 General information

The type of the problem depends on the parameter C_s. If $C_s \neq 0$, then it is a stiff system of 15 non-linear ordinary differential equations. For $C_s = 0$ we have a DAE of index 2, consisting of 11 differential equations and 4 algebraic equations. The numerical results presented here refer to $C_s = 2 \cdot 10^{-12}$. The problem has been taken from [KRS92], where the approach of Horneber [Hor76] is followed. The parallel-IVP-algorithm group of CWI contributed this problem to the test set.

The software part of the problem is in the file ringmod.f available at [MM08].

3.2 Mathematical description of the problem

For the ODE case, the problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \quad y(0) = y_0,$$

with

$$y \in \mathbb{R}^{15}, \quad 0 \leq t \leq 10^{-3}.$$

The function f is defined by

$$f(t,y) = \begin{pmatrix} C^{-1}(y_8 - 0.5y_{10} + 0.5y_{11} + y_{14} - R^{-1}y_1) \\ C^{-1}(y_9 - 0.5y_{12} + 0.5y_{13} + y_{15} - R^{-1}y_2) \\ C_s^{-1}(y_{10} - q(U_{D1}) + q(U_{D4})) \\ C_s^{-1}(-y_{11} + q(U_{D2}) - q(U_{D3})) \\ C_s^{-1}(y_{12} + q(U_{D1}) - q(U_{D3})) \\ C_s^{-1}(-y_{13} - q(U_{D2}) + q(U_{D4})) \\ C_p^{-1}(-R_p^{-1}y_7 + q(U_{D1}) + q(U_{D2}) - q(U_{D3}) - q(U_{D4})) \\ -L_h^{-1}y_1 \\ -L_h^{-1}y_2 \\ L_{s2}^{-1}(0.5y_1 - y_3 - R_{g2}y_{10}) \\ L_{s3}^{-1}(-0.5y_1 + y_4 - R_{g3}y_{11}) \\ L_{s2}^{-1}(0.5y_2 - y_5 - R_{g2}y_{12}) \\ L_{s3}^{-1}(-0.5y_2 + y_6 - R_{g3}y_{13}) \\ L_{s1}^{-1}(-y_1 + U_{in1}(t) - (R_i + R_{g1})y_{14}) \\ L_{s1}^{-1}(-y_2 - (R_c + R_{g1})y_{15}) \end{pmatrix}. \tag{II.3.1}$$

The auxiliary functions $U_{D1}, U_{D2}, U_{D3}, U_{D4}, q, U_{in1}$ and U_{in2} are given by

$$\begin{aligned} U_{D1} &= y_3 - y_5 - y_7 - U_{in2}(t), \\ U_{D2} &= -y_4 + y_6 - y_7 - U_{in2}(t), \\ U_{D3} &= y_4 + y_5 + y_7 + U_{in2}(t), \\ U_{D4} &= -y_3 - y_6 + y_7 + U_{in2}(t), \\ q(U) &= \gamma(e^{\delta U} - 1), \\ U_{in1}(t) &= 0.5\sin(2000\pi t), \\ U_{in2}(t) &= 2\sin(20000\pi t). \end{aligned} \tag{II.3.2}$$

The values of the parameters are:

C	=	$1.6 \cdot 10^{-8}$	R	=	25000
C_s	=	$2 \cdot 10^{-12}$	R_p	=	50
C_p	=	10^{-8}	R_{g1}	=	36.3
L_h	=	4.45	R_{g2}	=	17.3
L_{s1}	=	0.002	R_{g3}	=	17.3
L_{s2}	=	$5 \cdot 10^{-4}$	R_i	=	50
L_{s3}	=	$5 \cdot 10^{-4}$	R_c	=	600
γ	=	$40.67286402 \cdot 10^{-9}$	δ	=	17.7493332

The initial vector y_0 is given by

$$y_0 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)^{\mathrm{T}}.$$

The definition of the function $q(U)$ in (II.3.2) may cause overflow if δU becomes too large. In the Fortran subroutine that defines f, we set `IERR=-1` if $\delta U > 300$ to prevent this situation. See page *IV-ix* of the description of the software part of the test set for more details on `IERR`.

3.3 Origin of the problem

The problem originates from electrical circuit analysis. It describes the behavior of the ring modulator, of which the circuit diagram is given in Figure II.3.1. Given a low-frequency signal U_{in1} and a high-frequency signal U_{in2}, the ring modulator produces a mixed signal in U_2.

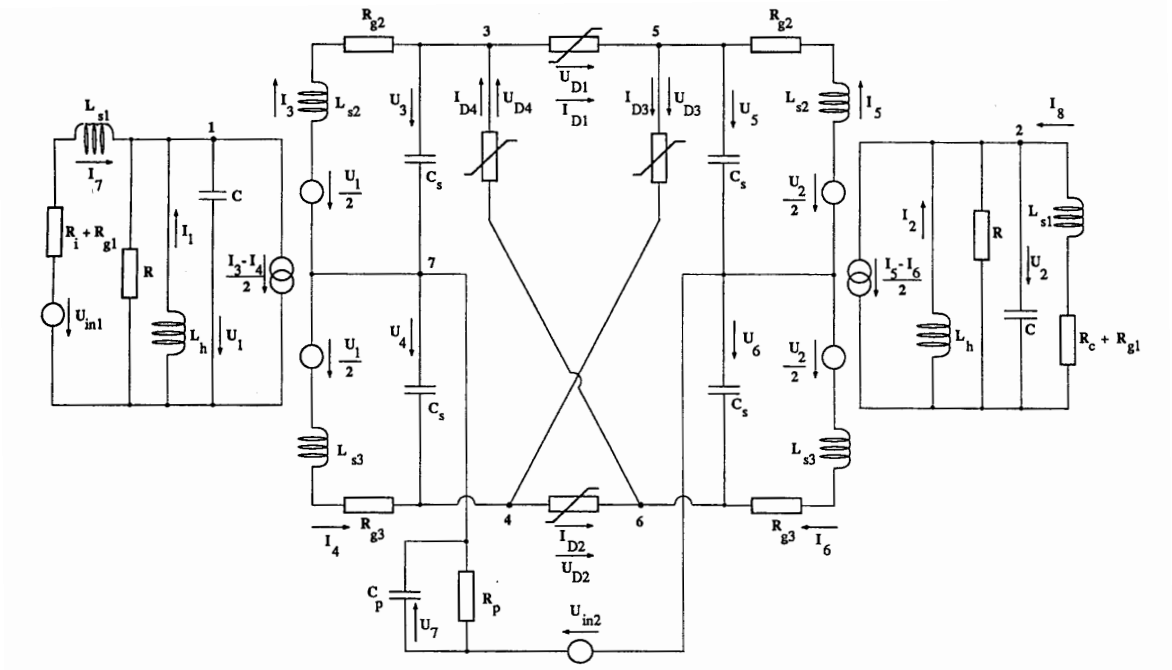

FIGURE II.3.1: *Circuit diagram for Ring Modulator (taken from [KRS92]).*

Every capacitor in the diagram leads to a differential equation:

$$C\dot{U} = I.$$

Applying Kirchhoff's Current Law yields the following differential equations:

$$\begin{aligned}
C\dot{U}_1 &= I_1 \quad -0.5I_3 + 0.5I_4 + I_7 - R^{-1}U_1,\\
C\dot{U}_2 &= I_2 \quad -0.5I_5 + 0.5I_6 + I_8 - R^{-1}U_2,\\
C_s\dot{U}_3 &= I_3 \quad -q(U_{D1}) + q(U_{D4}),\\
C_s\dot{U}_4 &= -I_4 \quad +q(U_{D2}) - q(U_{D3}),\\
C_s\dot{U}_5 &= I_5 \quad +q(U_{D1}) - q(U_{D3}),\\
C_s\dot{U}_6 &= -I_6 \quad -q(U_{D2}) + q(U_{D4}),\\
C_p\dot{U}_7 &= -R_p^{-1}U_7 + q(U_{D1}) + q(U_{D2}) - q(U_{D3}) - q(U_{D4}),
\end{aligned}$$

where U_{D1}, U_{D2}, U_{D3} and U_{D4} stand for:

$$\begin{aligned}
U_{D1} &= U_3 - U_5 - U_7 - U_{in2},\\
U_{D2} &= -U_4 + U_6 - U_7 - U_{in2},\\
U_{D3} &= U_4 + U_5 + U_7 + U_{in2},\\
U_{D4} &= -U_3 - U_6 + U_7 + U_{in2}.
\end{aligned}$$

The diode function q is given by

$$q(U) = \gamma(e^{\delta U} - 1),$$

where γ and δ are fixed constants.

Every inductor leads to a differential equation as well:

$$L\dot{I} = U.$$

Applying Kirchoff's Voltage Law to closed loops that contains an inductor, results in another 8 differential equations:

$$\begin{aligned}
L_h\dot{I}_1 &= -U_1,\\
L_h\dot{I}_2 &= -U_2,\\
L_{s2}\dot{I}_3 &= 0.5U_1 - U_3 - R_{g2}I_3,\\
L_{s3}\dot{I}_4 &= -0.5U_1 + U_4 - R_{g3}I_4,\\
L_{s2}\dot{I}_5 &= 0.5U_2 - U_5 - R_{g2}I_5,\\
L_{s3}\dot{I}_6 &= -0.5U_2 + U_6 - R_{g3}I_6,\\
L_{s1}\dot{I}_7 &= -U_1 + U_{in1} - (R_i + R_{g1})I_7,\\
L_{s1}\dot{I}_8 &= -U_2 - (R_c + R_{g1})I_8.
\end{aligned}$$

Initially, all voltages and currents are zero.

Identifying the voltages with $y_1, \ldots, y_7$ and the currents with $y_8, \ldots, y_{15}$, we obtain the 15 differential equations (II.3.1). From the plot of $y_2 = U_2$ in Figure II.3.2 we see how the low and high frequency input signals are mixed by the ring modulator.

3.4 Numerical solution of the problem

Tables II.3.2–II.3.3 and Figures II.3.2–II.3.7 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed using PSIDE with `atol = rtol =` 10^{-13}. For the work-precision diagrams, we used: rtol = $10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = rtol; h0 = $10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.3.1; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

TABLE II.3.1: *Failed runs.*

solver	m	reason
RADAU	$0, 1, \ldots, 26$	solver cannot handle `IERR=-1`.
RADAU5	$0, 1, \ldots, 9$	solver cannot handle `IERR=-1`.
VODE	0	solver cannot handle `IERR=-1`.
VODE	2	error test failed repeatedly.

TABLE II.3.2: *Reference solution at the end of the integration interval.*

y_1	$-0.2339057358486745 \cdot 10^{-1}$	y_9	$-0.2840029933642329 \cdot 10^{-7}$
y_2	$-0.7367485485540825 \cdot 10^{-2}$	y_{10}	$0.7267198267264553 \cdot 10^{-3}$
y_3	0.2582956709291169	y_{11}	$0.7929487196960840 \cdot 10^{-3}$
y_4	-0.4064465721283450	y_{12}	$-0.7255283495698965 \cdot 10^{-3}$
y_5	-0.4039455665149794	y_{13}	$-0.7941401968526521 \cdot 10^{-3}$
y_6	0.2607966765422943	y_{14}	$0.7088495416976114 \cdot 10^{-4}$
y_7	0.1106761861269975	y_{15}	$0.2390059075236570 \cdot 10^{-4}$
y_8	$0.2939904342435596 \cdot 10^{-6}$		

TABLE II.3.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	2.89	2.20	19415	19089	455877	17614	19127	3.0793
	10^{-7}	10^{-7}	10^{-9}	7.08	6.28	26590	25880	824318	25865	26585	5.5886
DDASSL	10^{-4}	10^{-4}		1.18	0.49	88627	86091	116778	3538		1.4230
	10^{-7}	10^{-7}		3.22	2.53	252827	249239	318196	7777		4.0123
GAMD	10^{-4}	10^{-4}	10^{-6}	2.34	1.65	12420	11264	474866	11264	12420	2.7572
	10^{-7}	10^{-7}	10^{-9}	6.11	5.42	18798	16913	1049423	16909	18793	6.0502
MEBDFI	10^{-4}	10^{-4}	10^{-6}	2.54	1.85	61426	61208	201899	5374	5374	1.6416
	10^{-7}	10^{-7}	10^{-9}	5.28	4.59	148609	148298	483689	12471	12471	3.9831
PSIDE-1	10^{-4}	10^{-4}		1.29	0.60	9791	8241	267721	6834	38184	1.6709
	10^{-7}	10^{-7}		5.21	4.53	55345	45636	886724	3984	111508	5.4656
RADAU5	10^{-7}	10^{-7}	10^{-9}	4.49	3.80	102515	93113	545282	12316	54746	3.7742
VODE	10^{-7}	10^{-7}		2.84	2.15	217383	207569	261396	3605	22598	2.4019

References

[Hor76] E.H. Horneber. *Analyse nichtlinearer RLCÜ-Netzwerke mit Hilfe der gemischten Potentialfunktion mit einer systematischen Darstellung der Analyse nichtlinearer dynamischer Netzwerke*. PhD thesis, Universität Kaiserslautern, 1976.

[KRS92] W. Kampowski, P. Rentrop, and W. Schmidt. Classification and numerical simulation of electric circuits. *Surveys on Mathematics for Industry*, 2(1):23–65, 1992.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

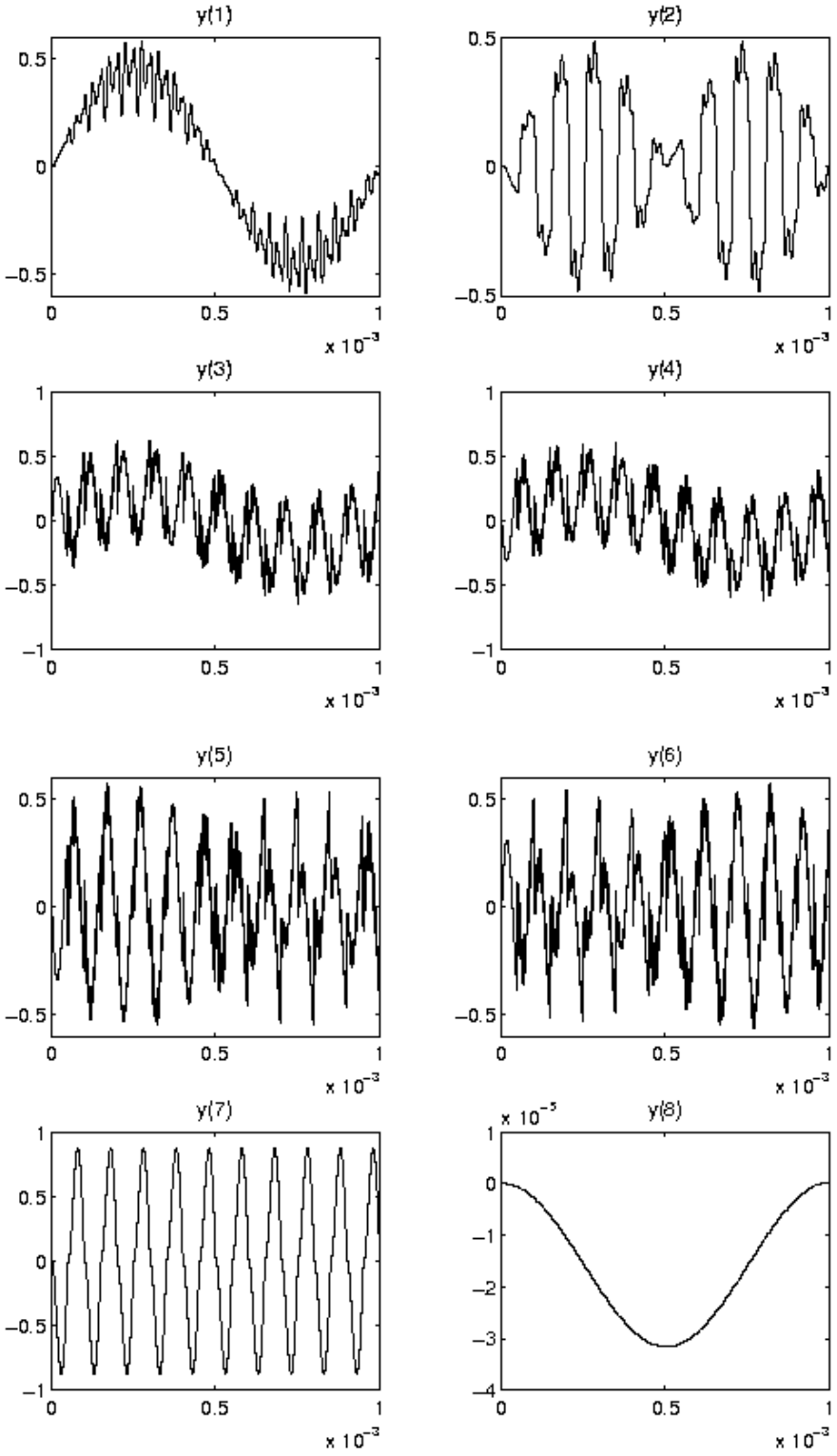

FIGURE II.3.2: *Behavior of the first eight solution components solution over the integration interval.*

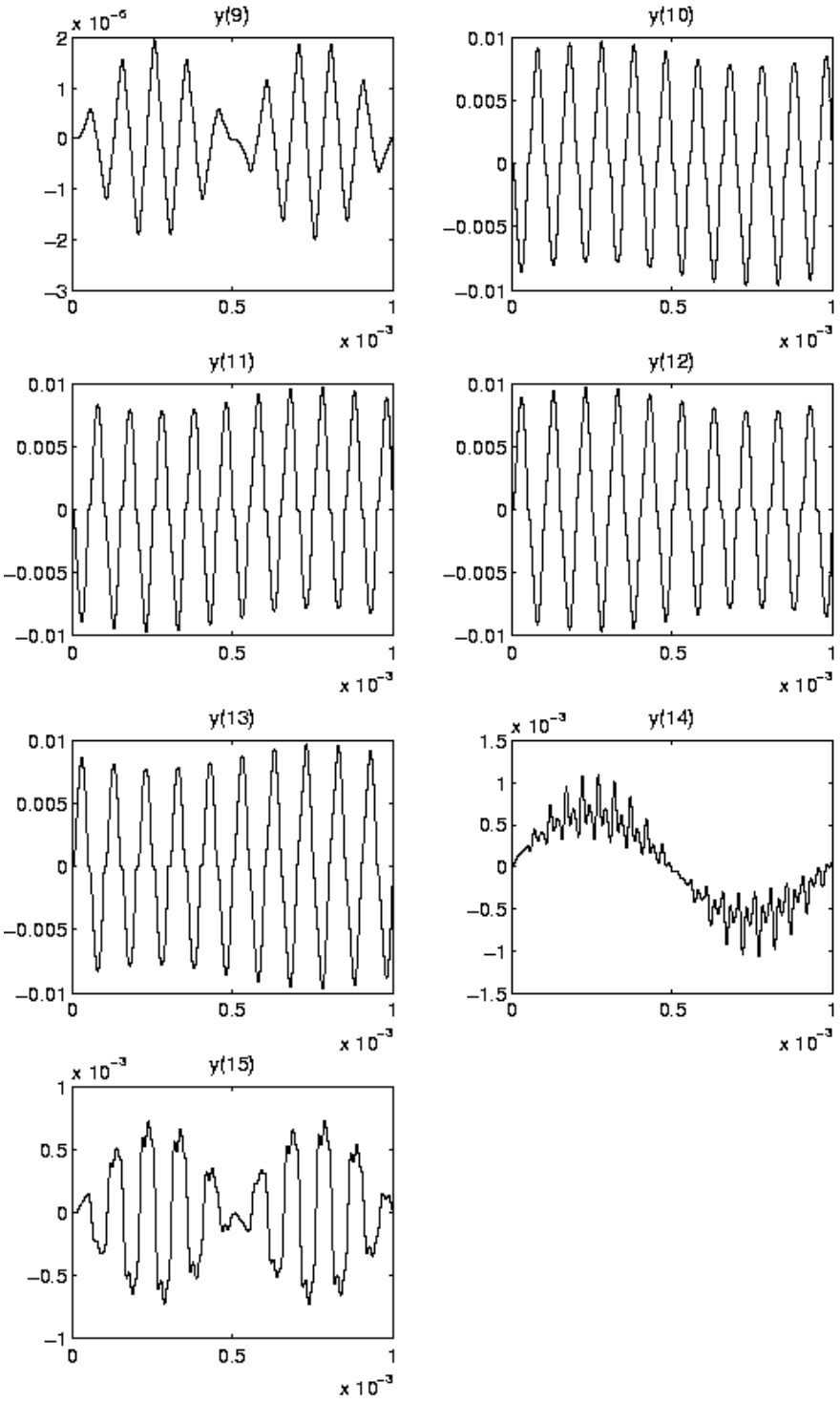

FIGURE II.3.3: *Behavior of the last seven solution components solution over the integration interval.*

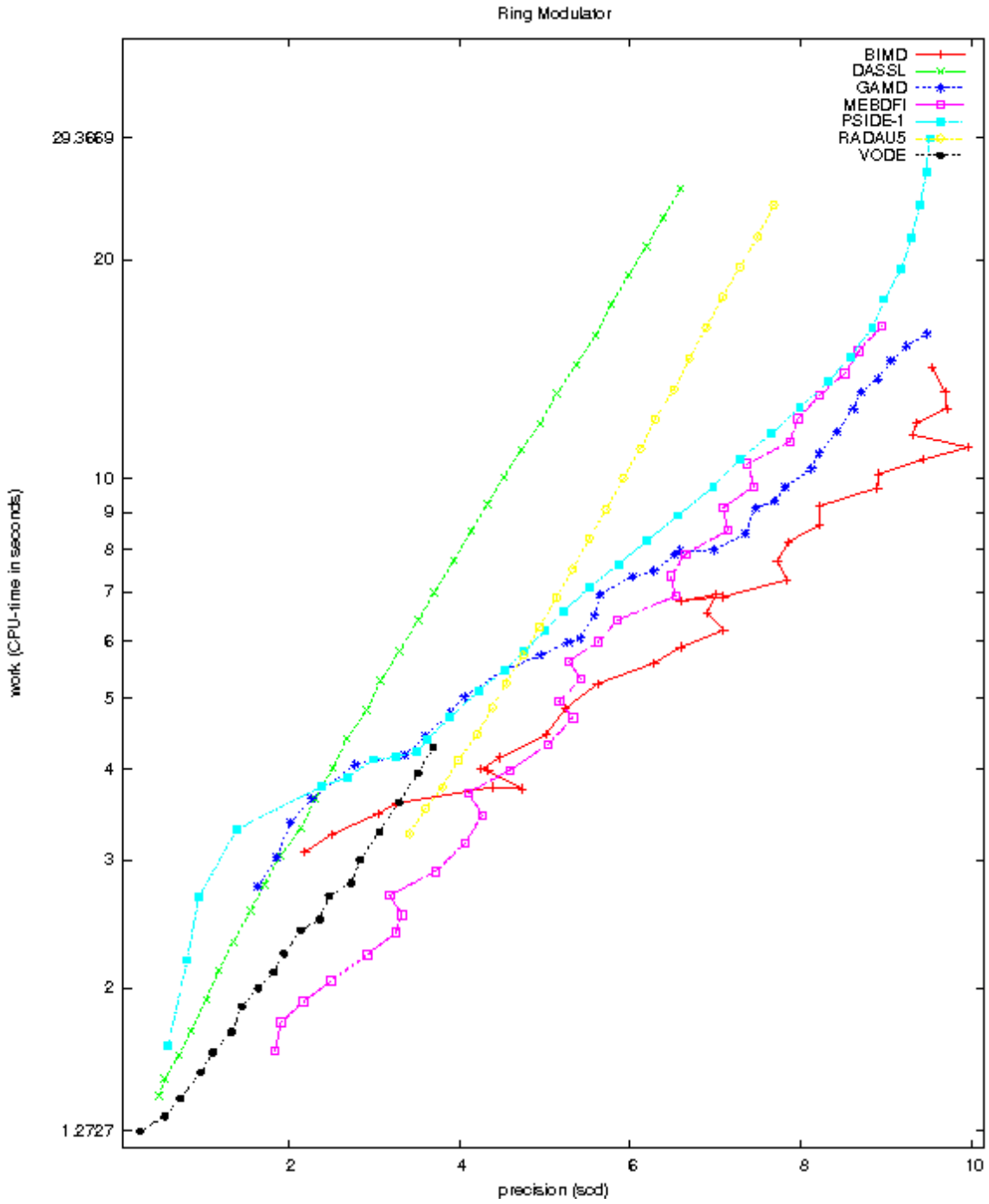

FIGURE II.3.4: *Work-precision diagram (scd versus CPU-time).*

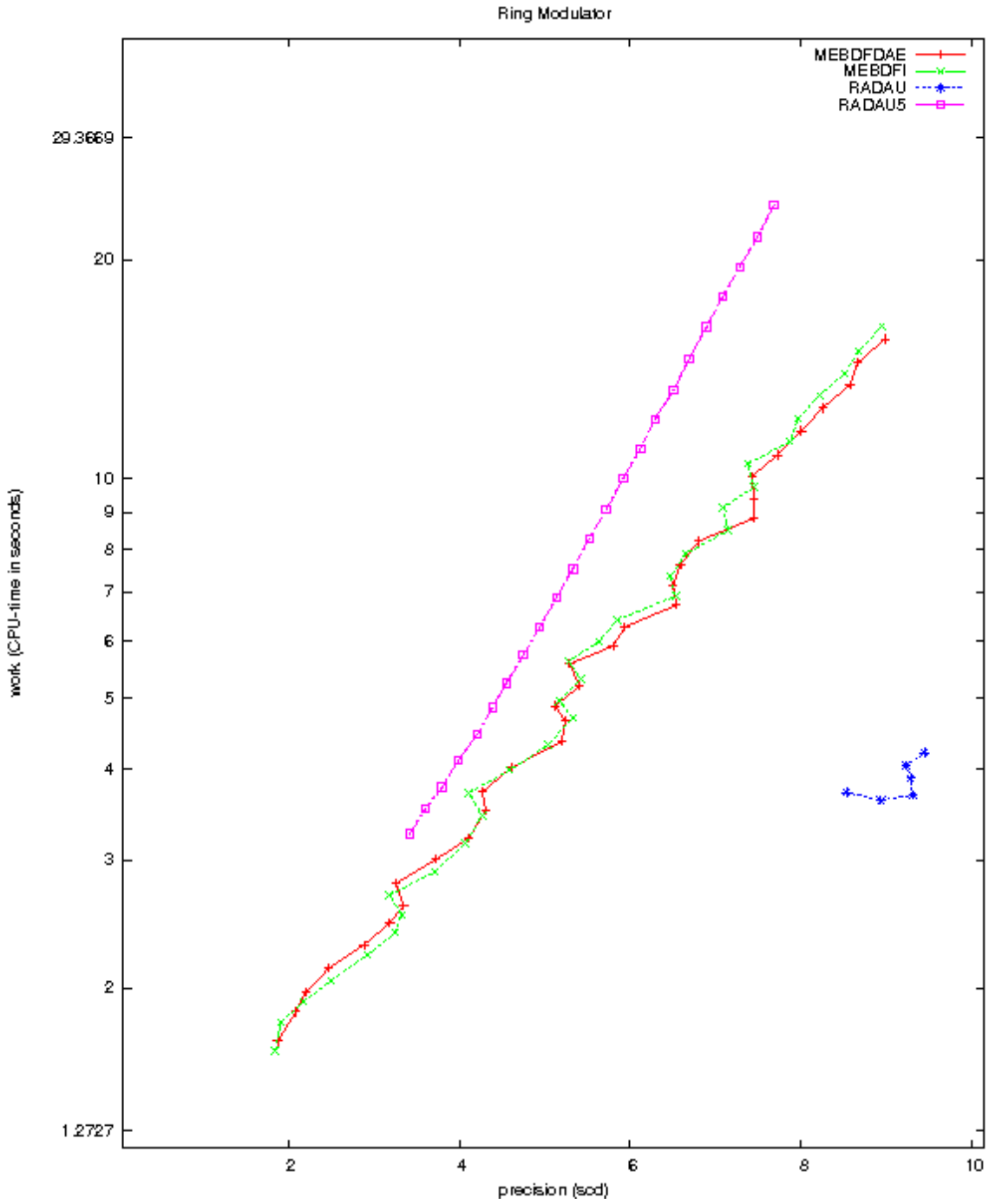

FIGURE II.3.5: *Work-precision diagram (scd versus CPU-time).*

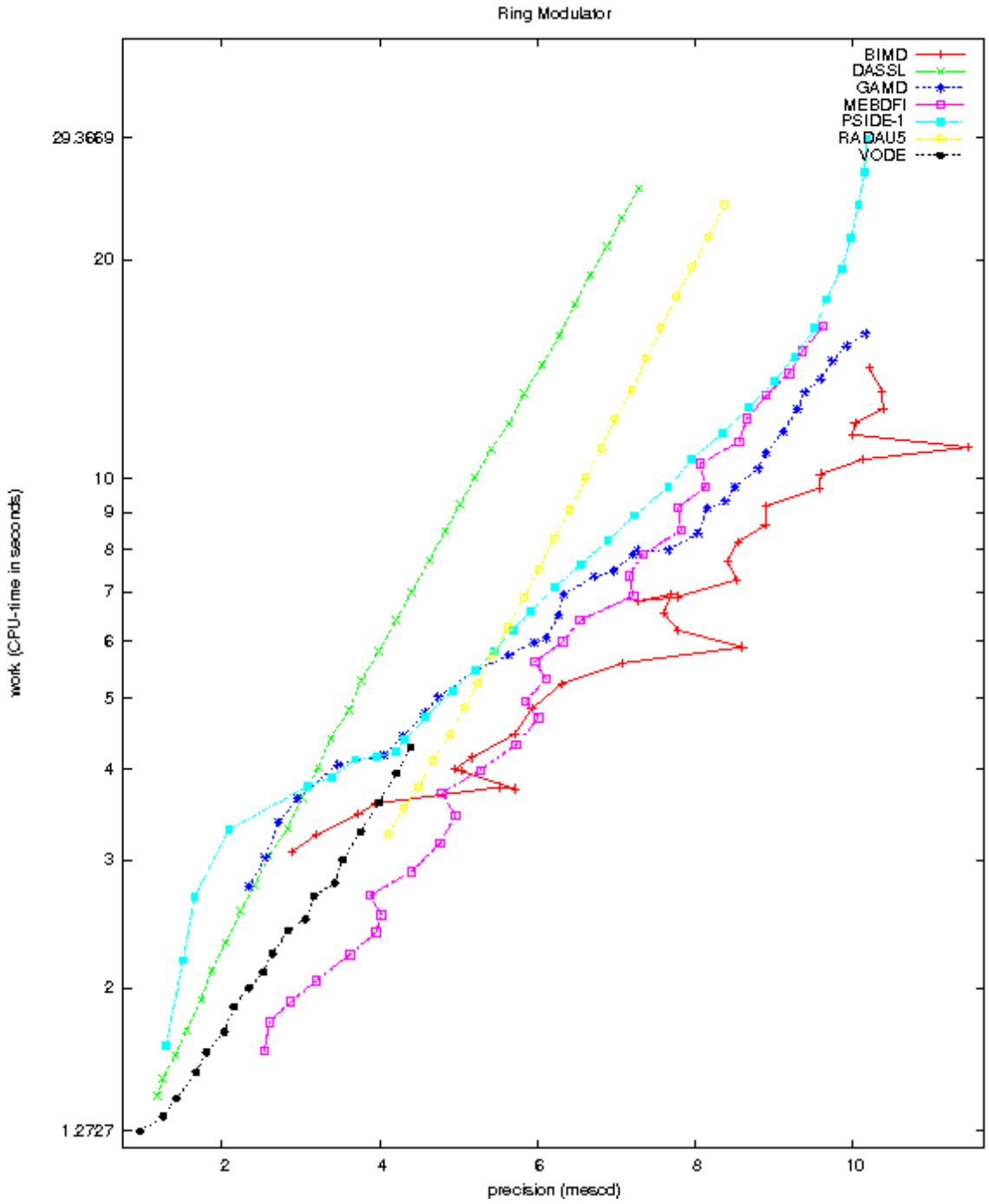

FIGURE II.3.6: *Work-precision diagram (mescd versus CPU-time).*

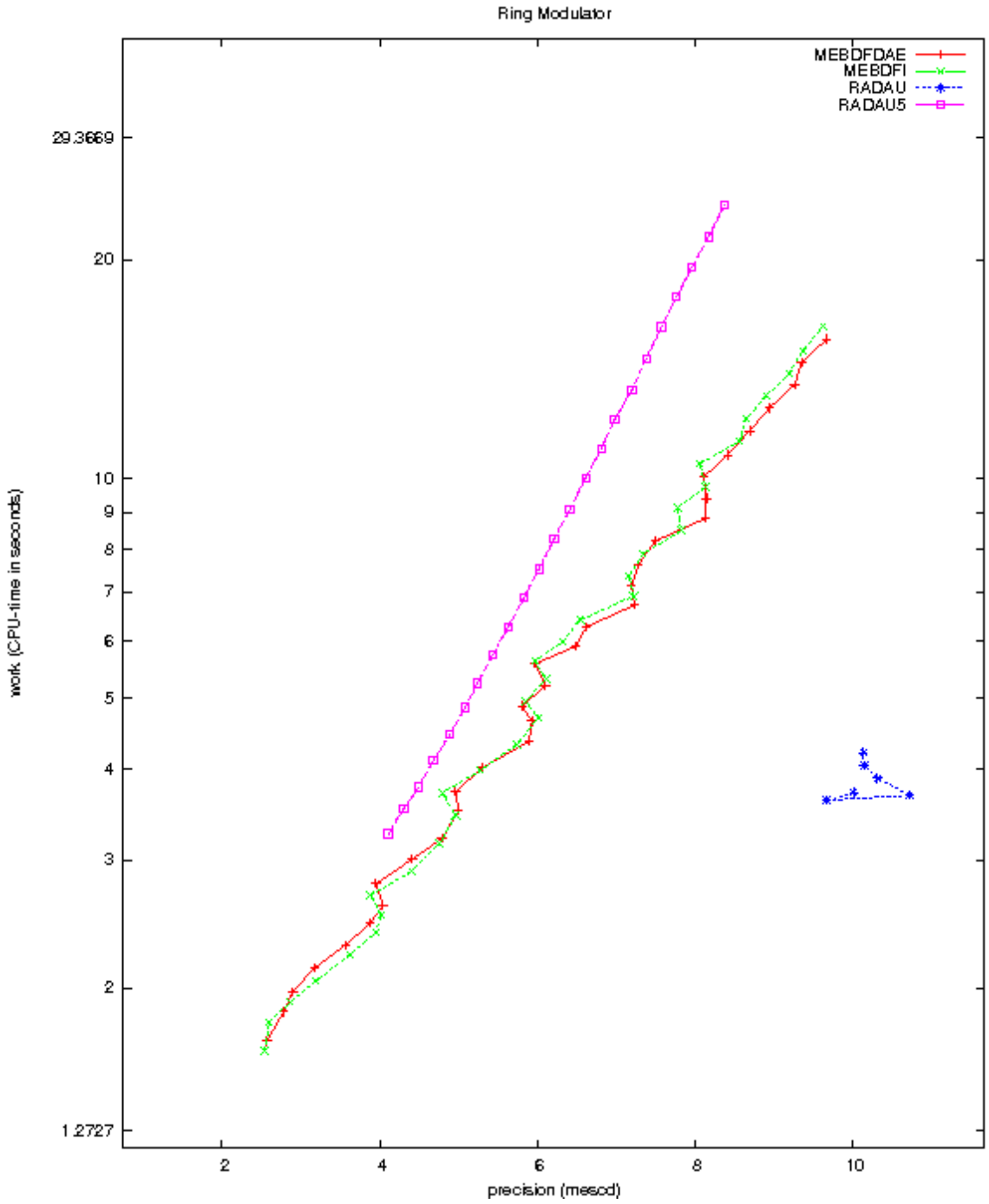

FIGURE II.3.7: *Work-precision diagram (mescd versus CPU-time).*

4 Medical Akzo Nobel problem

4.1 General information

The problem consists of 2 partial differential equations. Semi-discretization of this system yields a stiff ODE. The parallel-IVP-algorithm group of CWI contributed this problem to the test set in collaboration with R. van der Hout from Akzo Nobel Central Research.

The software part of the problem is in the file medakzo.f available at [MM08].

4.2 Mathematical description of the problem

The problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \quad y(0) = g, \tag{II.4.1}$$

with

$$y \in \mathbb{R}^{2N}, \quad 0 \le t \le 20.$$

Here, the integer N is a user-supplied parameter. The function f is given by

$$\begin{aligned} f_{2j-1} &= \alpha_j \frac{y_{2j+1} - y_{2j-3}}{2\Delta\zeta} + \beta_j \frac{y_{2j-3} - 2y_{2j-1} + y_{2j+1}}{(\Delta\zeta)^2} - k\, y_{2j-1} y_{2j}, \\ f_{2j} &= -k\, y_{2j} y_{2j-1}, \end{aligned}$$

where

$$\begin{aligned} \alpha_j &= \frac{2(j\Delta\zeta - 1)^3}{c^2}, \\ \beta_j &= \frac{(j\Delta\zeta - 1)^4}{c^2}. \end{aligned}$$

Here, j ranges from 1 to N, $\Delta\zeta = \frac{1}{N}$, $y_{-1}(t) = \phi(t)$, $y_{2N+1} = y_{2N-1}$ and $g \in \mathbb{R}^{2N}$ is given by

$$g = (0, v_0, 0, v_0, \ldots, 0, v_0)^{\mathrm{T}}.$$

The function ϕ is given by

$$\phi(t) = \begin{cases} 2 & \text{for} \quad t \in (0,5], \\ 0 & \text{for} \quad t \in (5,20]. \end{cases}$$

which means that f undergoes a discontinuity in time at $t = 5$. Suitable values for the parameters k, v_0 and c are 100, 1 and 4, respectively.

4.3 Origin of the problem

The Akzo Nobel research laboratories formulated this problem in their study of the penetration of radio-labeled antibodies into a tissue that has been infected by a tumor [Hou94]. This study was carried out for diagnostic as well as therapeutic purposes.

Let us consider a reaction diffusion system in one spatial dimension:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} - kuv, \tag{II.4.2}$$

$$\frac{\partial v}{\partial t} = -kuv, \tag{II.4.3}$$

which originates from the chemical reaction

$$A + B \xrightarrow{k} C.$$

Here A, the radio-labeled antibody, reacts with substrate B, the tissue with the tumor, and k denotes the rate constant. The concentrations of A and B are denoted by u and v, respectively. In the derivation of the equations (II.4.2) and (II.4.3) it was assumed that the reaction is governed by mass action kinetics and in addition that the chemical A is mobile while B is immobile.

Consider a clean semi-infinite slab, in which the substrate B is uniformly distributed. When the slab is exposed at its surface to the chemical A, this chemical starts to penetrate into the slab.

To model this penetration, the equations (II.4.2) and (II.4.3) are considered in the strip

$$S_T = \{(x,t) : 0 < x < \infty,\ 0 < t < T\} \quad \text{for some } T,$$

along with the following initial and boundary conditions:

$$u(x,0) = 0, \quad v(x,0) = v_0 \quad \text{for } x > 0,$$

where v_0 is a constant, and

$$u(0,t) = \phi(t) \quad \text{for } 0 < t < T.$$

In order to solve the problem numerically, we transform the variable x in such a way that the semi-infinite slab is transformed into a finite one. A suitable transformation is provided by the following special family of Möbius transformations:

$$\zeta = \frac{x}{x+c}, \quad \text{with } c > 0.$$

Each transformation in this class transforms S_T into the slab:

$$\{(\zeta,t) : 0 < \zeta < 1,\ 0 < t < T\}.$$

In terms of ζ the problem now reads:

$$\frac{\partial u}{\partial t} = \frac{(\zeta-1)^4}{c^2}\frac{\partial^2 u}{\partial \zeta^2} + \frac{2(\zeta-1)^3}{c^2}\frac{\partial u}{\partial \zeta} - kuv, \tag{II.4.4}$$

$$\frac{\partial v}{\partial t} = -kuv, \tag{II.4.5}$$

with initial conditions

$$u(\zeta,0) = 0, \quad v(\zeta,0) = v_0 \quad \text{for } \zeta > 0, \tag{II.4.6}$$

and boundary conditions

$$u(0,t) = \phi(t), \quad \frac{\partial u}{\partial \zeta}(1,t) = 0 \quad \text{for } 0 < t < T. \tag{II.4.7}$$

The last boundary condition is derived from $\frac{\partial u}{\partial x}(\infty,t) = 0$.

The system consisting of (II.4.4), (II.4.5), (II.4.6) and (II.4.7) will be written as a system of ordinary differential equations by using the method of lines, i.e. by discretizing the spatial derivatives. We use the uniform grid $\{\zeta_j\}_{j=1,\ldots,N}$ defined by:

$$\zeta_j = j \cdot \Delta\zeta, \quad j = 1,\ldots,N, \quad \Delta\zeta = \frac{1}{N}.$$

Let u_j and v_j denote the approximations of $u(\zeta_j, t)$ and $v(\zeta_j, t)$, respectively. Obviously, u_j and v_j are functions of t. In terms of the function u_j, our choices for the discretization of the spatial first and second order derivatives read

$$\frac{\partial u_j}{\partial \zeta} = \frac{u_{j+1} - u_{j-1}}{2\Delta\zeta} \quad \text{and} \quad \frac{\partial^2 u_j}{\partial \zeta^2} = \frac{u_{j-1} - 2u_j + u_{j+1}}{(\Delta\zeta)^2},$$

respectively, where $j = 1, \ldots, N$. Suitable values for u_0 and u_{N+1} are obtained from the boundary conditions. They are given by $u_0 = \phi(t)$ and $u_{N+1} = u_N$.

Defining $y(t)$ by $y = (u_1, v_1, u_2, v_2, \ldots, u_N, v_N)^{\mathrm{T}}$, and choosing $T = 20$, this semi-discretized problem is precisely the ODE (II.4.1).

To give an idea of the solution to the PDE (II.4.4)–(II.4.7), Figure II.4.1 plots u and v as function of x and t. We nicely see that injection of chemical A (locally) destroys B.

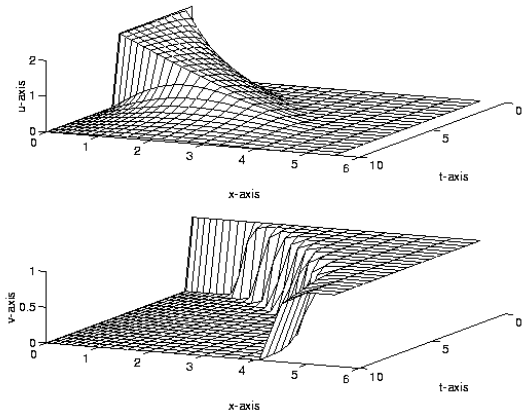

FIGURE II.4.1: *u and v as function of time and space.*

4.4 Numerical solution of the problem

The numerical experiments were done for the case $N = 200$. In Table II.4.1 we give the value of some components of the reference solution at the end of the integration interval. These components correspond to the values of u and v in $x = 1$, 2.4, 4.0 and 6.0. For the complete reference solution we refer to the Fortran subroutine `solut`. Figure II.4.2 plots the behavior of the solution components y_i for $i \in \{79, 80, 133, 134, 171, 172, 199, 200\}$, which correspond to approximations of the PDE solutions u and v on the grid lines $x = 1$, 2, 3 and 4. Table II.4.2 and Figures II.4.3–II.4.6 show the run

TABLE II.4.1: *Reference solution at the end of the integration interval.*

y_{79}	$0.2339942217046434 \cdot 10^{-3}$	y_{199}	$0.11737412926802 \cdot 10^{-3}$
y_{80}	$-0.1127916494884468 \cdot 10^{-141}$	y_{200}	$0.61908071460151 \cdot 10^{-5}$
y_{149}	$0.3595616017506735 \cdot 10^{-3}$	y_{239}	$0.68600948191191 \cdot 10^{-11}$
y_{150}	$0.1649638439865233 \cdot 10^{-86}$	y_{240}	0.99999973258552

TABLE II.4.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-9}	4.94	4.92	110	110	1565	90	110	0.1932
	10^{-7}	10^{-7}	10^{-12}	8.19	8.13	125	125	3496	115	125	0.4451
DDASSL	10^{-4}	10^{-4}		3.41	3.35	381	373	550	46		0.1200
	10^{-7}	10^{-7}		5.69	5.69	1378	1369	1700	62		0.3972
GAMD	10^{-4}	10^{-4}	10^{-9}	5.03	5.01	66	66	2116	66	66	0.2235
	10^{-7}	10^{-7}	10^{-12}	7.79	7.78	104	104	4760	104	104	0.5290
MEBDFI	10^{-4}	10^{-4}	10^{-9}	3.95	3.94	375	361	1238	70	70	0.2235
	10^{-7}	10^{-7}	10^{-12}	7.44	7.43	826	803	2749	104	104	0.5046
PSIDE-1	10^{-4}	10^{-4}		5.16	5.00	118	83	1263	34	456	0.1776
	10^{-7}	10^{-7}		7.18	7.12	159	145	2838	109	624	0.3445
RADAU	10^{-4}	10^{-4}	10^{-9}	3.87	3.82	93	93	747	60	93	0.0859
	10^{-7}	10^{-7}	10^{-12}	6.93	6.92	100	100	1807	58	100	0.1972
VODE	10^{-4}	10^{-4}		2.84	2.84	364	359	506	10	62	0.0625
	10^{-7}	10^{-7}		5.62	5.61	1036	1023	1217	19	101	0.1571

characteristics, and the work-precision diagrams, respectively. The reference solution was computed on the Cray C90, using PSIDE with Cray double precision and atol = rtol = 10^{-10}. For the work-precision diagrams, we used: rtol = $10^{-(4+m/8)}$, $m = 0, 1, \ldots, 40$; atol = rtol; h0 = $10^{-5} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. Since some solution components are zero, all scd values presented here denote absolute precision.

References

[Hou94] R. van der Hout, 1994. Private communication.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

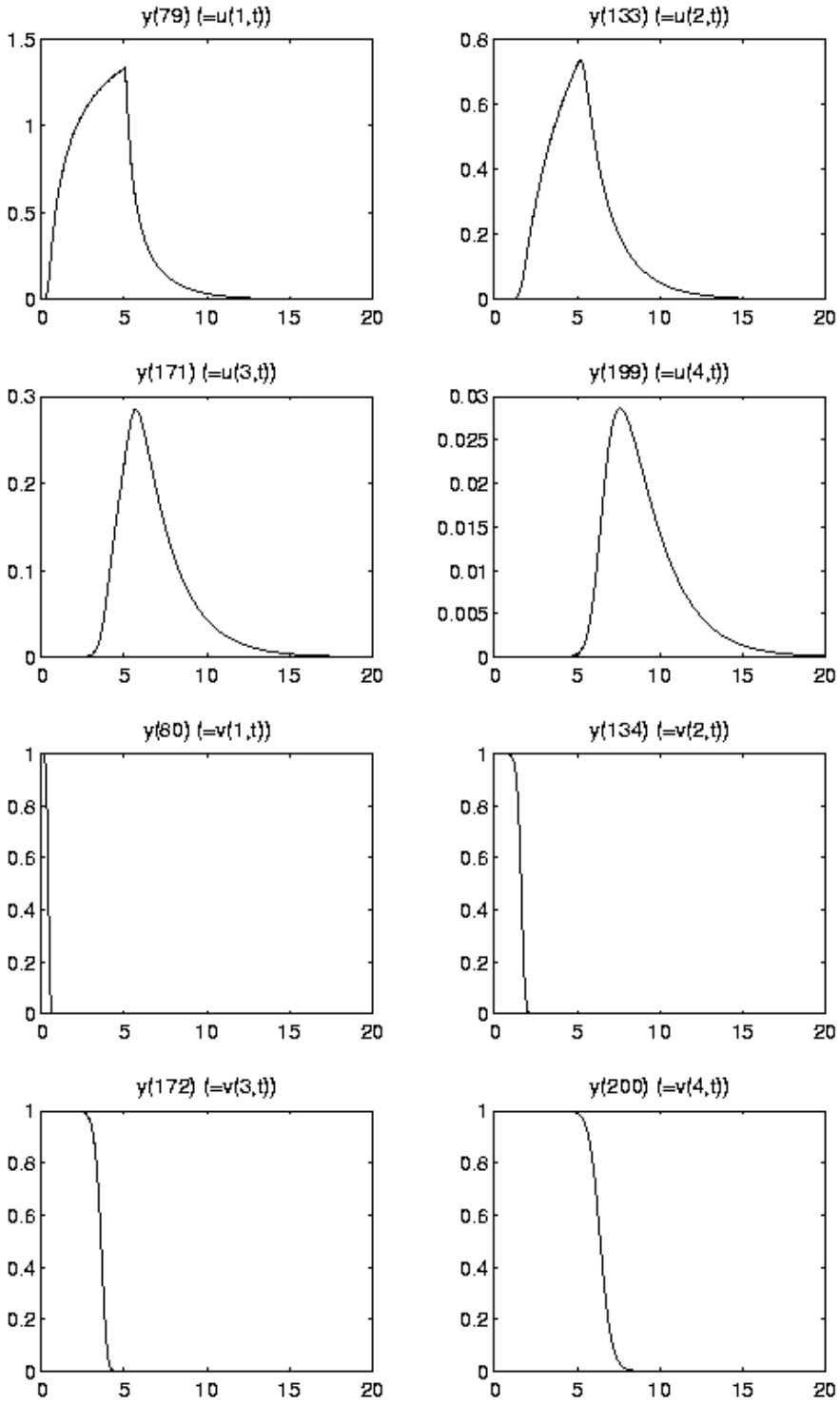

FIGURE II.4.2: *Behavior of some solution components over the integration interval.*

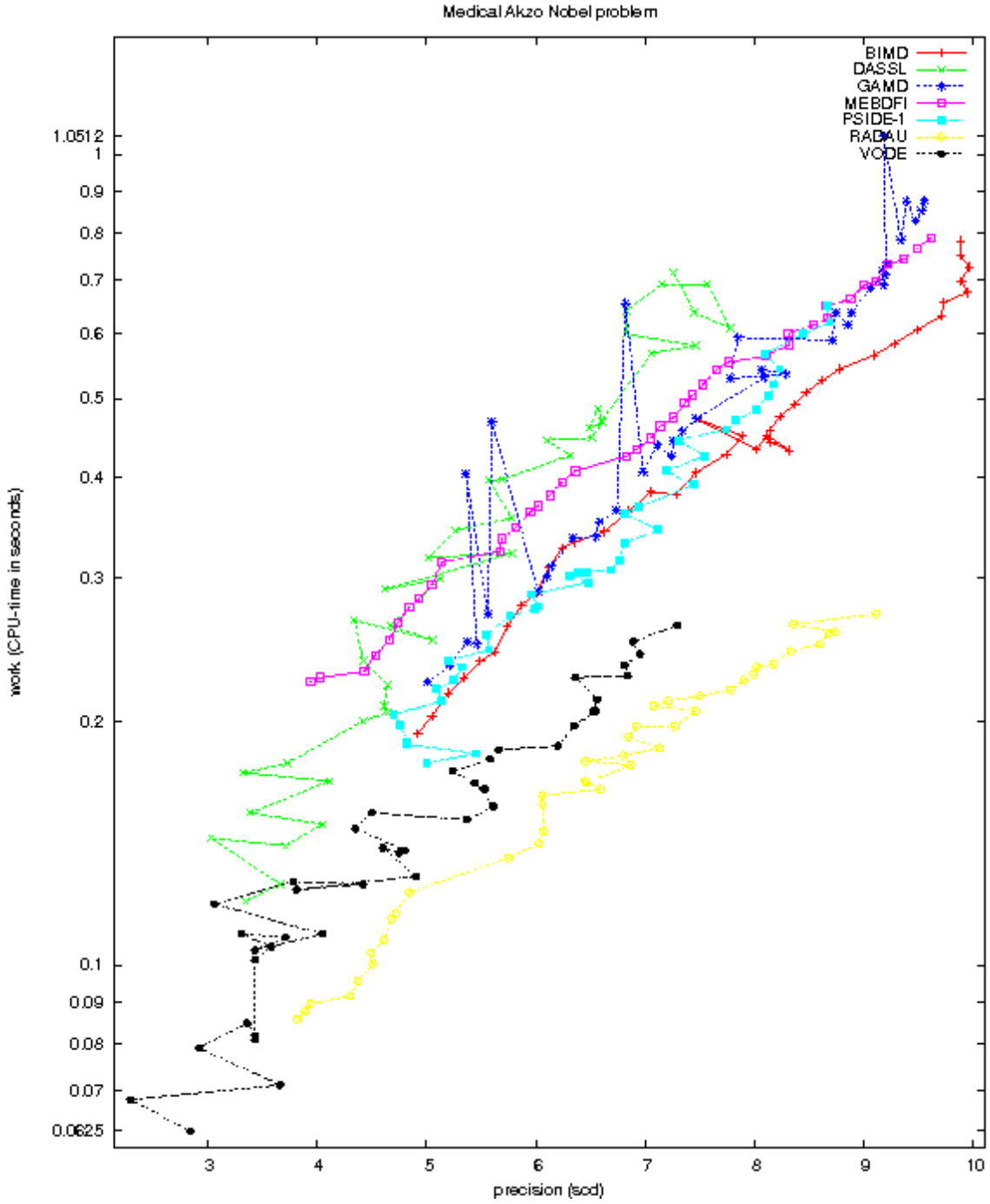

FIGURE II.4.3: *Work-precision diagram (scd versus CPU-time).*

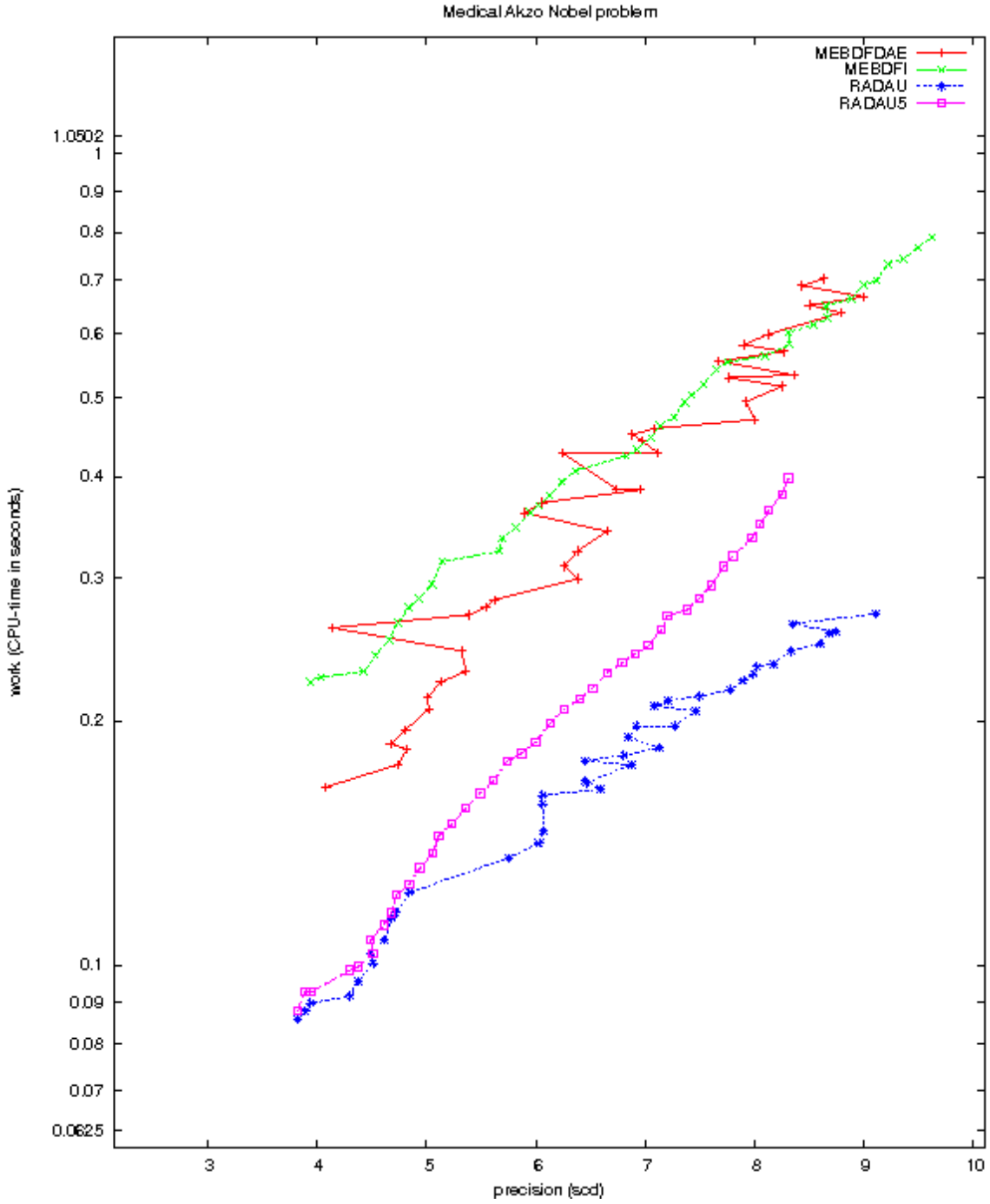

FIGURE II.4.4: *Work-precision diagram (scd versus CPU-time).*

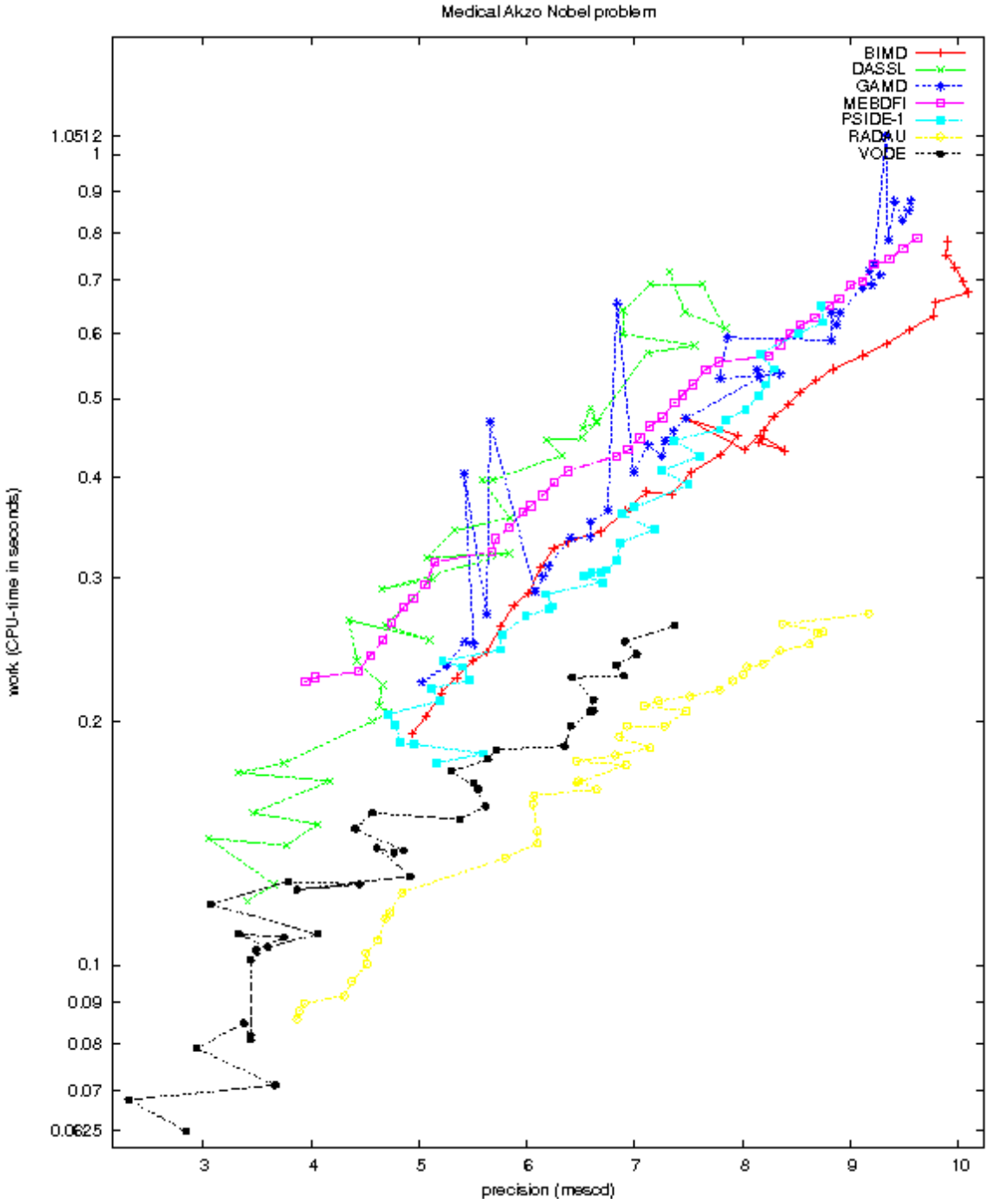

FIGURE II.4.5: *Work-precision diagram (mescd versus CPU-time).*

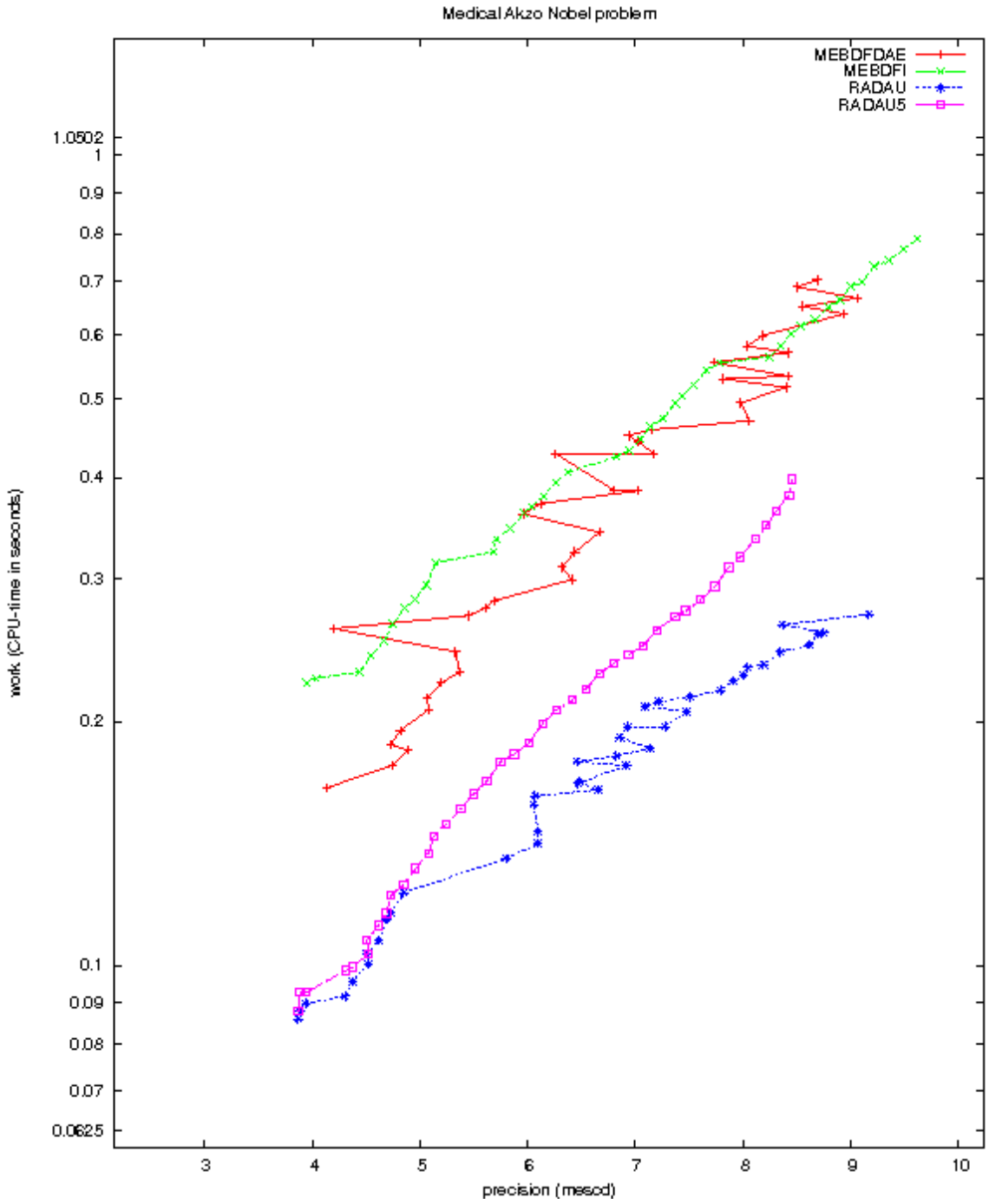

FIGURE II.4.6: *Work-precision diagram (mescd versus CPU-time).*

5 EMEP problem

5.1 General information

The problem is a stiff system of 66 ordinary differential equations. The 'Mathematics and the Environment' project group at CWI contributed this problem to the test set. The software part of the problem is in the file emep.f available at [MM08].

5.2 Mathematical description of the problem

The problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \quad y(0) = g,$$

with

$$y \in \mathbb{R}^{66}, \quad 14400 \le t \le 417600.$$

The initial vector $g = (g_i)$ is given by

$$g_i = \begin{cases} 1.0 \cdot 10^{9} & \text{for} \quad i = 1, \\ 5.0 \cdot 10^{9} & \text{for} \quad i \in \{2,3\}, \\ 3.8 \cdot 10^{12} & \text{for} \quad i = 4, \\ 3.5 \cdot 10^{13} & \text{for} \quad i = 5, \\ 1.0 \cdot 10^{7} & \text{for} \quad i \in \{6,7,\ldots,13\}, \\ 5.0 \cdot 10^{11} & \text{for} \quad i = 14, \\ 1.0 \cdot 10^{2} & \text{for} \quad i \in \{15,16,\ldots,37\}, \\ 1.0 \cdot 10^{-3} & \text{for} \quad i = 38, \\ 1.0 \cdot 10^{2} & \text{for} \quad i \in \{39,40,\ldots,66\}. \end{cases}$$

The function f has discontinuities in time at $t = 3600(4{+}24i)$ and $t = 3600(-4{+}24i)$ for $i = 1,2,3,4,5$. Since f is too voluminous to be described here, we refer to the Fortran subroutine `feval` and to [VS94] to get more insight in the function.

5.3 Origin of the problem

The problem is the chemistry part of the EMEP MSC-W ozone chemistry model, which is in development at the Norwegian Meteorological Institute in Oslo, Norway. About 140 reactions with a total of 66 species are involved. Below we give the correspondence between the solution vector y and the chemical species.

$y = ($ NO, NO2, SO2, CO, CH4, C2H6,
NC4H10, C2H4, C3H6, OXYL, HCHO, CH3CHO,
MEK, O3, HO2, HNO3, H2O2, H2,
CH3O2, C2H5OH, SA, CH3O2H, C2H5O2, CH3COO,
PAN, SECC4H, MEKO2, R2OOH, ETRO2, MGLYOX,
PRRO2, GLYOX, OXYO2, MAL, MALO2, OP,
OH, OD, NO3, N2O5, ISOPRE, NITRAT,
ISRO2, MVK, MVKO2, CH3OH, RCO3H, OXYO2H,
BURO2H, ETRO2H, PRRO2H, MEKO2H, MALO2H, MACR,
ISNI, ISRO2H, MARO2, MAPAN, CH2CCH3, ISONO3,
ISNIR, MVKO2H, CH2CHR, ISNO3H, ISNIRH, MARO2H $)^{\mathrm{T}}$.

TABLE II.5.1: *Reference solution at the end of the integration interval.*

NO	$=0.2564580511140732 \cdot 10^{8}$	CH4	$=0.3459285326034955 \cdot 10^{14}$
NO2	$=0.5146134770952715 \cdot 10^{11}$	O3	$=0.3150308585365321 \cdot 10^{13}$
SO2	$=0.2315679957701715 \cdot 10^{12}$	N2O5	$=0.7684596616753747 \cdot 10^{9}$

The integration interval covers 112 hours. Rate coefficients are often variable. Some of them undergo a discontinuity at sunrise and sunset, which correspond to $t = 3600(\pm 4 + 24i)$, respectively, for $i = 1, 2, 3, 4, 5$. The unit of the species is number of molecules per cm^3, the time t is in seconds. The test problem corresponds to the rural case in [VS94]. From the plot of O3 versus time in Figure II.5.1 we see that in this model the ozone concentration steadily grows over the integration interval. A more elaborate description of the model can be found in [VS94], [Sim93] and [SASJ93].

5.4 Numerical solution of the problem

Table II.5.1 and Figure II.5.1 present the value of reference solution at the end of the integration interval $t = 417600$ and the behavior of the solution over the integration interval of the components of y corresponding to NO, NO2, SO2, CH4, O3 and N2O5 (i.e. y_1, y_2, y_3, y_5, y_{14} and y_{40}). For the complete reference solution at the end of the integration interval we refer to the Fortran subroutine `solut`. The values at the horizontal axis in Figure II.5.1 denote the time t in hours modulo 24 hours. Table II.5.2 and Figures II.5.2–II.5.5 contain the run characteristics and the work-precision diagrams, respectively. Since components y_{36} and y_{38} are relatively very small and physically unimportant, we did not include these components in the computation of the scd value. The reference solution was computed using RADAU5 with rtol $= 10^{-14}$, atol $= 10^{-8}$, h0 $= 10^{-8}$. For the work-precision diagrams, we used: rtol $= 10^{-(2+m/8)}$, $m = 0, 1, \ldots, 32$; atol $= 1$ and h0 $= 10^{-7}$ for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

References

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[SASJ93] D. Simpson, Y. Andersson-Skold, and M.E. Jenkin. Updating the chemical scheme for the EMEP MSC-W model: Current status. Report EMEP MSC-W Note 2/93, The Norwegian Meteorological Institute, Oslo, 1993.

[Sim93] D. Simpson. Photochemical model calculations over Europe for two extended summer periods: 1985 and 1989. model results and comparisons with observations. *Atmospheric Environment*, 27A:921–943, 1993.

[VS94] J.G. Verwer and D. Simpson. Explicit methods for stiff ODEs from atmospheric chemistry. Report NM-R9409, CWI, Amsterdam, 1994.

TABLE II.5.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-2}	1	10^{-7}	1.63	2.48	300	278	4797	212	296	0.3250
	10^{-4}	1	10^{-7}	2.39	2.39	509	485	8292	445	501	0.5553
	10^{-6}	1	10^{-7}	5.12	5.12	808	748	17116	639	793	1.0873
DDASSL	10^{-2}	1		1.57	1.82	741	701	1340	171		0.1200
	10^{-4}	1		3.48	3.48	1938	1880	3322	254		0.2557
	10^{-6}	1		5.35	5.35	3964	3851	6221	404		0.4714
GAMD	10^{-2}	1	10^{-7}	2.46	2.46	347	283	10656	283	347	0.4851
	10^{-4}	1	10^{-7}	2.92	2.92	335	300	13551	300	335	0.6188
	10^{-6}	1	10^{-7}	4.66	4.64	607	503	28488	504	607	1.2629
MEBDFI	10^{-2}	1	10^{-7}	1.17	1.17	649	597	2537	130	130	0.1454
	10^{-4}	1	10^{-7}	3.53	3.53	1320	1252	4834	216	216	0.2772
	10^{-6}	1	10^{-7}	4.80	4.80	2621	2458	9214	406	406	0.5407
PSIDE-1	10^{-2}	1		1.58	2.39	490	438	6954	175	1908	0.8462
	10^{-4}	1		2.29	2.29	509	447	9241	213	1980	0.9516
	10^{-6}	1		3.97	3.95	769	650	15861	335	2716	1.4240
RADAU	10^{-2}	1	10^{-7}	1.59	2.57	398	325	3510	224	398	0.6159
	10^{-4}	1	10^{-7}	2.68	2.68	542	492	4815	377	542	0.8433
	10^{-6}	1	10^{-7}	3.62	3.60	463	390	10241	281	463	1.3566
VODE	10^{-2}	1		0.87	0.87	884	859	1409	62	272	0.1396
	10^{-4}	1		2.49	2.49	2296	2199	3547	64	383	0.2586
	10^{-6}	1		4.51	4.49	4302	4078	6090	82	637	0.4431

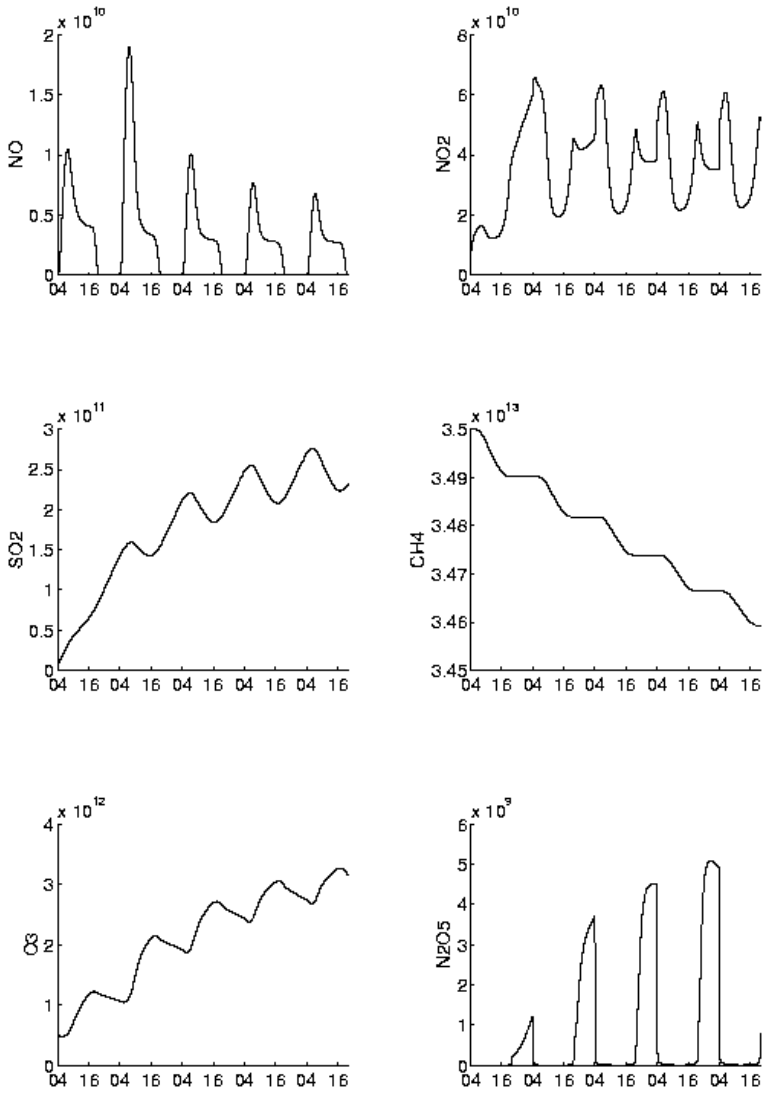

FIGURE II.5.1: *Behavior of the solution over the integration interval.*

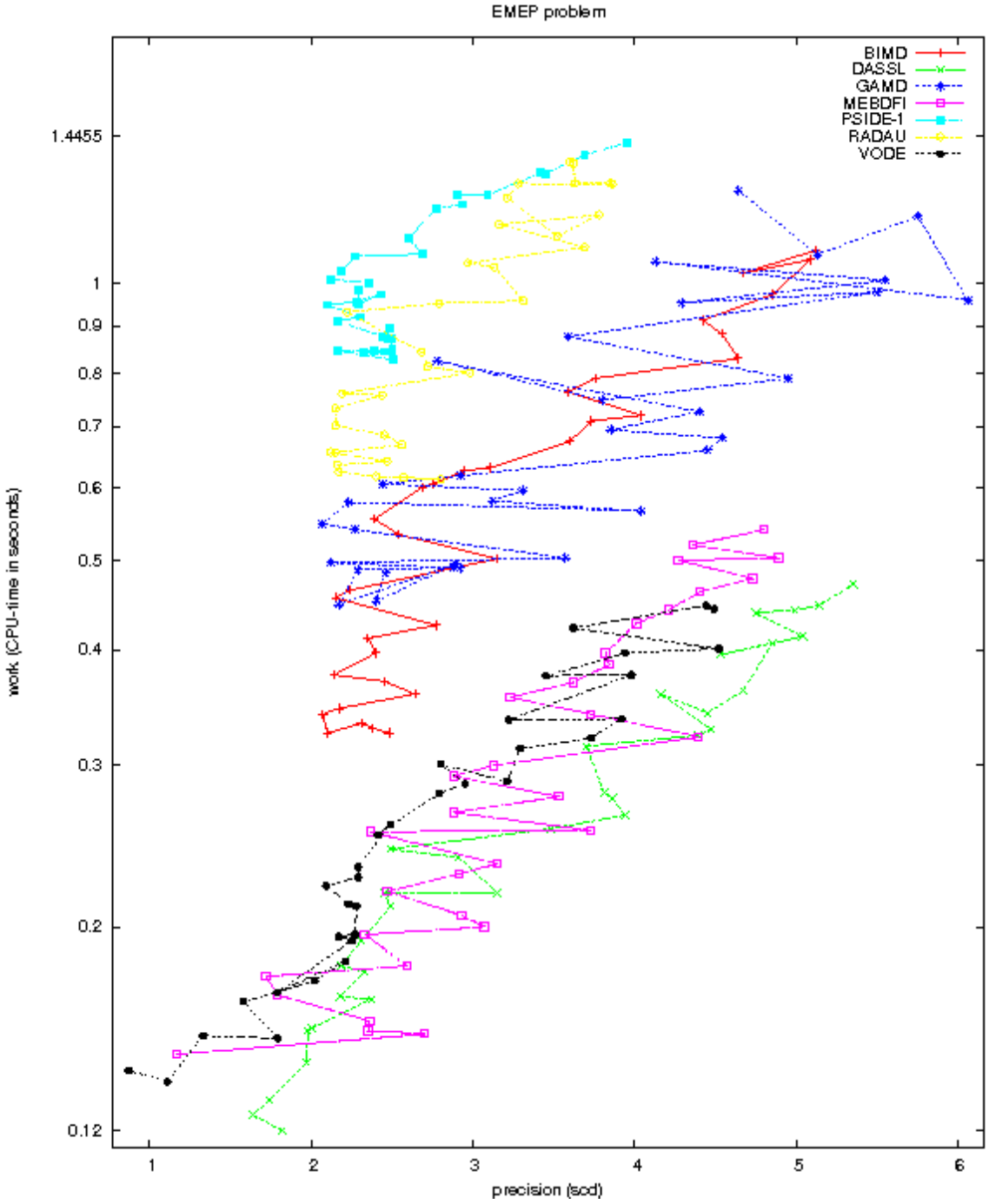

FIGURE II.5.2: *Work-precision diagram (scd versus CPU-time).*

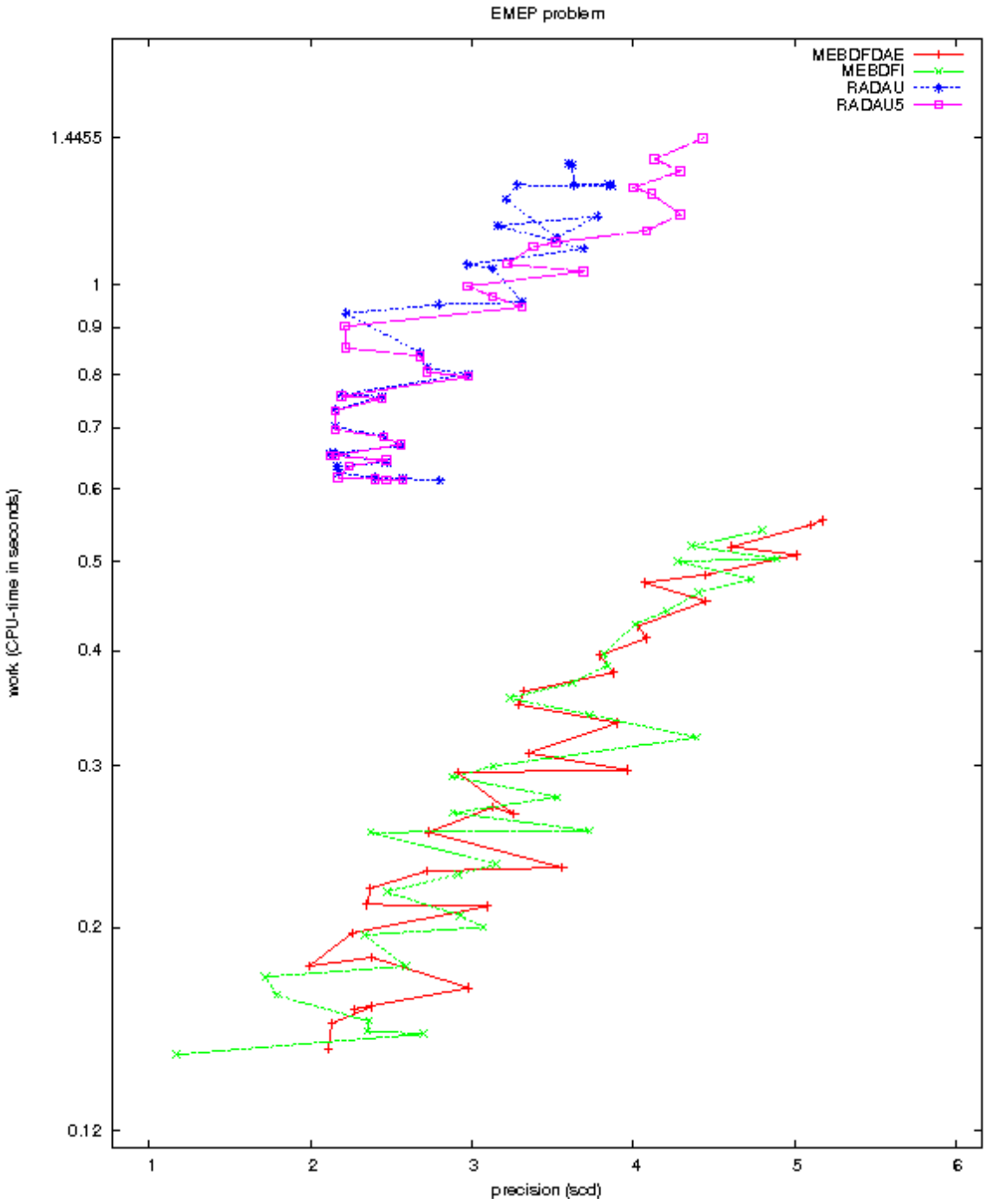

FIGURE II.5.3: *Work-precision diagram (scd versus CPU-time).*

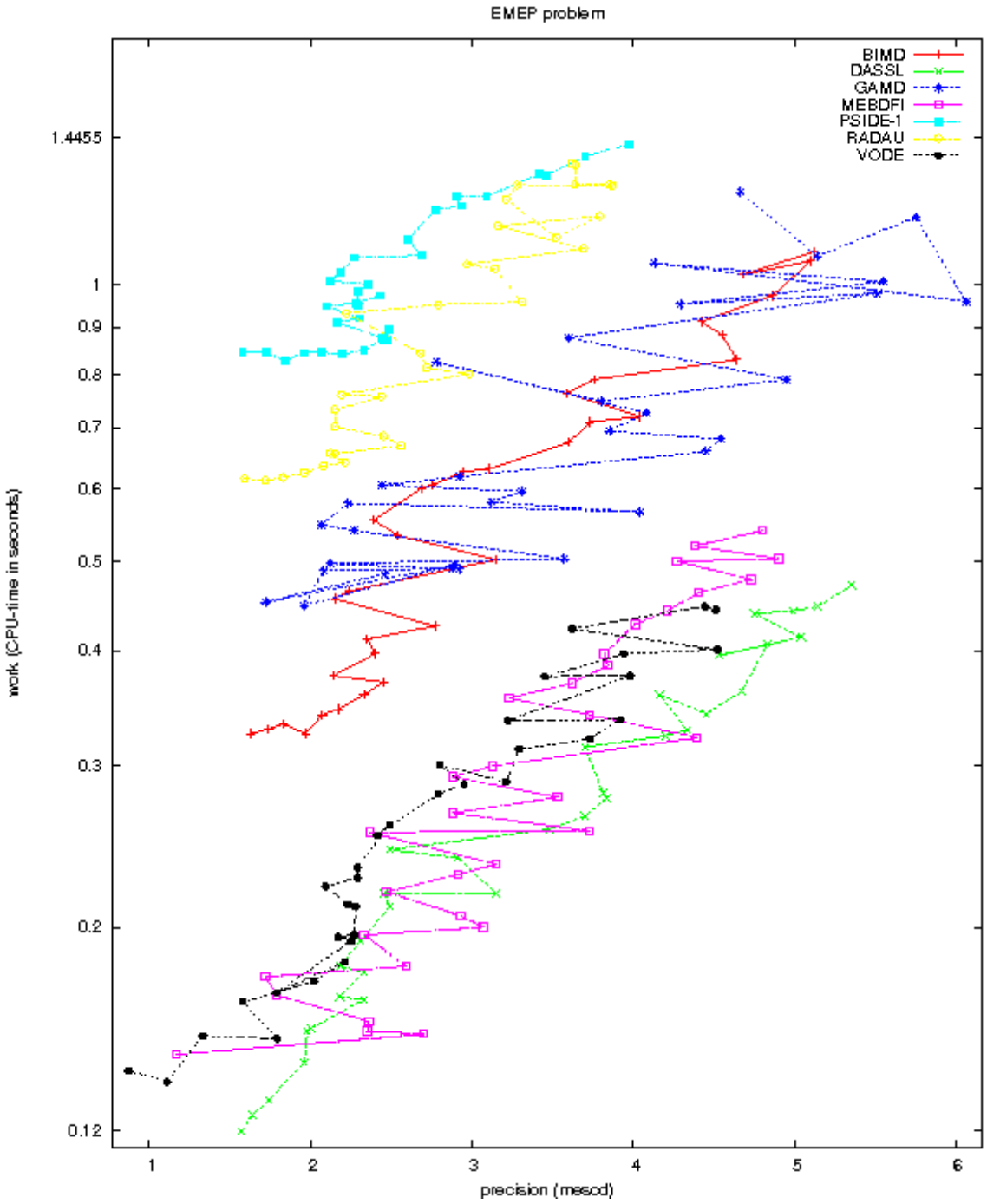

FIGURE II.5.4: *Work-precision diagram (mescd versus CPU-time).*

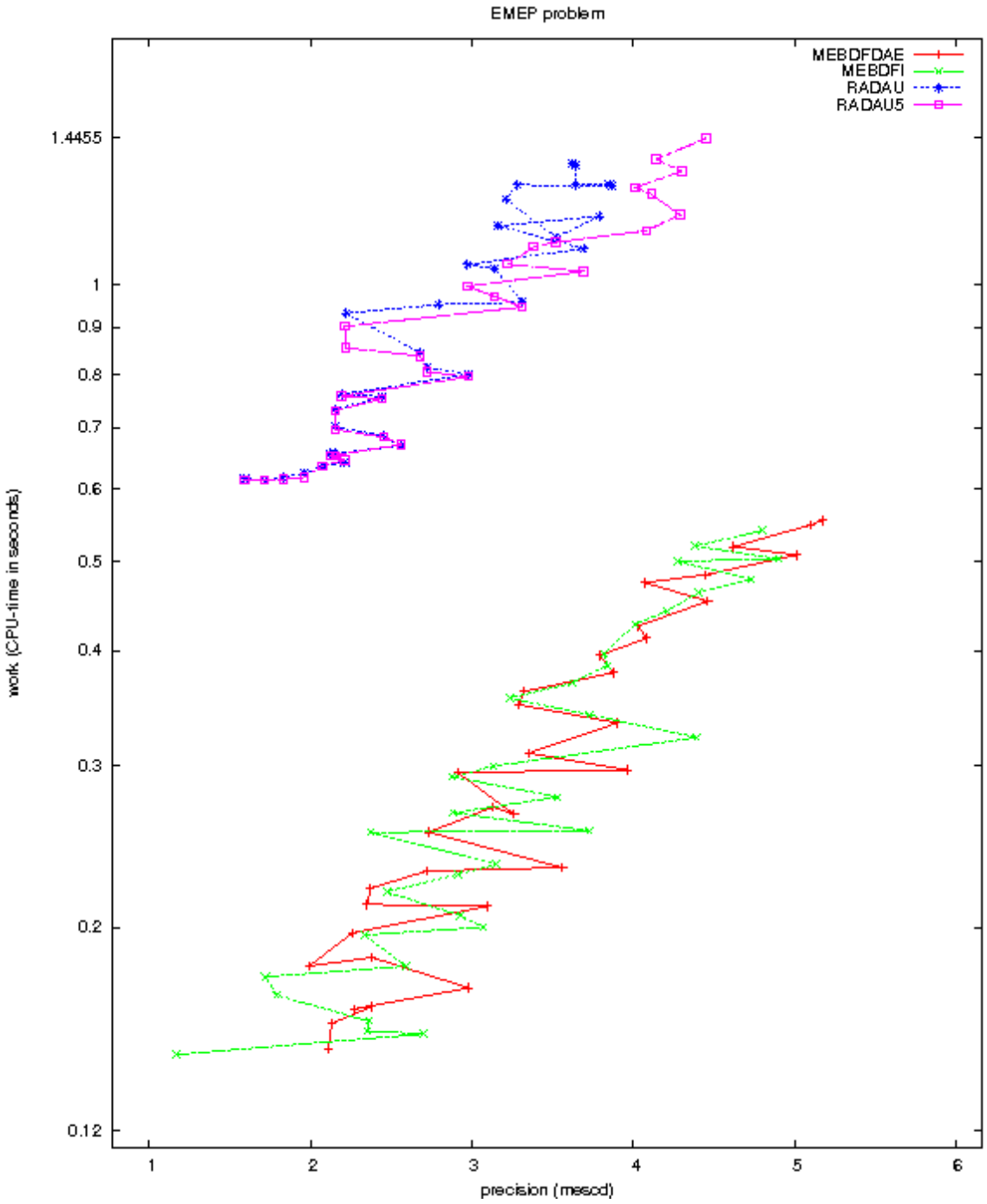

FIGURE II.5.5: *Work-precision diagram (mescd versus CPU-time).*

6 Pleiades problem

6.1 General information

The problem consists of a nonstiff system of 14 special second order differential equations rewritten to first order form, thus providing a nonstiff system of ordinary differential equations of dimension 28. The formulation and data have been taken from [HNW93]. E. Messina contributed this problem to the test set. Comments to `eleonora.messina@unina.it`.

The software part of the problem is in the file plei.f available at [MM08].

6.2 Mathematical description of the problem

The problem is of the form

$$z'' \quad = \quad f(z), \quad z(0) = z_0, \quad z'(0) = z'_0, \tag{II.6.1}$$

with

$$z \in \mathbb{R}^{14}, \quad 0 \le t \le 3.$$

Defining $z := (x^{\mathrm{T}}, y^{\mathrm{T}})^{\mathrm{T}}$, $x, y \in \mathbb{R}^7$, the function $f : \mathbb{R}^{14} \to \mathbb{R}^{14}$ is given by $f(z) = f(x, y) = (f^{(1)}(x, y)^{\mathrm{T}}, f^{(2)}(x, y)^{\mathrm{T}})^{\mathrm{T}}$, where $f^{(1,2)} : \mathbb{R}^{14} \to \mathbb{R}^7$ read

$$f_i^{(1)} = \sum_{j \neq i} m_j (x_j - x_i)/r_{ij}^{\frac{3}{2}}, \qquad f_i^{(2)} = \sum_{j \neq i} m_j (y_j - y_i)/r_{ij}^{\frac{3}{2}}, \qquad i = 1, \ldots, 7. \tag{II.6.2}$$

Here, $m_i = i$ and

$$r_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2.$$

We write this problem to first order form by defining $w = z'$, yielding a system of 28 non-linear differential equations of the form

$$\begin{pmatrix} z \\ w \end{pmatrix}' = \begin{pmatrix} w \\ f(z) \end{pmatrix} \tag{II.6.3}$$

with

$$(z^{\mathrm{T}}, w^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{28}, \quad 0 \le t \le 3.$$

The initial values are

$$\begin{pmatrix} z_0 \\ w_0 \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ x'_0 \\ y'_0 \end{pmatrix}, \quad \text{where} \quad \begin{cases} x_0 &= (3, 3, -1, -3, 2, -2, 2)^{\mathrm{T}}, \\ y_0 &= (3, -3, 2, 0, 0, -4, 4)^{\mathrm{T}}, \\ x'_0 &= (0, 0, 0, 0, 0, 1.75, -1.5)^{\mathrm{T}}, \\ y'_0 &= (0, 0, 0, -1.25, 1, 0, 0)^{\mathrm{T}}. \end{cases}$$

6.3 Origin of the problem

The Pleiades problem is a celestial mechanics problem of seven stars in the plane of coordinates x_i, y_i and masses $m_i = i$ $(i = 1, \ldots, 7)$. We obtain the formulation of the problem by means of some mechanical considerations. Let us consider the body i. According to the second law of Newton this star is subjected to the action

$$F_i = m_i p_i'', \tag{II.6.4}$$

where $p_i := (x_i, y_i)^{\mathrm{T}}$. On the other hand, the law of gravity states that the force working on body i implied by body j, denoted by F_{ij}, is

$$F_{ij} = g \frac{m_i \cdot m_j}{\|p_i - p_j\|_2^2} d_{ij}. \tag{II.6.5}$$

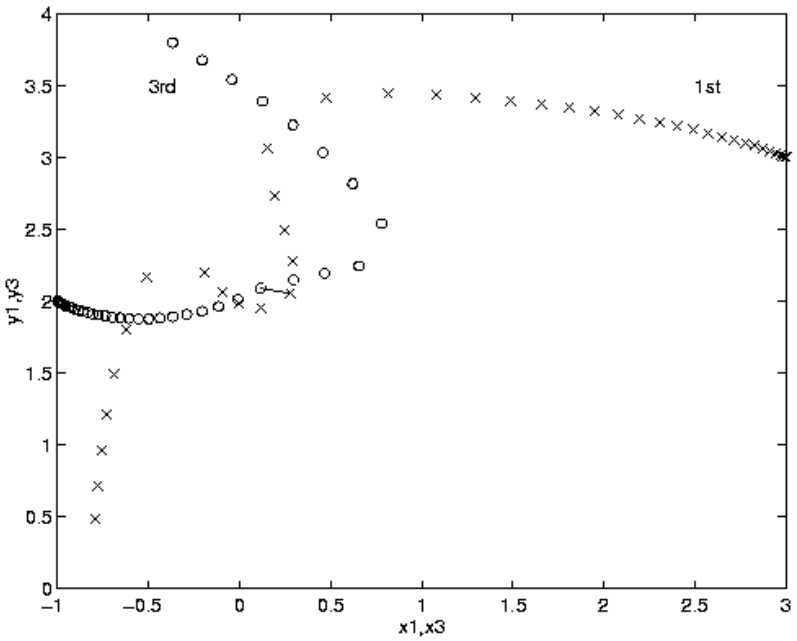

FIGURE II.6.1: *Trajectories of the first and third body on* $[0, 2]$.

TABLE II.6.1: *Quasi-collisions in Pleiades problem. The squared distance between body* i *and body* j *at* $t = \tau$ *is listed (values taken from [HNW93]).*

i	1	1	3	1	2	5
j	7	3	5	7	6	7
τ	1.23	1.46	1.63	1.68	1.94	2.14
$\|p_i - p_j\|_2^2$	0.0129	0.0193	0.0031	0.0011	0.1005	0.0700

Here, F_i, $F_{ij} \in \mathbb{R}^2$, g is the gravitational constant, which is assumed to be one here, and $d_{ij} = \frac{p_j - p_i}{\|p_j - p_i\|_2}$ represents the direction of the distance between the two stars. According to the principle of superposition of actions, F_i will be the sum of the interactions between body i and all the others,

$$F_i = \sum_{i \neq j} F_{ij}. \tag{II.6.6}$$

It is easily checked that (II.6.4)–(II.6.6) and (II.6.2) are the same.

During the movement of the 7 bodies several quasi-collisions occur which are displayed in Table II.6.1. In Figure II.6.1 the behaviors of the bodies 1 and 3 in the interval $[0, 2]$ are shown; the circles and the crosses represent data obtained every 0.05 sec, the link '—' indicates the distance occurring between the two stars at $t = 1.45$.

6.4 Numerical solution of the problem

One should be aware of the fact that the Pleiades problem is a nonstiff ODE. Therefore we also include the results obtained by the nonstiff solver DOPRI5[HW96], which is based on an explicit Runge–Kutta method.

TABLE II.6.2: *Reference solution at the end of the integration interval.*

x_1	0.3706139143970502	y_1	$-0.3943437585517392 \cdot 10$
x_2	$0.3237284092057233 \cdot 10$	y_2	$-0.3271380973972550 \cdot 10$
x_3	$-0.3222559032418324 \cdot 10$	y_3	$0.5225081843456543 \cdot 10$
x_4	0.6597091455775310	y_4	$-0.2590612434977470 \cdot 10$
x_5	0.3425581707156584	y_5	$0.1198213693392275 \cdot 10$
x_6	$0.1562172101400631 \cdot 10$	y_6	-0.2429682344935824
x_7	-0.7003092922212495	y_7	$0.1091449240428980 \cdot 10$
x'_1	$0.3417003806314313 \cdot 10$	y'_1	$-0.3741244961234010 \cdot 10$
x'_2	$0.1354584501625501 \cdot 10$	y'_2	0.3773459685750630
x'_3	$-0.2590065597810775 \cdot 10$	y'_3	0.9386858869551073
x'_4	$0.2025053734714242 \cdot 10$	y'_4	0.3667922227200571
x'_5	$-0.1155815100160448 \cdot 10$	y'_5	-0.3474046353808490
x'_6	-0.8072988170223021	y'_6	$0.2344915448180937 \cdot 10$
x'_7	0.5952396354208710	y'_7	$-0.1947020434263292 \cdot 10$

Tables II.6.2–II.6.3 and Figures II.6.2–II.6.4 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution components x_1 and y_1 over the integration interval and the work-precision diagrams, respectively. The computation of the scd values is based on the first 14 components, since they refer to the physically important quantities. The reference solution was computed on the Cray C90, using PSIDE with Cray double precision and atol = rtol = 10^{-16}. For the work-precision diagrams, we used: rtol = $10^{-(4+m/4)}$, $m = 0, 1, \ldots, 24$; atol = rtol; h0 = $10^{-2} \cdot$ rtol for BIMD, GAMD, RADAU, RADAU5 and MEBDFDAE.

.

With respect to the RADAU and RADAU5 results in Table II.6.3 and Figures II.6.3–II.6.4, we remark that for generality of the test set drivers, we did not use the facility to exploit the special structure of problems of the form (II.6.3). By setting the input parameter `IWORK(9)=14`, and adjusting the Jacobian routine appropriately, RADAU and RADAU5 produces considerably better results.

These results are listed for RADAU in Table II.6.4.

References

[HNW93] E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems.* Springer-Verlag, second revised edition, 1993.

[HW96] E. Hairer and G. Wanner. *DOPRI5*, April 25, 1996. Bug fix release sep 18, 1998. Available at http://www.unige.ch/~hairer/prog/nonstiff/dopri5.f.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

TABLE II.6.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	2.69	2.12	113	105	1955	79	110	0.0449
	10^{-7}	10^{-7}	10^{-9}	5.38	4.81	138	127	4013	123	138	0.0888
	10^{-10}	10^{-10}	10^{-12}	8.60	8.42	154	138	6947	129	152	0.1562
DDASSL	10^{-4}	10^{-4}		0.80	0.23	428	390	589	49		0.0185
	10^{-7}	10^{-7}		3.43	3.24	1237	1224	1674	59		0.0517
	10^{-10}	10^{-10}		5.88	5.72	3778	3773	4709	61		0.1425
DOPRI5	10^{-4}	10^{-4}		1.06	0.50	100	74	602			0.0059
	10^{-7}	10^{-7}		4.06	3.49	295	244	1772			0.0176
	10^{-10}	10^{-10}		8.06	7.83	940	940	5642			0.0566
GAMD	10^{-4}	10^{-4}	10^{-6}	1.54	0.97	85	69	2751	69	85	0.0566
	10^{-7}	10^{-7}	10^{-9}	4.81	4.57	122	104	5163	104	122	0.1083
	10^{-10}	10^{-10}	10^{-12}	7.65	7.30	183	177	7927	173	183	0.1649
MEBDFI	10^{-4}	10^{-4}	10^{-6}	1.12	0.56	387	366	1339	56	56	0.0303
	10^{-7}	10^{-7}	10^{-9}	3.84	3.62	835	816	2764	86	86	0.0654
	10^{-10}	10^{-10}	10^{-12}	7.14	6.94	1868	1868	6119	189	189	0.1454
PSIDE-1	10^{-4}	10^{-4}		2.23	1.82	102	76	1710	27	364	0.0410
	10^{-7}	10^{-7}		5.26	4.70	248	223	3187	1	592	0.0712
	10^{-10}	10^{-10}		8.12	7.55	807	807	9095	1	604	0.1786
RADAU	10^{-4}	10^{-4}	10^{-6}	2.67	2.11	151	138	1053	132	151	0.0303
	10^{-7}	10^{-7}	10^{-9}	6.20	6.17	112	95	2153	83	112	0.0547
	10^{-10}	10^{-10}	10^{-12}	9.41	9.20	130	119	3001	91	130	0.0742
VODE	10^{-4}	10^{-4}		0.40	−0.17	352	325	468	6	57	0.0117
	10^{-7}	10^{-7}		2.76	2.57	1081	1043	1232	18	94	0.0303
	10^{-10}	10^{-10}		5.41	5.20	3120	3079	3351	51	203	0.0830

TABLE II.6.4: *Run characteristics obtained by RADAU with exploited special structure.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
RADAU	10^{-4}	10^{-4}	10^{-6}	1.72	2.11	151	138	1053	132	151	0.0234
	10^{-7}	10^{-7}	10^{-9}	5.13	6.17	112	95	2153	83	112	0.0429
	10^{-10}	10^{-10}	10^{-12}	8.27	9.20	130	119	3001	91	130	0.0586

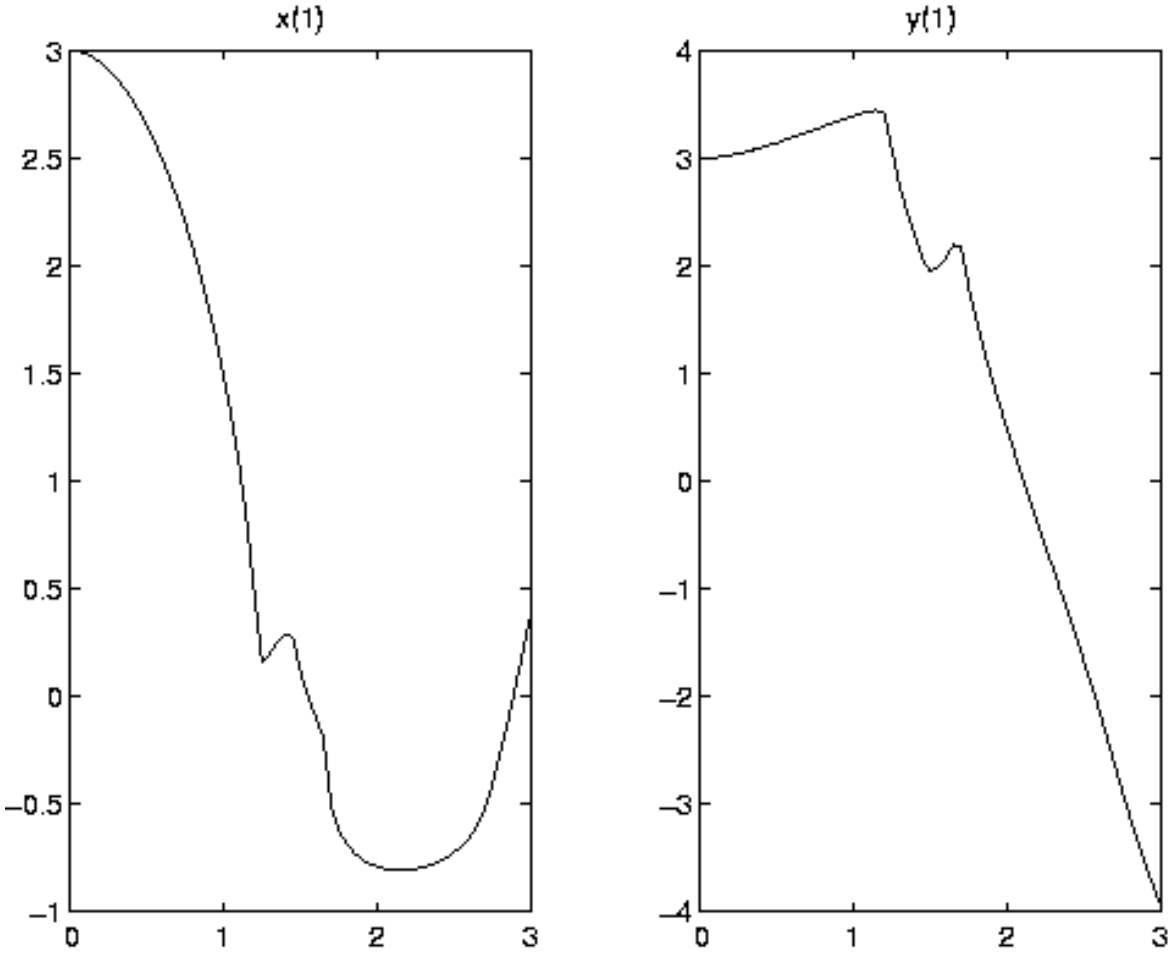

FIGURE II.6.2: *Behavior of the two solution components corresponding to the first body over the integration interval.*

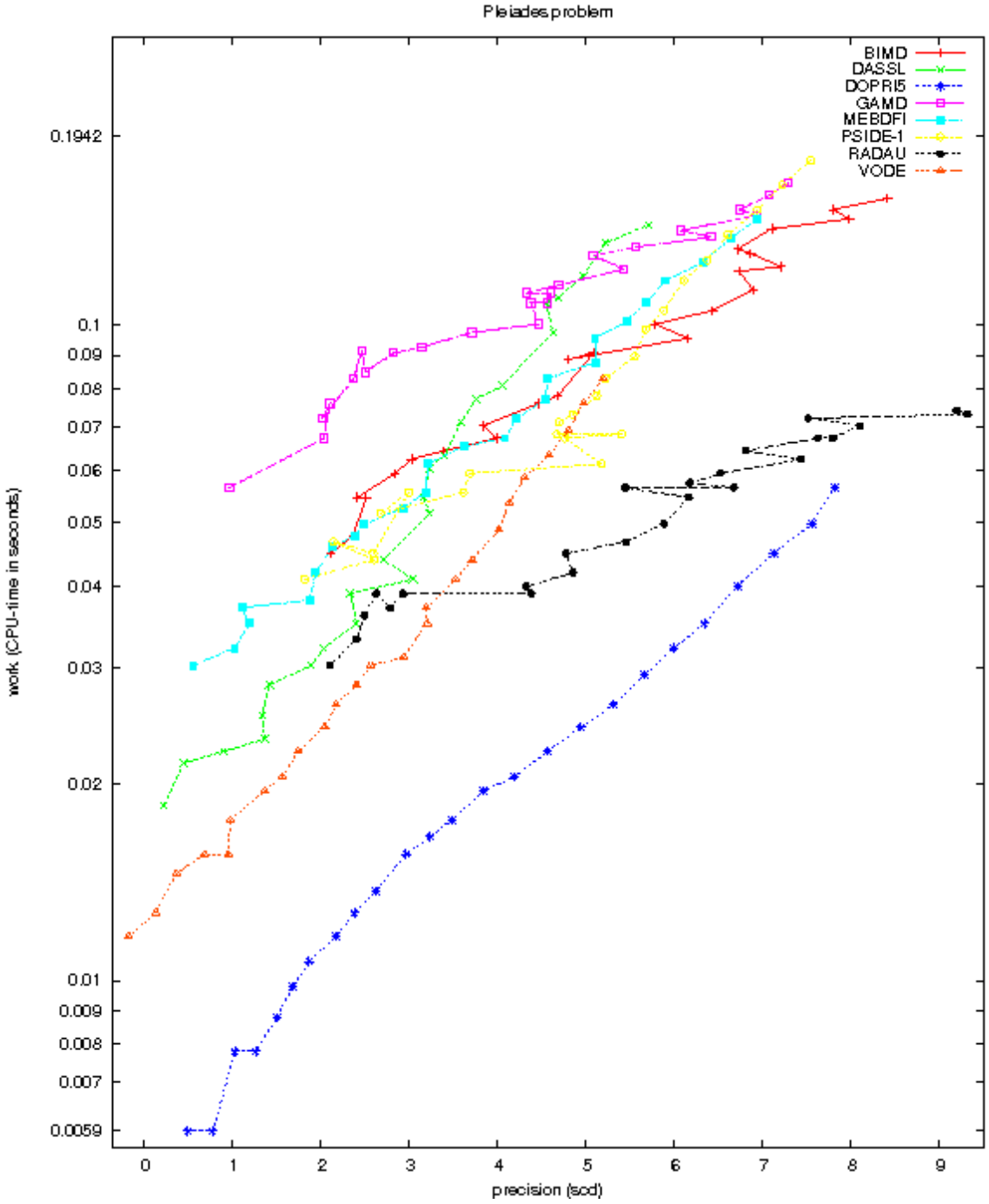

FIGURE II.6.3: *Work-precision diagram (scd versus CPU-time).*

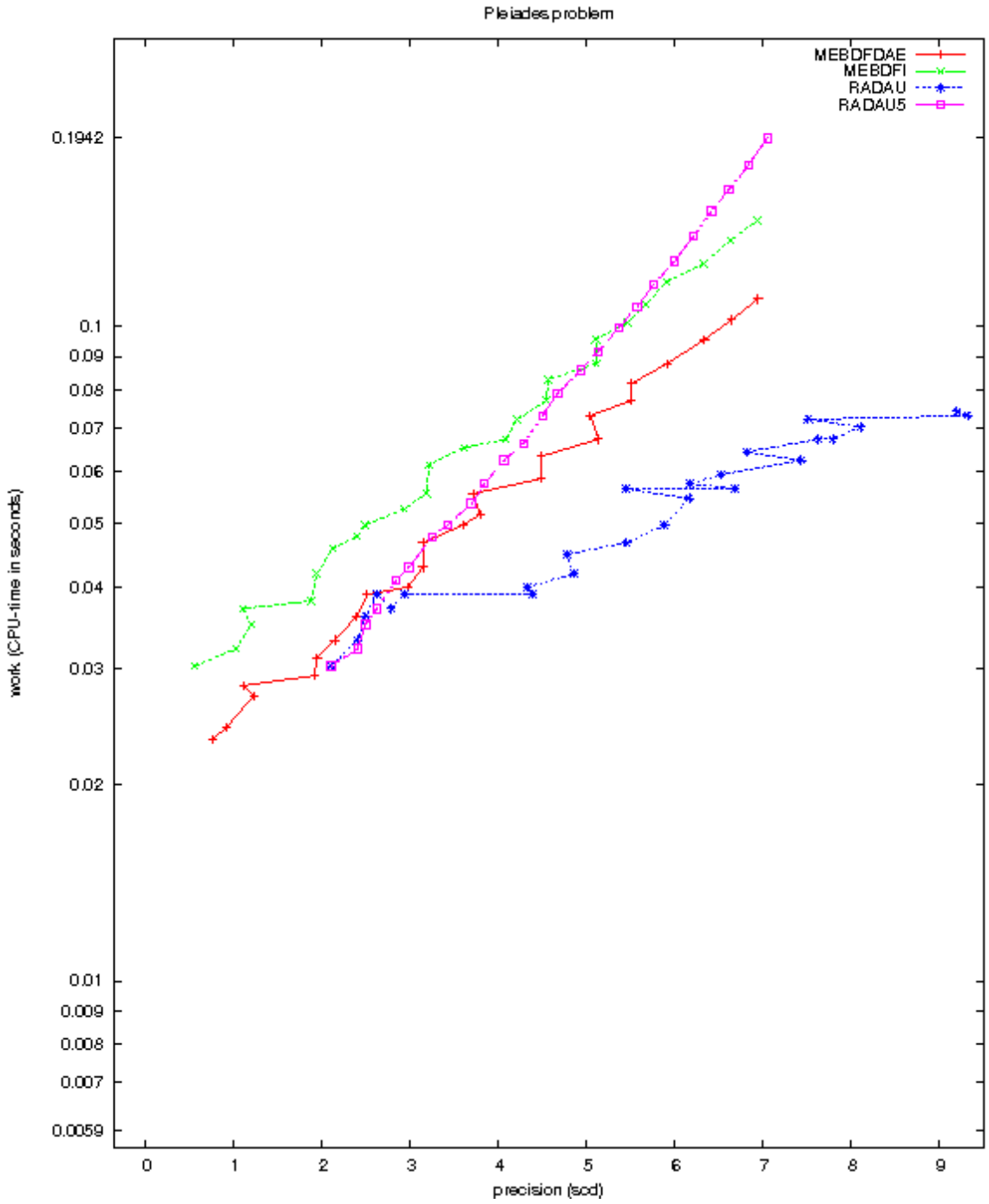

FIGURE II.6.4: *Work-precision diagram (scd versus CPU-time).*

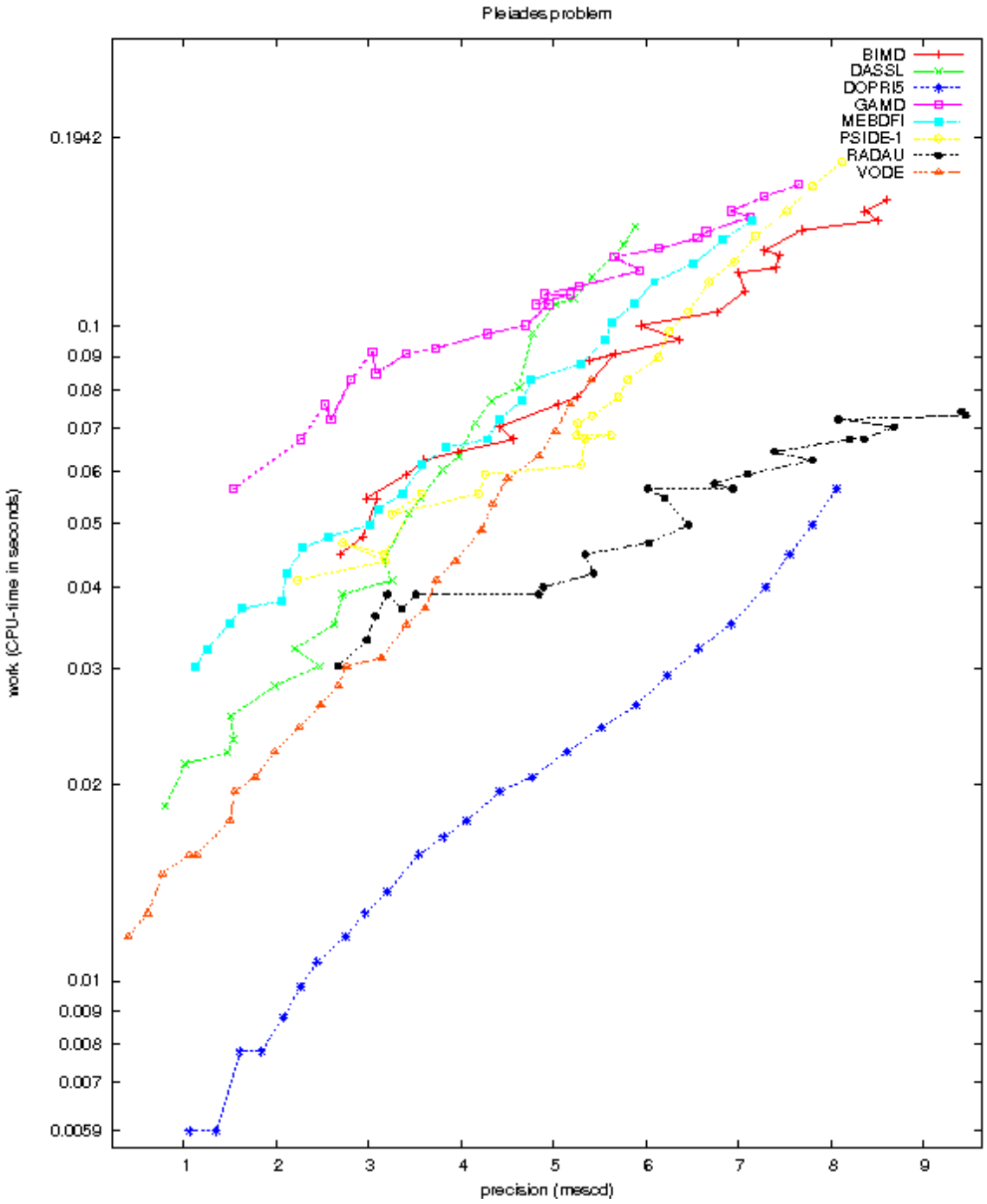

FIGURE II.6.5: *Work-precision diagram (mescd versus CPU-time).*

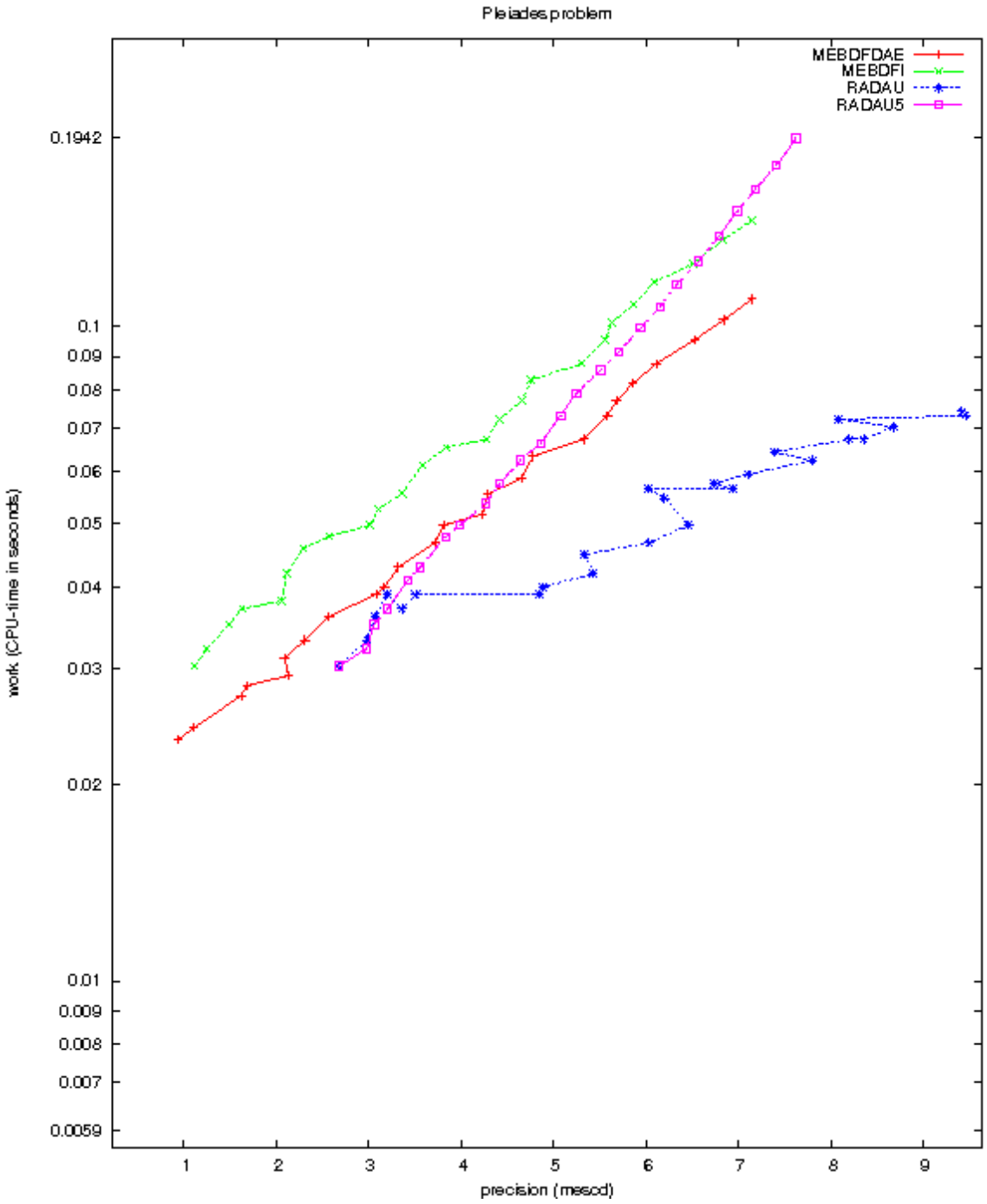

FIGURE II.6.6: *Work-precision diagram (mescd versus CPU-time).*

7 Problem BEAM

7.1 General information

The problem is originally described by a partial differential equation subject to boundary conditions. The semi-discretization in space of this equation leads to a stiff system of n non-linear second order differential equations which is rewritten to first order form, thus providing a stiff system of ordinary differential equations of dimension $2n$. The formulation and data have been taken from [HW96].The INdAM-Bari Test Set group contributed this problem to the test set.The software part of the problem is in the files beam.f available at [MM08].

7.2 Mathematical description of the problem

The problem is of the form

$$z'' \quad = \quad f(t,z,z'), \quad z(0)=z_0 \quad z'(0)=z'_0,$$

with

$$z \in R^n, \quad t \geq 0.$$

The function $f : R^n \to R^n$ is defined by

$$f(t,z,z') = Cv + Du.$$

Here C is the tridiagonal $n \times n$ matrix whose entries are given by

$$\begin{cases} (C)_{11}=1,\ (C)_{nn}=3, \text{ and } (C)_{ll}=2, & l=2,\ldots,n-1, \\ (C)_{l,l+1} = -\cos(z_l - z_{l+1}), & l=1,\ldots,n-1, \\ (C)_{l,l-1} = -\cos(z_l - z_{l-1}), & l=2,\ldots,n, \end{cases}$$

and D is the $n \times n$ bidiagonal matrix whose lower and upper diagonal entries are

$$\begin{cases} (D)_{l,l+1} = -\sin(z_l - z_{l+1}), & l=1,\ldots,n-1, \\ (D)_{l,l-1} = -\sin(z_l - z_{l-1}), & l=2,\ldots,n, \end{cases}$$

$v = (v_1, v_2, \ldots, v_n)^{\mathrm{T}}$ is defined by

$$v_l = n^4(z_{l-1} - 2z_l + z_{l+1}) + n^2(\cos(z_l)F_y - \sin(z_l)F_x), \quad l=1,\ldots,n$$

with $z_0 = -z_1$, $z_{n+1} = z_n$, and u is the column vector of size n solution of the tridiagonal system

$$Cu = g$$

with $g = Dv + (z'^2_1, z'^2_2, \ldots, z'^2_n)^{\mathrm{T}}$.

We write this problem to first order form by defining $w = z'$, yelding a system of $2n$ non-linear differential equations of the form

$$\begin{pmatrix} z \\ w \end{pmatrix}' = \begin{pmatrix} w \\ f(t,z,w) \end{pmatrix}$$

with

$$(z,w)^{\mathrm{T}} \in R^{2n}, \quad t \geq 0.$$

The initial values are

$$\begin{pmatrix} z_0 \\ w_0 \end{pmatrix} = \begin{pmatrix} z_0 \\ z'_0 \end{pmatrix}, \quad \text{where} \quad \begin{cases} z_0 = (0,0\ldots,0)^T \\ z'_0 = (0,0,\ldots,0)^T \end{cases}$$

7.3 Origin of the problem

The BEAM problem originates from mechanics and describes the motion of an elastic beam which is supposed inextensible, of length 1 and thin. Moreover, it is assumed that the beam is clamped at one end and a force $F = (F_u, F_v)$ acts at the free end. As coordinate system it is used the angle θ as a function of arc length s and time t. The beam is then described by the equations

$$u(s,t) = \int_0^s \cos\theta(\sigma,t)\, d\sigma, \quad v(s,t) = \int_0^s \sin\theta(\sigma,t)\, d\sigma.$$

In order to obtain the equations of motion for this problem, the Lagrange theory is applied. Let T be the kinetic and U the potential energy defined respectively as follows

$$\begin{aligned} T &= \frac{1}{2}\int_0^1 ((\dot{u}(s,t))^2 + (\dot{v}(s,t))^2)\, ds \\ U &= \frac{1}{2}\int_0^1 ((\theta^{'}(s,t))^2\, ds - F_u(t)u(1,t) - F_v(t)v(1,t). \end{aligned}$$

Here dots and primes denote derivatives with respect to t and s, respectively.
Using the Hamilton principle, the equations of motion are derived. They are

$$\begin{aligned} & \int_0^1 G(s,\sigma)\cos(\theta(s,t)-\theta(\sigma,t))\ddot{\theta}(\sigma,t)\, d\sigma = \\ = \;& \theta^{''}(s,t) + \cos\theta(s,t)F_v(t) - \sin\theta(s,t)F_u(t) \\ - \;& \int_0^1 G(s,\sigma)\sin(\theta(s,t)-\theta(\sigma,t))(\dot{\theta}(\sigma,t))^2\, d\sigma \\ & \theta(0,t) = 0, \qquad \theta^{'}(1,t) = 0 \end{aligned} \tag{II.7.1}$$

where

$$G(s,\sigma) = 1 - \max(s,\sigma)$$

is the Green function for the problem $-w^{''}(s) = g(s), w^{'}(0) = w(1) = 0$.
We discretize the integrals with the midpoint rule:

$$\int_0^1 f(\theta(\sigma,t))\, d\sigma = \frac{1}{n}\sum_{k=1}^{n} f(\theta_k), \qquad \theta_k = \theta\Big((k-\frac{1}{2})\frac{1}{n}, t\Big), \quad k = 1,\ldots,n.$$

Equations (II.7.1) then become

$$\begin{aligned} & \sum_{k=1}^{n} a_{\ell k}\ddot{\theta}_k = n^4(\theta_{\ell-1} - 2\theta_\ell + \theta_{\ell+1}) + n^2(\cos\theta_\ell\ F_v - \sin\theta_\ell\ F_u) \\ & -\sum_{k=1}^{n} g_{\ell k}\sin(\theta_\ell - \theta_k)\dot{\theta}_k^2, \qquad \ell = 1,\ldots,n, \\ & \theta_0 = -\theta_1, \quad \theta_{n+1} = \theta_n, \end{aligned}$$

where

$$a_{\ell k} = g_{\ell k}\cos(\theta_\ell - \theta_k), \qquad g_{\ell k} = n + \frac{1}{2} - \max(\ell, k).$$

TABLE II.7.1: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-4}	3.53	3.58	60	60	1249	58	59	0.2137
	10^{-7}	10^{-7}	10^{-7}	5.61	6.97	777	777	16197	722	744	2.7308
DDASSL	10^{-4}	10^{-4}		1.83	2.29	29120	28928	30700	243		3.1544
	10^{-7}	10^{-7}		4.63	5.25	51757	51160	56908	768		6.4455
GAMD	10^{-4}	10^{-4}	10^{-4}	3.58	3.59	49	49	1715	49	49	0.2030
	10^{-7}	10^{-7}	10^{-7}	5.49	6.28	459	458	21156	458	459	2.2321
MEBDFI	10^{-4}	10^{-4}	10^{-4}	2.56	1.92	578	559	6447	55	55	0.2284
	10^{-7}	10^{-7}	10^{-7}	5.20	5.26	38693	38645	292234	2054	2054	12.7690
PSIDE-1	10^{-4}	10^{-4}		2.52	2.14	42	36	1096	29	168	0.2303
	10^{-7}	10^{-7}		4.28	5.44	241	208	8006	192	964	1.4806
RADAU	10^{-4}	10^{-4}	10^{-4}	3.57	2.49	62	55	406	43	61	0.2645
	10^{-7}	10^{-7}	10^{-7}	4.24	5.72	71	71	1653	46	60	0.5632
VODE	10^{-4}	10^{-4}		−0.25	1.09	60537	60519	145514	1009	3041	7.3727
	10^{-7}	10^{-7}		4.40	6.48	58132	57793	139394	967	3338	7.8080

In Hairer & Wanner [HW96] the exterior forces are chosen as

$$F_u = -\varphi(t), \quad F_v = \varphi(t), \qquad \varphi(t) = \begin{cases} 1.5\ \sin^2 t, & 0 \le t \le \pi, \\ 0, & \pi \le t, \end{cases}$$

and the initial conditions are taken to be

$$\theta(s,0) = 0, \qquad \dot\theta(s,0) = 0.$$

7.4 Numerical solution of the problem

The resulting system of ODEs is integrated for $0 \le t \le 5$, using $n = 40$. Table II.7.1 and Figures II.7.1-II.7.3 present the run characteristics, the behavior of the solution components z_{10}, z_{20}, z_{30} and z_{40} over the interval and the work-precision diagrams, respectively. The computation of the scd values is based on the first 40 components, since they refer to the physically important quantities. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` $= 1.01\cdot 10^{-19}$, `rtol = atol = h0` $= 1.1\cdot 10^{-18}$. For the work-precision diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, \ldots, 16$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

With respect to the RADAU and RADAU5 results in Table II.7.1 and Figures II.7.2–II.7.5, we remark that for generality of the test set drivers, we did not use the facility to exploit the special structure of problems. By setting the input parameter `IWORK(9)=40`, and adjusting the Jacobian routine appropriately, RADAU and RADAU5 produce considerably better results.

These results are listed for RADAU in Table II.7.2.

References

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

TABLE II.7.2: *Run characteristics obtained by RADAU with exploited special structure.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
RADAU	10^{-4}	10^{-4}	10^{-4}	3.42	2.49	62	55	406	43	61	0.0869
	10^{-7}	10^{-7}	10^{-7}	4.24	5.72	71	71	1653	46	60	0.1728

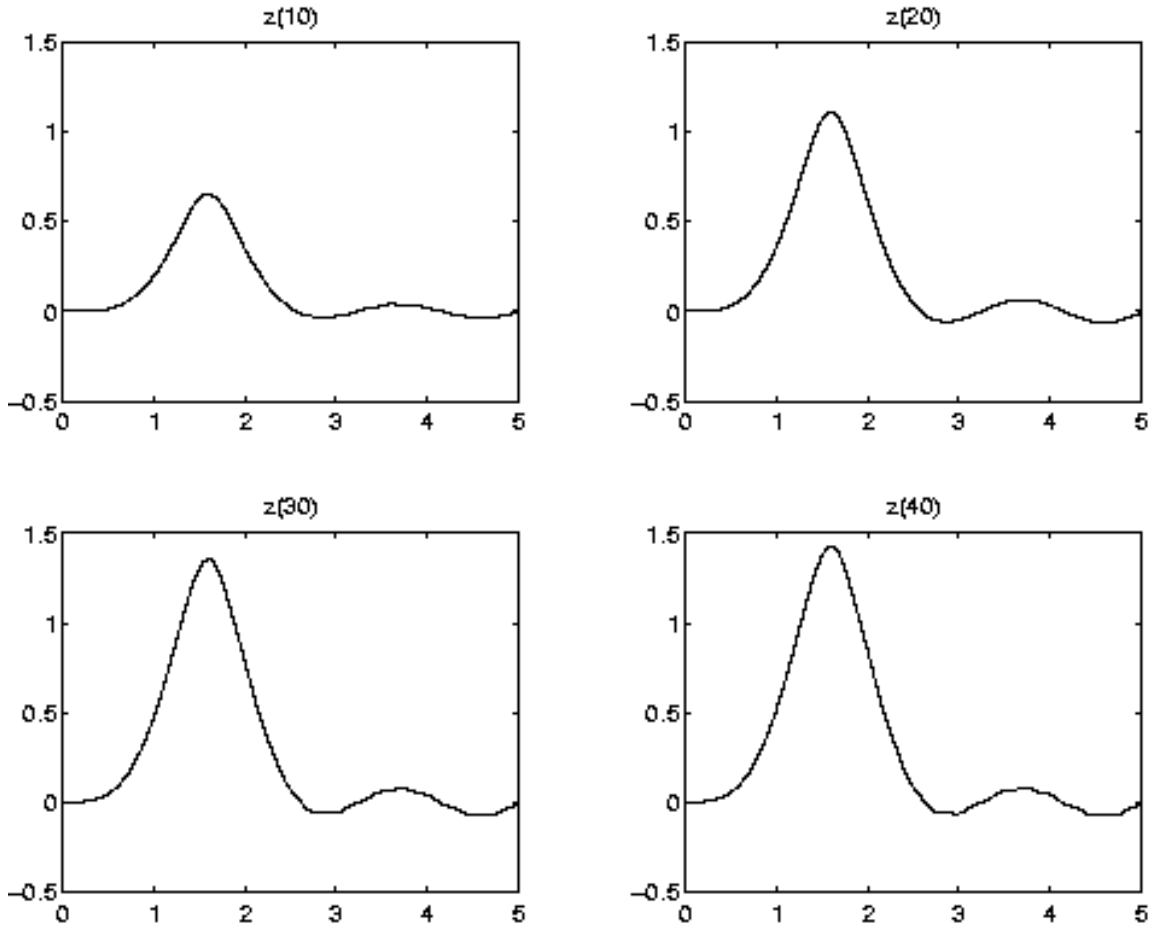

FIGURE II.7.1: Behavior of the solution components z_{10}, z_{20}, z_{30} and z_{40} over the integration interval

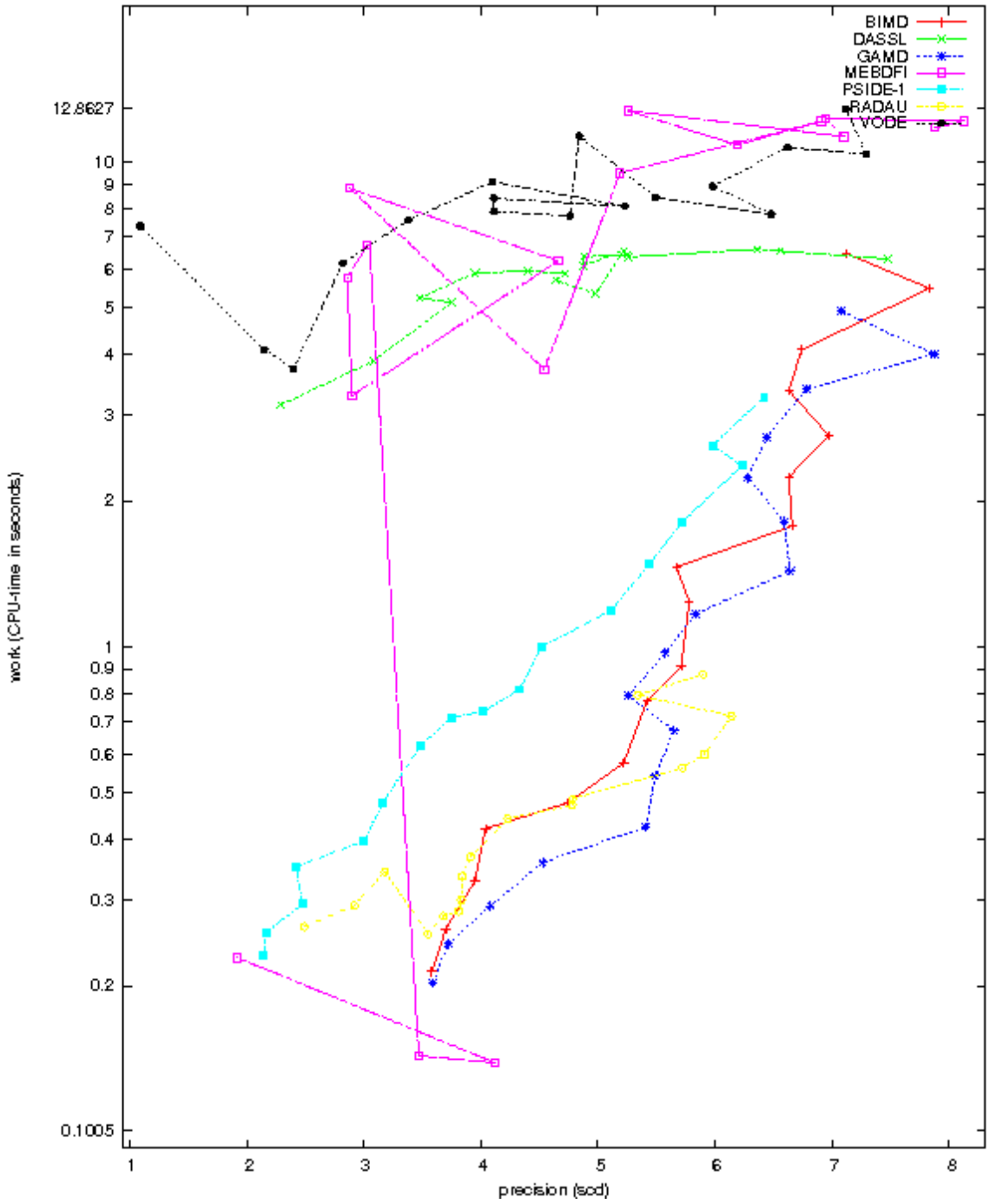

FIGURE II.7.2: *Work-precision diagram (scd versus CPU-time).*

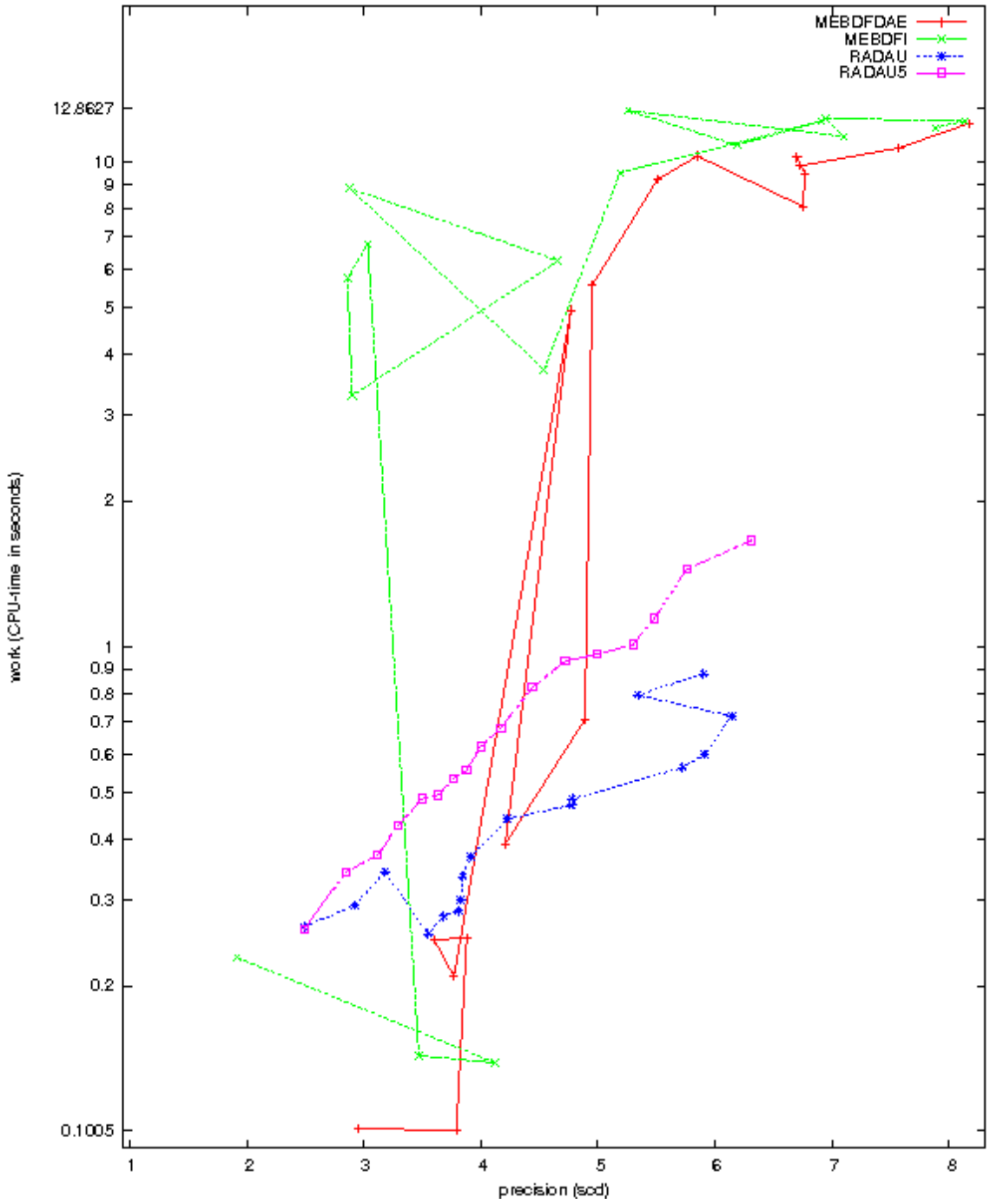

FIGURE II.7.3: *Work-precision diagram (scd versus CPU-time).*

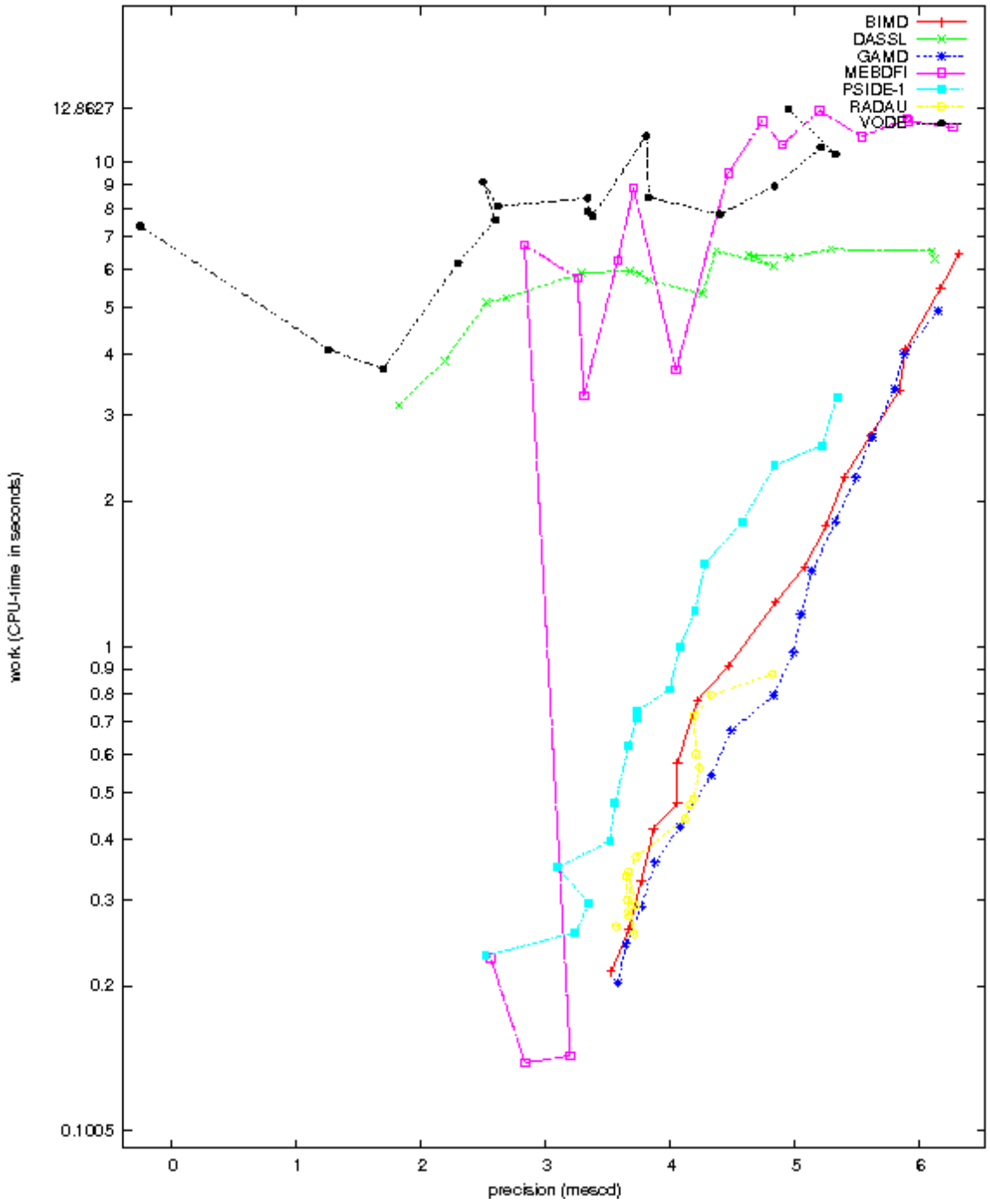

FIGURE II.7.4: *Work-precision diagram(mescd versus CPU-time) .*

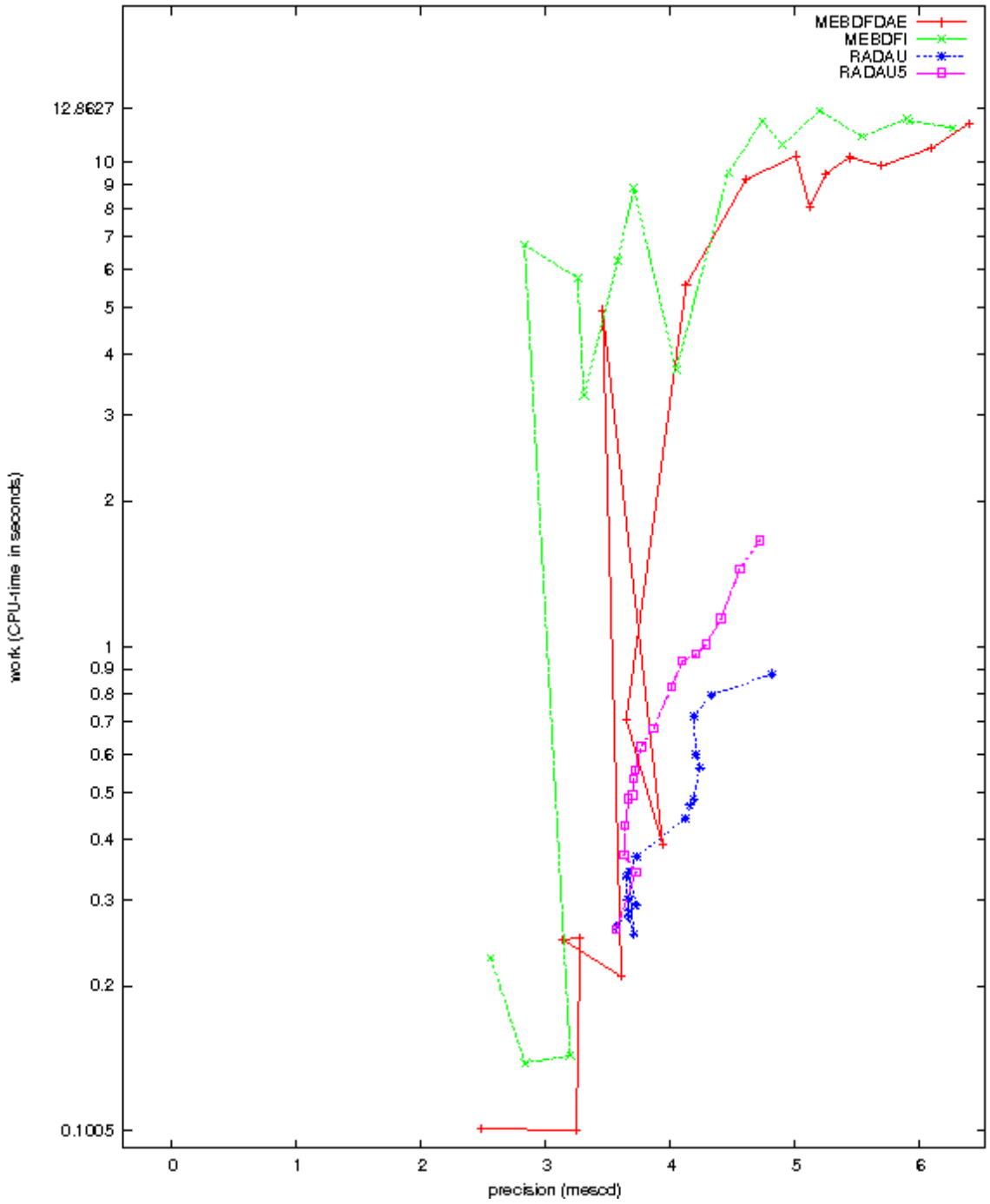

FIGURE II.7.5: *Work-precision diagram (mescd versus CPU-time) .*

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

8 Problem VDPOL

8.1 General information

The problem consists of a second order differential equation rewritten to first order form, thus providing a system of ordinary differential equations of dimension 2. It was proposed by B. van der Pol in the 1920's [vdP20], [vdP26]. The INdAM-Bari Test Set group contributed this problem to the test set. Most of the documentation about this problem has been retrieved from [EP02]. The software part of the problem is in the files vdpol.f and vdpolm.f available at [MM08].

8.2 Mathematical description of the problem

The problem is of the form

$$z'' \quad = \quad f(z, z'), \quad z(0) = z_0 \quad z'(0) = z'_0,$$

with

$$z \in \mathbb{R}, \quad t \in [0, T],$$

where the function f is given by

$$f(z, z') = \mu(1 - z^2)z' - z, \quad \mu > 0. \tag{II.8.1}$$

We write this problem to first order form by defining $y_1 = z$ and $y_2 = z'$, yelding a system of 2 nonlinear differential equations of the form

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}' = \begin{pmatrix} y_2 \\ f(y_1, y_2) \end{pmatrix} \tag{II.8.2}$$

with

$$(y_1, y_2)^T \in \mathbb{R}^2, \quad t \in [0, T].$$

A rescaling of the solutions of (II.8.2) results in the following formulation

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}' = \begin{pmatrix} y_2 \\ \widetilde{f}(y_1, y_2) \end{pmatrix}, \tag{II.8.3}$$

where

$$\widetilde{f}(y_1, y_2) = ((1 - y_1^2)y_2 - y_1)/\epsilon, \quad \epsilon > 0.$$

Problem (II.8.2) will be referred to as vdpol_μ and problem (II.8.3) as vdpol_ϵ. The initial values are

$$\begin{pmatrix} y_1(0) \\ y_2(0) \end{pmatrix} = \begin{pmatrix} z_0 \\ z'_0 \end{pmatrix} \quad \text{where} \quad \begin{cases} z_0 = 2 \\ z'_0 = 0 \end{cases}.$$

8.3 Origin of the problem

The VDPOL problem originates from electronics and describes the behaviour of nonlinear vacuum tube circuits. The circuit scheme, designed by Balthazar van der Pol in the 1920's, is given in Figure II.8.1. This is an RLC loop, but with the passive resistor of Ohm's Law replaced by an active element which would pump energy into the circuit whenever the amplitude of the current falls below a certain level. In the 1920's this active element was an array of vacuum tubes, now it is a semiconductor device. The voltage drop at the semiconductor (which used to be RI) is given by a nonlinear function $f(I)$ of

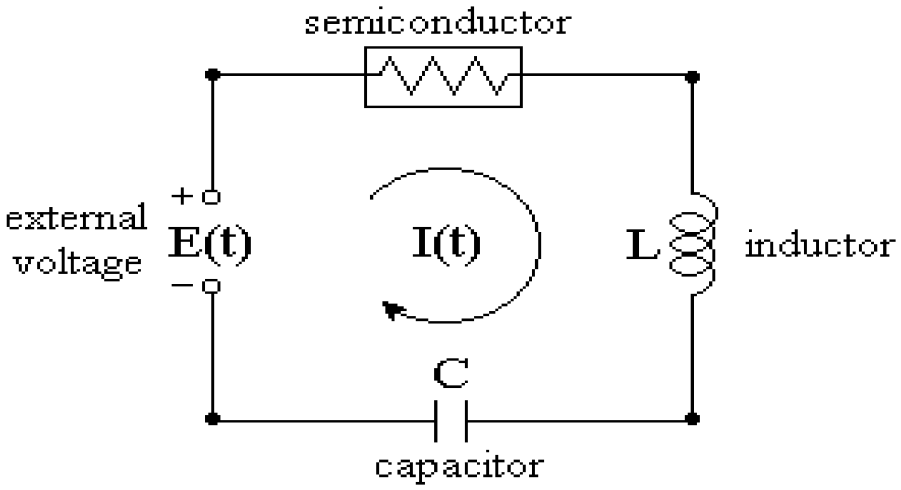

FIGURE II.8.1: *Negative resistance oscillatory circuit*

the current I. If we substitute $f(I)$ for RI in the standard RLC-circuit equation $LI'' + RI' + I/C = 0$, the current in the circuit turns out to be modeled by

$$LI'' + f'(I)I' + I/C = 0. \tag{II.8.4}$$

In a 1924 study of oscillator circuits in early commercial radios (at Philips research laboratories), B. van der Pol assumed the voltage drop to be represented by the nonlinear function $f(I) = bI^3 - aI$, which with equation (II.8.4) leads to

$$LI'' + (3bI^2 - a)I' + I/C = 0. \tag{II.8.5}$$

This equation is also closely related to the equation introduced by the British mathematical physicist Lord Rayleigh (John William Strutt, 1842 - 1919) to model the oscillations of a clarinet reed. For more details see [EP02].

If we denote by τ the time variable in Eq. (II.8.5) and make the substitutions $I = pz$ and $t = \tau/\sqrt{LC}$, the result is

$$\frac{d^2z}{dt^2} + (3bp^2z^2 - a)\sqrt{\frac{C}{L}}\frac{dz}{dt} + z = 0.$$

With $p = \sqrt{a/(3b)}$ and $\mu = a\sqrt{C/L}$ this gives the standard form

$$z'' + \mu(z^2 - 1)z' + z = 0$$

of the *van der Pol's equation*.

The van der Pol equation is often used as a test problem for ODEs solvers. It has two periodic solutions, the constant solution, $z(t) \equiv 0$, that is unstable, and the nontrivial periodic solution (roughly corresponding to the initial conditions $z(0) = 2,\ z'(0) = 0$), that is named 'limit cycle' because all the other nontrivial solutions converge to this one as $t \to \infty$.

This qualitative behavior is well shown in the phase plane plot in Figure II.8.2 (for $\mu = 2$), where outward and inward spiral trajectories converge to the limit cycle (the closed curve).

The parameter $\mu > 0$ weights the importance of the nonlinear part of the equation. When μ is 'large' the approach to the limit cycle is quite rapid (see Figure II.8.3 for $\mu = 10^3$) and the van der Pol equation is more interesting because of the non negligible influence of the nonlinear term. From

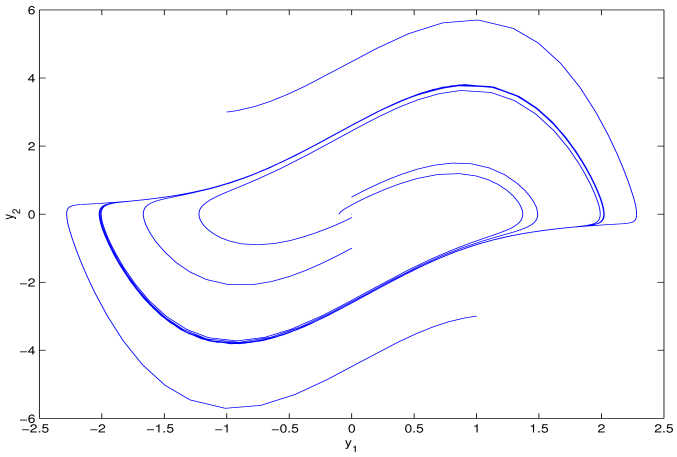

FIGURE II.8.2: *Limit cycle for* $\mu = 2$

an analysis of the behavior of the limit cycle [Sha94] it turns out that it can be described in terms of portions where the solution components change slowly and the problem is quite stiff, alternating with regions of very sharp change (quasi-discontinuities) where it is non-stiff. Thus, the problem switches from stiff to non stiff with a very sharp changing solution that makes the equation quite challenging for ODEs solvers.

The van der Pol equation may be treated in different ways, the most straightforward is to split the equation into a system of two first order differential equations as in (II.8.2). Note that if the second of the equations is divided by μ we get an equation that has the character of a singular perturbation problem. Several other approaches may show other aspects on the nature of this problem. For example Hairer and Wanner [HW96] introduce the following scaling transformation of (II.8.2) to make the steady-state approximation independent of μ:

$$x = t/\mu, \quad w_1(x) = y_1(t), \quad w_2(x) = \mu y_2(t)$$

Substituting in (II.8.2) and using again y for w and t for x, the equation (II.8.3) is obtained with $\varepsilon = 1/\mu^2$. The scaled version (II.8.3) has the advantage that a small interval independent of the parameter value can be considered to track at least one period of the solution.

8.4 Numerical solution of the problem

8.4.1 vdpol$_\mu$ with $\mu = 10^3$ and $t \in [0, 2\mu]$

Tables II.8.1, II.8.2 and Figures II.8.4, II.8.6–II.8.9 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the first component of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol = h0` $= 1.1 \cdot 10^{-18}$. For the work-precision diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = rtol; h0 $= 10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

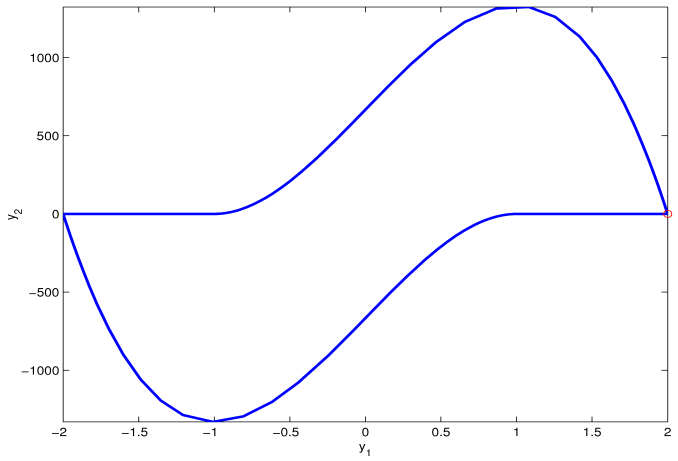

FIGURE II.8.3: *Limit cycle for* $\mu = 10^3$

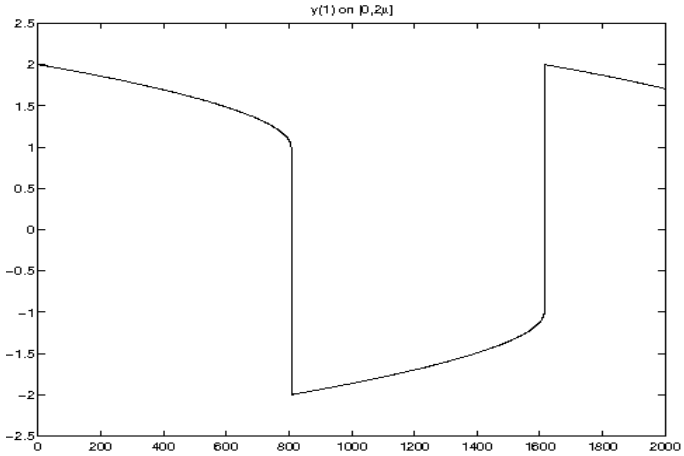

FIGURE II.8.4: *Behavior of the solution component* y_1 *over the integration interval*

8.4.2 vdpol$_\epsilon$ with $\epsilon = 10^{-6}$ and $t \in [0, 2]$

Tables II.8.3, II.8.4 and Figures II.8.5, II.8.10–II.8.13 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the first component of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol = h0` $= 1.1 \cdot 10^{-18}$. For the work-precision

TABLE II.8.1: *Reference solution at the end of the integration interval.*

y_1	$0.1706167732170469 \cdot 10^{1}$
y_2	$-0.8928097010248125 \cdot 10^{-3}$

TABLE II.8.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	4.05	3.57	133	112	2801	104	133	0.0020
	10^{-7}	10^{-7}	10^{-9}	9.18	8.80	224	219	5072	209	224	0.0029
	10^{-10}	10^{-10}	10^{-12}	11.17	10.32	250	248	10151	237	250	0.0078
DDASSL	10^{-4}	10^{-4}		2.88	2.37	549	507	940	122		0.0020
	10^{-7}	10^{-7}		5.57	5.06	1342	1296	1980	129		0.0049
	10^{-10}	10^{-10}		8.25	7.73	4484	4445	5943	168		0.0166
GAMD	10^{-4}	10^{-4}	10^{-6}	4.86	4.30	129	90	5133	91	129	0.0039
	10^{-7}	10^{-7}	10^{-9}	7.55	6.71	173	137	9422	141	173	0.0078
	10^{-10}	10^{-10}	10^{-12}	9.53	9.17	235	197	16067	201	235	0.0127
MEBDFI	10^{-4}	10^{-4}	10^{-6}	3.31	2.86	477	435	1761	83	83	0.0029
	10^{-7}	10^{-7}	10^{-9}	6.11	5.60	1134	1083	3818	118	118	0.0059
	10^{-10}	10^{-10}	10^{-12}	9.06	8.55	2135	2098	7215	208	208	0.0107
PSIDE-1	10^{-4}	10^{-4}		6.42	3.43	181	149	2811	57	648	0.0029
	10^{-7}	10^{-7}		7.20	6.32	310	293	6141	52	756	0.0059
	10^{-10}	10^{-10}		9.99	9.14	1000	990	15536	109	1156	0.0156
RADAU	10^{-4}	10^{-4}	10^{-6}	4.48	4.28	210	172	1822	144	208	0.0010
	10^{-7}	10^{-7}	10^{-9}	8.56	8.18	240	222	3508	187	238	0.0020
	10^{-10}	10^{-10}	10^{-12}	10.63	9.24	209	176	6240	130	207	0.0039
VODE	10^{-4}	10^{-4}		3.29	3.08	545	487	779	19	117	0.0020
	10^{-7}	10^{-7}		5.20	4.73	1614	1502	2145	30	223	0.0049
	10^{-10}	10^{-10}		7.49	7.07	4350	4120	5266	72	516	0.0146

diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = rtol; h0 $= 10^{-2} \cdot$ rtol for GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

TABLE II.8.3: *Reference solution at the end of the integration interval.*

y_1	$0.1706167732170483 \cdot 10^{1}$
y_2	$-0.8928097010247975 \cdot 10^{0}$

References

[EP02] C. H. Edwards and D. E. Penney. *Differential Equations and Linear Algebra.* Prentice Hall, 2002.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

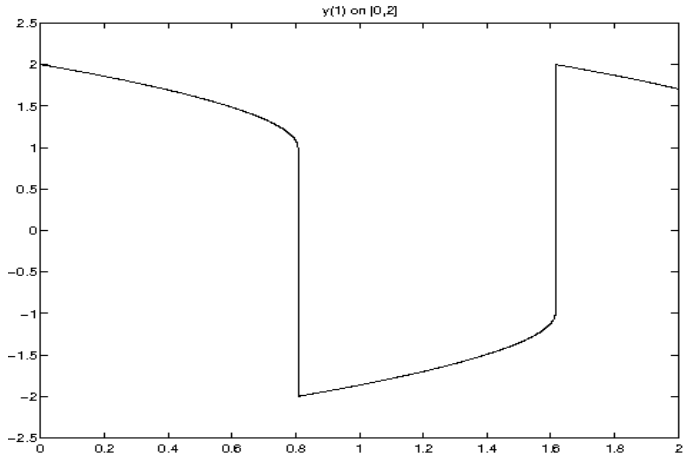

FIGURE II.8.5: *Behavior of the solution component y_1 over the integration interval (scaled equation)*

TABLE II.8.4: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	4.31	3.98	170	155	3684	151	170	0.0029
	10^{-7}	10^{-7}	10^{-9}	9.06	8.73	301	293	7631	280	301	0.0059
	10^{-10}	10^{-10}	10^{-12}	11.17	10.84	307	304	13339	292	307	0.0088
DDASSL	10^{-4}	10^{-4}		2.89	2.56	796	776	1260	127		0.0029
	10^{-7}	10^{-7}		5.89	5.57	1943	1912	2796	149		0.0078
	10^{-10}	10^{-10}		8.96	8.64	6166	6110	7973	223		0.0234
GAMD	10^{-4}	10^{-4}	10^{-6}	5.40	5.08	148	105	6999	105	148	0.0049
	10^{-7}	10^{-7}	10^{-9}	6.52	6.19	163	133	12727	131	163	0.0098
	10^{-10}	10^{-10}	10^{-12}	10.16	9.84	244	216	18095	215	244	0.0137
MEBDFI	10^{-4}	10^{-4}	10^{-6}	3.86	3.53	638	591	2179	92	92	0.0029
	10^{-7}	10^{-7}	10^{-9}	6.99	6.67	1369	1317	4735	132	132	0.0078
	10^{-10}	10^{-10}	10^{-12}	10.80	10.47	2862	2858	9489	287	287	0.0146
PSIDE-1	10^{-4}	10^{-4}		5.70	5.38	235	166	4402	73	780	0.0049
	10^{-7}	10^{-7}		8.72	8.39	414	386	7896	75	908	0.0078
	10^{-10}	10^{-10}		11.40	11.07	1388	1365	23066	131	1360	0.0224
RADAU	10^{-4}	10^{-4}	10^{-6}	4.77	4.44	242	207	2214	165	231	0.0020
	10^{-7}	10^{-7}	10^{-9}	8.28	7.95	186	149	5212	102	173	0.0029
	10^{-10}	10^{-10}	10^{-12}	11.47	11.14	245	215	7589	148	224	0.0049
VODE	10^{-4}	10^{-4}		2.93	2.61	788	702	1186	21	181	0.0029
	10^{-7}	10^{-7}		5.65	5.32	2375	2200	3091	41	345	0.0088
	10^{-10}	10^{-10}		8.42	8.09	6426	6058	7814	106	794	0.0215

[Sha94] Lawrence F. Shampine. *Numerical solution of ordinary differential equations.* Chapman & Hall, New York, 1994.

[vdP20] B. van der Pol. *Radio Rev.*, 1:704–754, 1920.

[vdP26] B. van der Pol. On relaxation oscillations. *Phil. Mag.*, 2:978–992, 1926. reproduced in: B.

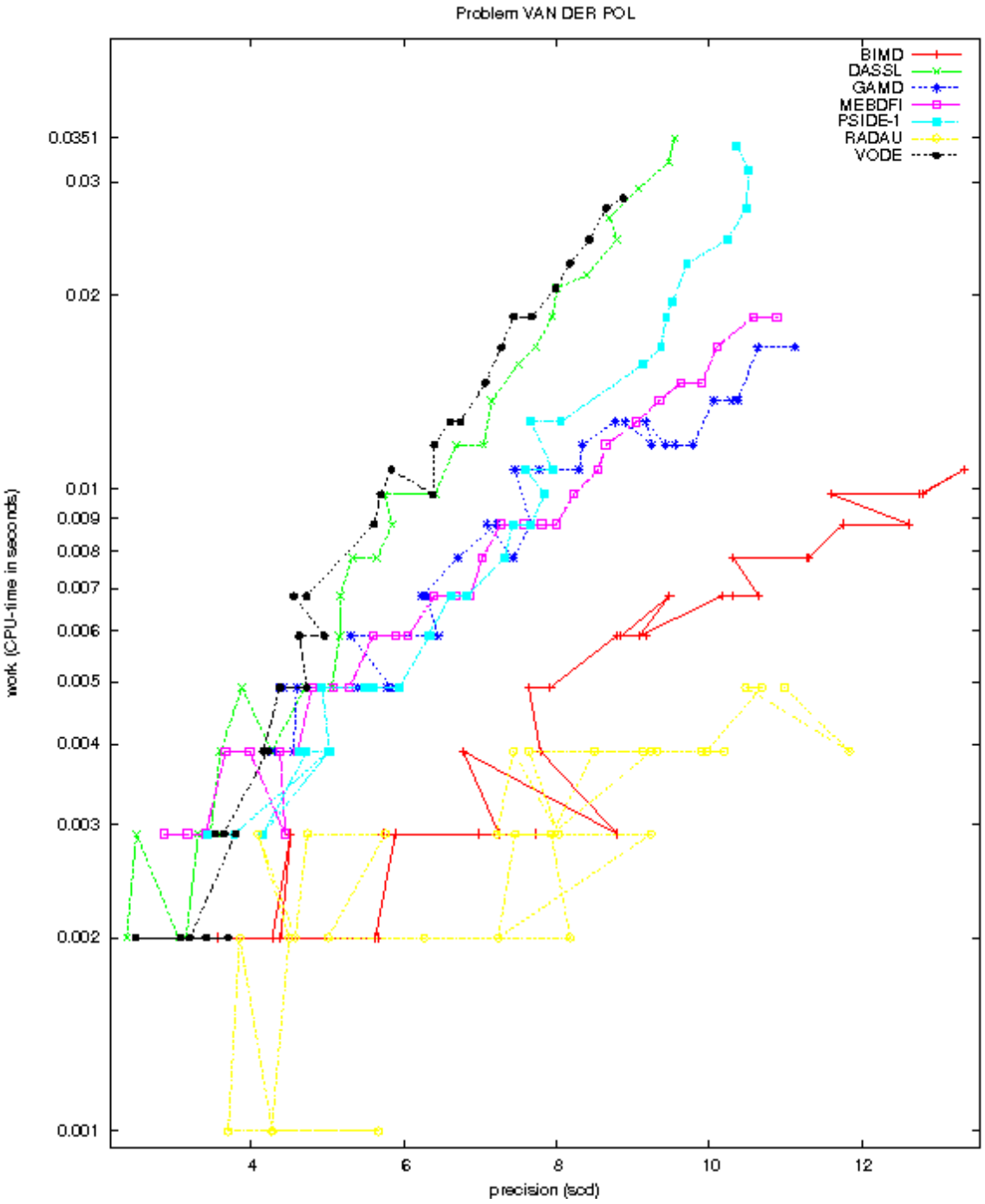

FIGURE II.8.6: *Work-precision diagram (scd versus CPU-time).*

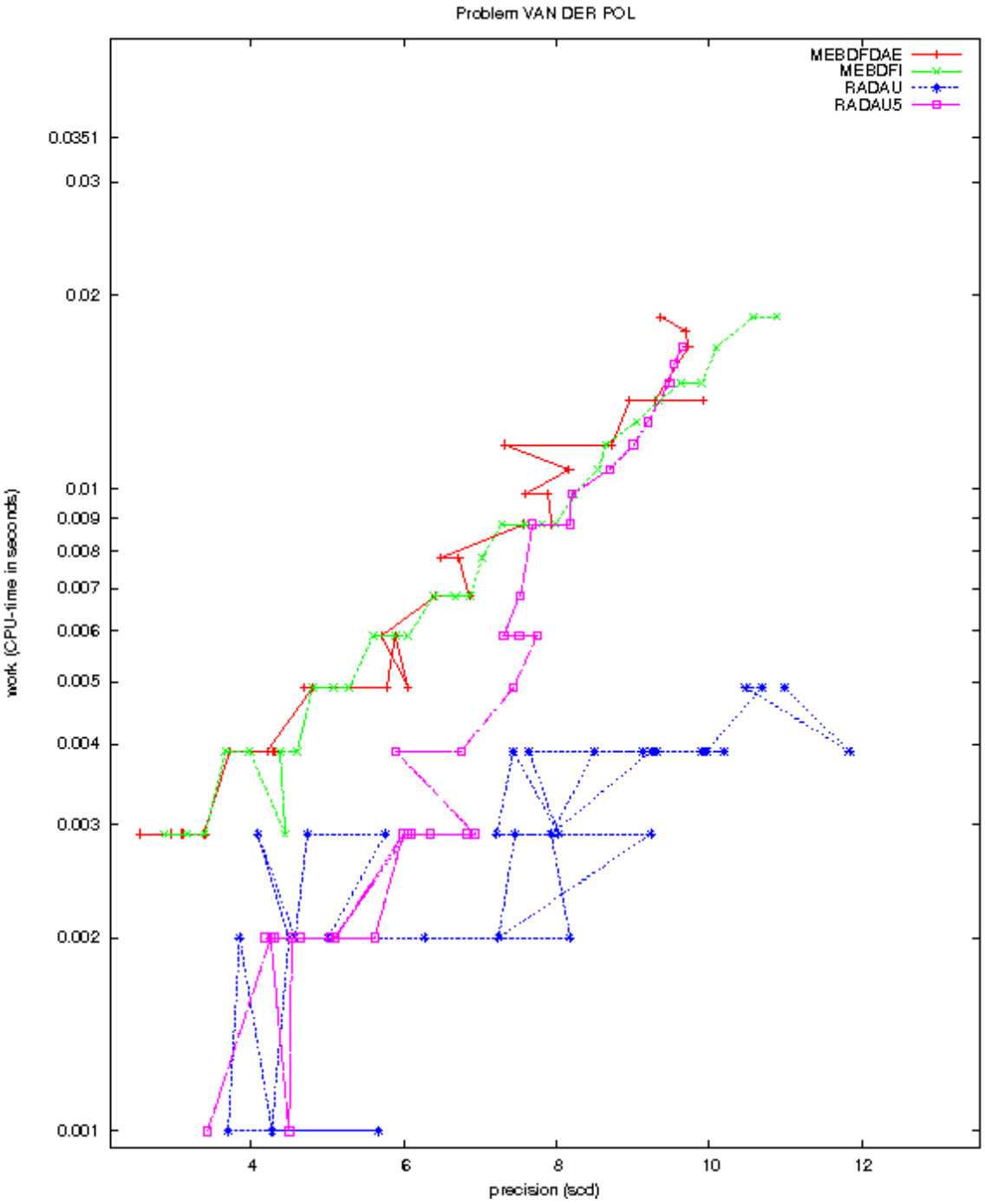

FIGURE II.8.7: *Work-precision diagram (scd versus CPU-time).*

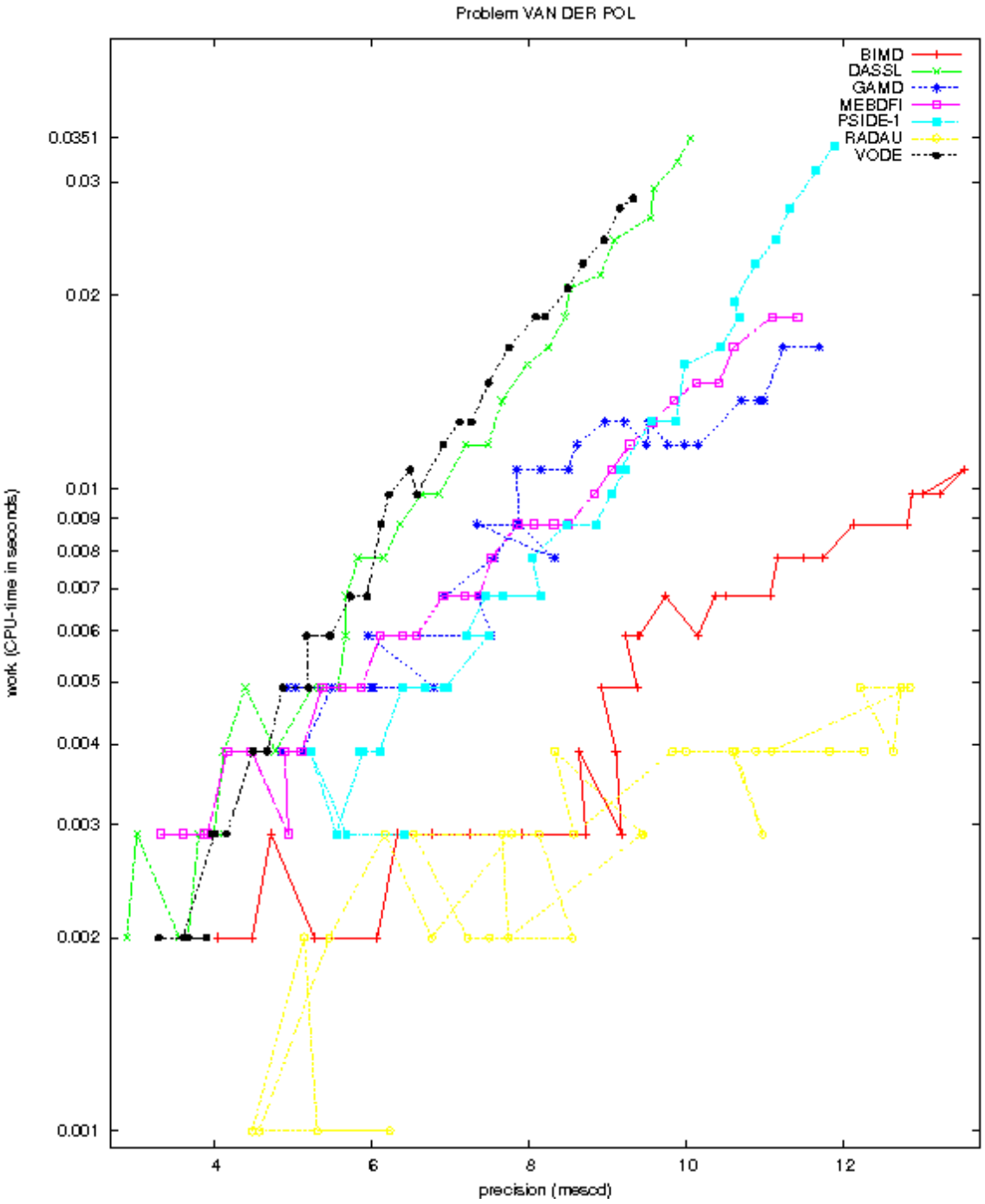

FIGURE II.8.8: *Work-precision diagram (mescd versus CPU-time).*

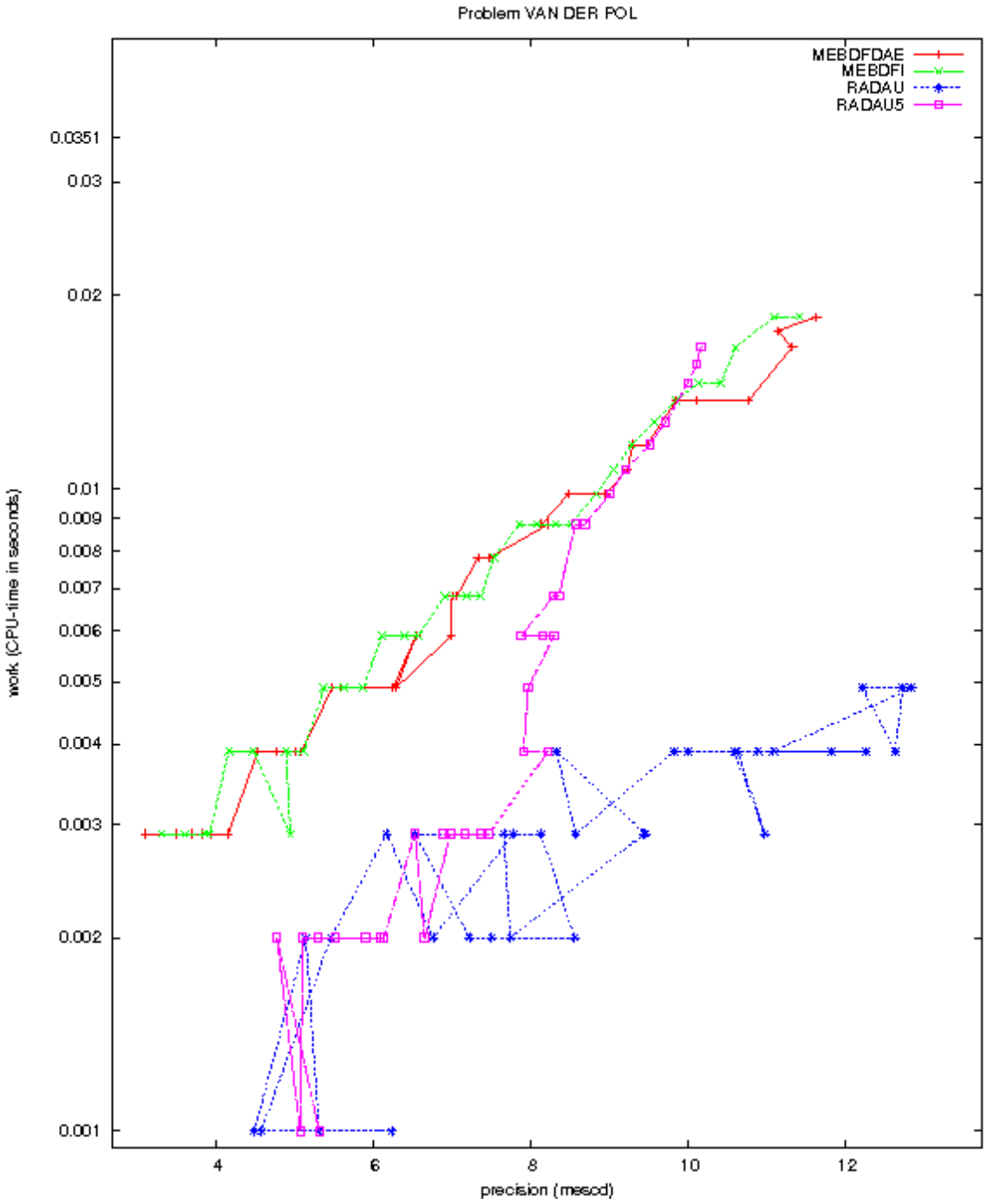

FIGURE II.8.9: *Work-precision diagram (mescd versus CPU-time).*

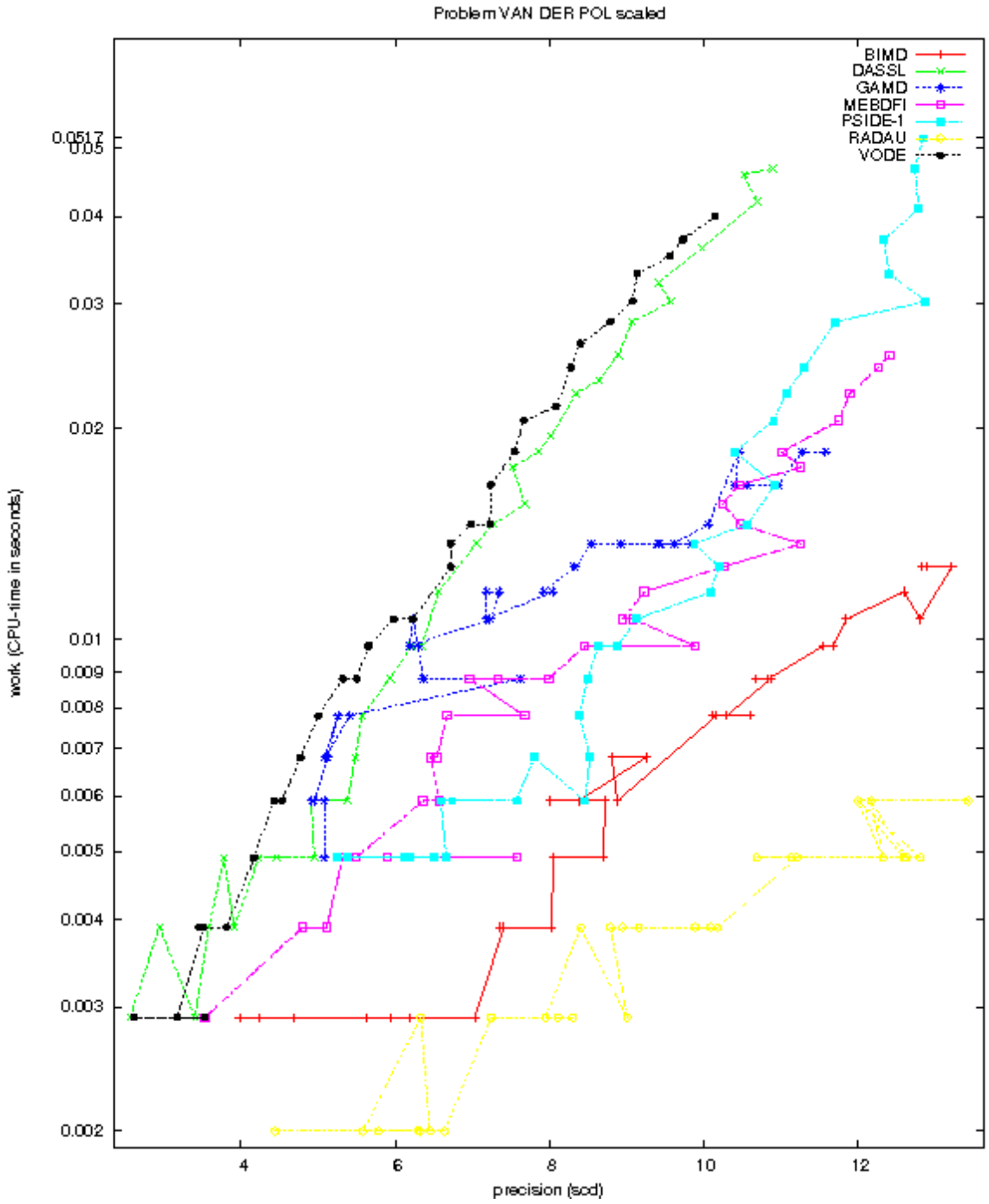

FIGURE II.8.10: *Work-precision diagram (scd versus CPU-time).*

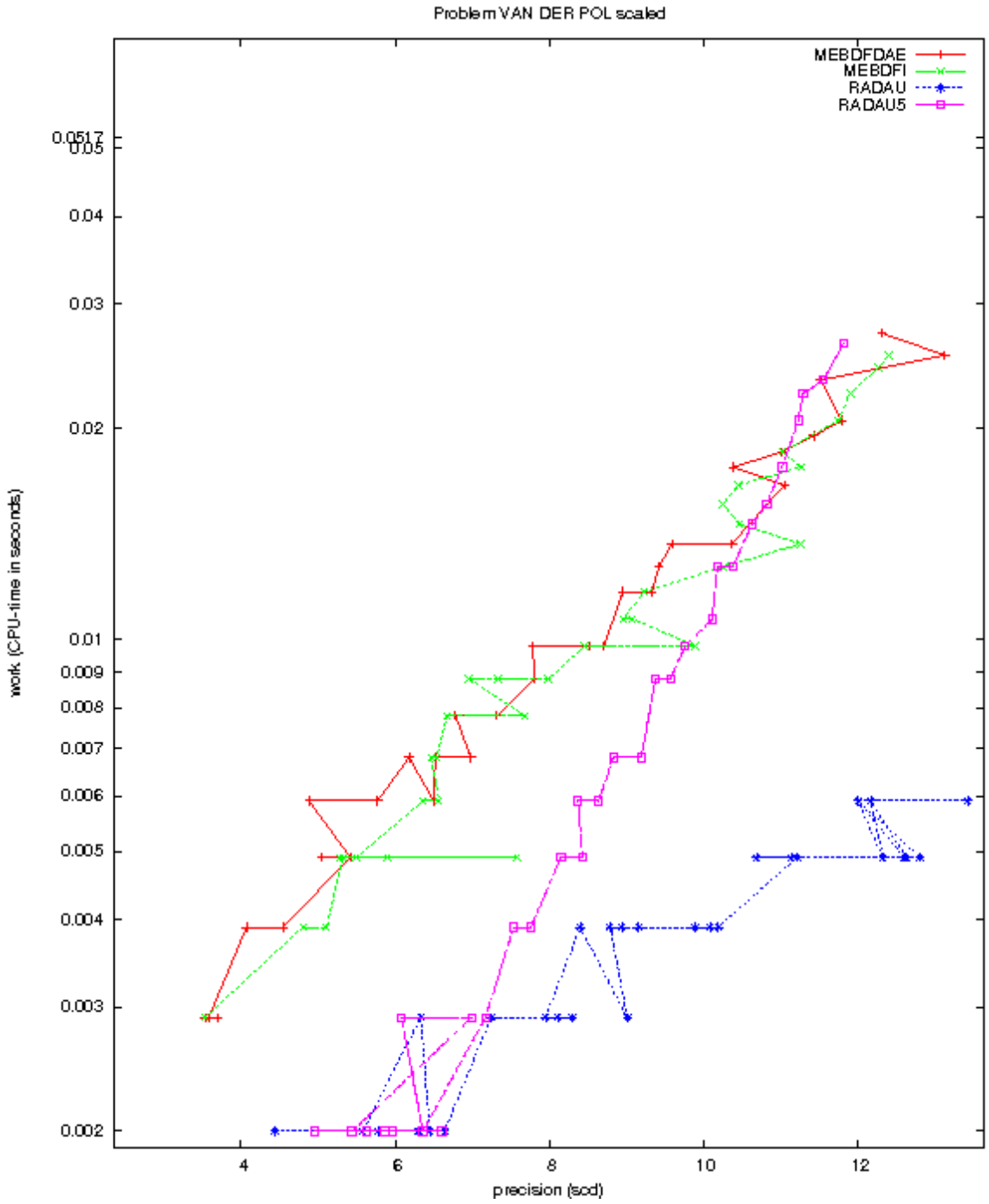

FIGURE II.8.11: *Work-precision diagram (scd versus CPU-time).*

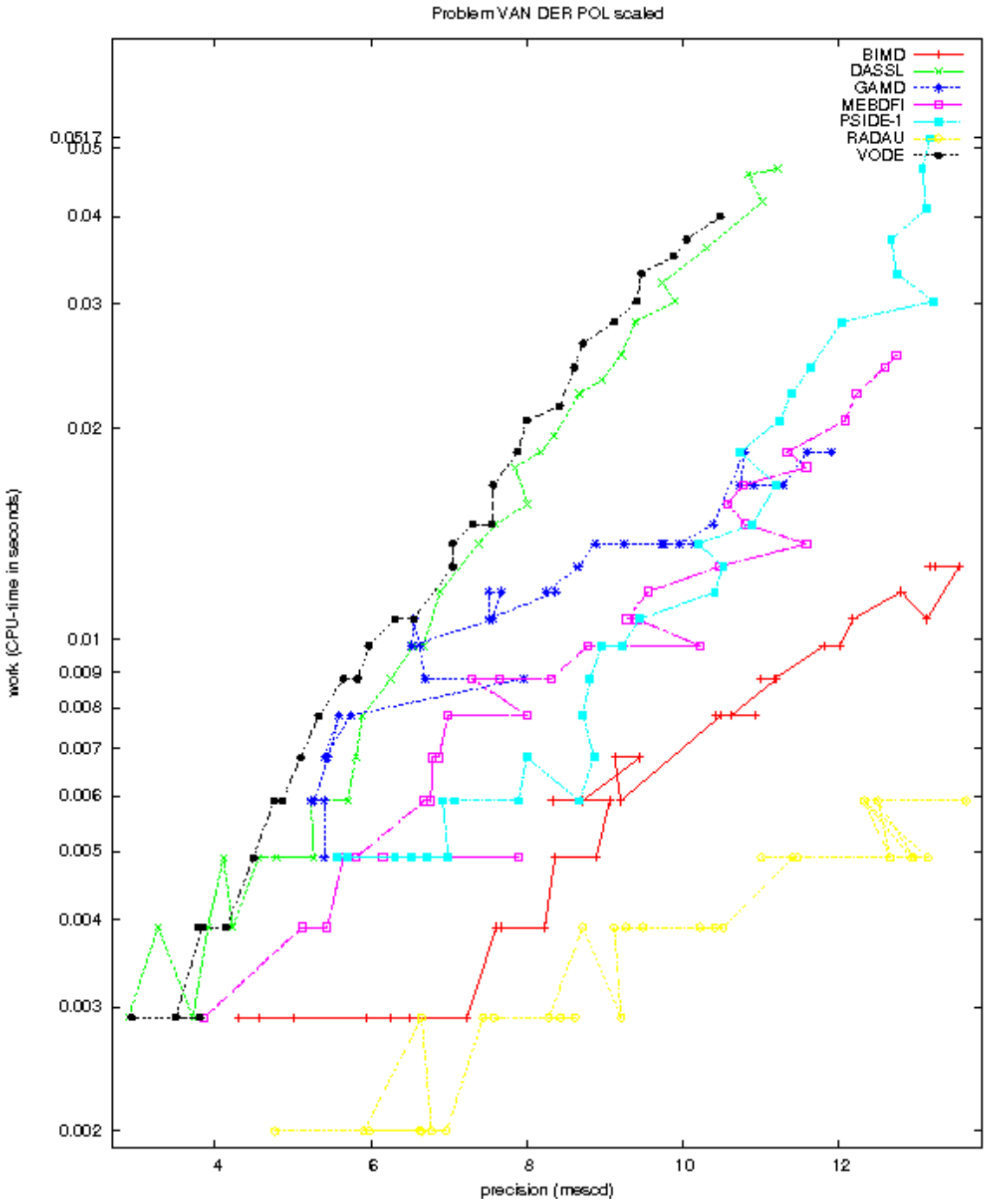

FIGURE II.8.12: *Work-precision diagram (mescd versus CPU-time).*

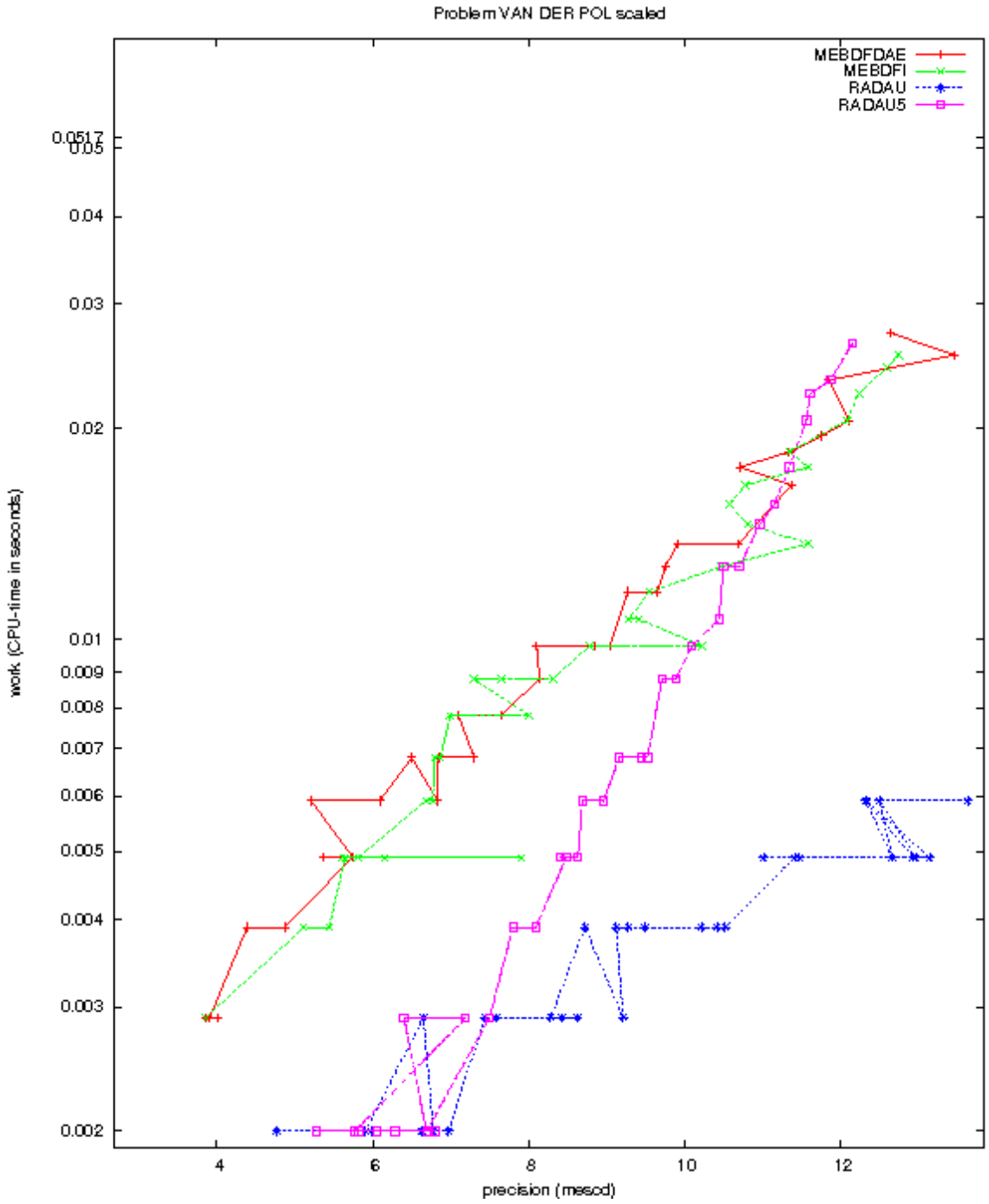

FIGURE II.8.13: *Work-precision diagram (mescd versus CPU-time).*

van der Pol, Selected Scientific Papers, vol. I, North Holland Publ. Comp. Amsterdam, 1960.

9 Problem OREGO

9.1 General information

The problem consists of a stiff system of 3 non-linear Ordinary Differential Equations. The name Orego was given by Hairer & Wanner [HW96] and refers to the Oregonator model which is described by this ODE. The Oregonator model takes its name from the University of Oregon where in the 1972 Field, Körös & Noyes [FKN72] proposed this model for the Belousov–Zhabotinskii reaction. The INdAM-Bari Test Set group contributed this problem to the test set. The software part of the problem is in the file orego.f available at [MM08].

9.2 Mathematical description of the problem

The problem is of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \quad y(0) = y_0,$$

with

$$y \in \mathcal{R}^3, \quad 0 \leq t \leq 360.$$

The function f is defined by

$$f(y) = \begin{pmatrix} s(y_2 - y_1 y_2 + y_1 - q y_1^2) \\ \frac{1}{s}(-y_2 - y_1 y_2 + y_3) \\ w(y_1 - y_3) \end{pmatrix}.$$

The values of the parameters s, q and w are

$$\begin{aligned} s &= 77.27 \\ w &= 0.161 \\ q &= 8.375 \cdot 10^{-6}. \end{aligned}$$

The initial vector y_0 is given by $(1, 2, 3)^{\mathrm{T}}$.

9.3 Origin of the problem

The OREGO problem originates from the celebrated Belousov–Zhabotinskii (BZ) reaction. When certain reactans, like bromous acid, bromide ion and cerium ion, are combined, they exhibit a chemical reaction which, after an induction period of inactivity, oscillates with change in structure and in color, from red to blue and viceversa.

The color changes are caused by alternating oxidation–reductions in which the cerium switches its oxidation state from Ce(III) to Ce(IV).

Field, Körös and Noyes formulated the following model for the most important parts of the kinetic mechanism that gives rice to oscillation in the BZ reaction. This mechanism can be summarized as three concurrent processes [Gra02]:

- the reduction of bromate (BrO_3^-) to bromine (Br) via the reducing agent bromide (Br^-). Bromomalonic acid (BrMA) is produced;
- the increase of hypobromous acid ($HBrO_2$) at an accelerating rate and the production of Ce(IV). Here we have a sudden change in color from red to blue;
- the reduction of Cerium catalyst Ce(IV) to Ce(III). Here we have a gradual change in color from blue to red.

TABLE II.9.1: Reference solution at the end of the integration interval.

t	X $= y_1$	Y $= y_2$	Z $= y_3$
360	$0.1000814870318523 \cdot 10^1$	$0.1228178521549917 \cdot 10^4$	$0.1320554942846706 \cdot 10^3$

TABLE II.9.2: *Failed runs.*

solver	m	reason
VODE	2,4	error test failed repeatedly

Then, from this mechanism the following Oregonator scheme is obtained

A+Y$\rightarrow$ X+P	r=k_3AY
X+Y$\rightarrow$ 2P	r=k_2XY
A+X$\rightarrow$ 2X+2Z	r=k_5AX
2X$\rightarrow$ A+P	r=k_4X^2
B+Z$\rightarrow$ $\frac{1}{2}f$Y	r=k_cBZ

Here using the conventional notation as in [FKN72] the assignments and the effective concentration are

hypobromous acid	$[HBrO_2] = X$	5.025×10^{-11}
bromide	$[Br^-] = Y$	3.0×10^{-7}
cerium $-$ 4	$[CE(IV)] = Z$	2.412×10^{-8}
bromate	$[BrO_3^-] = A$	
all oxidizable organic species	$[Org] = B$	
	$[HOBr] = P$	.

The reaction rate equations for the intermediate species X, Y, and Z are

$$\begin{aligned} \frac{dX}{dt} &= s\left(Y - XY + X - qX^2\right) \\ \frac{dY}{dt} &= \frac{1}{s}\left(-Y - XY + fZ\right) \\ \frac{dZ}{dt} &= w\left(X - Z\right). \end{aligned}$$

with $f = 1$, and s, w, and q as in the previous subsection.

9.4 Numerical solution of the problem

Tables II.9.1, II.9.3 and Figures **II.9.1–II.9.7** present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol = h0` $= 1.1 \cdot 10^{-18}$, `atol = h0` $= 1.1 \cdot 10^{-40}$. For the work-precision diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = rtol; `h0` $= 10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.9.2; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

TABLE II.9.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	3.85	3.85	235	224	4393	215	235	0.0049
	10^{-7}	10^{-7}	10^{-9}	7.87	7.86	347	339	9629	334	347	0.0107
	10^{-10}	10^{-10}	10^{-12}	11.29	11.29	373	367	16863	359	373	0.0176
DDASSL	10^{-4}	10^{-4}		2.62	2.62	889	813	1505	124		0.0039
	10^{-7}	10^{-7}		5.58	5.57	2725	2671	4210	189		0.0137
	10^{-10}	10^{-10}		8.66	8.66	8192	8098	11119	274		0.0381
GAMD	10^{-4}	10^{-4}	10^{-6}	3.61	3.61	219	162	8510	163	219	0.0088
	10^{-7}	10^{-7}	10^{-9}	6.90	6.89	251	205	16050	208	251	0.0176
	10^{-10}	10^{-10}	10^{-12}	9.50	9.50	291	268	22034	270	291	0.0234
MEBDFI	10^{-4}	10^{-4}	10^{-6}	3.34	3.33	733	687	2707	103	103	0.0049
	10^{-7}	10^{-7}	10^{-9}	6.39	6.39	1586	1529	5399	174	174	0.0107
	10^{-10}	10^{-10}	10^{-12}	9.59	9.59	3248	3232	10754	345	345	0.0205
PSIDE-1	10^{-4}	10^{-4}		4.74	4.73	221	178	4696	128	836	0.0059
	10^{-7}	10^{-7}		7.06	7.06	441	407	9235	148	1164	0.0117
	10^{-10}	10^{-10}		10.77	10.47	1450	1412	26255	219	1788	0.0332
RADAU	10^{-4}	10^{-4}	10^{-6}	3.42	3.12	268	222	3416	200	267	0.0029
	10^{-7}	10^{-7}	10^{-9}	7.48	7.48	267	216	6859	192	265	0.0059
	10^{-10}	10^{-10}	10^{-12}	9.83	9.82	261	202	12917	176	257	0.0098
VODE	10^{-4}	10^{-4}		2.15	2.15	1196	1101	1820	38	236	0.0049
	10^{-7}	10^{-7}		4.73	4.73	3083	2858	4348	64	454	0.0117
	10^{-10}	10^{-10}		7.51	7.51	7890	7430	9903	133	970	0.0293

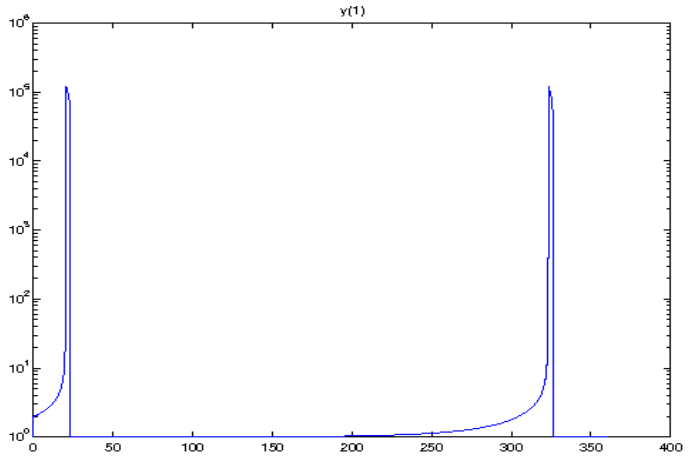

FIGURE II.9.1: Behavior of the solution component y_1 over the integration interval

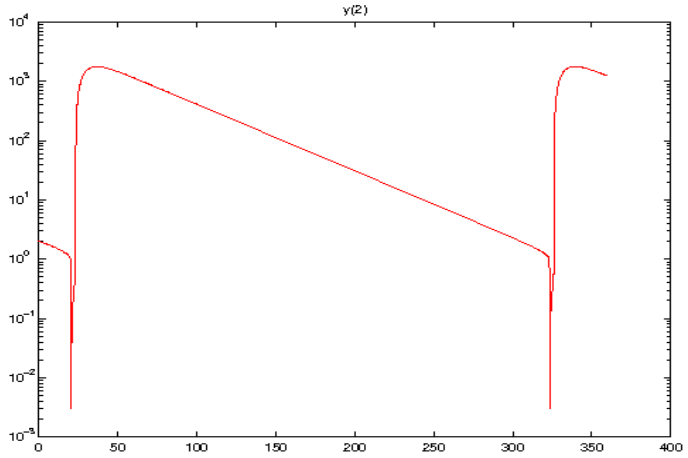

FIGURE II.9.2: Behavior of the solution component y_2 over the integration interval

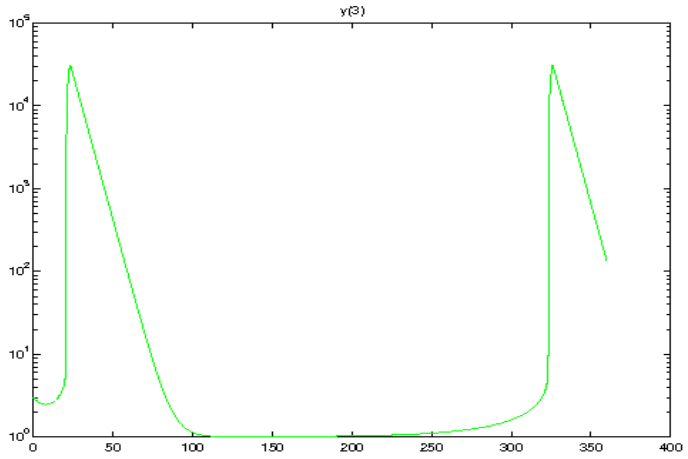

FIGURE II.9.3: Behavior of the solution component y_3 over the integration interval

References

[FKN72] R. J. Field, E. Körös, and R.M Noyes. Oscillation in chemical systems, part. 2. thorough analysis of temporal oscillations in the bromate–cerium–malonic acid system. *Journal of the American Society*, 94:8649–8664, 1972.

[Gra02] C. Gray. An analysis of the Belousov-Zhabotinskii reaction. *Rose-Hulman Undergraduate Mathematics Journal*, 3(1), 2002. http://www.rose-hulman.edu/mathjournal/.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems*. Springer-Verlag, second revised edition, 1996.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

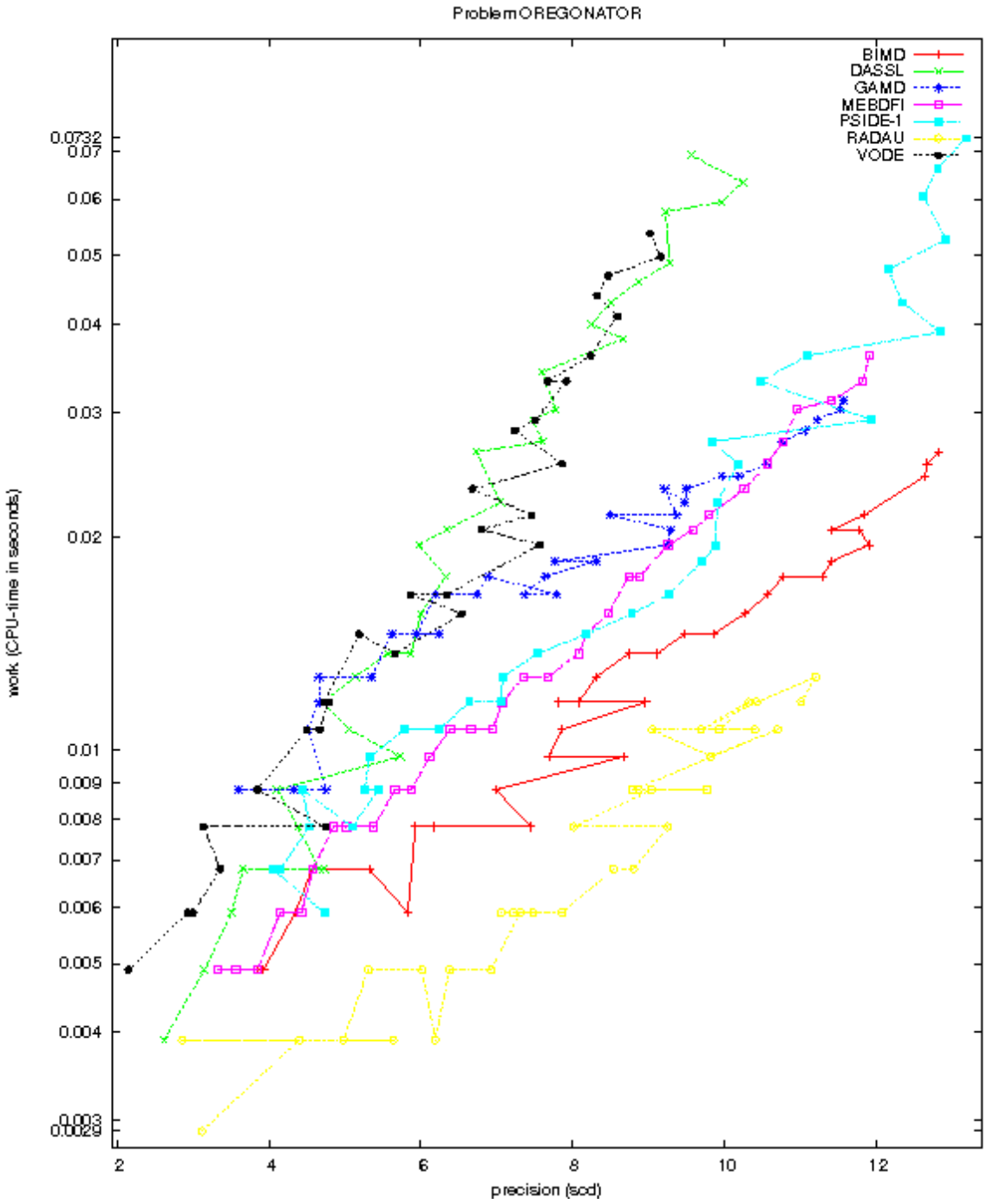

FIGURE II.9.4: *Work-precision diagram (scd versus CPU-time).*

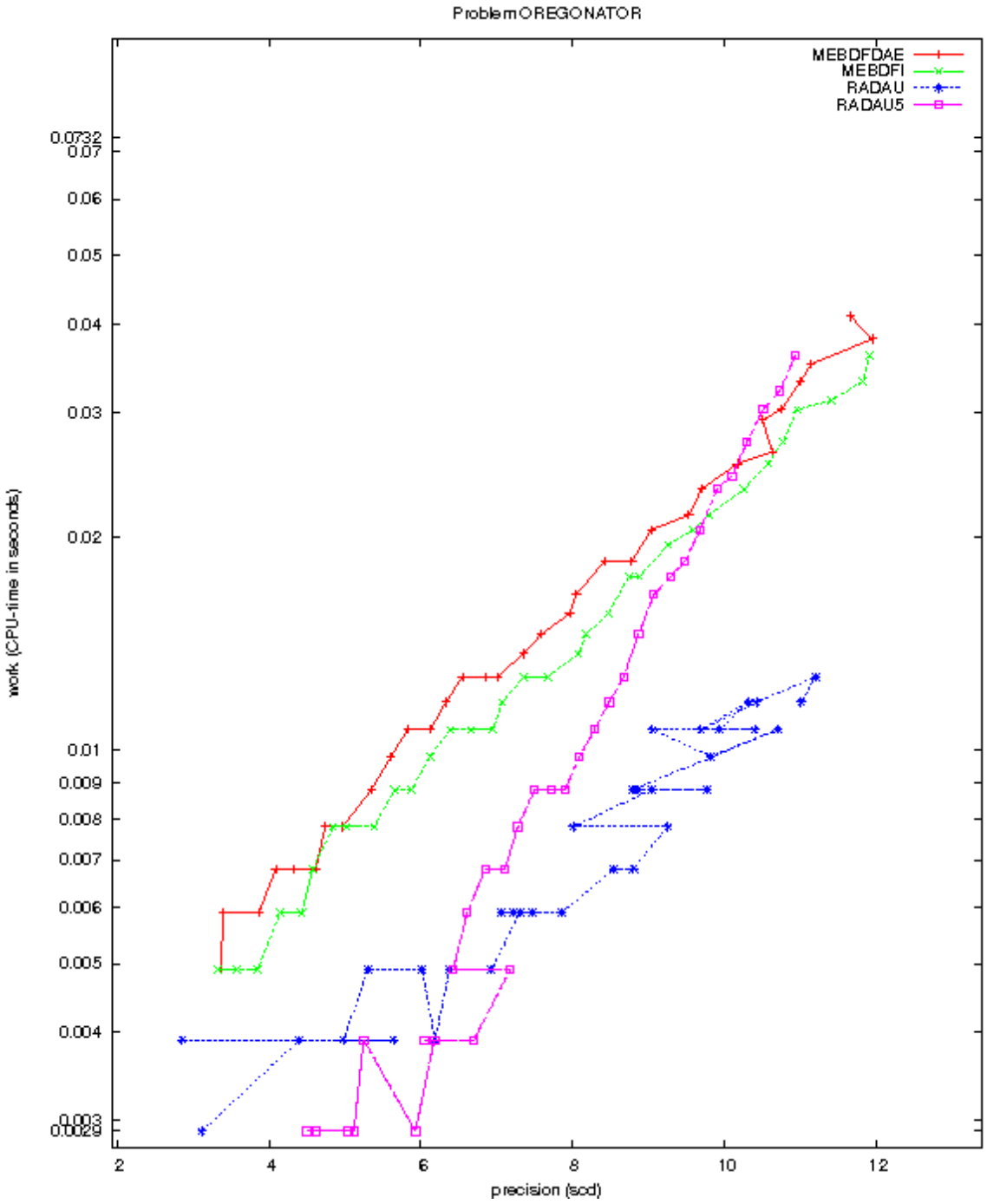

FIGURE II.9.5: *Work-precision diagram (scd versus CPU-time).*

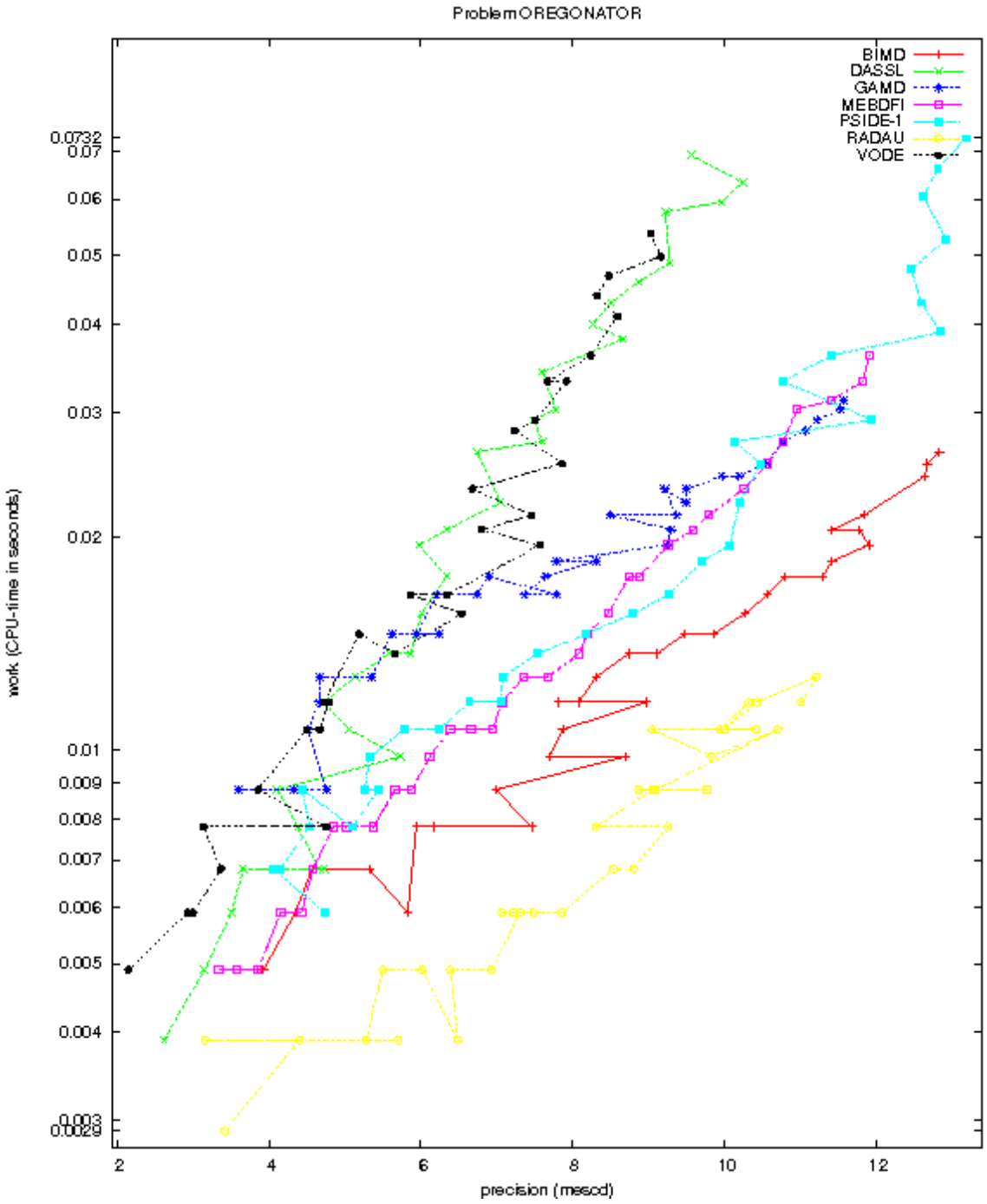

FIGURE II.9.6: *Work-precision diagram (mescd versus CPU-time).*

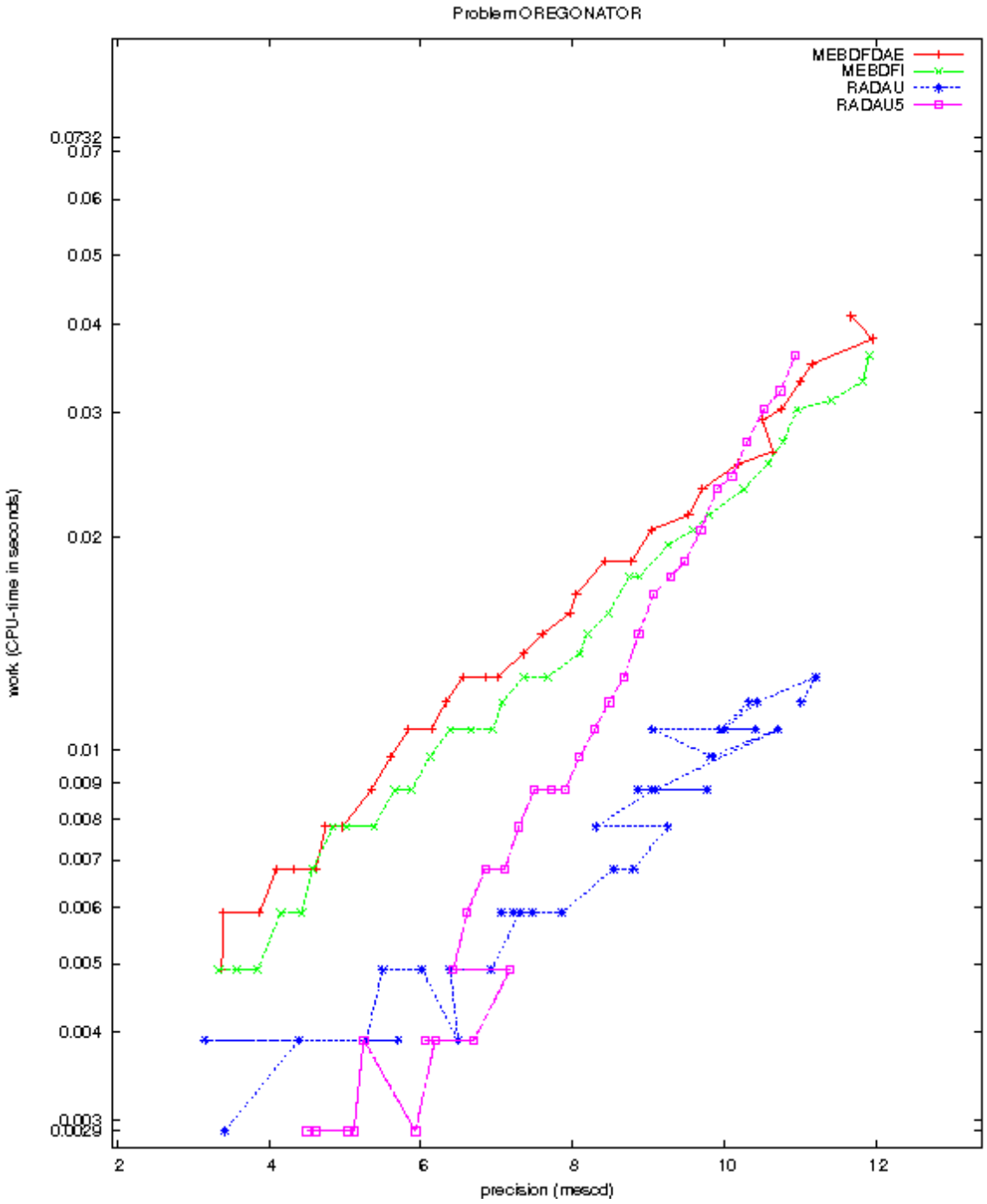

FIGURE II.9.7: *Work-precision diagram (mescd versus CPU-time).*

10 Problem ROBER

10.1 General information

The problem consists of a stiff system of 3 non-linear ordinary differential equations. It was proposed by H.H. Robertson in 1966 [Rob66]. The name ROBER was given by Hairer & Wanner [HW96]. The INdAM-Bari Test Set group contributed this problem to the test set. The software part of the problem is in the file rober.f available at [MM08].

10.2 Mathematical description of the problem

The problem is of the form

$$\frac{dy}{dt} = f(y), \quad y(0) = y_0,$$

with

$$y \in I\!R^3, \quad t \in [0, T],$$

The function f is defined by

$$f(y) = \begin{pmatrix} -0.04y_1 + 10^4 y_2 y_3 \\ 0.04y_1 - 10^4 y_2 y_3 - 3 \cdot 10^7 y_2^2 \\ 3 \cdot 10^7 y_2^2 \end{pmatrix} \tag{II.10.1}$$

The initial vector y_0 is given by $(1, 0, 0)^T$.

10.3 Origin of the problem

The ROBER problem describes the kinetics of an autocatalytic reaction given by Robertson (1966) [Rob66]. The structure of the reactions is given in Table II.10.1, where k_1, k_2, k_3 are the rate constants and A, B and C are the chemical species involved. Under some idealized conditions [Aik85] and the

1.	A	$\xrightarrow{k_1}$	B
2.	$B + B$	$\xrightarrow{k_2}$	$C + B$
3.	$B + C$	$\xrightarrow{k_3}$	$A + C$

TABLE II.10.1: *Reaction scheme for problem ROBER*

assumption that the mass action law is applied for the rate functions, the following mathematical model consisting of a set of three ODEs can be set up

$$\begin{pmatrix} y_1' \\ y_2' \\ y_3' \end{pmatrix} = \begin{pmatrix} -k_1 y_1 + k_3 y_2 y_3 \\ k_1 y_1 - k_2 y_2^2 - k_3 y_2 y_3 \\ k_2 y_2^2 \end{pmatrix}, \tag{II.10.2}$$

with $(y_1(0),\ y_2(0),\ y_3(0))^T = (y_{01},\ y_{02},\ y_{03})^T$, where y_1, y_2, y_3 denote the concentrations of A, B and C respectively and y_{01}, y_{02}, y_{03} are the concentrations at time $t = 0$.

The ROBER problem is very popular in numerical studies [Eds74] and it is often used as a test problem in the stiff integrators comparisons. The numerical values of the rate constants used in the test problem are $k_1 = 0.04$, $k_2 = 3 \cdot 10^7$and $k_3 = 10^4$, and the initial concentrations $y_{01} = 1$, $y_{02} =$

0, $y_{03} = 0$. The large difference among the reaction rate constants is the reason for stiffness. As is typical for problems arising in chemical kinetics this special system has a small very quick initial transient. This phase is followed by a very smooth variation of the components where a large stepsize would be appropriate for a numerical method.

Originally the problem was posed on the interval $0 \leq t \leq 40$, but it is convenient to integrate it on much longer intervals. As a matter of fact Hindmarsh discovered that many codes fail if t becomes very large. In this case if y_2 accidentally becomes negative, it then tends to $-\infty$, causing overflow (see [HW96]).

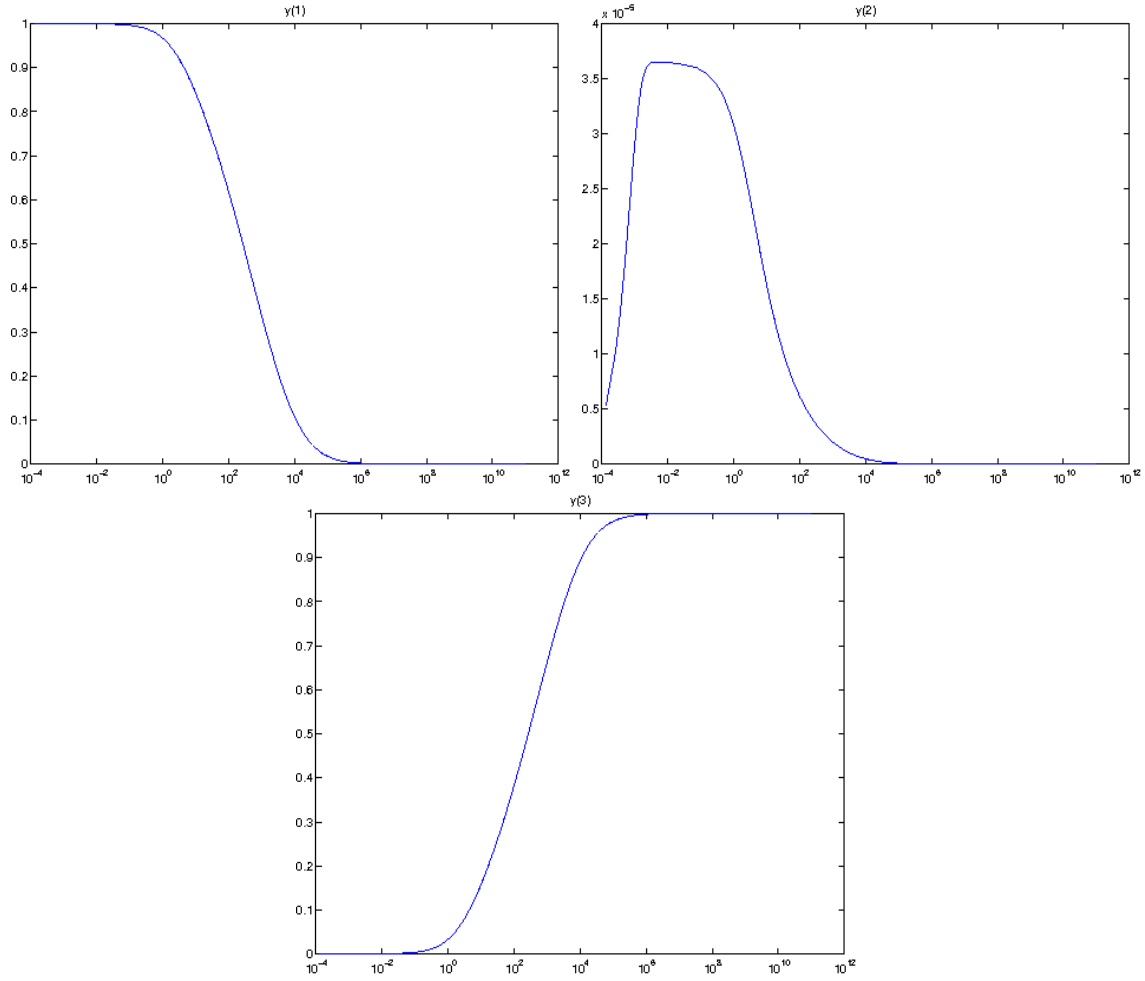

FIGURE II.10.1: *Behavior of the solution on* $[0, 10^{11}]$

10.4 Numerical solution of the problem

The system of ODEs is integrated for $t \in [0, 10^{11}]$. Tables II.10.3–II.10.4 and Figures II.10.1–II.10.5 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the components of the solution over part of the integration interval and the work-precision diagrams, respectively. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` $= 1.01 \cdot 10^{-19}$, `rtol = atol =`

TABLE II.10.2: *Failed runs.*

solver	m	reason
DASSL	$5, \cdots, 8, 10, 11, 13, \ldots, 32$	error test failed repeatedly

`h0` = $1.1 \cdot 10^{-18}$. For the work-precision diagrams, we used: rtol = $10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = 10^{-4}rtol; h0 = $10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.10.2; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

TABLE II.10.3: *Reference solution at the end of the integration interval.*

y_1	$0.2083340149701255 \cdot 10^{-7}$
y_2	$0.8333360770334713 \cdot 10^{-13}$
y_3	0.9999999791665050

TABLE II.10.4: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-8}	10^{-6}	6.70	3.02	101	100	1904	97	101	0.0020
	10^{-7}	10^{-11}	10^{-9}	10.07	6.39	132	131	3883	125	132	0.0039
	10^{-10}	10^{-14}	10^{-12}	13.70	10.12	159	157	6529	148	159	0.0068
DDASSL	10^{-4}	10^{-8}		4.51	0.83	473	453	682	62		0.0020
	10^{-7}	10^{-11}		7.15	3.47	1278	1252	1549	108		0.0059
GAMD	10^{-4}	10^{-8}	10^{-6}	6.27	2.59	62	62	2165	62	62	0.0020
	10^{-7}	10^{-11}	10^{-9}	9.94	6.05	93	91	4883	89	92	0.0059
	10^{-10}	10^{-14}	10^{-12}	12.41	8.73	169	169	9427	166	169	0.0107
MEBDFI	10^{-4}	10^{-8}	10^{-6}	6.25	2.56	401	398	1299	72	72	0.0029
	10^{-7}	10^{-11}	10^{-9}	8.95	5.27	804	802	2611	98	98	0.0049
	10^{-10}	10^{-14}	10^{-12}	11.53	7.85	1614	1612	5252	186	186	0.0107
PSIDE-1	10^{-4}	10^{-8}		5.75	2.07	56	55	1295	36	224	0.0020
	10^{-7}	10^{-11}		9.03	5.35	158	154	3128	39	496	0.0039
	10^{-10}	10^{-14}		11.29	7.61	570	563	9772	50	744	0.0127
RADAU	10^{-4}	10^{-8}	10^{-6}	6.74	3.06	114	112	811	108	113	0.0010
	10^{-7}	10^{-11}	10^{-9}	9.35	5.67	112	110	1852	104	112	0.0020
	10^{-10}	10^{-14}	10^{-12}	11.21	7.53	108	106	3420	92	108	0.0029
VODE	10^{-4}	10^{-8}		3.66	−0.02	593	576	830	12	100	0.0020
	10^{-7}	10^{-11}		6.70	3.02	1292	1220	1686	22	199	0.0049
	10^{-10}	10^{-14}		9.59	5.91	3306	3138	3873	56	408	0.0127

References

[Aik85] R.C. Aiken. *Stiff Computation*. Oxford University Press, 1985.

[Eds74] L. Edsberg. *Integration Package for Chemical Kinetics*, pages 81–94. Plenum Press, New York, 1974.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems*. Springer-Verlag, second revised edition, 1996.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Rob66] H.H. Robertson. *The solution of a set of reaction rate equations*, pages 178–182. Academ Press, 1966.

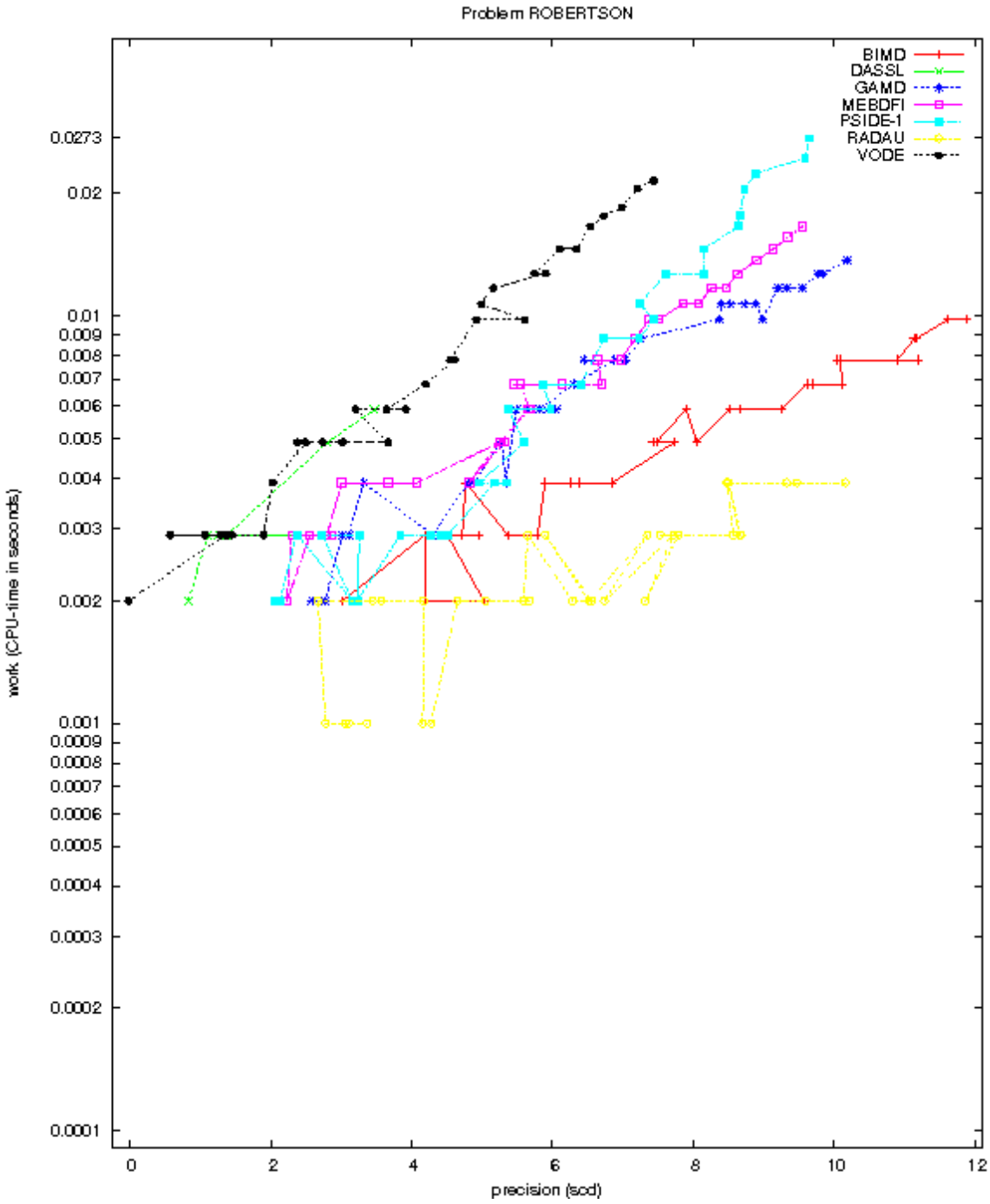

FIGURE II.10.2: *Work-precision diagram (scd versus CPU-time.*

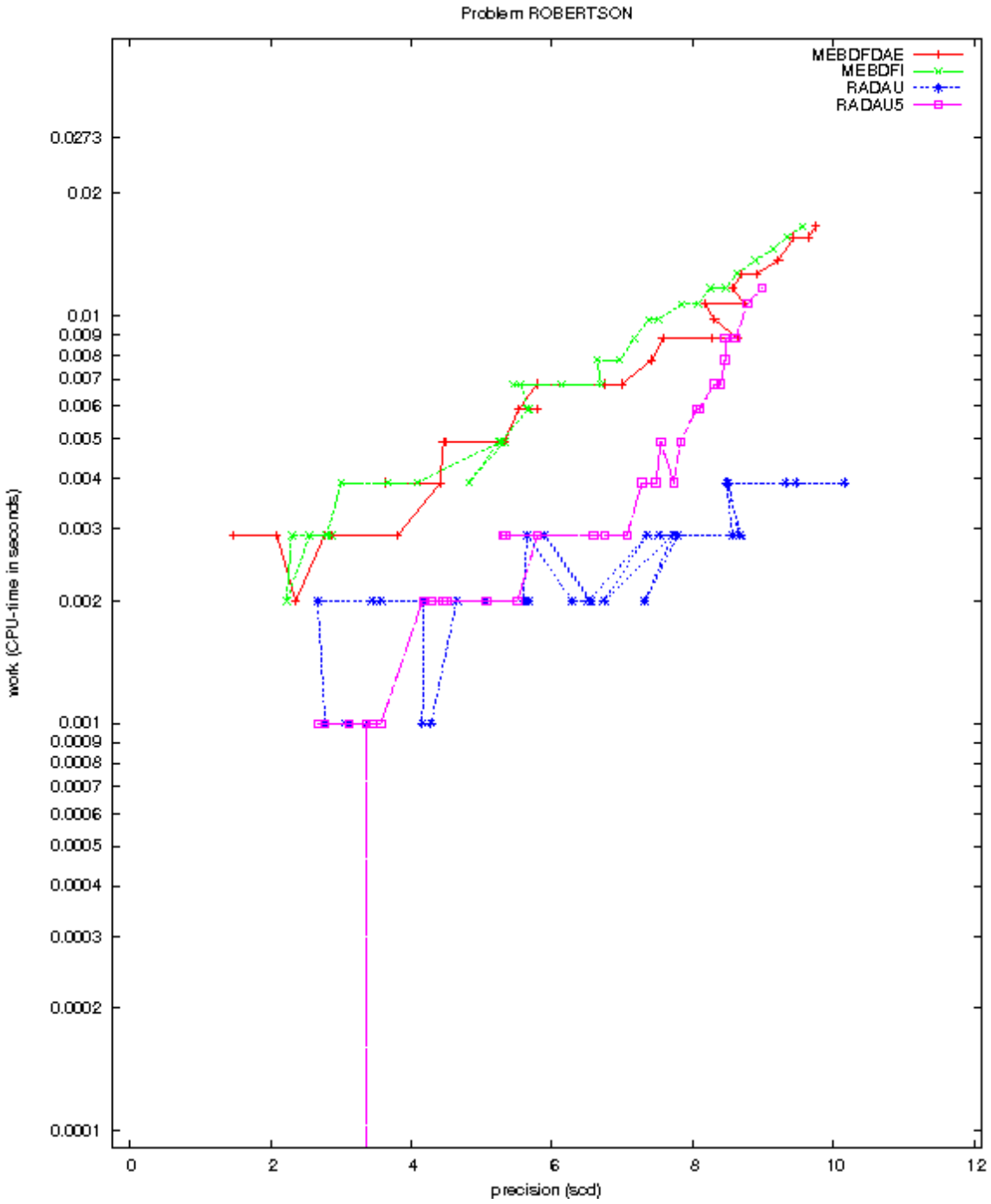

FIGURE II.10.3: *Work-precision diagram (scd versus CPU-time.*

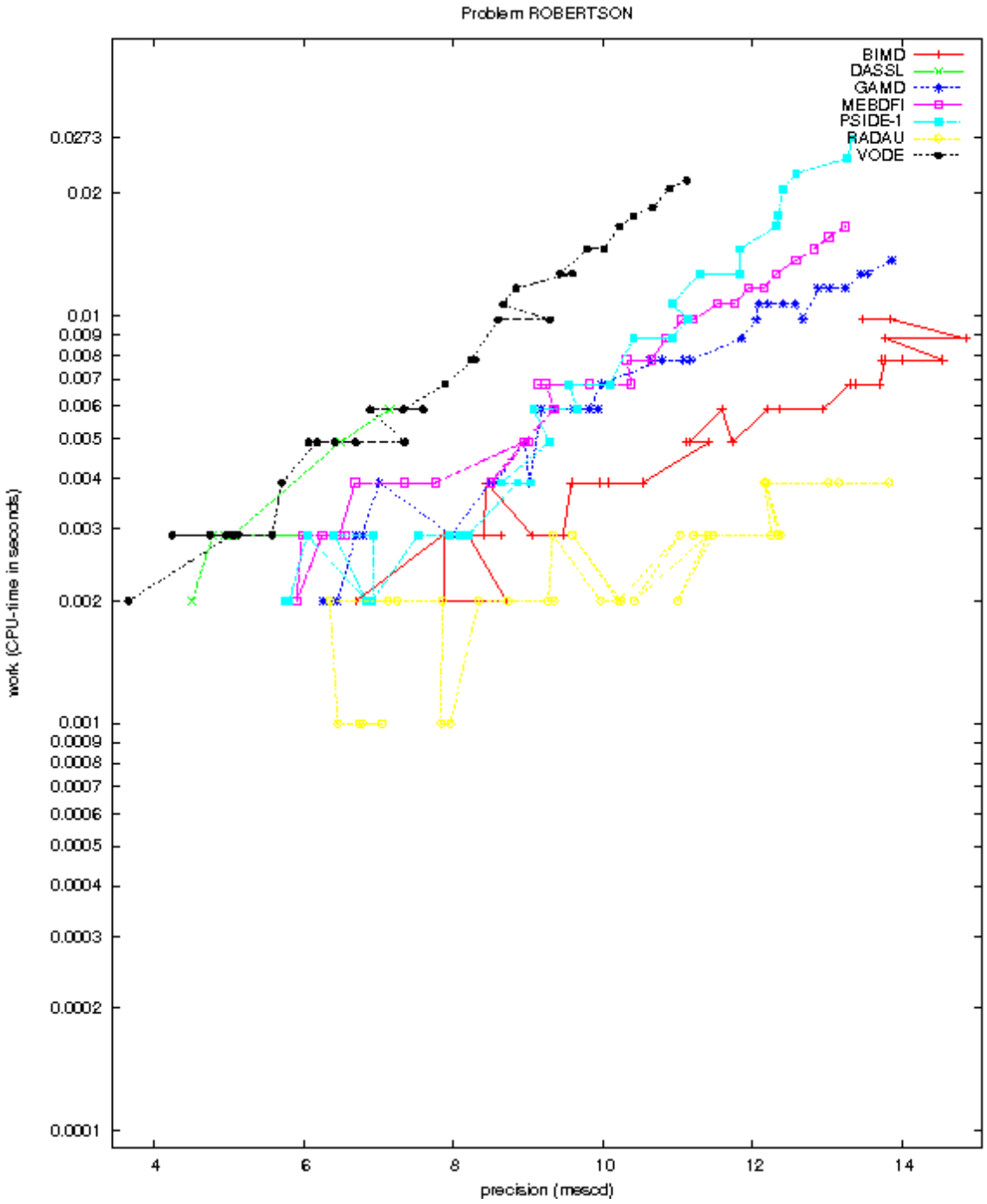

FIGURE II.10.4: *Work-precision diagram (mescd versus CPU-time.*

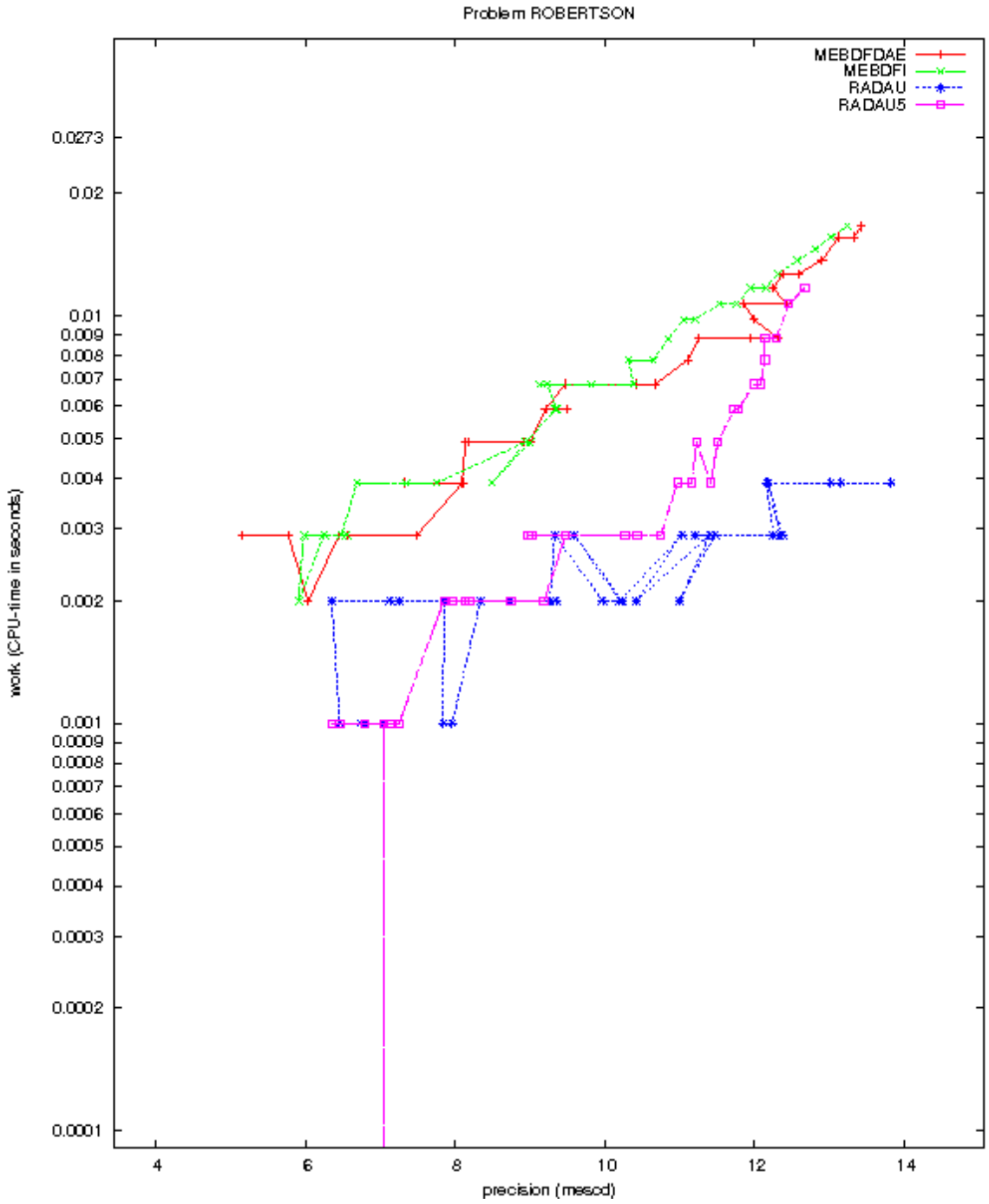

FIGURE II.10.5: *Work-precision diagram (mescd versus CPU-time.*

11 Problem E5

11.1 General information

The problem consists of a stiff system of 4 non-linear ordinary differential equations. It was proposed by Datta in 1967. The name E5 was given by Enright, Hull and Lindberg (1975) [EHL75]. The formulation and data have been taken from [HW96]. The Bari Test Set group contributed this problem to the test set. The software part of the problem is in the file e5.f available at [MM08].

11.2 Mathematical description of the problem

The problem is of the form

$$\frac{dy}{dt} = f(y), \quad y(0) = y_0,$$

with

$$y \in I\!R^4, \quad t \in [0, T],$$

The function f is defined by

$$f(y) = \begin{pmatrix} -Ay_1 - By_1y_3 \\ Ay_1 - MCy_2y_3 \\ Ay_1 - By_1y_3 - MCy_2y_3 + Cy_4 \\ By_1y_3 - Cy_4 \end{pmatrix} \tag{II.11.1}$$

where $A = 7.89 \cdot 10^{-10}$, $B = 1.1 \cdot 10^7$, $C = 1.13 \cdot 10^3$, and $M = 10^6$.
The initial vector y_0 is given by $(1.76 \cdot 10^{-3}, 0, 0, 0)^T$.

11.3 Origin of the problem

The E5 problem is a model for chemical pyrolysis studied by Datta in 1967 and describes a reaction involving six reactants. The reaction scheme is given in Table II.11.1, where A_i, $i = 1, \ldots, 6$ are the chemical species and k_1, k_2, k_3, k_4 the rate of reaction constants. According to mass action kinetics,

A_1	$\xrightarrow{k_1}$	$A_2 + A_3$
$A_2 + A_3$	$\xrightarrow{k_2}$	A_5
$A_1 + A_3$	$\xrightarrow{k_3}$	A_4
A_4	$\xrightarrow{k_4}$	$A_3 + A_6$

TABLE II.11.1: *Reaction scheme for problem E5*

the corresponding mathematical model is the following

$$\begin{cases} y_1' = -k_1y_1 - k_3y_1y_3 \\ y_2' = k_1y_1 - k_2y_2y_3 \\ y_3' = k_1y_1 - k_2y_2y_3 - k_3y_1y_3 + k_4y_4 \\ y_4' = k_3y_1y_3 - k_4y_4 \\ y_5' = k_2y_2y_3 \\ y_6' = k_4y_4 \end{cases} \tag{II.11.2}$$

TABLE II.11.2: *Failed runs.*

solver	m	reason
DASSL	0,1,2,6,7,8,9,11,13, 14,16,...,32	error test failed repeatedly

where y_i are the concentrations of the reactants A_i. This set of ODEs is one of the test problems in the stiff integrator comparison by Enright, Hull and Lindberg (1975) [EHL75]. The rate constants used in the test problem were $k_1 = 7.89 \cdot 10^{-10}$, $k_2 = 1.13 \cdot 10^9$, $k_3 = 1.1 \cdot 10^7$, $k_4 = 1.13 \cdot 10^3$ and the initial values were all set to zero except for $y_1(0) = 1.76 \cdot 10^{-3}$. The fastly different rates of reaction that occur in the same system are the cause for stiffness. With rate constants inserted in (II.11.2) the system (II.11.1) is obtained [Aik85]. Note that the differential equation possesses the invariant $y_2 - y_3 - y_4 = 0$ and it is recommended to use the relation $y_3' = y_2' - y_4'$ in the function subroutine in order to avoid eventual cancellation of digits [HW96].

Although the problem was originally posed on the interval $0 \le t \le 1000$, it is often integrated on a much longer interval because of the interesting properties of the solutions for t large [HW96]. In 1981 Shampine [Sha81] observed that since the solution components are badly scaled ($|y_1| \le 2 \cdot 10^{-3}$ and the magnitude of all the other components doesn't exceed $4 \cdot 10^{-10}$), a scalar absolute error control is quite unsuitable and a componentwise scaled absolute error control would be recommendable for this problem.

11.4 Numerical solution of the problem

The system of ODEs is integrated for $t \in [0, 10^{13}]$. Tables II.11.3–II.11.4 present the reference solution at the end of the integration interval and the run characteristics, Figures II.11.1–II.11.3 present the behavior of the components of the solution over the integration interval and the work-precision diagrams, respectively. The work precision diagrams were computed using the mescd since the solution at the end of the integration interval is very close to zero. For the same reason, the scd column in Table II.11.4 has been skipped. The reference solution was computed by RADAU on an Alphaserver DS20E, with a 667 MHz EV67 processor, using double precision `work(1) = uround` = $1.01 \cdot 10^{-19}$, `rtol = h0` = $1.1 \cdot 10^{-18}$,`atol` = $1.1 \cdot 10^{-40}$. For the work-precision diagrams, we used: rtol = $10^{-(4+m/4)}$, $m = 0, 1, \ldots, 32$; atol = $1.7 \cdot 10^{-24}$; h0 = $10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDF-DAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.11.2; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

TABLE II.11.3: *Reference solution at the end of the integration interval.*

y_1	$0.1152903278711829 \cdot 10^{-290}$	y_3	$0.8854814626268838 \cdot 10^{-22}$
y_2	$0.8867655517642120 \cdot 10^{-22}$	y_4	0.0000000000000000000

References

[Aik85] R.C. Aiken. *Stiff Computation.* Oxford University Press, 1985.

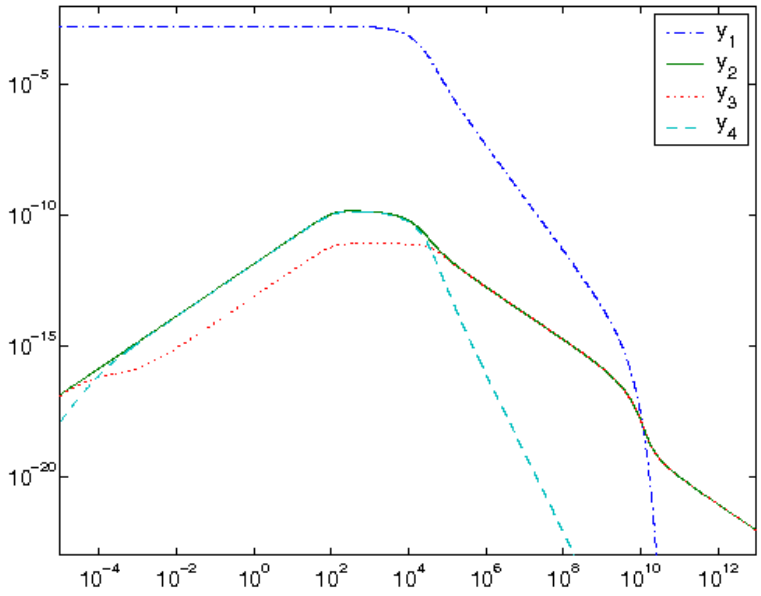

FIGURE II.11.1: - *Behavior of the solution over the integration interval in double logarithmic scale.*

[EHL75] W.H. Enright, T.E. Hull, and B. Lindberg. Comparing numerical methods for stiff systems of ODEs. *BIT*, 15:10–48, 1975.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems*. Springer-Verlag, second revised edition, 1996.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Sha81] L.F. Shampine. Evaluation of a test set for stiff ode solvers. *ACM Trans. Math. Soft.*, 8:93–113, 1981.

TABLE II.11.4: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	1.110^{-24}	10^{-6}	4.98	2.70	169	169	3438	162	169	0.0049
	10^{-7}	1.110^{-24}	10^{-9}	8.34	3.05	174	174	6409	168	174	0.0088
	10^{-10}	1.110^{-24}	10^{-12}	11.77	3.48	287	287	10726	282	287	0.0156
DDASSL	10^{-7}	1.110^{-24}		7.55	2.26	2516	2468	3443	148		0.0137
GAMD	10^{-4}	1.110^{-24}	10^{-6}	5.52	3.24	103	101	4977	99	103	0.0068
	10^{-7}	1.110^{-24}	10^{-9}	8.19	2.90	125	125	9167	122	125	0.0117
	10^{-10}	1.110^{-24}	10^{-12}	11.13	2.84	154	154	13497	154	154	0.0166
MEBDFI	10^{-4}	1.110^{-24}	10^{-6}	5.16	2.87	653	644	2145	86	86	0.0049
	10^{-7}	1.110^{-24}	10^{-9}	8.13	2.85	1048	1043	3423	122	122	0.0088
	10^{-10}	1.110^{-24}	10^{-12}	10.56	2.27	1782	1779	5823	188	188	0.0137
PSIDE-1	10^{-4}	1.110^{-24}		3.94	1.65	137	112	3160	69	544	0.0049
	10^{-7}	1.110^{-24}		7.99	2.71	255	243	5181	173	944	0.0078
	10^{-10}	1.110^{-24}		11.46	3.18	707	704	13278	286	1512	0.0195
RADAU	10^{-4}	1.110^{-24}	10^{-6}	4.72	2.43	100	99	2220	80	100	0.0029
	10^{-7}	1.110^{-24}	10^{-9}	8.42	3.14	148	145	3123	118	144	0.0039
	10^{-10}	1.110^{-24}	10^{-12}	11.79	3.51	142	132	5733	106	141	0.0059
VODE	10^{-4}	1.110^{-24}		3.17	0.88	1238	1149	1718	27	260	0.0059
	10^{-7}	1.110^{-24}		6.67	1.39	2655	2484	3464	47	397	0.0107
	10^{-10}	1.110^{-24}		9.69	1.41	4003	3836	4776	70	458	0.0156

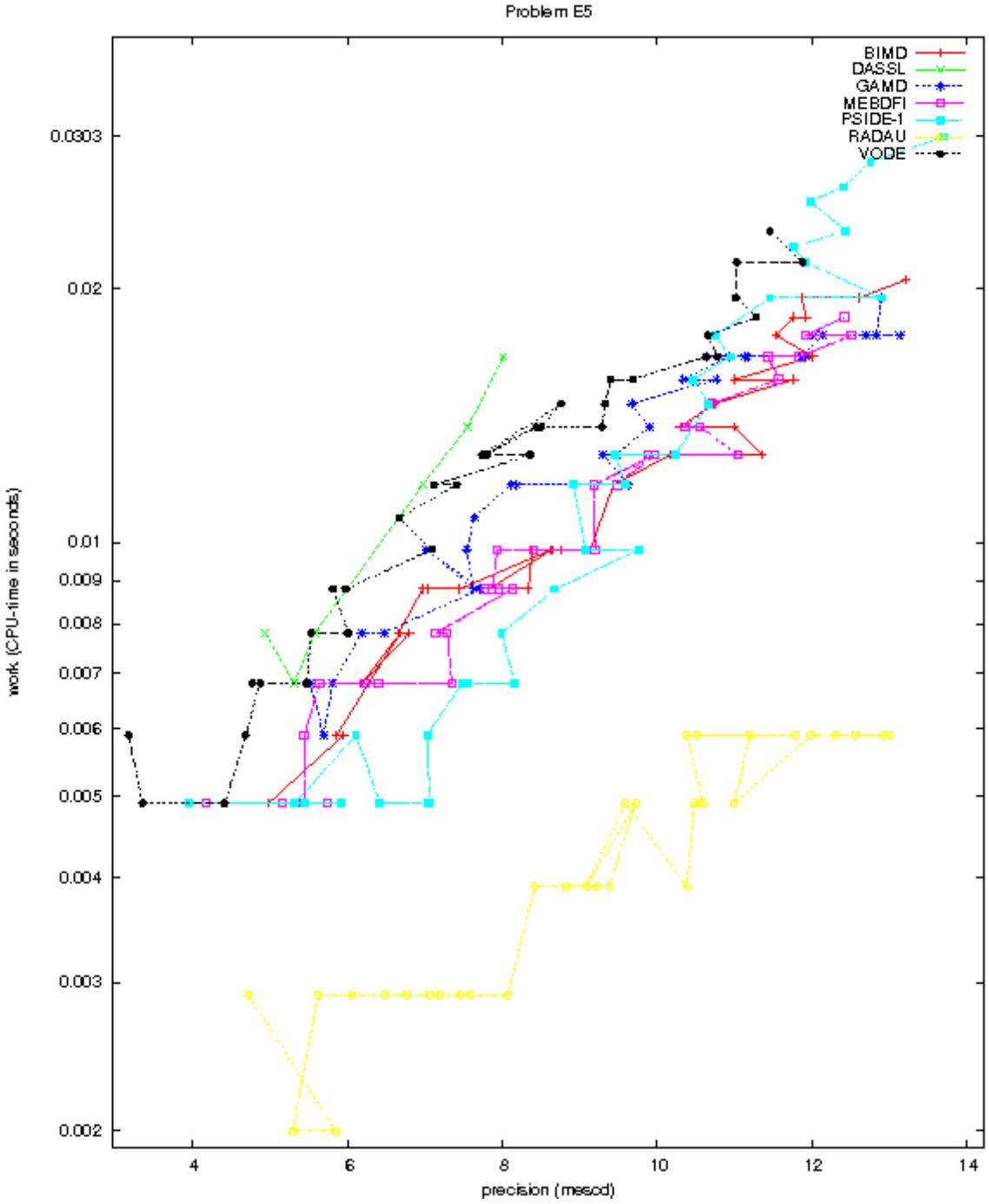

FIGURE II.11.2: *Work-precision diagram (mescd versus CPU-time).*

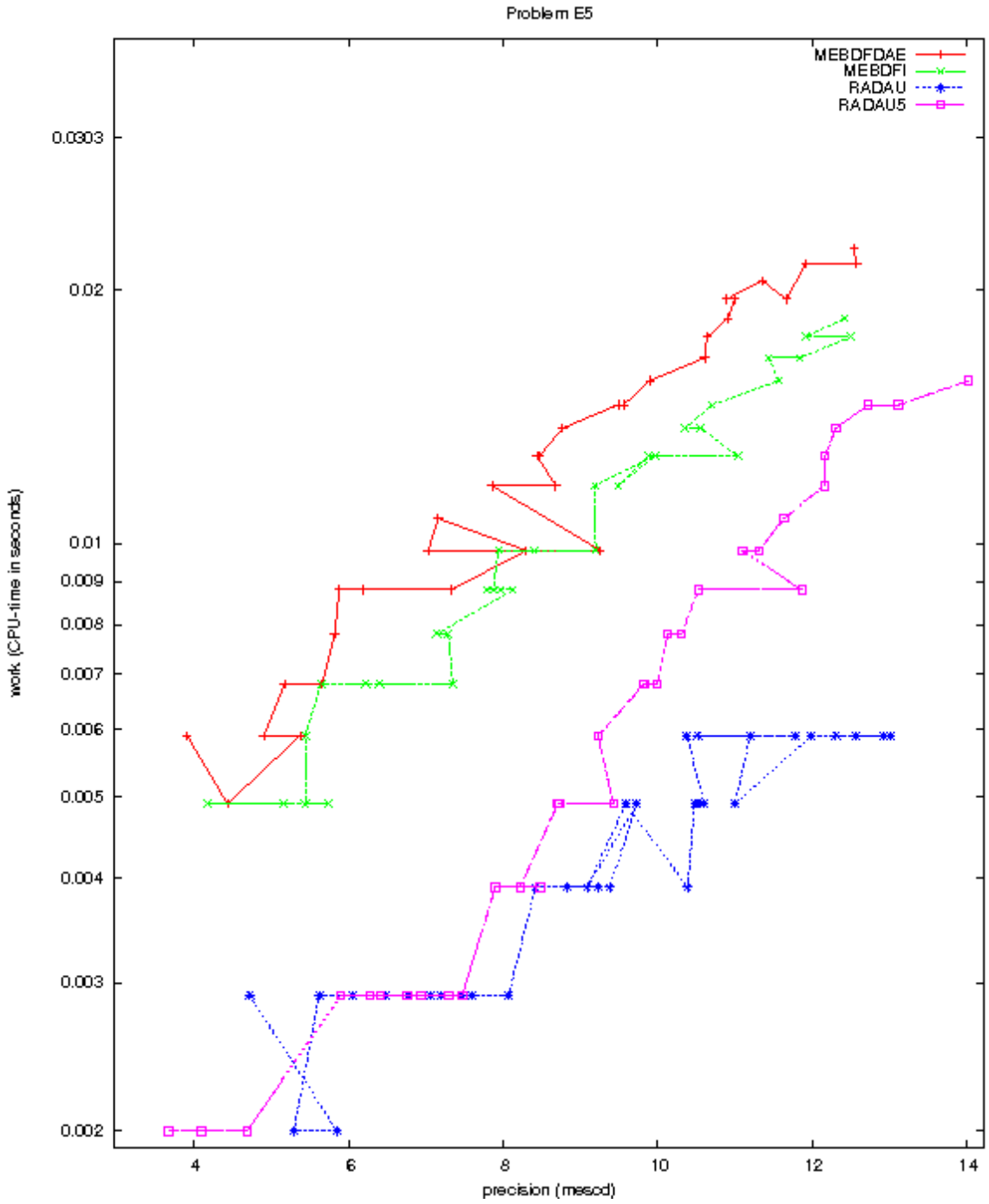

FIGURE II.11.3: *Work-precision diagram (mescd versus CPU-time).*

12 Chemical Akzo Nobel problem

12.1 General information

This IVP is a stiff system of 6 non-linear DAEs of index 1 and has been taken from [Sto98]. The parallel-IVP-algorithm group of CWI contributed this problem to the test set in collaboration with W.J.H. Stortelder. We acknowledge the remarks of Dotsikas Ioannis, which improved the formulation of this problem considerably. The software part of the problem is in the file chemakzo.f available at [MM08].

12.2 Mathematical description of the problem

The problem is of the form

$$M\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \qquad y(0) = y_0, \quad y'(0) = y'_0,$$

with

$$y \in I\!R^6, \quad 0 \le t \le 180.$$

The matrix M is of rank 5 and given by

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

and the function f by

$$f(y) = \begin{pmatrix} -2r_1 & +r_2 & -r_3 & -r_4 & & \\ -\frac{1}{2}r_1 & & & -r_4 & -\frac{1}{2}r_5 & +F_{in} \\ r_1 & -r_2 & +r_3 & & & \\ & -r_2 & +r_3 & -2r_4 & & \\ & r_2 & -r_3 & & +r_5 & \\ K_s \cdot y_1 \cdot y_4 - y_6 & & & & & \end{pmatrix},$$

where the r_i and F_{in} are auxiliary variables, given by

$$\begin{aligned} r_1 &= k_1 \cdot y_1^4 \cdot y_2^{\frac{1}{2}}, \\ r_2 &= k_2 \cdot y_3 \cdot y_4, \\ r_3 &= \frac{k_2}{K} \cdot y_1 \cdot y_5, \\ r_4 &= k_3 \cdot y_1 \cdot y_4^2, \\ r_5 &= k_4 \cdot y_6^2 \cdot y_2^{\frac{1}{2}}, \\ F_{in} &= klA \cdot (\frac{p(\mathrm{CO}_2)}{H} - y_2). \end{aligned}$$

The values of the parameters k_1, k_2, k_3, k_4, K, klA, $p(\mathrm{CO}_2)$ and H are

$$\begin{array}{rclrclrcl} k_1 &=& 18.7, & k_4 &=& 0.42, & K_s &=& 115.83, \\ k_2 &=& 0.58, & K &=& 34.4, & p(\mathrm{CO}_2) &=& 0.9, \\ k_3 &=& 0.09, & klA &=& 3.3, & H &=& 737. \end{array}$$

The consistent initial vectors are

$$y_0 = \left(\; 0.444,\, 0.00123,\, 0,\, 0.007,\, 0,\, K_s \cdot y_{0,1} \cdot y_{0,4} \; \right)^{\mathrm{T}} \qquad y_0' = f(y_0).$$

It is clear from the definition of r_1 and r_5 that the function f can not be evaluated for negative values of y_2. In the Fortran subroutine that defines f, we set `IERR=-1` if $y_2 < 0$ to prevent this situation. See page *IV-ix* of the description of the software part of the test set for more details on `IERR`.

12.3 Origin of the problem

The problem originates from Akzo Nobel Central Research in Arnhem, The Netherlands. It describes a chemical process, in which 2 species, FLB and ZHU, are mixed, while carbon dioxide is continuously added. The resulting species of importance is ZLA. In the interest of commercial competition, the names of the chemical species are fictitious. The reaction equations, as given by Akzo Nobel [CBS93], are given in Figure II.12.1. The last reaction equation describes an equilibrium

$$\begin{aligned}
2\,\mathrm{FLB} + \frac{1}{2}\mathrm{CO_2} &\xrightarrow{k_1} \mathrm{FLBT} + \mathrm{H_2O} \\
\mathrm{ZLA} + \mathrm{FLB} &\underset{k_2}{\overset{k_2/K}{\rightleftharpoons}} \mathrm{FLBT} + \mathrm{ZHU} \\
\mathrm{FLB} + 2\,\mathrm{ZHU} + \mathrm{CO_2} &\xrightarrow{k_3} \mathrm{LB} + \mathrm{nitrate} \\
\mathrm{FLB.ZHU} + \frac{1}{2}\mathrm{CO_2} &\xrightarrow{k_4} \mathrm{ZLA} + \mathrm{H_2O} \\
\mathrm{FLB} + \mathrm{ZHU} &\rightleftharpoons \mathrm{FLB.ZHU}
\end{aligned}$$

FIGURE II.12.1: *Reaction scheme for Chemical Akzo Nobel problem.*

$$K_s = \frac{[\mathrm{FLB.ZHU}]}{[\mathrm{FLB}] \cdot [\mathrm{ZHU}]}.$$

The value of K_s plays a role in parameter estimation. The other equations describe reactions with velocities given by

$$\begin{aligned}
r_1 &= k_1 \cdot [\mathrm{FLB}]^4 \cdot [\mathrm{CO_2}]^{\frac{1}{2}}, && \text{(II.12.1)} \\
r_2 &= k_2 \cdot [\mathrm{FLBT}] \cdot [\mathrm{ZHU}], \\
r_3 &= \frac{k_2}{K} \cdot [\mathrm{FLB}] \cdot [\mathrm{ZLA}], \\
r_4 &= k_3 \cdot [\mathrm{FLB}] \cdot [\mathrm{ZHU}]^2, && \text{(II.12.2)} \\
r_5 &= k_4 \cdot [\mathrm{FLB.ZHU}]^2 \cdot [\mathrm{CO_2}]^{\frac{1}{2}}, && \text{(II.12.3)}
\end{aligned}$$

respectively. Here the square brackets '[ ]' denote concentrations. One would expect from the reaction scheme in Figure II.12.1, that reaction velocities r_1, r_4 and r_5 would read

$$\begin{aligned}
r_1 &= k_1 \cdot [\mathrm{FLB}]^2 \cdot [\mathrm{CO_2}]^{\frac{1}{2}}, \\
r_4 &= k_3 \cdot [\mathrm{FLB}] \cdot [\mathrm{ZHU}]^2 \cdot [\mathrm{CO_2}], \\
r_5 &= k_4 \cdot [\mathrm{FLB.ZHU}] \cdot [\mathrm{CO_2}]^{\frac{1}{2}}.
\end{aligned}$$

However, it turns out that the chemical process under consideration is modeled more appropriately using (II.12.1)–(II.12.3).

The inflow of carbon dioxide per volume unit is denoted by F_{in}, and satisfies

$$F_{in} = klA \cdot (\frac{p(CO_2)}{H} - [CO_2]),$$

where klA is the mass transfer coefficient, H is the Henry constant and $p(CO_2)$ is the partial carbon dioxide pressure. $p(CO_2)$ is assumed to be independent of $[CO_2]$. The parameters k_1, k_2, k_3, k_4, K, klA, K_s, H and $p(CO_2)$ are given constants*.

The process is started by mixing 0.444 mol/liter FLB with 0.007 mol/liter ZHU. The concentration of carbon dioxide at the beginning is 0.00123 mol/liter. Initially, no other species are present. The simulation is performed on the time interval [0, 180 minutes].

Identifying the concentrations [FLB], $[CO_2]$, [FLBT], [ZHU], [ZLA], [FLB.ZHU] with $y_1, \ldots, y_6$, respectively, one easily arrives at the mathematical formulation of the preceding section.

12.4 Numerical solution of the problem

Tables II.12.1–II.12.2 and Figures II.12.2–II.12.6 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed by PSIDE on a Cray C90, using double precision, `rtol = atol` 10^{-19}. To get more insight in the exact behavior of the second component, we included a plot of y_2 on $[0, 3]$ in Figure II.12.2. For the work-precision diagrams, we used: rtol = $10^{-(8+m/4)}$, $m = 0, 1, \ldots, 20$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.12.3; listed are the

TABLE II.12.1: *Reference solution at the end of the integration interval.*

y_1	0.1150794920661702	y_4	$0.3656156421249283 \cdot 10^{-3}$
y_2	$0.1203831471567715 \cdot 10^{-2}$	y_5	$0.1708010885264404 \cdot 10^{-1}$
y_3	0.1611562887407974	y_6	$0.4873531310307455 \cdot 10^{-2}$

TABLE II.12.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-10}	10^{-10}	10^{-10}	12.39	10.61	41	41	1177	41	41	0.0039
DDASSL	10^{-10}	10^{-10}		10.04	8.33	522	515	649	38		0.0039
GAMD	10^{-10}	10^{-10}	10^{-10}	11.89	9.82	35	35	1737	35	35	0.0039
MEBDFI	10^{-10}	10^{-10}	10^{-10}	11.42	9.76	274	273	916	32	32	0.0029
PSIDE-1	10^{-10}	10^{-10}		11.41	9.91	87	85	1671	15	204	0.0039
RADAU	10^{-10}	10^{-10}	10^{-10}	10.71	8.39	43	41	696	30	43	0.0010

name of the solver that failed, for which values of m this happened, and the reason for failing.

*Apart from H, which is generally known, all parameters have been estimated by W. Stortelder [Sto95].

TABLE II.12.3: *Failed runs.*

solver	m	reason
PSIDE-1	14,16,17,18,19,20	stepsize too small

References

[CBS93] CBS-reaction-meeting Köln. Handouts, May 1993. Br/ARLO-CRC.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Sto95] W.J.H. Stortelder, 1995. Private communication.

[Sto98] W.J.H. de Stortelder. *Parameter Estimation in Nonlinear Dynamical Systems.* PhD thesis, University of Amsterdam, March 12, 1998.

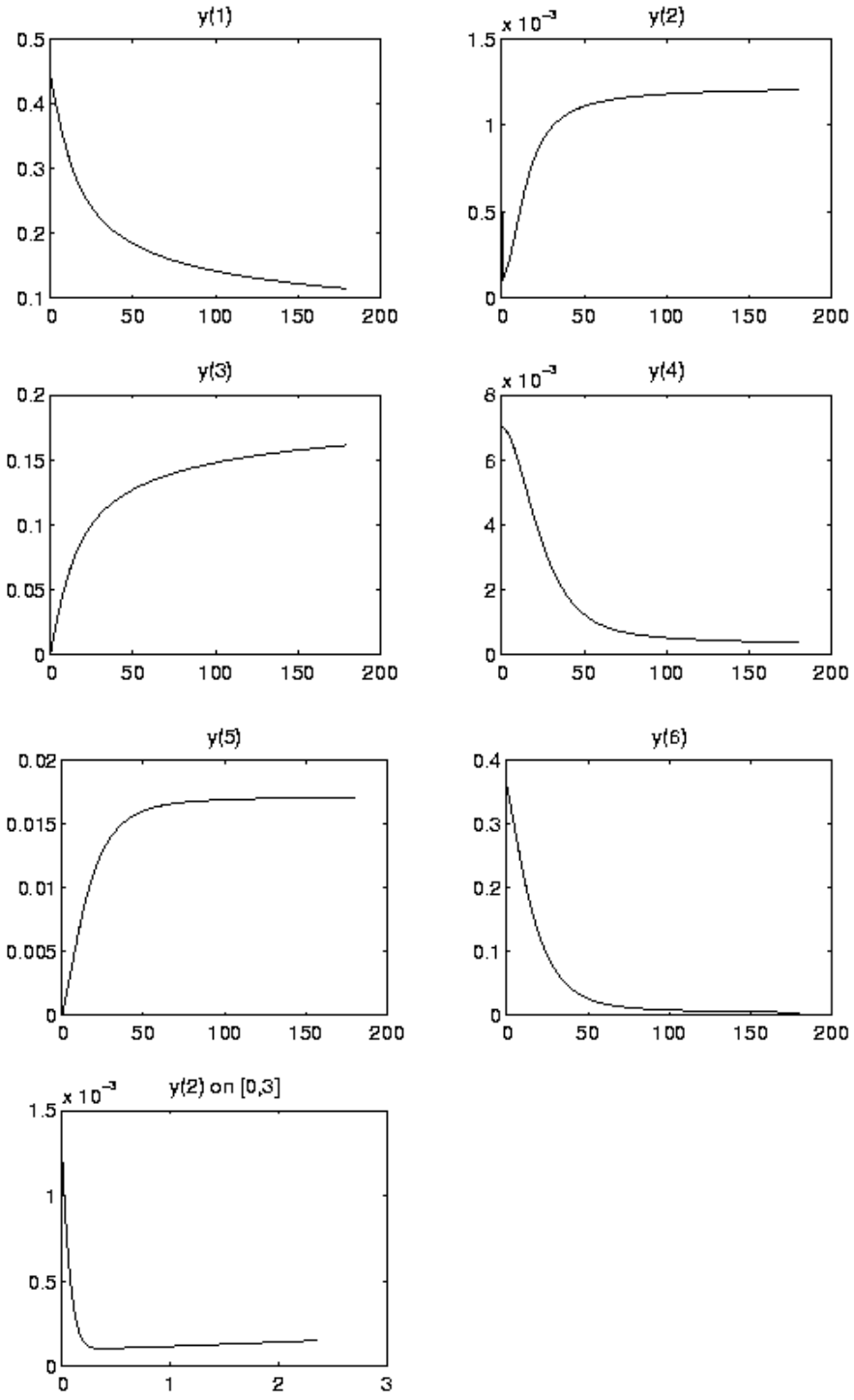

FIGURE II.12.2: *Behavior of the solution over the integration interval.*

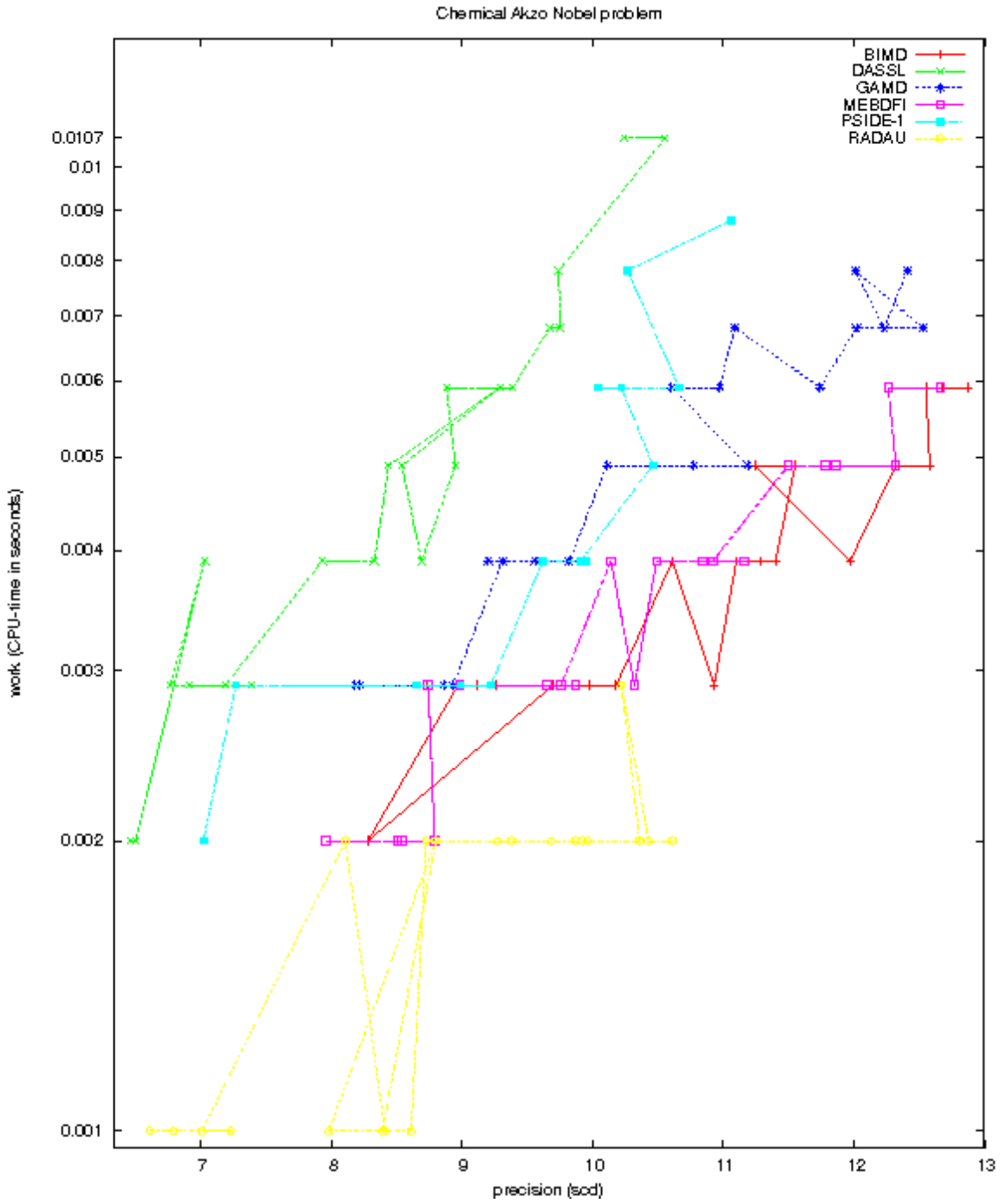

FIGURE II.12.3: *Work-precision diagram (scd versus CPU-time).*

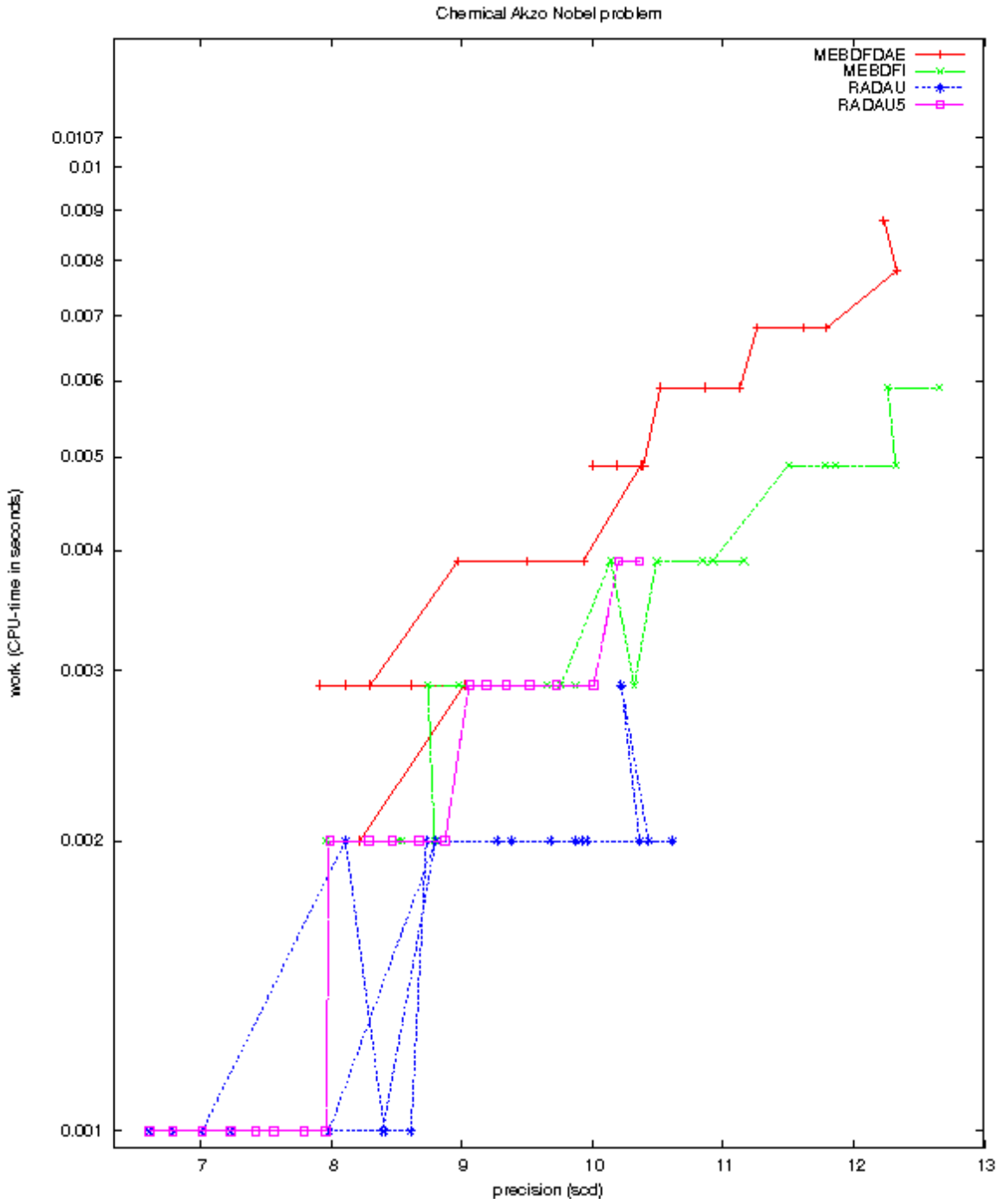

FIGURE II.12.4: *Work-precision diagram (scd versus CPU-time).*

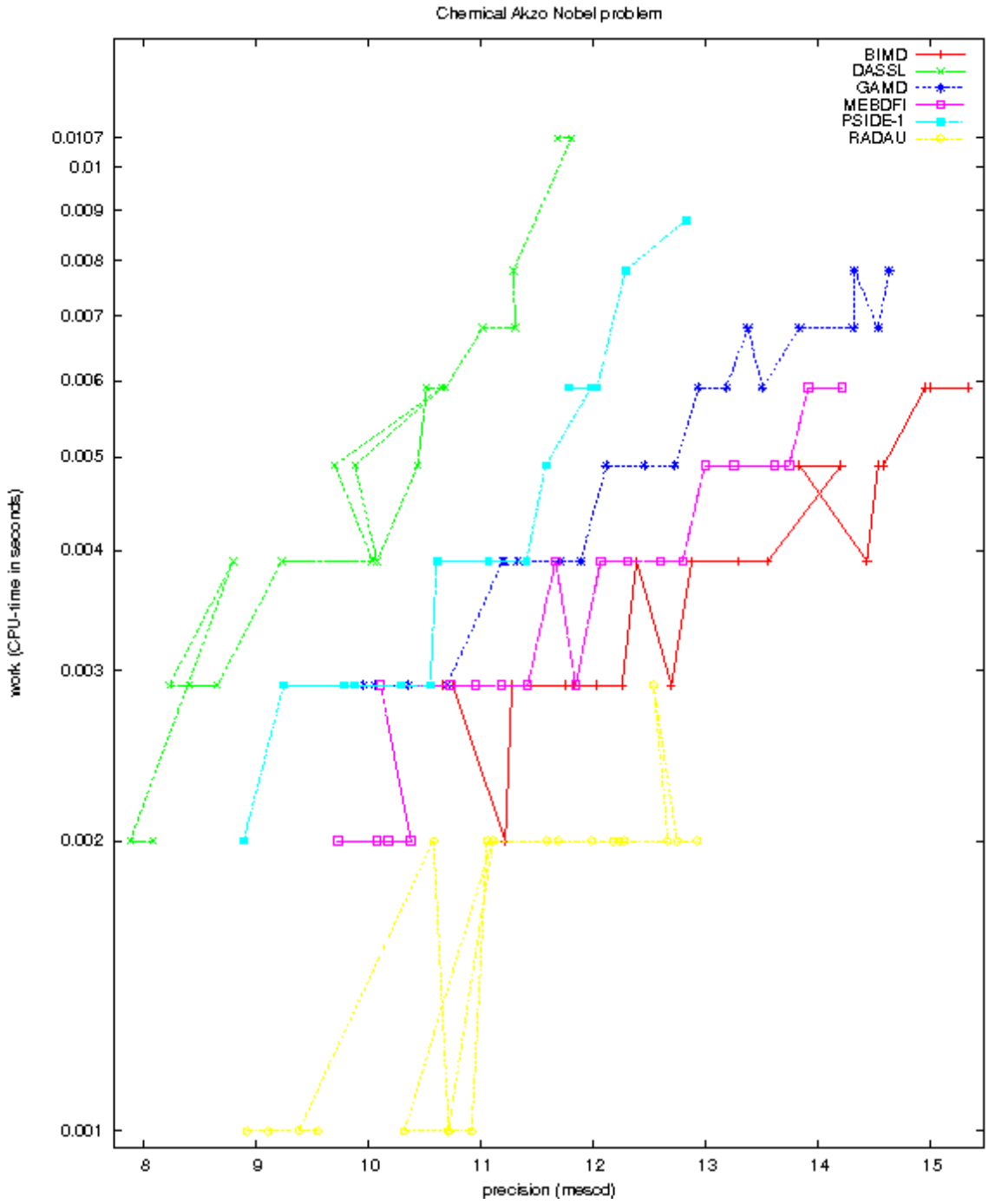

FIGURE II.12.5: *Work-precision diagram (mescd versus CPU-time).*

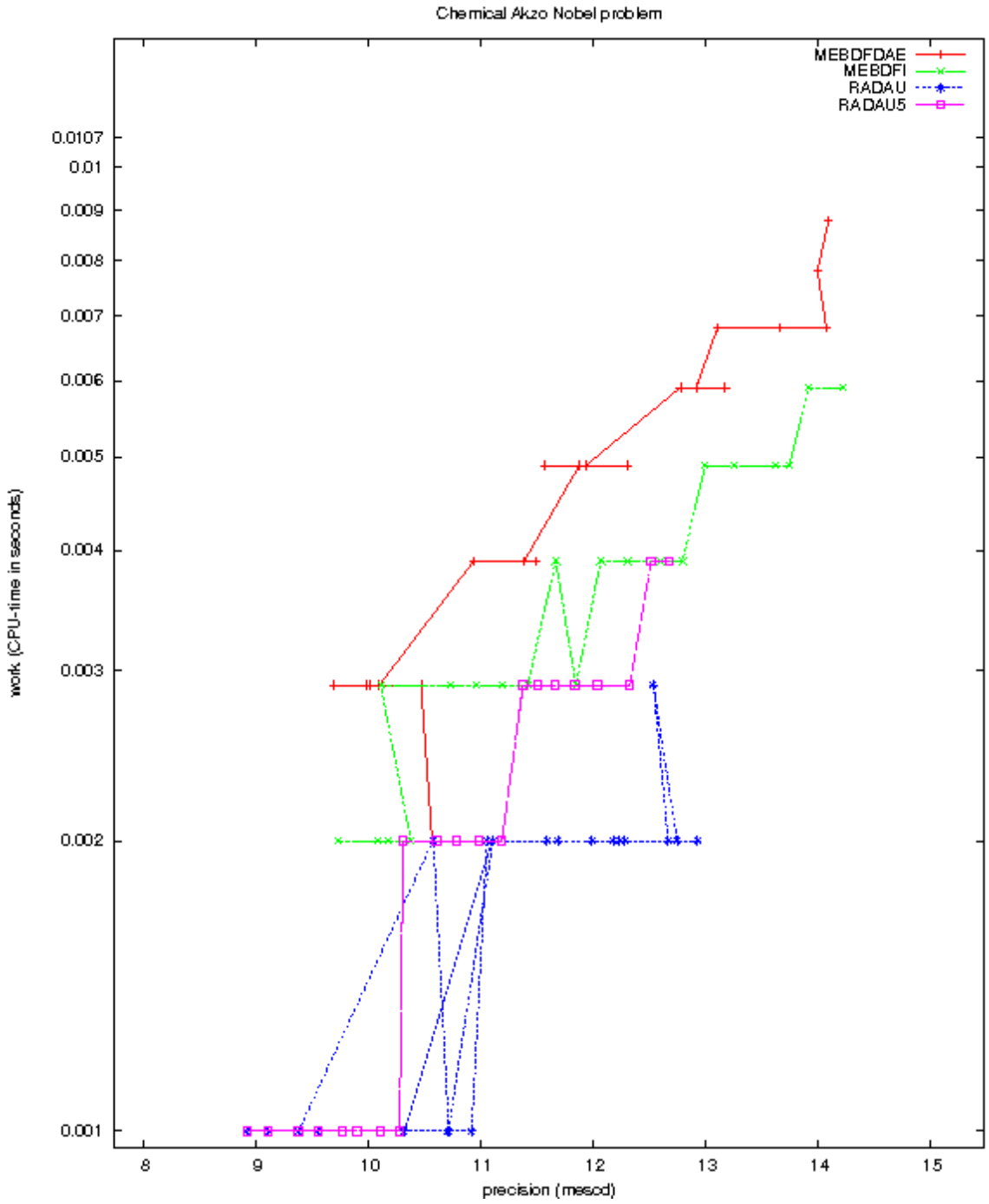

FIGURE II.12.6: *Work-precision diagram (mescd versus CPU-time).*

13 Andrews' squeezing mechanism

13.1 General information

The problem is a non-stiff second order DAE of index 3, consisting of 14 differential and 13 algebraic equations. It has been promoted as a test problem by Giles [Gil78] and Manning [Man81]. The formulation here corresponds to the one presented in Hairer & Wanner [HW96]. The parallel-IVP-algorithm group of CWI contributed this problem to the test set. The software part of the problem is in the file andrews.f available at [MM08].

13.2 Mathematical description of the problem

The problem is of the form

$$K\frac{\mathrm{d}y}{\mathrm{d}t} = \phi(y), \qquad y(0) = y_0, \quad y'(0) = y_0', \tag{II.13.1}$$

where

$$y = \begin{pmatrix} q \\ \dot{q} \\ \ddot{q} \\ \lambda \end{pmatrix}, \quad K = \begin{bmatrix} I & O & O & O \\ O & I & O & O \\ O & O & O & O \\ O & O & O & O \end{bmatrix}, \quad \phi(y) = \begin{pmatrix} \dot{q} \\ \ddot{q} \\ M(q)\ddot{q} - f(q,\dot{q}) + G^{\mathrm{T}}(q)\lambda \\ g(q) \end{pmatrix}.$$

Here,

$$\begin{aligned} 0 &\le t \le 0.03, \\ q &\in I\!R^7, \\ \lambda &\in I\!R^6, \\ M &: I\!R^7 \to I\!R^{7\times 7}, \\ f &: I\!R^{14} \to I\!R^7, \\ g &: I\!R^7 \to I\!R^6, \\ G &= \frac{\partial g}{\partial q}. \end{aligned}$$

The function $M(q) = (M_{ij}(q))$ is given by:

$$\begin{aligned} M_{11}(q) &= m_1 \cdot ra^2 + m_2(rr^2 - 2da \cdot rr \cdot \cos q_2 + da^2) + I_1 + I_2, \\ M_{21}(q) &= M_{12}(q) = m_2(da^2 - da \cdot rr \cdot \cos q_2) + I_2, \\ M_{22}(q) &= m_2 \cdot da^2 + I_2, \\ M_{33}(q) &= m_3(sa^2 + sb^2) + I_3, \\ M_{44}(q) &= m_4(e - ea)^2 + I_4, \\ M_{54}(q) &= M_{45}(q) = m_4((e - ea)^2 + zt(e - ea)\sin q_4) + I_4, \\ M_{55}(q) &= m_4(zt^2 + 2zt(e - ea)\sin q_4 + (e - ea)^2) + m_5(ta^2 + tb^2) + I_4 + I_5, \\ M_{66}(q) &= m_6(zf - fa)^2 + I_6, \\ M_{76}(q) &= M_{67}(q) = m_6((zf - fa)^2 - u(zf - fa)\sin q_6) + I_6, \\ M_{77}(q) &= m_6((zf - fa)^2 - 2u(zf - fa)\sin q_6 + u^2) + m_7(ua^2 + ub^2) + I_6 + I_7, \\ M_{ij}(q) &= 0 \quad \text{for all other cases.} \end{aligned}$$

The function $f = (f_i(q, \dot{q}))$ reads:

$$\begin{aligned}
f_1(q,\dot{q}) &= mom - m_2 \cdot da \cdot rr \cdot \dot{q}_2(\dot{q}_2 + 2\dot{q}_1)\sin q_2,\\
f_2(q,\dot{q}) &= m_2 \cdot da \cdot rr \cdot \dot{q}_1^2 \cdot \sin q_2,\\
f_3(q,\dot{q}) &= F_x(sc \cdot \cos q_3 - sd \cdot \sin q_3) + F_y(sd \cdot \cos q_3 + sc \cdot \sin q_3),\\
f_4(q,\dot{q}) &= m_4 \cdot zt(e - ea)\dot{q}_5^2 \cdot \cos q_4,\\
f_5(q,\dot{q}) &= -m_4 \cdot zt(e - ea)\dot{q}_4(\dot{q}_4 + 2\dot{q}_5)\cos q_4,\\
f_6(q,\dot{q}) &= -m_6 \cdot u(zf - fa)\dot{q}_7^2 \cdot \cos q_6,\\
f_7(q,\dot{q}) &= m_6 \cdot u(zf - fa)\dot{q}_6(\dot{q}_6 + 2\dot{q}_7)\cos q_6.
\end{aligned}$$

F_x and F_y are defined by:

$$\begin{aligned}
F_x &= F(xd - xc),\\
F_y &= F(yd - yc),\\
F &= -c_0(L - l_0)/L,\\
L &= \sqrt{(xd - xc)^2 + (yd - yc)^2},\\
xd &= sd \cdot \cos q_3 + sc \cdot \sin q_3 + xb,\\
yd &= sd \cdot \sin q_3 - sc \cdot \cos q_3 + yb.
\end{aligned}$$

The function $g = (g_i(q))$ is given by:

$$\begin{aligned}
g_1(q) &= rr \cdot \cos q_1 - d \cdot \cos(q_1 + q_2) - ss \cdot \sin q_3 - xb,\\
g_2(q) &= rr \cdot \sin q_1 - d \cdot \sin(q_1 + q_2) + ss \cdot \cos q_3 - yb,\\
g_3(q) &= rr \cdot \cos q_1 - d \cdot \cos(q_1 + q_2) - e \cdot \sin(q_4 + q_5) - zt \cdot \cos q_5 - xa,\\
g_4(q) &= rr \cdot \sin q_1 - d \cdot \sin(q_1 + q_2) + e \cdot \cos(q_4 + q_5) - zt \cdot \sin q_5 - ya,\\
g_5(q) &= rr \cdot \cos q_1 - d \cdot \cos(q_1 + q_2) - zf \cdot \cos(q_6 + q_7) - u \cdot \sin q_7 - xa,\\
g_6(q) &= rr \cdot \sin q_1 - d \cdot \sin(q_1 + q_2) - zf \cdot \sin(q_6 + q_7) + u \cdot \cos q_7 - ya.
\end{aligned}$$

The constants arising in these formulas are given by:

m_1	=	0.04325	I_1	=	$2.194 \cdot 10^{-6}$	ss	=	0.035
m_2	=	0.00365	I_2	=	$4.410 \cdot 10^{-7}$	sa	=	0.01874
m_3	=	0.02373	I_3	=	$5.255 \cdot 10^{-6}$	sb	=	0.01043
m_4	=	0.00706	I_4	=	$5.667 \cdot 10^{-7}$	sc	=	0.018
m_5	=	0.07050	I_5	=	$1.169 \cdot 10^{-5}$	sd	=	0.02
m_6	=	0.00706	I_6	=	$5.667 \cdot 10^{-7}$	ta	=	0.02308
m_7	=	0.05498	I_7	=	$1.912 \cdot 10^{-5}$	tb	=	0.00916
xa	=	−0.06934	d	=	0.028	u	=	0.04
ya	=	−0.00227	da	=	0.0115	ua	=	0.01228
xb	=	−0.03635	e	=	0.02	ub	=	0.00449
yb	=	0.03273	ea	=	0.01421	zf	=	0.02
xc	=	0.014	rr	=	0.007	zt	=	0.04
yc	=	0.072	ra	=	0.00092	fa	=	0.01421
c_0	=	4530	l_0	=	0.07785	mom	=	0.033

Consistent initial values are

$$y_0 = (q_0, \dot{q}_0, \ddot{q}_0, \lambda_0)^{\mathrm{T}} \text{ and } y_0' = (\dot{q}_0, \ddot{q}_0, \dddot{q}_0, \dot{\lambda}_0)^{\mathrm{T}},$$

where

$$q_0 = \begin{pmatrix} -0.06171389001427644963589484580 01 \\ 0 \\ 0.45527981916307038025591238244 9 \\ 0.22266839016588588467447318560 9 \\ 0.48736497954384255022559895353 0 \\ -0.22266839016588588467447318560 9 \\ 1.23054744454982119249735015568 \end{pmatrix},$$

$$\dot{q}_0 = \ddot{q}_0 = (0,0,0,0,0,0,0)^{\mathrm{T}},$$

$$\ddot{q}_0 = \begin{pmatrix} 14222.44391995411387059116258 87 \\ -10666.83293996558540294337194 15 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$\lambda_0 = \begin{pmatrix} 98.56687039624108960576549821 70 \\ -6.12268834425566265503114393122 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$\dot{\lambda}_0 = (0,0,0,0,0,0)^{\mathrm{T}}.$$

The index of the q, $\dot{q}$, $\ddot{q}$ and λ components in y is 1, 2, 3 and 3, respectively.

13.3 Origin of the problem

Formulation (II.13.1) can be rewritten as

$$\begin{aligned} M(q)\ddot{q} &= f(q,\dot{q}) - G^{\mathrm{T}}(q)\lambda, \\ 0 &= g(q), \end{aligned}$$

which is the general form of a constrained mechanical system. More precisely, the problem describes the motion of 7 rigid bodies connected by joints without friction. It was promoted by [Gil78] and [Man81] as a test problem for numerical codes. [HW96, pp. 530–536] describes the system and the modeling process in full detail.

13.4 Numerical solution of the problem

The Jacobian $\partial\phi/\partial y$, needed by the numerical solver, was approximated by

$$\begin{bmatrix} O & I & O & O \\ O & O & I & O \\ O & O & M & G^{\mathrm{T}} \\ G & O & O & O \end{bmatrix},$$

which means that we neglect the derivatives of $f(q,\dot{q})$ as well as those of $M(q)$ and $G(q)$. Note that the evaluation of such a Jacobian does not cost anything, because M and G are already computed in the evaluation of ϕ. However, we did not exploit this in the numerical computations.

Tables II.13.2–II.13.3 and Figures II.13.1–II.13.5 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively. In computing the scd values, only the first seven components were considered, since they refer to the physically important quantities, in computing the mescd values all the components were considered. The reference solution was computed on the Cray C90, using PSIDE with Cray double precision and atol = rtol = 10^{-14}. For the work-precision diagrams, we used: rtol = $10^{-(4+m/8)}$, $m = 0, 1, \ldots, 48$; atol = rtol; h0 = rtol for BIMD, GAMD. MEBDFDAE, MEBDFI, RADAU and RADAU5.

The failed runs are in Table II.13.1; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

TABLE II.13.1: *Failed runs.*

solver	m	reason
GAMD	3,4,6	stepsize too small
RADAU	55, 56	stepsize too small

References

[Gil78] D.R.A. Giles. An algebraic approach to A-stable linear multistep-multiderivative integration formulas. *BIT*, 14:382–406, 1978.

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

[Man81] D.W. Manning. *A computer technique for simulating dynamic multibody systems based on dynamic formalism.* PhD thesis, Univ. Waterloo, Ontario, 1981.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

TABLE II.13.2: *Reference solution (first 7 components) at the end of the integration interval.*

y_1	$0.1581077119629904 \cdot 10^{2}$	y_4	-0.5347301163226948	y_6	0.5347301163226948
y_2	$-0.1575637105984298 \cdot 10^{2}$	y_5	0.5244099658805304	y_7	$0.1048080741042263 \cdot 10$
y_3	$0.4082224013073101 \cdot 10^{-1}$				

TABLE II.13.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-4}	0.27	3.05	46	41	1034	41	46	0.0185
	10^{-7}	10^{-7}	10^{-7}	2.82	5.38	122	122	2553	122	122	0.0459
GAMD	10^{-4}	10^{-4}	10^{-4}	0.35	2.82	82	58	2281	58	82	0.0293
	10^{-7}	10^{-7}	10^{-7}	1.53	4.54	128	116	5176	116	128	0.0693
MEBDFI	10^{-4}	10^{-4}	10^{-4}	−1.11	0.37	118	108	466	23	23	0.0078
	10^{-7}	10^{-7}	10^{-7}	1.25	3.50	300	287	1222	38	38	0.0195
PSIDE-1	10^{-4}	10^{-4}		0.22	2.95	92	75	1675	52	368	0.0410
	10^{-7}	10^{-7}		2.10	4.98	113	93	2637	63	428	0.0615
RADAU	10^{-4}	10^{-4}	10^{-4}	−0.84	1.36	96	56	810	54	96	0.0137
	10^{-7}	10^{-7}	10^{-7}	0.47	4.45	114	95	1292	90	114	0.0195

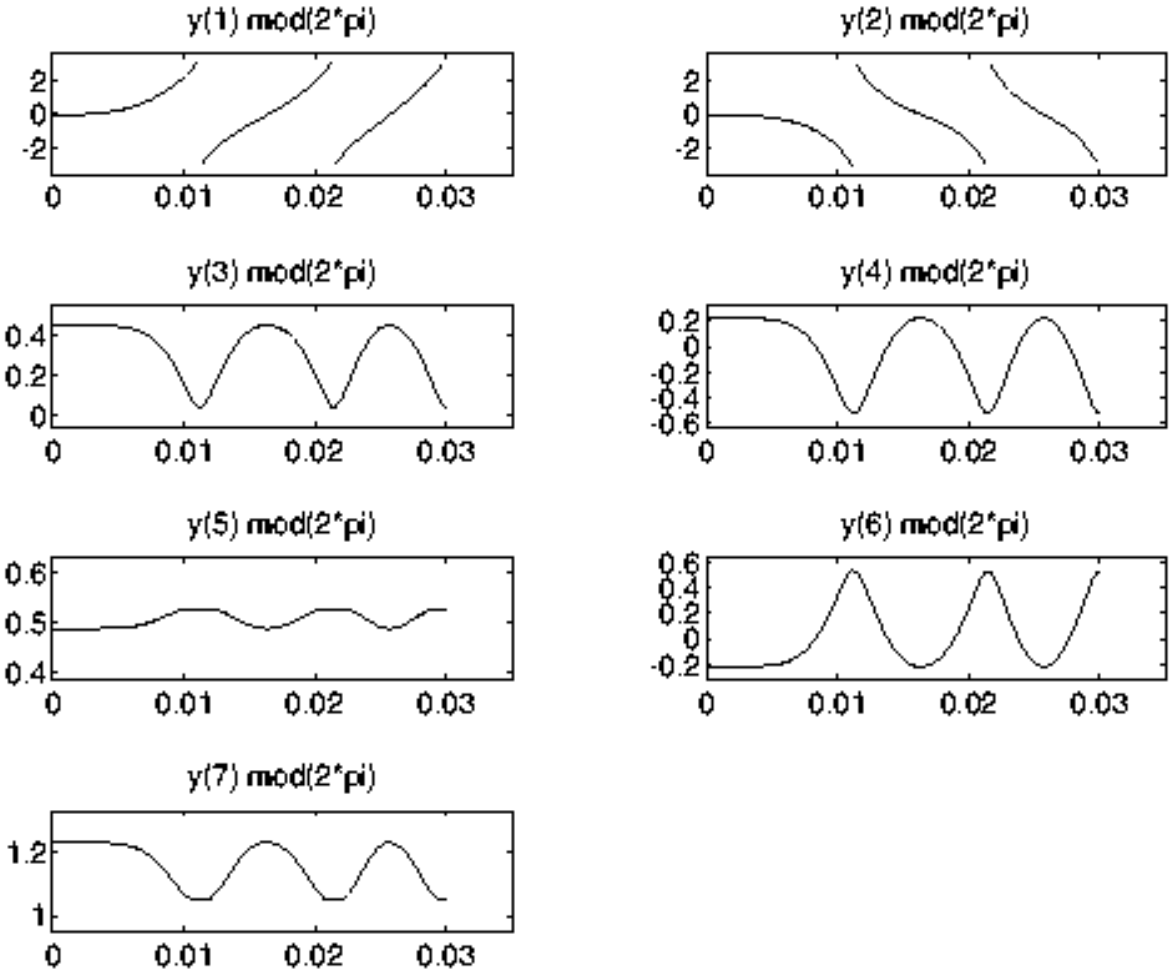

FIGURE II.13.1: *Behavior of the solution modulo 2π over the integration interval.*

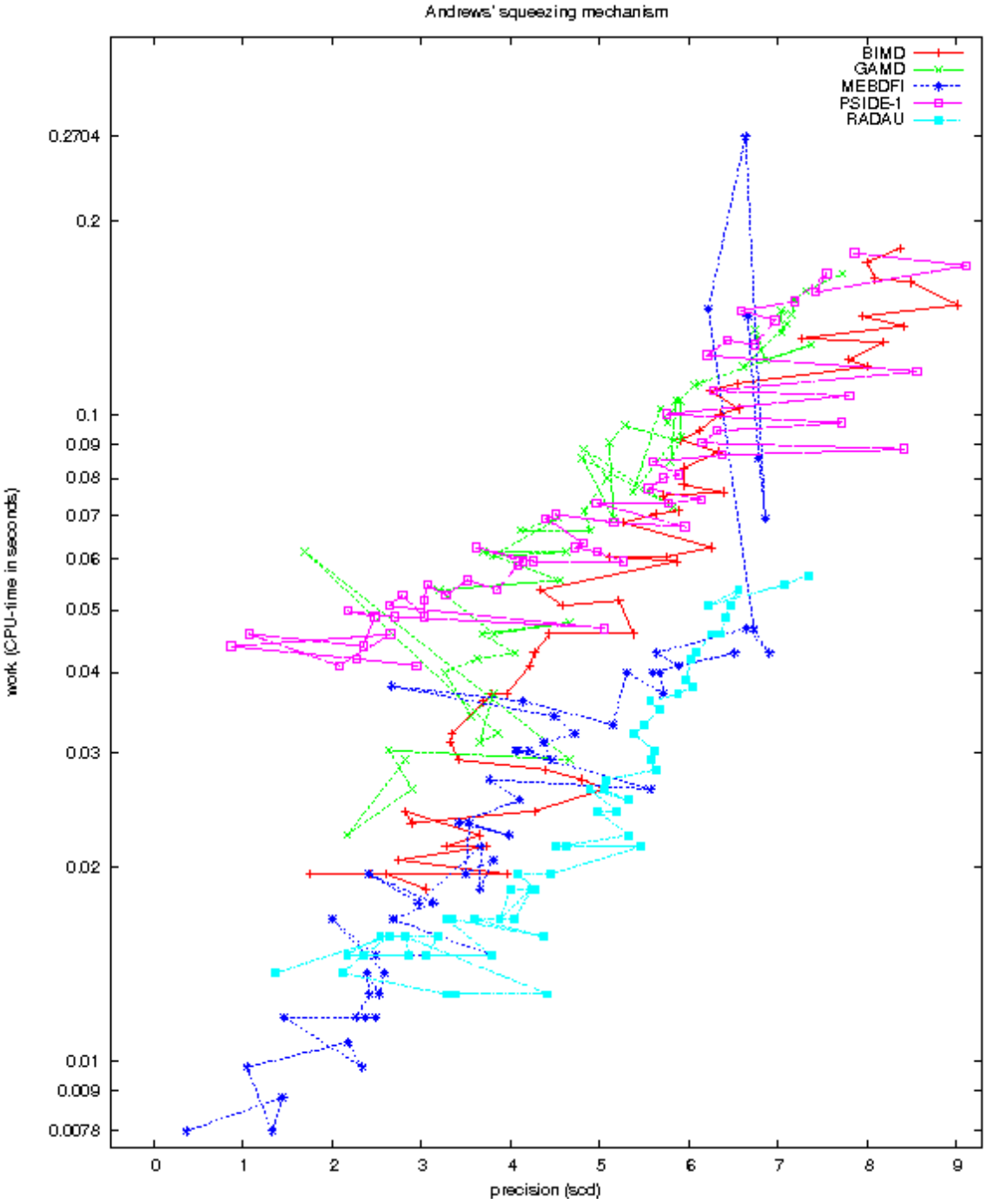

FIGURE II.13.2: *Work-precision diagram (scd versus CPU-time).*

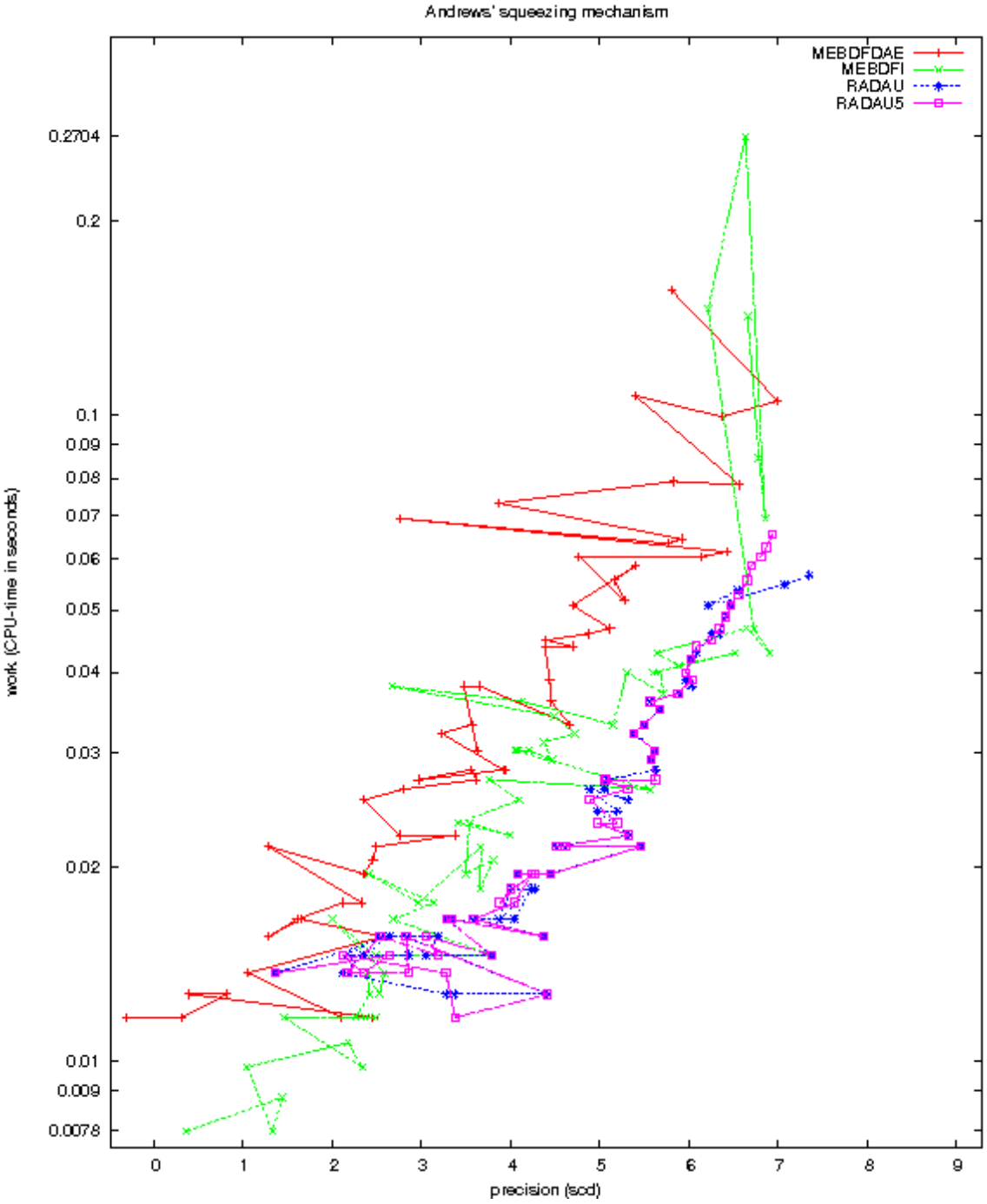

FIGURE II.13.3: *Work-precision diagram (scd versus CPU-time).*

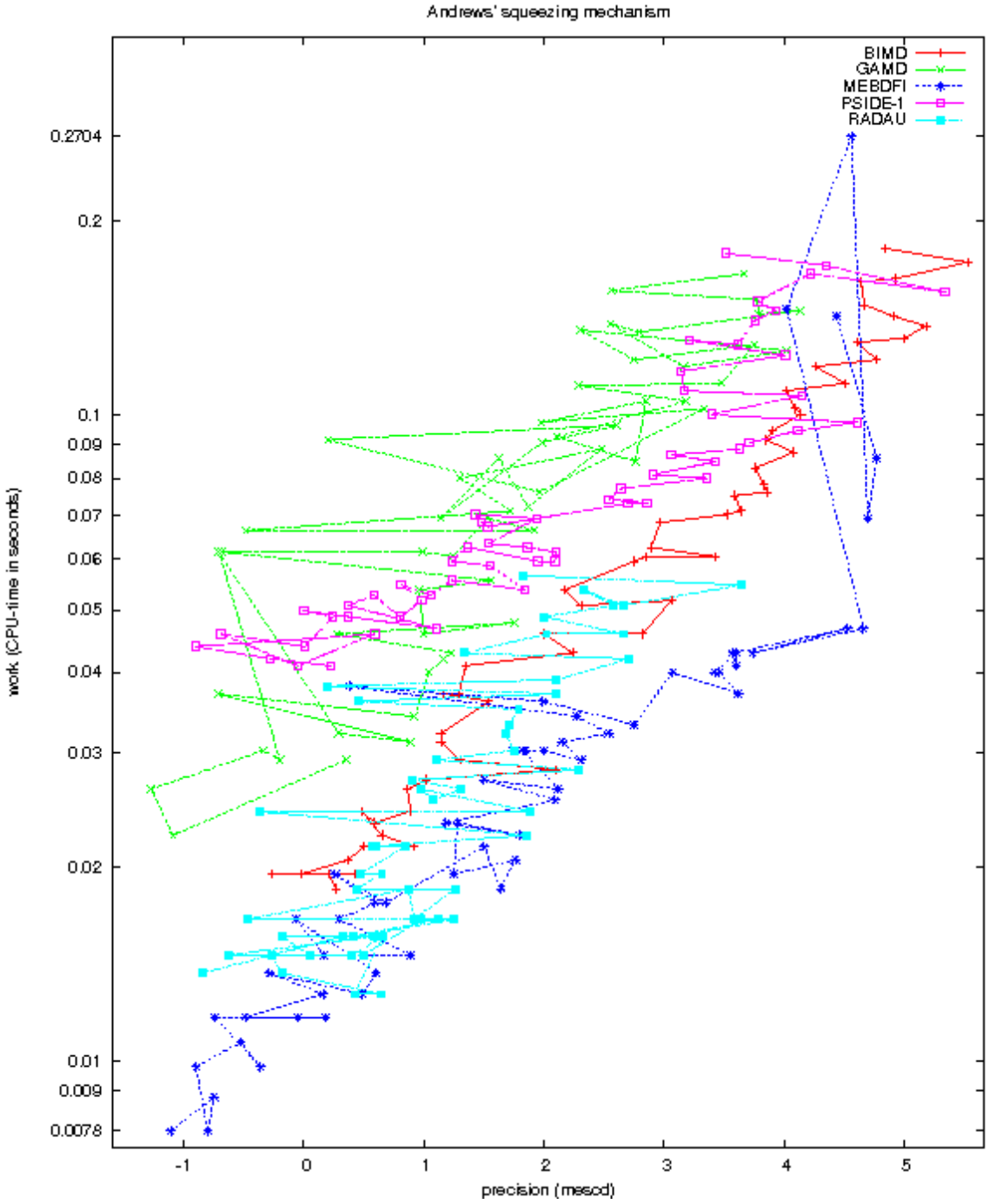

FIGURE II.13.4: *Work-precision diagram (mescd versus CPU-time).*

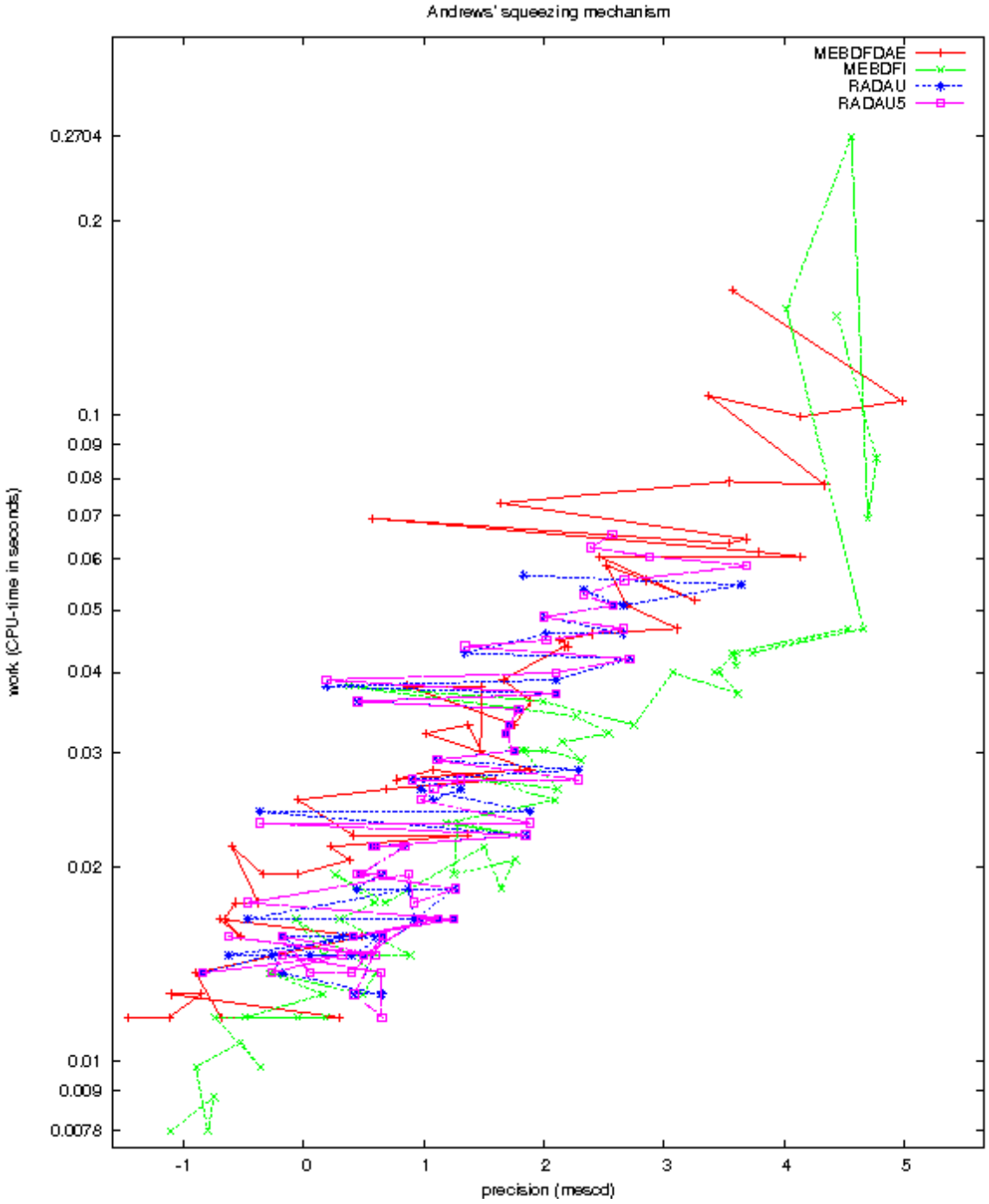

FIGURE II.13.5: *Work-precision diagram (mescd versus CPU-time).*

14 Transistor amplifier

14.1 General information

The problem is a stiff DAE of index 1 consisting of 8 equations P. Rentrop has received it from K. Glashoff & H.J. Oberle and has documented it in [RRS89]. The formulation presented here has been taken from [HLR89]. The parallel-IVP-algorithm group of CWI contributed this problem to the test set.

The software part of the problem is in the file transamp.f available at [MM08].

14.2 Mathematical description of the problem

The problem is of the form

$$M\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \qquad y(0)=y_0, \quad y'(0)=y'_0,$$

with

$$y \in I\!R^8, \quad 0 \le t \le 0.2.$$

The matrix M is of rank 5 and given by

$$M = \begin{pmatrix} -C_1 & C_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ C_1 & -C_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -C_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -C_3 & C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_3 & -C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -C_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -C_5 & C_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & C_5 & -C_5 \end{pmatrix},$$

and the function f by

$$f(t,y) = \begin{pmatrix} -\frac{U_e(t)}{R_0} + \frac{y_1}{R_0} \\ -\frac{U_b}{R_2} + y_2(\frac{1}{R_1}+\frac{1}{R_2}) - (\alpha-1)g(y_2-y_3) \\ -g(y_2-y_3) + \frac{y_3}{R_3} \\ -\frac{U_b}{R_4} + \frac{y_4}{R_4} + \alpha g(y_2-y_3) \\ -\frac{U_b}{R_6} + y_5(\frac{1}{R_5}+\frac{1}{R_6}) - (\alpha-1)g(y_5-y_6) \\ -g(y_5-y_6) + \frac{y_6}{R_7} \\ -\frac{U_b}{R_8} + \frac{y_7}{R_8} + \alpha g(y_5-y_6) \\ \frac{y_8}{R_9} \end{pmatrix},$$

where g and U_e are auxiliary functions given by

$$g(x) = \beta(e^{\frac{x}{U_F}} - 1) \quad \text{and} \quad U_e(t) = 0.1\sin(200\pi t). \tag{II.14.1}$$

The values of the technical parameters are:

$$\begin{array}{|rcl|} \hline U_b &=& 6, \\ U_F &=& 0.026, \\ \alpha &=& 0.99, \\ \beta &=& 10^{-6}, \\ \hline \end{array} \qquad \begin{array}{|rcll|} \hline R_0 &=& 1000, & \\ R_k &=& 9000 & \text{for } k = 1, \ldots, 9, \\ C_k &=& k \cdot 10^{-6} & \text{for } k = 1, \ldots, 5. \\ \hline \end{array}$$

Consistent initial values at $t = 0$ are

$$y_0 = \begin{pmatrix} 0 \\ U_b/(\frac{R_2}{R_1}+1) \\ U_b/(\frac{R_2}{R_1}+1) \\ U_b \\ U_b/(\frac{R_6}{R_5}+1) \\ U_b/(\frac{R_6}{R_5}+1) \\ U_b \\ 0 \end{pmatrix}, \qquad y_0' = \begin{pmatrix} 51.338775 \\ 51.338775 \\ -U_b/((\frac{R_2}{R_1}+1)(C_2 \cdot R_3)) \\ -24.9757667 \\ -24.9757667 \\ -U_b/((\frac{R_6}{R_5}+1)(C_4 \cdot R_7)) \\ -10.00564453 \\ -10.00564453 \end{pmatrix}.$$

The first, fourth and seventh component of y_0' were determined numerically. All components of y are of index 1.

The definition of the function $g(x)$ in (II.14.1) may cause overflow if $\frac{x}{U_F}$ becomes too large. In the Fortran subroutines `feval` and `jeval` that define the function f and the partial derivatives of f with respect to y, respectively, we set `IERR=-1` if $\frac{x}{U_F} > 300$ to prevent this situation. See page *IV-ix* of the description of the software part of the test set for more details on `IERR`.

14.3 Origin of the problem

The problem originates from electrical circuit analysis. It is a model for the transistor amplifier. The diagram of the circuit is given in Figure II.14.1. Here U_e is the input signal and U_8 is the amplified

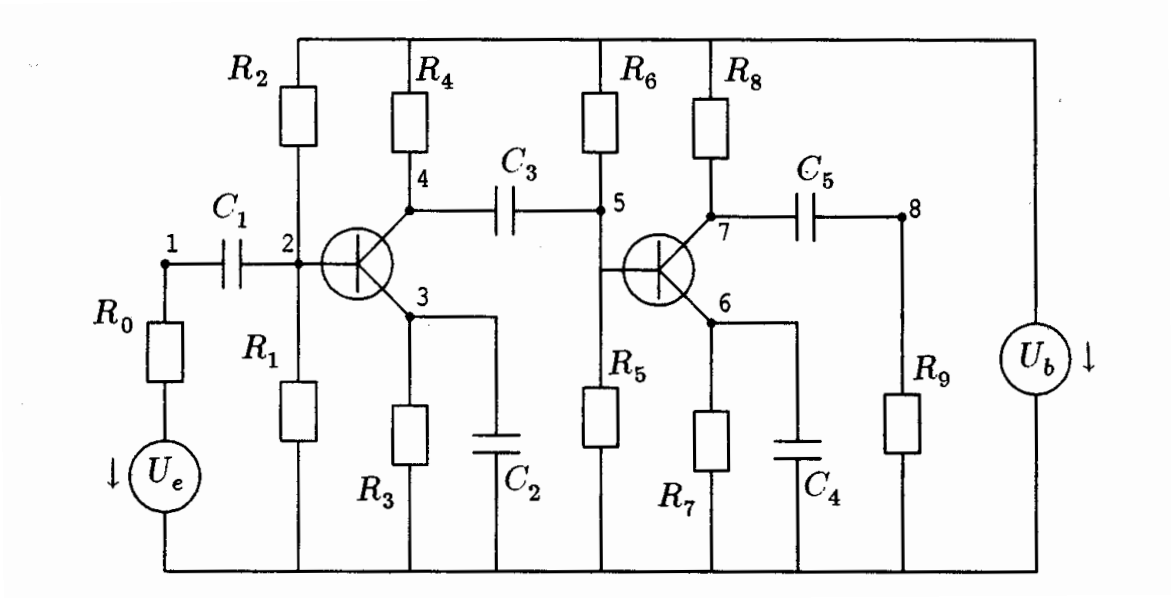

FIGURE II.14.1: *Circuit diagram of Transistor Amplifier (taken from [HLR89]).*

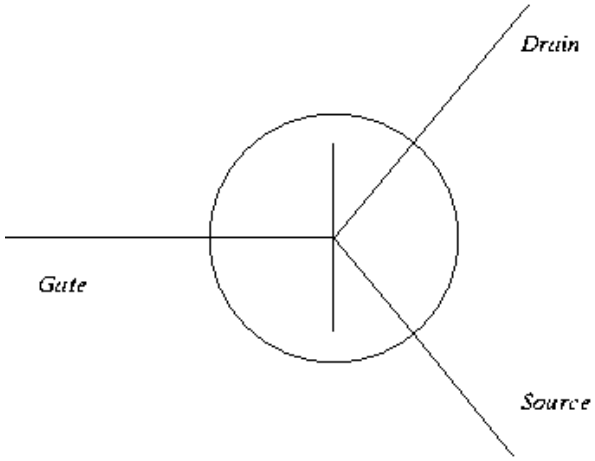

FIGURE II.14.2: *Schematic representation of a transistor.*

output voltage. The circuit contains two transistors of the form depicted in Figure II.14.2. As a simple model for the behavior of the transistors we assume that the currents through the gate, drain and source, which are denoted by I_G, I_D and I_S, respectively, are

$$\begin{aligned} I_G &= (1-\alpha)g(U_G - U_S), \\ I_D &= \alpha g(U_G - U_S), \\ I_S &= g(U_G - U_S), \end{aligned}$$

where U_G and U_S denote the voltage at the gate and source, respectively, and $\alpha = 0.99$. For the function g we take

$$g(U_i - U_j) = \beta(\mathrm{e}^{\frac{U_i - U_j}{U_F}} - 1),$$

where $\beta = 10^{-6}$ and $U_F = 0.026$.

To formulate the governing equations, Kirchoff's Current Law is used in each numbered node. This law states that the total sum of all currents entering a node must be zero. All currents passing through the circuit components can be expressed in terms of the unknown voltages $U_1, \ldots, U_8$. Consider for instance node 1. The current I_{C_1} passing through capacitor C_1 is given by

$$I_{C_1} = \frac{\mathrm{d}}{\mathrm{d}t}(C_1(U_2 - U_1)),$$

and the current I_{R_0} passing through the resistor R_0 by

$$I_{R_0} = \frac{U_e - U_1}{R_0}.$$

Here, the currents are directed towards node 1 if the current is positive. A similar derivation for the

TABLE II.14.1: *Failed runs.*

solver	m	reason
RADAU	$0,\ldots,8,30$	solver cannot handle `IERR=-1`.
RADAU5	$0,\ldots,8$	solver cannot handle `IERR=-1`.

other nodes gives the system:

$$\begin{aligned}
\text{node 1:} \quad & \tfrac{\mathrm{d}}{\mathrm{d}t}(C_1(U_2-U_1)) + \tfrac{U_e(t)}{R_0} - \tfrac{U_1}{R_0} && = 0,\\
\text{node 2:} \quad & \tfrac{\mathrm{d}}{\mathrm{d}t}(C_1(U_1-U_2)) + \tfrac{U_b}{R_2} - U_2(\tfrac{1}{R_1}+\tfrac{1}{R_2}) + (\alpha-1)g(U_2-U_3) && = 0,\\
\text{node 3:} \quad & -\tfrac{\mathrm{d}}{\mathrm{d}t}(C_2U_3) + g(U_2-U_3) - \tfrac{U_3}{R_3} && = 0,\\
\text{node 4:} \quad & -\tfrac{\mathrm{d}}{\mathrm{d}t}(C_3(U_4-U_5)) + \tfrac{U_b}{R_4} - \tfrac{U_4}{R_4} - \alpha g(U_2-U_3) && = 0,\\
\text{node 5:} \quad & \tfrac{\mathrm{d}}{\mathrm{d}t}(C_3(U_4-U_5)) + \tfrac{U_b}{R_6} - U_5(\tfrac{1}{R_5}+\tfrac{1}{R_6}) + (\alpha-1)g(U_5-U_6) && = 0,\\
\text{node 6:} \quad & -\tfrac{\mathrm{d}}{\mathrm{d}t}(C_4U_6) + g(U_5-U_6) - \tfrac{U_6}{R_7} && = 0,\\
\text{node 7:} \quad & -\tfrac{\mathrm{d}}{\mathrm{d}t}(C_5(U_7-U_8)) + \tfrac{U_b}{R_8} - \tfrac{U_7}{R_8} - \alpha g(U_5-U_6) && = 0.\\
\text{node 8:} \quad & -\tfrac{\mathrm{d}}{\mathrm{d}t}(C_5(U_7-U_8)) + \tfrac{U_8}{R_9} && = 0,
\end{aligned}$$

The input signal $U_e(t)$ is

$$U_e(t) = 0.1\sin(200\pi t).$$

To arrive at the mathematical formulation of the preceding subsection, one just has to identify U_i with y_i.

From the plot of output signal $U_8 = y(8)$ in Figure II.14.2 we see that the amplitude of the input signal U_e is indeed amplified.

14.4 Numerical solution of the problem

Tables II.14.2–II.14.3 and Figures II.14.3–II.14.4 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively. The reference solution was computed on the Cray C90, using PSIDE with Cray double precision and atol = rtol = 10^{-14}. For the work-precision diagrams, we used: rtol = $10^{-(4+m/8)}$, $m = 0,1,\ldots,40$; atol = rtol; h0 = $10^{-2}\cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

The failed runs are in Table II.14.1; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

References

[HLR89] E. Hairer, C. Lubich, and M. Roche. *The Numerical Solution of Differential-Algebraic Systems by Runge–Kutta Methods.* Lecture Notes in Mathematics 1409. Springer-Verlag, 1989.

TABLE II.14.2: *Reference solution at the end of the integration interval.*

y_1	$-0.5562145012262709 \cdot 10^{-2}$	y_5	$0.2704617865010554 \cdot 10$
y_2	$0.3006522471903042 \cdot 10$	y_6	$0.2761837778393145 \cdot 10$
y_3	$0.2849958788608128 \cdot 10$	y_7	$0.4770927631616772 \cdot 10$
y_4	$0.2926422536206241 \cdot 10$	y_8	$0.1236995868091548 \cdot 10$

TABLE II.14.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	5.85	5.63	466	408	8423	408	466	0.0322
	10^{-7}	10^{-7}	10^{-9}	8.62	8.34	618	575	19632	575	618	0.0752
DDASSL	10^{-4}	10^{-4}		4.60	3.08	9759	6026	18381	7359		0.1113
	10^{-7}	10^{-7}		7.24	5.49	40810	23859	77402	33678		0.4743
GAMD	10^{-4}	10^{-4}	10^{-6}	6.30	5.83	373	276	17204	276	373	0.0517
	10^{-7}	10^{-7}	10^{-9}	8.58	7.37	374	325	34320	326	374	0.1064
MEBDFI	10^{-4}	10^{-4}	10^{-6}	5.06	4.80	1580	1486	5949	256	256	0.0303
	10^{-7}	10^{-7}	10^{-9}	7.25	6.99	3628	3513	13324	419	419	0.0703
PSIDE-1	10^{-4}	10^{-4}		5.02	4.76	516	362	9742	253	2008	0.0351
	10^{-7}	10^{-7}		7.50	7.23	835	653	21914	419	2724	0.0732
RADAU	10^{-7}	10^{-7}	10^{-9}	7.11	6.83	1775	1551	17582	1541	1775	0.0517

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[RRS89] P. Rentrop, M. Roche, and G. Steinebach. The application of Rosenbrock-Wanner type methods with stepsize control in differential-algebraic equations. *Numer. Math.*, 55:545–563, 1989.

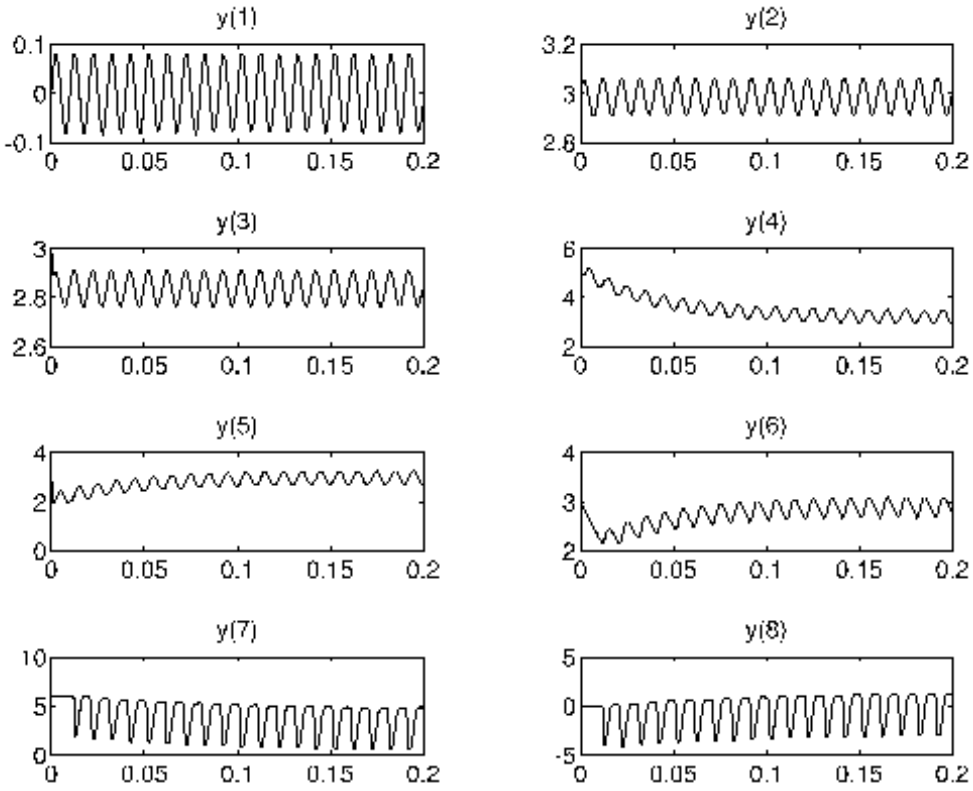

FIGURE II.14.3: *Behavior of the solution over the integration interval.*

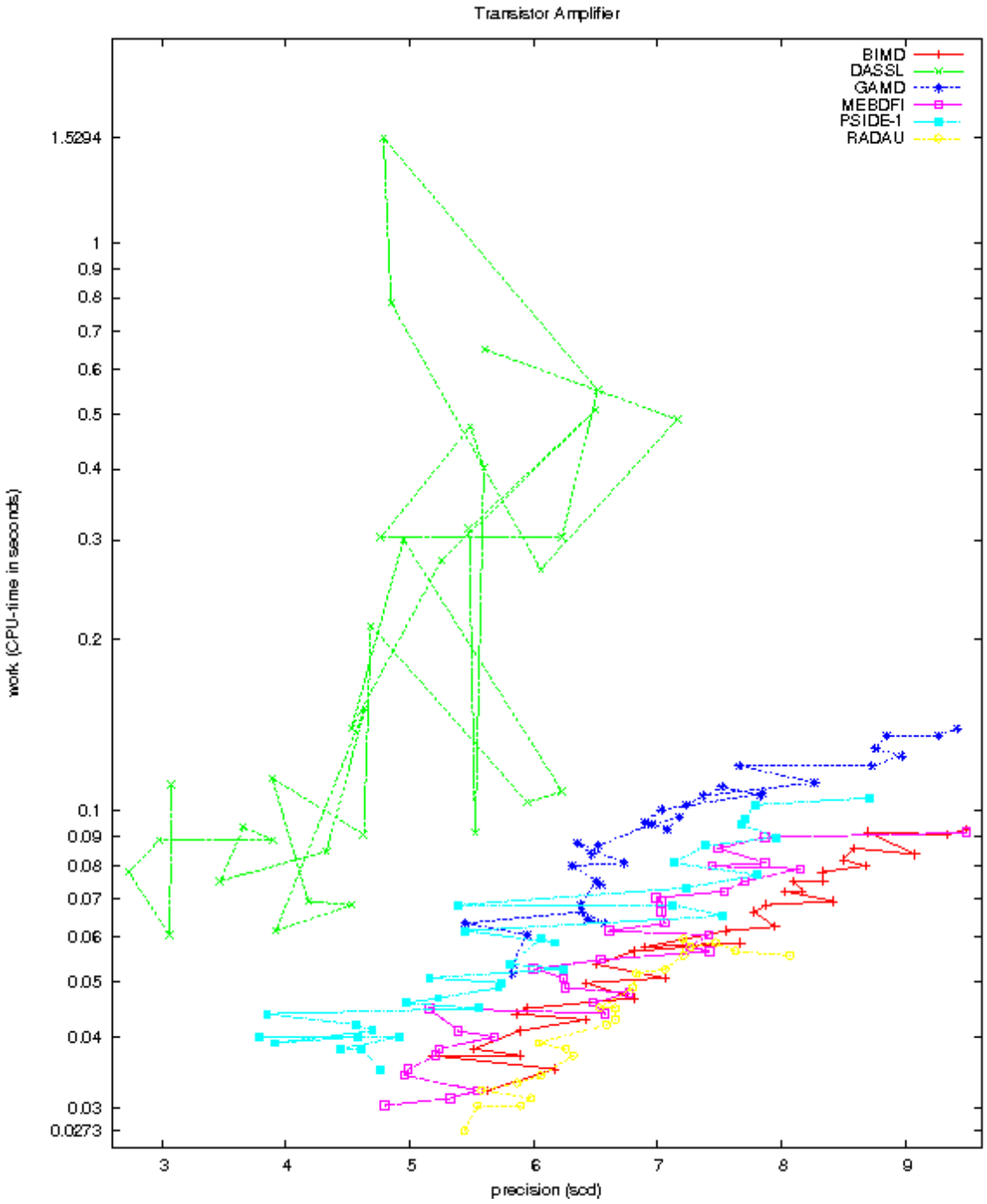

FIGURE II.14.4: *Work-precision diagram (scd versus CPU-time).*

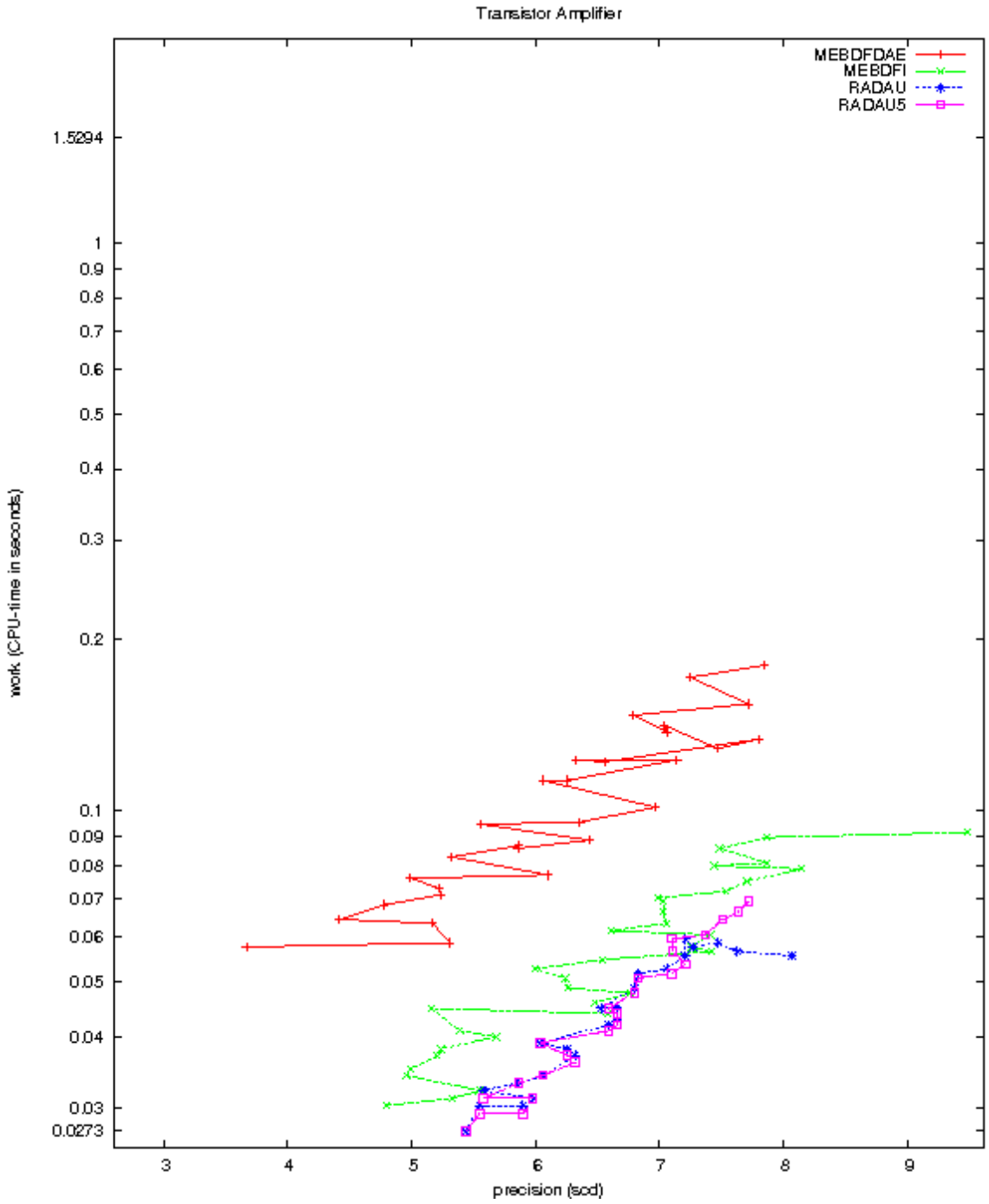

FIGURE II.14.5: *Work-precision diagram (scd versus CPU-time).*

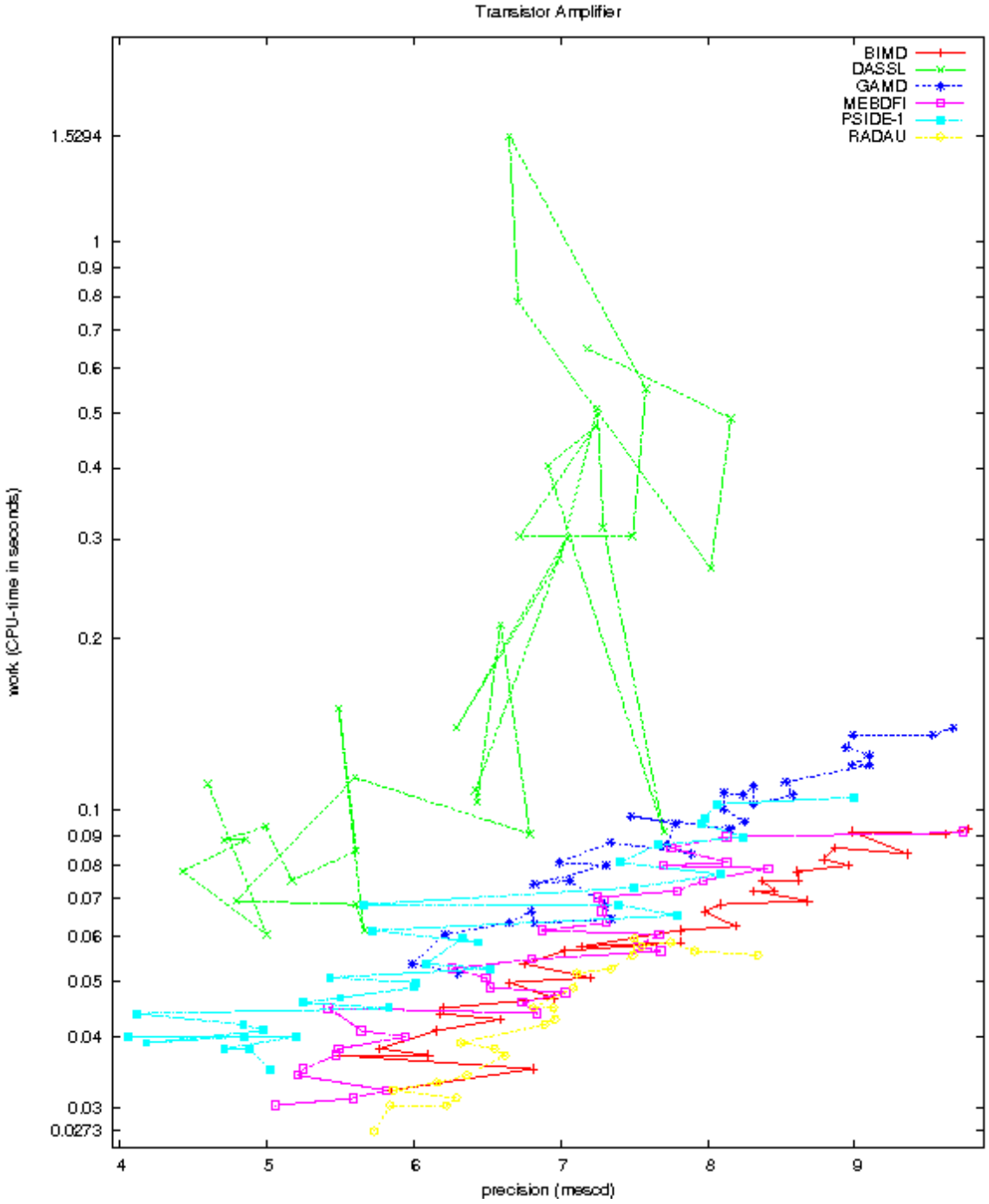

FIGURE II.14.6: *Work-precision diagram (mescd versus CPU-time).*

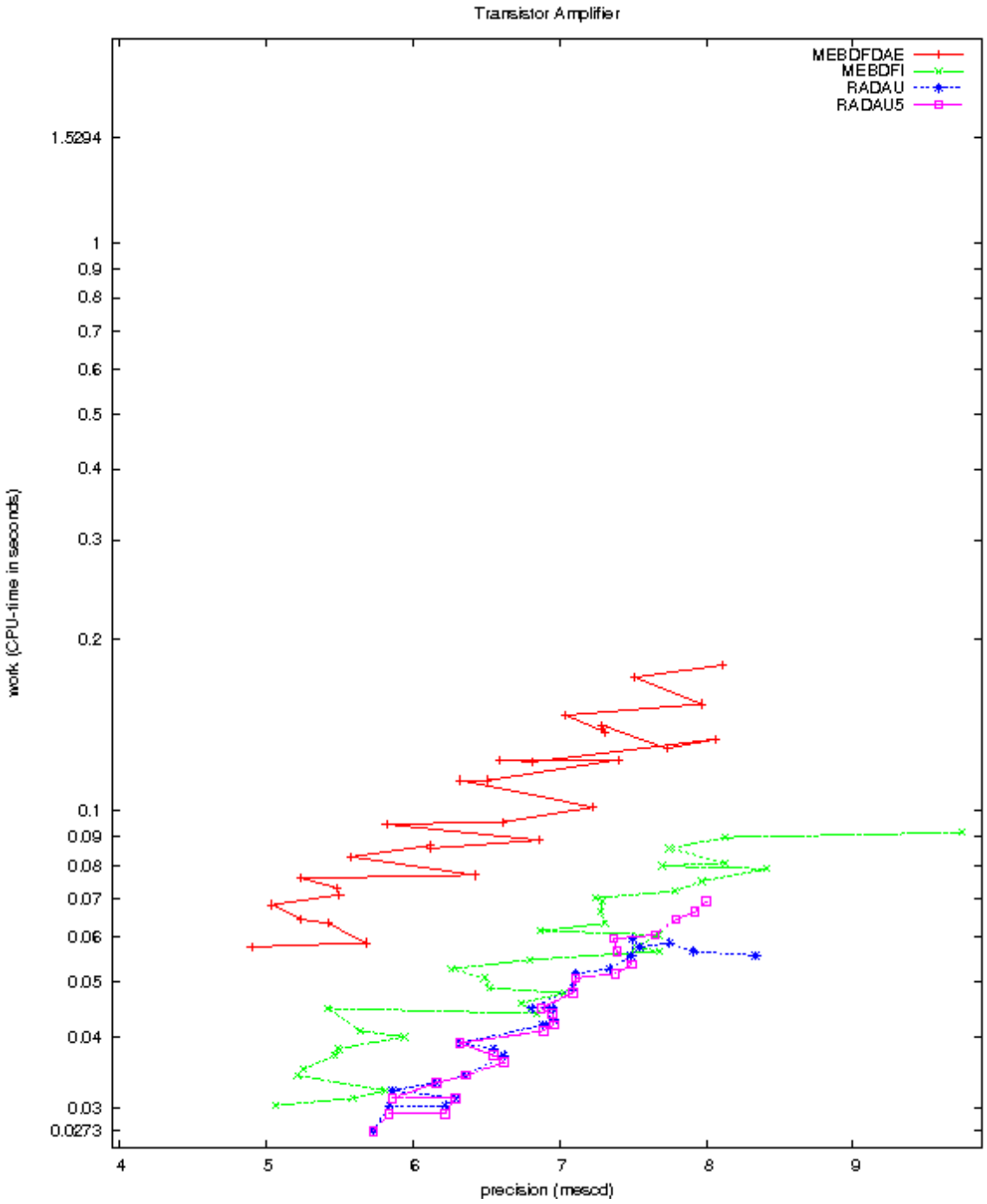

FIGURE II.14.7: *Work-precision diagram (mescd versus CPU-time).*

15 Charge pump

15.1 General information

The problem is a stiff DAE of index 2, consisting of 3 differential and 6 algebraic equations. It has been contributed by Michael Günther, Georg Denk and Uwe Feldmann [GDF95].

The software part of the problem is in the file pump.f available at [MM08].

15.2 Mathematical description

The problem is of the form

$$M\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y(t)), \qquad y(0) = y_0, \quad y'(0) = y'_0,$$

with

$$y \in I\!R^9, \qquad 0 \le t \le 1.2 \cdot 10^{-6}.$$

The 9×9 matrix M is the zero matrix except for the the minor $M_{1..3,1..5}$, that is given by

$$M_{1..3,1..5} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

The function f is defined by

$$f(t,y) = \begin{pmatrix} -y_9 \\ 0 \\ 0 \\ -y_6 + V_{in}(t) \\ y_1 - Q_G(v) \\ y_2 - C_S \cdot y_7 \\ y_3 - Q_S(v) \\ y_4 - C_D \cdot y_8 \\ y_5 - Q_D(v) \end{pmatrix},$$

with $v := (v_1, v_2, v_3) = (y_6, y_6 - y_7, y_6 - y_8)$, $C_D = 0.4 \cdot 10^{-12}$ and $C_S = 1.6 \cdot 10^{-12}$. The functions Q_G, Q_S and Q_D are given by:

1. If $v_1 \le V_{FB} := U_{T0} - \gamma\sqrt{\Phi} - \Phi$, then

$$\begin{aligned} Q_G(v) &= C_{ox}(v_1 - V_{FB}), \\ Q_S(v) &= Q_D(v) = 0, \end{aligned}$$

   with $C_{ox} = 4 \cdot 10^{-12}$, $U_{T0} = 0.2$, $\gamma = 0.035$ and $\Phi = 1.01$.

2. If $v_1 > V_{FB}$ and $v_2 \le U_{TE} := U_{T0} + \gamma(\sqrt{\Phi - U_{BS}} - \sqrt{\Phi})$, then

$$\begin{aligned} Q_G(v) &= C_{ox}\gamma\left(\sqrt{(\gamma/2)^2 + v_1 - V_{FB}} - \gamma/2\right), \\ Q_S(v) &= Q_D(v) = 0. \end{aligned}$$

3. If $v_1 > V_{FB}$ and $v_2 > U_{TE}$, then

$$\begin{aligned} Q_G(v) &= C_{ox}\left(\frac{2}{3}(U_{GDT} + U_{GST} - \frac{U_{GDT}U_{GST}}{U_{GDT} + U_{GST}}) + \gamma\sqrt{\Phi - U_{BS}}\right), \\ Q_S(v) &= Q_D(v) = -\frac{1}{2}\left(Q_G - C_{ox}\gamma\sqrt{\Phi - U_{BS}}\right). \end{aligned}$$

Here, U_{BS}, U_{GST} and U_{GDT} are given by

$$\begin{aligned} U_{BS} &= v_2 - v_1, \\ U_{GST} &= v_2 - U_{TE}, \\ U_{GDT} &= \begin{cases} v_3 - U_{TE} & \text{for} \quad v_3 > U_{TE}, \\ 0 & \text{for} \quad v_3 \le U_{TE}. \end{cases} \end{aligned}$$

The function $V_{in}(t)$ is defined using $\tau = (10^9 \cdot t) \bmod 120$ by

$$V_{in}(t) = \begin{cases} 0 & \text{if} \quad \tau < 50, \\ 20(\tau - 50) & \text{if} \quad 50 \le \tau < 60, \\ 20 & \text{if} \quad 60 \le \tau < 110, \\ 20(120 - \tau) & \text{if} \quad \tau \ge 110. \end{cases}$$

This means that the function f has discontinuities in its derivative at $\tau = 50, 60, 90, 110, 120$.

Consistent initial values are

$$y_0 = (Q_G(0,0,0), 0, Q_S(0,0,0), 0, Q_D(0,0,0), 0, 0, 0, 0)^T \quad \text{and} \quad y_0' = (0,0,0,0,0,0,0,0,0)^T.$$

The index of the first eight variables is 1, whereas the index of y_9 is 2.

15.3 Origin of the problem

The Charge-pump circuit shown in Figure II.15.1 consists of two capacitors and an n-channel MOS-transistor. The nodes gate, source, gate, and drain of the MOS-transistor are connected with the nodes 1, 2, 3, and Ground, respectively. In formulating the circuit equations, the transistor is replaced by four non-linear current sources in each of the connecting branches. They model the transistor.

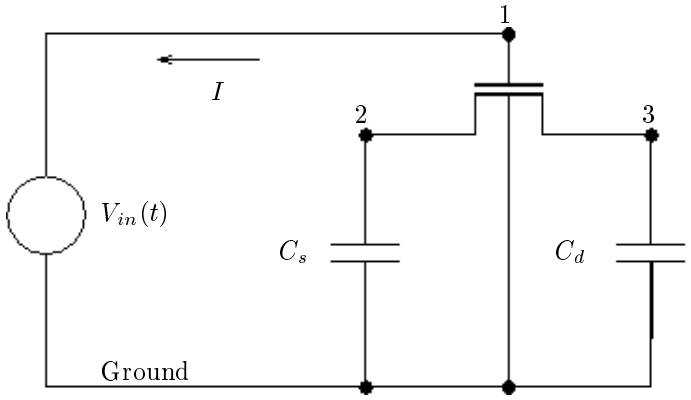

FIGURE II.15.1: *Circuit diagram of Charge-pump circuit (taken from [GDF95])*

After inserting the transistor model in the circuit, we get the final circuit, which can be obtained from the circuit in Figure II.15.1 by applying the following changes:

- Remove the transistor and replace it by a solid line between the nodes 2 and 3. The point where the lines 2–3 and 1–Ground cross each other becomes a node, which will be denoted by T.
- Add current sources between nodes 1 and T, between 2 and T and between 3 and T. There should also be a current source between the ground and node T, but as the node Ground does not enter the circuit equations, it will not be discussed. The currents produced by these sources are written as the derivatives of charges: current from 1 to T: Q_G', from T to 2: Q_S' and from T to 3: Q_D'. Here, the functions Q_G, Q_S and Q_D depend on the voltage drops U_1, $U_1 - U_2$ and $U_1 - U_3$, where U_i denotes the potential in node i.

The unknowns in the circuit are given by:

- The charges produced by the current sources: Y_{T1}, Y_{T2}, Y_{T3}. They are aliases for respectively Q_G, Q_S and Q_D. Consequently, Y'_{Ti} is the current between node T and node i.
- The charges Y_S and Y_D in the capacitors C_S and C_D.
- Potentials in nodes 1 to 3: U_1, U_2, U_3.
- The current through the voltage source $V_{in}(t)$: I.

In terms of these physical variables, the vector y introduced earlier reads

$$y = (Y_{T1}, Y_S, Y_{T2}, Y_D, Y_{T3}, U_1, U_2, U_3, I)^T.$$

Now, the following equations hold:

$$\begin{aligned} Y'_{T1} &= -I, \\ Y'_S + Y'_{T2} &= 0, \\ Y'_D + Y'_{T3} &= 0, \\ U_1 &= V_{in}(t). \end{aligned}$$

The charges depend on the potentials and are given by

$$\begin{aligned} Y_{T1} &= Q_G(U_1, U_1 - U_2, U_1 - U_3), \\ Y_S &= C_S \cdot U_2, \\ Y_{T2} &= Q_S(U_1, U_1 - U_2, U_1 - U_3), \\ Y_D &= C_D \cdot U_3, \\ Y_{T3} &= Q_D(U_1, U_1 - U_2, U_1 - U_3). \end{aligned}$$

The functions Q_G, Q_S and Q_D are given in the previous section.
Remark: the potential U_1 is known. Here, it is treated as an unknown in order to keep the formulation general and leaving open the possibility to extend the circuit. In addition, removing U_1 by hand contradicts a Computer Aided Design (CAD) approach in circuit simulation.

15.4 Numerical solution of the problem

The various components differ enormously in magnitude. Therefore, the absolute and relative input tolerances atol and rtol were chosen to be component-dependent. Furthermore, we neglect the index 2 variable y_9 in the error control of DASSL. This leads to the following input tolerances:

$$\begin{aligned} \text{atol}(i) &= \text{Tol} \cdot 10^{-6} \quad \text{for} \quad i = 1, \ldots, 5, \\ \text{atol}(i) &= \text{Tol} \quad \text{for} \quad i = 6, \ldots, 8, \\ \text{rtol}(i) &= \text{Tol} \quad \text{for} \quad i = 1, \ldots, 8, \\ \text{atol}(9) = \text{rtol}(9) &= 1000 \quad \text{for} \quad \text{DASSL}, \\ \text{atol}(9) = \text{rtol}(9) &= \text{Tol} \quad \text{for} \quad \text{other solvers}. \end{aligned}$$

The reference solution was computed using quadruple precision GAMD on an Alphaserver DS20E, with a 667 MHz EV67 processor, atol = rtol = 10^{-18}, $h_0 = 10^{-37}$.

Table II.15.1 and Figures II.15.3–II.15.4 present the run characteristics and the work-precision diagram, respectively. For the computation of the number of significant correct digits (scd), only the first component is taken into account. The second up to eighth component are ignored because these components are zero in the true solution; the ninth component is neglected because it was excluded

TABLE II.15.1: *Run characteristics.*

solver	Tol	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-5}	7.34	16.00	711	454	8827	454	711	0.0478
	10^{-7}	8.65	16.00	1125	688	15367	688	1125	0.0820
DDASSL	10^{-1}	0.93	0.14	447	438	604	369		0.0088
	10^{-3}	5.42	16.00	983	833	1659	853		0.0215
	10^{-5}	6.71	3.43	1737	1487	2903	1309		0.0361
	10^{-7}	6.09	3.32	3059	2587	4945	2058		0.0595
GAMD	10^{-1}	2.11	1.51	320	200	3735	200	320	0.0166
	10^{-3}	2.85	2.69	350	220	4786	220	350	0.0205
	10^{-5}	4.78	5.12	620	370	14890	320	570	0.0547
	10^{-7}	4.94	4.75	870	510	22340	410	770	0.0791
PSIDE-1	10^{-1}	1.17	0.37	938	839	9843	140	3752	0.0742
	10^{-5}	2.64	4.47	1366	1068	13424	160	5424	0.1005
	10^{-7}	9.05	16.00	2425	1555	24331	300	9616	0.1835

from DASSL's error control. For the mescd we consider all the components. The first component of the reference solution equals $0.1262800429876759 \cdot 10^{-12}$ at the end of the integration interval. We remark that the magnitude of this component is at most 10^{-10}. For the work-precision diagram, we used: Tol $= 10^{-(1+m/2)}$, $m = 0, 1, \ldots, 14$; h0 $= 10^{-6} \cdot$ Tol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. From Table II.15.1 and Figure II.15.3 we see that the numerical solution computed by DASSL results for some rather large values of Tol in an scd value of 15.4, which equals the accuracy of the reference solution.

Figure II.15.2 shows the behavior of the solution over the integration interval. Only the last four components have been plotted, since they are the physically important quantities. The other five components refer to charge flows inside the transistor, which are quantities the user is not interested in. These components have a similar behavior as the components 6, 7 and 8, but their magnitude is at most 10^{-10}.

The failed runs are in Table II.15.2; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

References

[GDF95] M. Günther, G. Denk, and U. Feldmann. How models for MOS transistors reflect charge distribution effects. Technical Report 1745, Technische Hochschule Darmstadt, Fachbereich Mathematik, Darmstadt, 1995.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

TABLE II.15.2: *Failed runs.*

solver	m	reason
BIMD	0	floating invalid
BIMD	4	too many consecutive Newton failures
BIMD	$3, 5, 7$	floating divide by zero
DASSL	2	error test failed repeatedly
DASSL	$4, 7$	floating overflow
DASSL	14	corrector failed to converge repeatedly
MEBDFDAE	$0, 1, \ldots, 14$	stepsize too small
MEBDFI	$0, 1, \ldots, 10$	floating invalid
MEBDFI	$11, 12, 13, 14$	stepsize too small
PSIDE-1	$4, 13, 14$	stepsize too small
RADAU	$0, 1, \ldots, 14$	stepsize too small
RADAU5	$0, 1, \ldots, 10$	floating invalid
RADAU5	$11, \ldots, 14$	stepsize too small

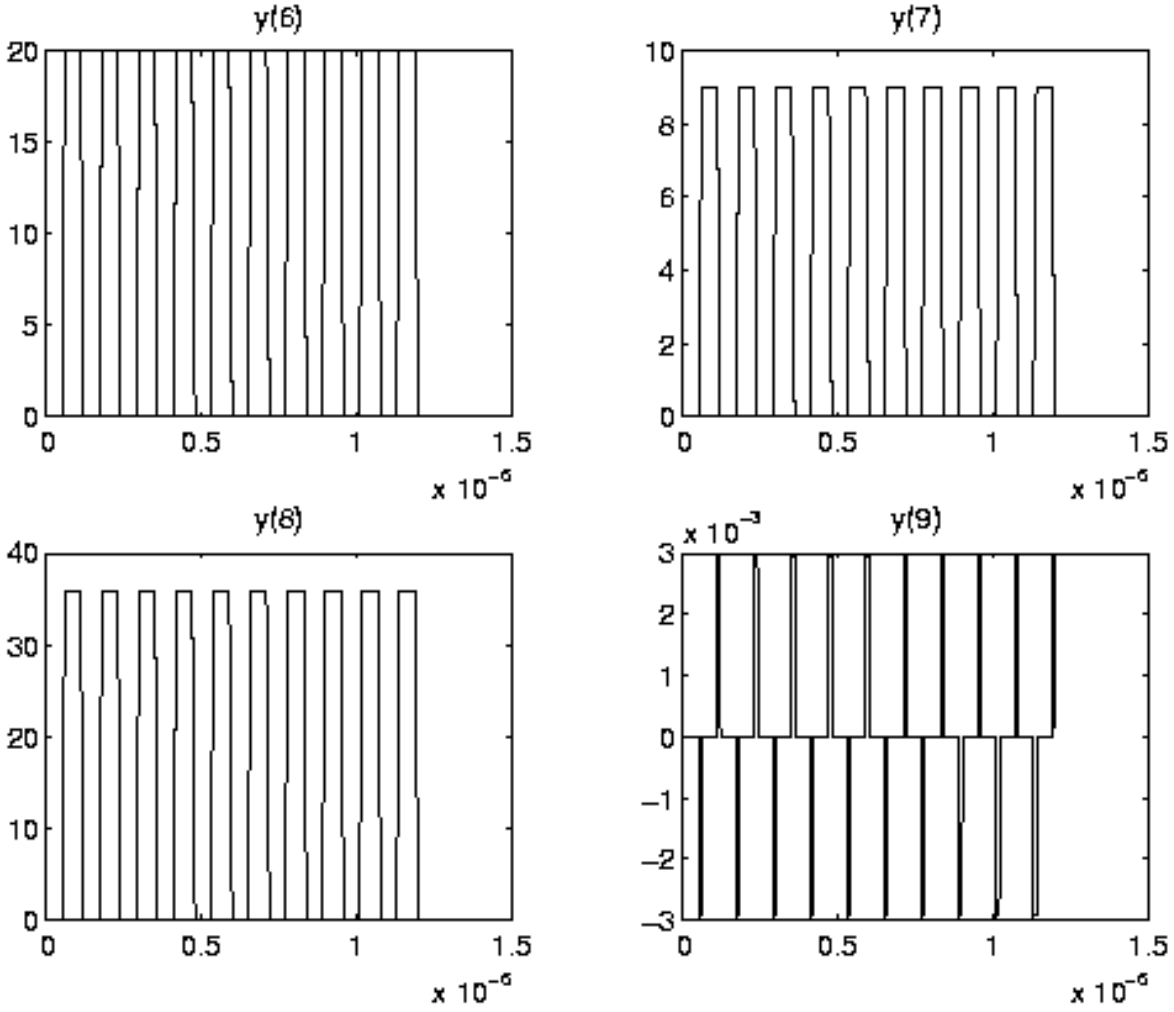

FIGURE II.15.2: *Behavior of the solution over the integration interval.*

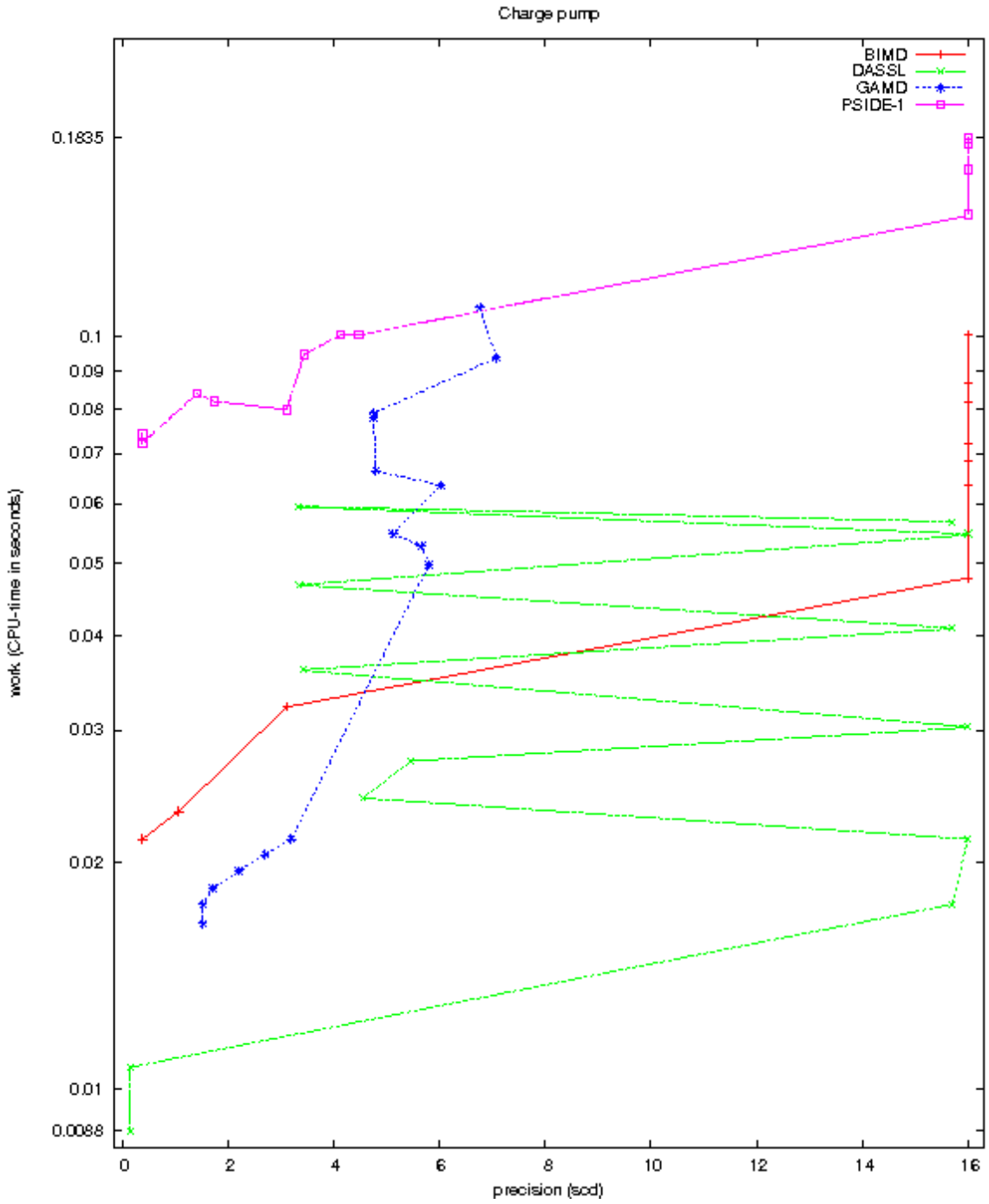

FIGURE II.15.3: *Work-precision diagram (scd versus CPU-time).*

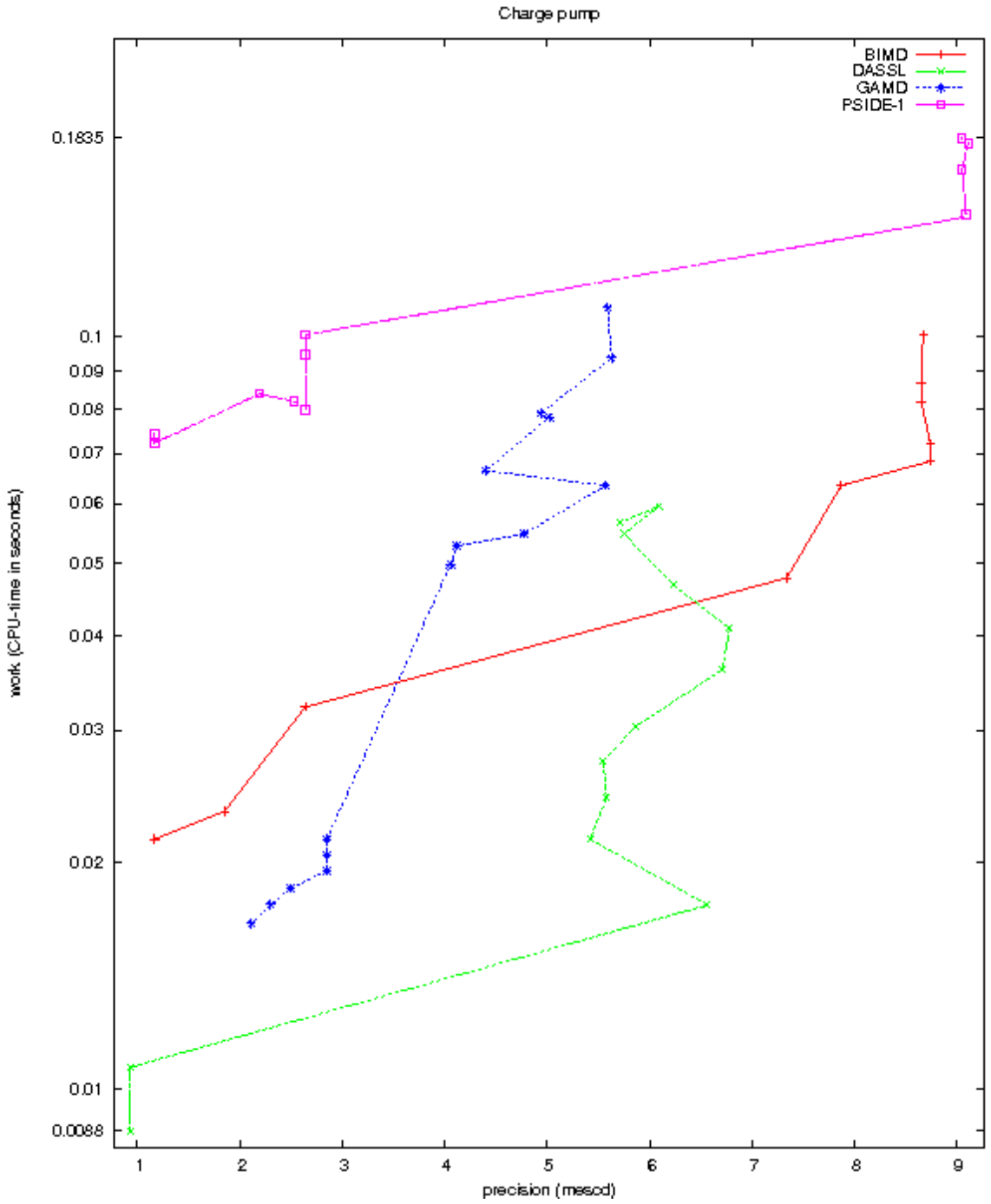

FIGURE II.15.4: *Work-precision diagram (mescd versus CPU-time).*

16 Two bit adding unit

16.1 General Information

The problem is a stiff DAE of index 1, consisting of 175 differential equations and 175 algebraic equations. It has been contributed by M. Günther [Gün95, Gün98].

The software part of the problem is in the file tba.f available at [MM08].

16.2 Mathematical description of the problem

The problem is of the form

$$\begin{aligned} \frac{\mathrm{d}y}{\mathrm{d}t} &= f(t,x), \\ 0 &= y - g(x), \end{aligned} \tag{II.16.1}$$

where

$$y, x \in I\!R^{175}, \quad f : I\!R^{351} \to I\!R^{350}, \; g : I\!R^{350} \to I\!R^{350}, \quad 0 \le t \le 320, \quad y(0) = y_0, \; x(0) = x_0.$$

Since the functions $f(t,x)$ and $g(x)$ and the (consistent) initial values y_0 and x_0 are too voluminous to be printed here, we refer to the subroutines `feval` and `init` for their definitions. The function f has discontinuities in its derivative at $t = 0, 5, 10, \ldots, 320$. The index of the components of x and y equals 1.

The function f contains several square roots. It is clear that the function can not be evaluated if one of the arguments of one of these square roots becomes negative. To prevent this situation, we set `IERR=-1` in the Fortran subroutine that defines f if this happens. See page *IV-ix* of the description of the software part of the test set for more details on `IERR`.

16.3 Origin of the problem

The two bit adding unit computes the sum of two base-2 numbers (each two digits long) and a carry bit. These numbers are fed into the circuit in the form of input signals. As a result the circuit gives their sum coded as three output signals.

The two bit adding unit circuit is a digital circuit. These circuits are used to compute boolean expressions. This is accomplished by associating voltages with boolean variables. By convention the boolean is true if the voltage exceeds $2V$, and false if it is lower than $0.8V$. In between the boolean is undefined. Using CMOS technique, however, sharper bounds are possible for the representation of booleans.

Digital circuits that compute elementary logical operations are called gates. An example of a gate is the NAND gate of test problem 9. This circuit is used to compute the logical expression $\neg(V_1 \wedge V_2)$, where V_1 and V_2 are the booleans that are fed into the circuit as input signals.

The two bit adding unit is depicted in Figure II.16.1. In this figure the symbols '&', '≥ 1' and a little white circle respectively stand for the AND, OR and NOT gate. A number of input signals and output signals enter and leave the circuit. Each signal is described by a time-dependent voltage and the boolean it represents. For these two quantities we shall use one symbol: the symbol of this boolean variable. Which one of the two quantities is meant by the symbol, is always clear from the context. With this convention, the input signals are referred to by the boolean variable they represent.

The circuit is designed to perform the addition

$$A_1 \; A_0 + B_1 \; B_0 + C_{in} = C \; S_1 \; S_0.$$

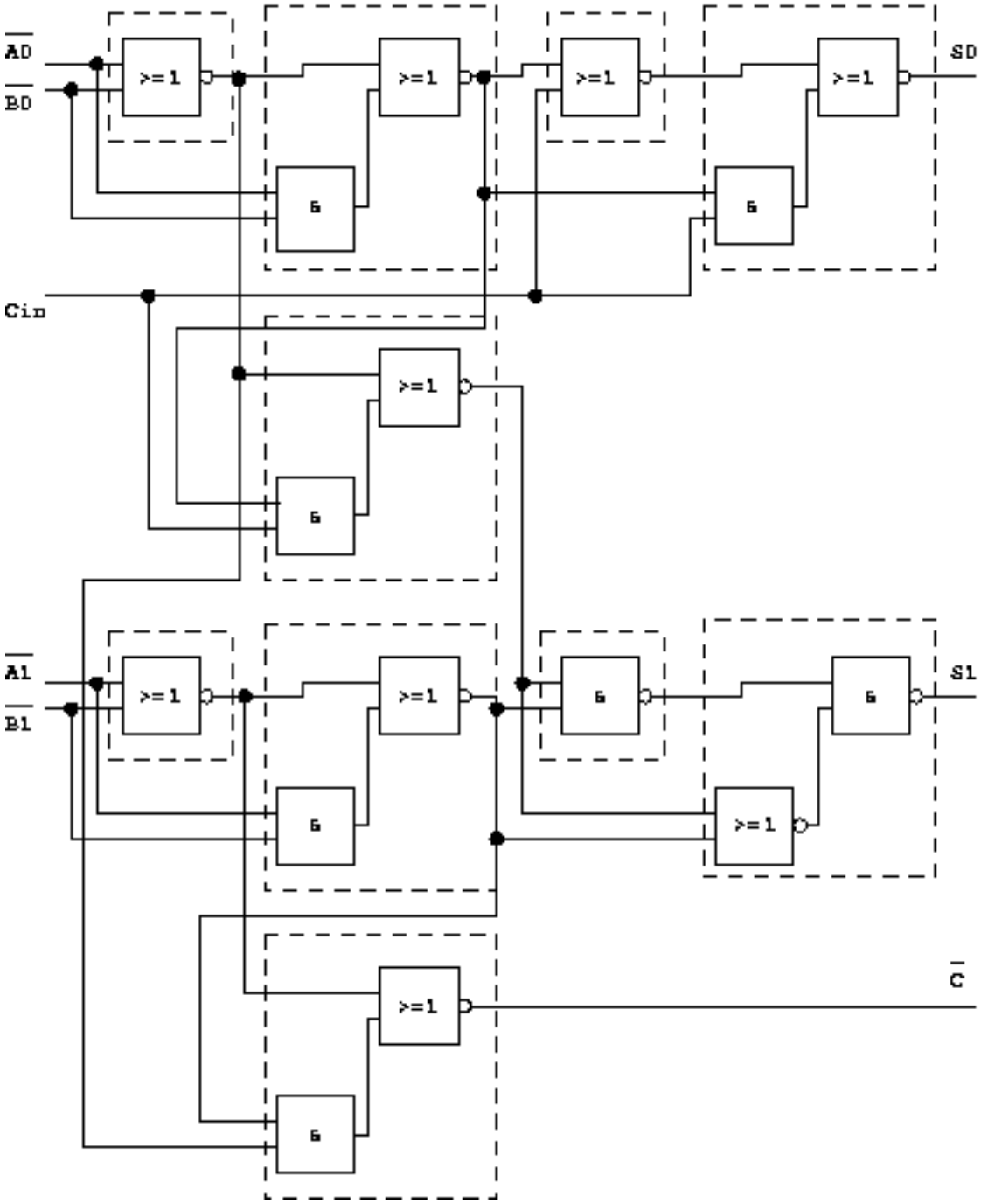

FIGURE II.16.1: *Circuit diagram of the two bit adder (taken from [Gün95]).*

The input signals representing the two numbers and the carry bit C_{in} are fed into the circuit at the nodes indicated by $\overline{A0}$, $\overline{A1}$, $\overline{B0}$, $\overline{B1}$ and Cin. Here, a bar denotes the logical inversion. The output signals are delivered by the nodes indicated by S0, S1 and $\overline{C}$.

In Figure II.16.1, a number of boxes are drawn using dashed lines. Each of them represents one

TABLE II.16.1: *Characteristics of the gates that occur in the two bit adding unit.*

Name	logical expression	# nodes	# times
NOR	$\neg(V_1 \vee V_2)$	$3 \cdot 4 + 1 = 13$	3
NAND	$\neg(V_1 \wedge V_2)$	$3 \cdot 4 + 2 = 14$	1
ANDOI	$\neg(V_1 \vee (V_2 \wedge V_3))$	$4 \cdot 4 + 2 = 18$	5
ORANI	$\neg(V_1 \wedge (V_2 \vee V_3))$	$4 \cdot 4 + 2 = 18$	1

of the following gates: the NOR (first box to the left in the top-row), the ORANI gate (the box besides S_1), the NAND (the box besides the ORANI gate) and the ANDOI(the box at the bottom). The circuit diagram of the NAND-gate is given in test problem 9. For the circuit diagrams of the NOR, ANDOI and ORANI gate see Figures II.16.2, II.16.3 and II.16.4. What logical expressions they compute, is listed in Table II.16.1. The fourth column in this table lists the number of times the gate occurs in the big circuit. The third column tabulates the number of nodes in the gate. These nodes consist of two types. The first type of nodes consists of the internal nodes of the transistors due to the MOS transistor model of Shichmann and Hodges [SH68]. Each transistor has four internal nodes that are also the links between transistor and the rest of the circuit. The second type of nodes comprises the usual nodes that are used to link circuit components together. These nodes are indicated by a number placed inside a square. To prevent any misunderstanding, we remark that the big dots in Figures II.16.2–II.16.4 do not represent nodes.

The connection of a gate with the rest of the circuit consists of the input nodes and the output node of the gate. The input signals enter the gate at the nodes with symbol V_1, V_2 and V_3. The output signal leaves the gate from one of the numbered nodes. To ensure stability of the circuit, such an output node is always connected to a capacitance (we refer to the Fortran driver: `CLOAD` denoting the value of a load capacitance for the logical gates, and `COUT` for the output nodes S_0, S_1 and $\overline{C}$). Finally, three enhancement transistors are coupled with the ANDOI gate at the bottom for a correct treatment of C_{in}. This yields 12 internal nodes and two additional nodes, because the three transistors are coupled in series. Counting all nodes we have $3 \cdot 13 + 1 \cdot 14 + 5 \cdot 18 + 1 \cdot 18 + 14 = 175$ nodes.

Applying Kirchoff's law to all nodes yields a system of 175 equations. This system is an integral form DAE of the special form

$$A \cdot \dot{q}(V) = f(t, V).$$

The function q is a generally nonlinear function of node potentials V, which describes the charges stored in all charge storing elements [GDF96]. Assembling the charge flow at each node by an incidence matrix A, the dynamic part $A \cdot \dot{q}(V)$ equals the contribution of static currents denoted by $f(t, V)$. If all load capacitances at the output nodes are nonzero, then the integral form DAE has differential index 0. If only one of the load capacitances equals zero, the generalized capacitance matrix $A \cdot \partial q(V)/\partial V$ is singular, yielding a system of differential index 1. This shows the regularization effects by applying additional capacitances. Here, we use `CLOAD=0` and `COUT=2.0`.

To make this problem suitable for the solvers used in this test set, the variable $Q = A \cdot q(V)$ of assembled charges is introduced leading to

$$\begin{aligned} \dot{Q} &= f(t, V), \\ 0 &= Q - Aq(V). \end{aligned}$$

This transformation of the integral form DAE into a linearly implicit system raises the differential index by one. However, in the case of singular load capacitances, no higher index effects are detected in the sense of an appropriate perturbation index [Gün98].

Some of the 175 variables have a special meaning. These are the voltage variables of the nodes

that deliver the output signals. The output signals S_0, S_1 and $\overline{C}$ are given by the variables x_{49}, x_{130} and x_{148}, respectively. Only these variables are of interest to the engineer.

In the next section we shall see the two bit adder in operation. Every 10 units of time the addition

$$A_1\ A_0 + B_1\ B_0 + C_{in} = C\ S_1\ S_0,$$

is carried out. The numbers that are added are represented by the input signals depicted in Figure II.16.5. The outcome of the addition is represented by output signals given in Figure II.16.6. Often the output signals need time to adjust to changes in the input signal. Therefore, only during certain periods the sum is correctly represented by the output signals. The two bit adding unit has been designed in such a way that after each 10 units of time the output signal represents the sum correctly.

To see the two bit adding unit performing an addition let us see what happens at $t = 200$. Then the input signals read:

$$\overline{A}_0 = 0,\ \overline{A}_1 = 1,\ \overline{B}_0 = 0, \overline{B}_1 = 0,\ C_{in} = 1,$$

and the output signals are

$$S_0 = 1,\ S_1 = 0,\ \overline{C} = 0.$$

Recall, that a bar denotes the logical inverse. Clearly, the addition 01+11+1=101 has been carried out.

16.4 Numerical solution of the problem

M. Günther provided the source code that defines the problem.

Table II.22.2 lists the voltages of the output signals in the reference solution. For the complete reference solution at $t = 320$ we refer to subroutine `solut`. Since these components refer to the output

TABLE II.16.2: *Value at the end of the integration interval of the components of the reference solution that correspond to the output signals.*

x_{49}	0.2040419147264534
x_{130}	$0.4997238455712048 \cdot 10$
x_{148}	0.2038985905095614

signals S_0, S_1 and $\overline{C}$, they are the physically relevant quantities.

Table II.16.4 and Figures II.16.6–II.16.10 present the run characteristics, the behavior of the output signals over the integration interval and the work-precision diagram, respectively. In computing the scd values, only x_{49}, x_{130} and x_{148} were considered, since they refer to the physically important quantities.

The reference solution was computed using RADAU5 without restarts in the discontinuities in time of the derivative of the problem defining function f, with rtol = atol = 10^{-5} and h0 = $4 \cdot 10^{-5}$.

For the work-precision diagram, we used: rtol = $10^{-(2+m/8)}$, $m = 0, 1, \ldots, 32$; atol = rtol; h0 = $10 \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5. The failed runs are in Table II.16.3; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

Remark

M. Günther also wrote a special purpose solver called CHORAL, which stands for CHarge-ORiented ALgorithm [Gün95, Gün98] for integrating equations of the form

$$\begin{aligned} \frac{dy}{dt} &= f(t, x), \\ 0 &= y - q(x). \end{aligned}$$

TABLE II.16.3: *Failed runs.*

solver	m	reason
BIMD	$27, \ldots, 32$	more than nmax steps are needed
DASSL	$30, 31, 32$	corrector failed to converge repeatedly
GAMD	$25, \ldots, 29$	stepsize too small
MEBDFDAE	$0, 1$	stepsize too small
MEBDFDAE	$2, \ldots, 18$	illegal function call
PSIDE-1	$0, \ldots, 24$	stepsize too small
RADAU	$0, 1, \ldots, 17$	solver cannot handle `IERR=-1`.
RADAU5	$0, 1, \ldots, 17$	solver cannot handle `IERR=-1`.

Most equations occurring in circuit analysis are of this form. In these equations the variables y and x represent respectively (assembled) charges and voltages. CHORAL is based on Rosenbrock-Wanner methods, while the special structure of the problem is exploited. The code eliminates the y variables, reducing the linear algebra work to solving systems of order 175 instead of 350. Correspondingly, a step size prediction and error control based directly on node potentials and currents is offered. For more information see

`http://www.math.uni-wuppertal.de/~guenther`.

References

[GDF96] M. Günther, G. Denk, and U. Feldmann. Modeling and simulating charge sensitive circuits. *Math. Modelling of Systems*, 2:69–81, 1996.

[Gün95] M. Günther. *Ladungsorientierte Rosenbrock–Wanner–Methoden zur numerischen Simulation digitaler Schaltungen.* Number 168 in Fortschritt-Berichte VDI Reihe 20. VDI-Verlag, Düsseldorf, 1995.

[Gün98] M. Günther. Simulating digital circuits numerically – a charge-oriented ROW approach. *Numer. Math.*, 79(2):203–212, 1998.

TABLE II.16.4: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-2}	10^{-2}	10^{-1}	2.11	3.23	836	679	12440	679	836	29.4664
	10^{-4}	10^{-4}	10^{-3}	4.44	6.58	1688	1621	24239	1621	1688	62.4786
DDASSL	10^{-2}	10^{-2}		1.55	2.40	1892	1779	3674	786		21.3276
	10^{-4}	10^{-4}		3.60	4.54	6036	5736	9380	866		27.7379
GAMD	10^{-2}	10^{-2}	10^{-1}	2.72	4.66	735	597	20213	597	735	28.1996
	10^{-4}	10^{-4}	10^{-3}	2.68	3.32	1332	1225	43250	1234	1332	61.4255
MEBDFI	10^{-2}	10^{-2}	10^{-1}	1.96	3.14	2065	1818	194700	533	533	19.1911
	10^{-4}	10^{-4}	10^{-3}	3.01	3.36	5269	4851	363601	982	982	38.9239

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[SH68] H. Shichman and D.A. Hodges. Insulated-gate field-effect transistor switching circuits. *IEEE J. Solid State Circuits*, SC-3:285–289, 1968.

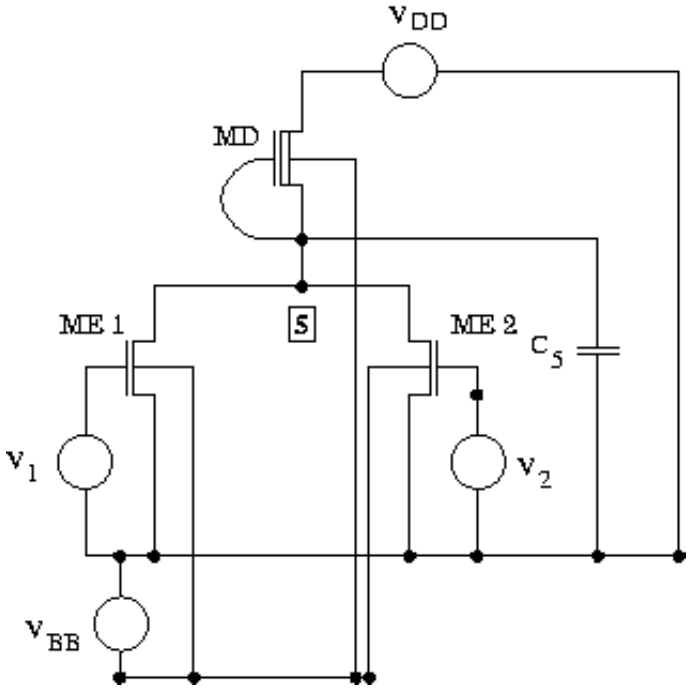

FIGURE II.16.2: *Circuit diagram of the NOR gate (taken from [Gün95]).*

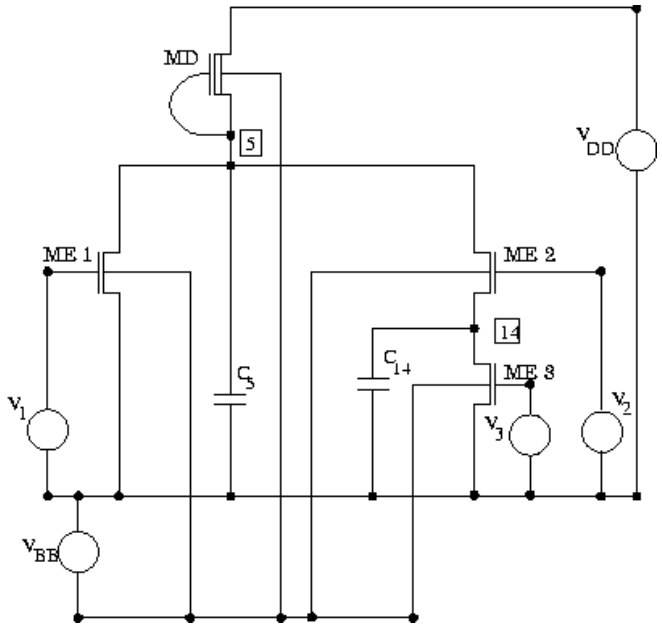

FIGURE II.16.3: *Circuit diagram of the ANDOI gate (taken from [Gün95]).*

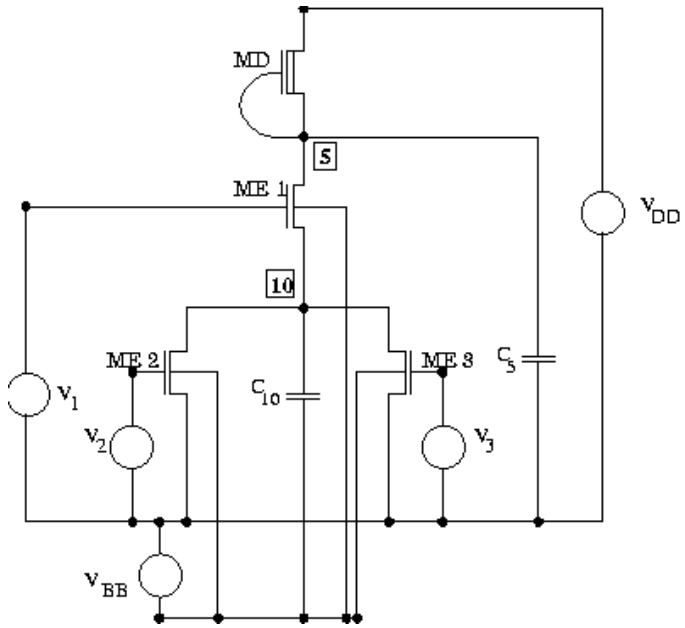

FIGURE II.16.4: *Circuit diagram of the ORANI gate (taken from [Gün95]).*

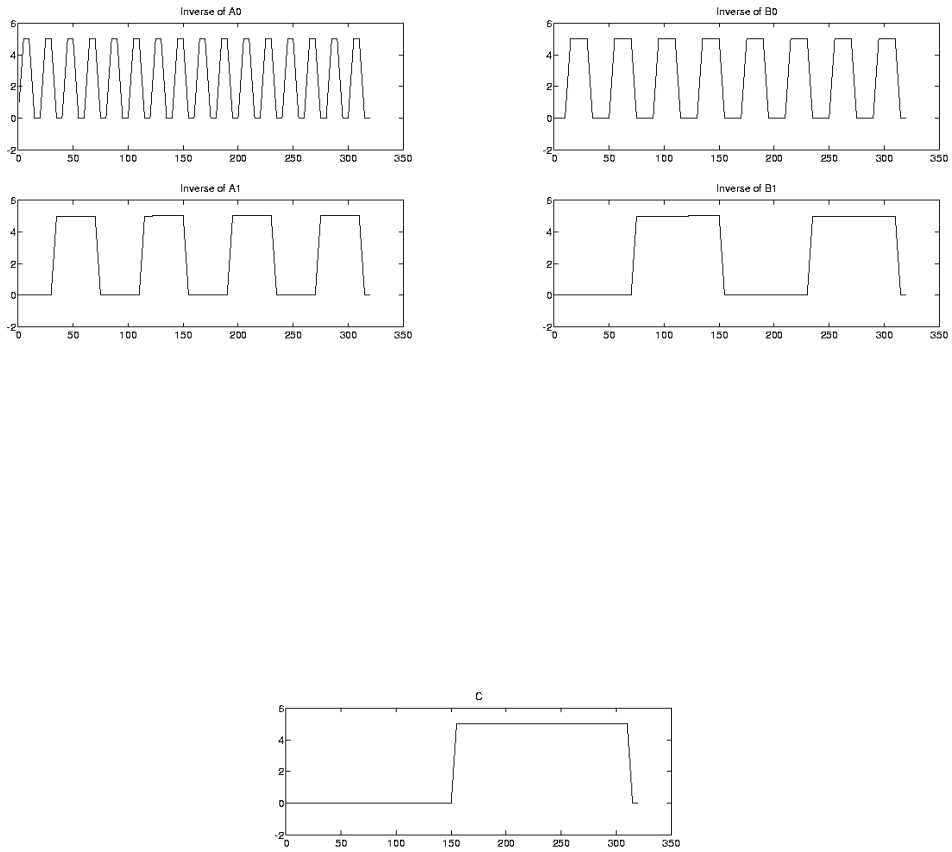

FIGURE II.16.5: *The input signals* $\overline{A}_0$, $\overline{A}_1$, $\overline{B}_0$, $\overline{B}_1$ *and* C.

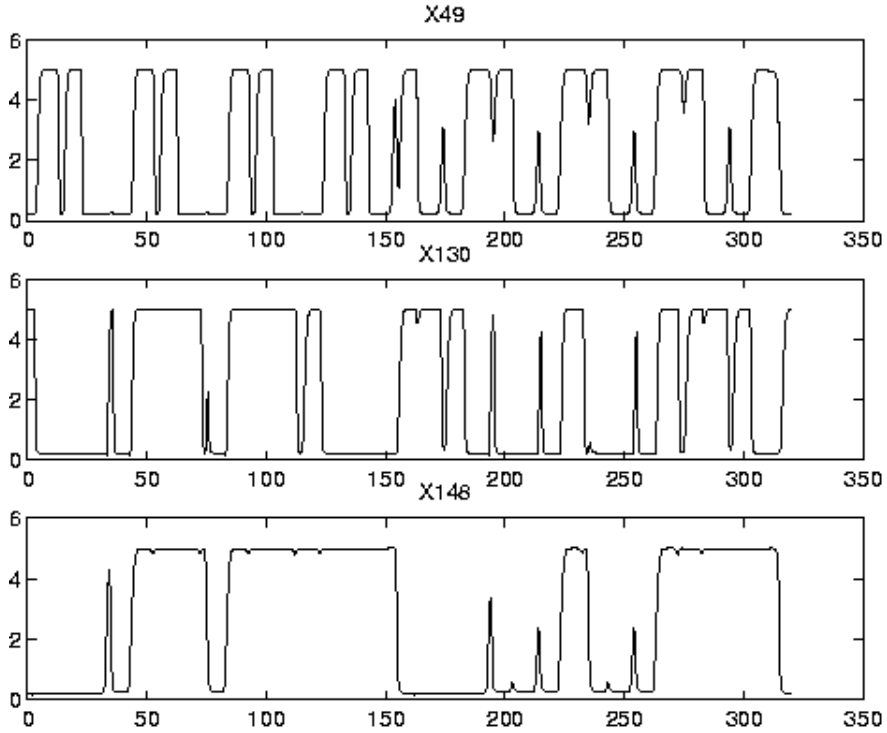

FIGURE II.16.6: *Behavior of the output signals S_0, S_1 and $\overline{C}$ over the integration interval.*

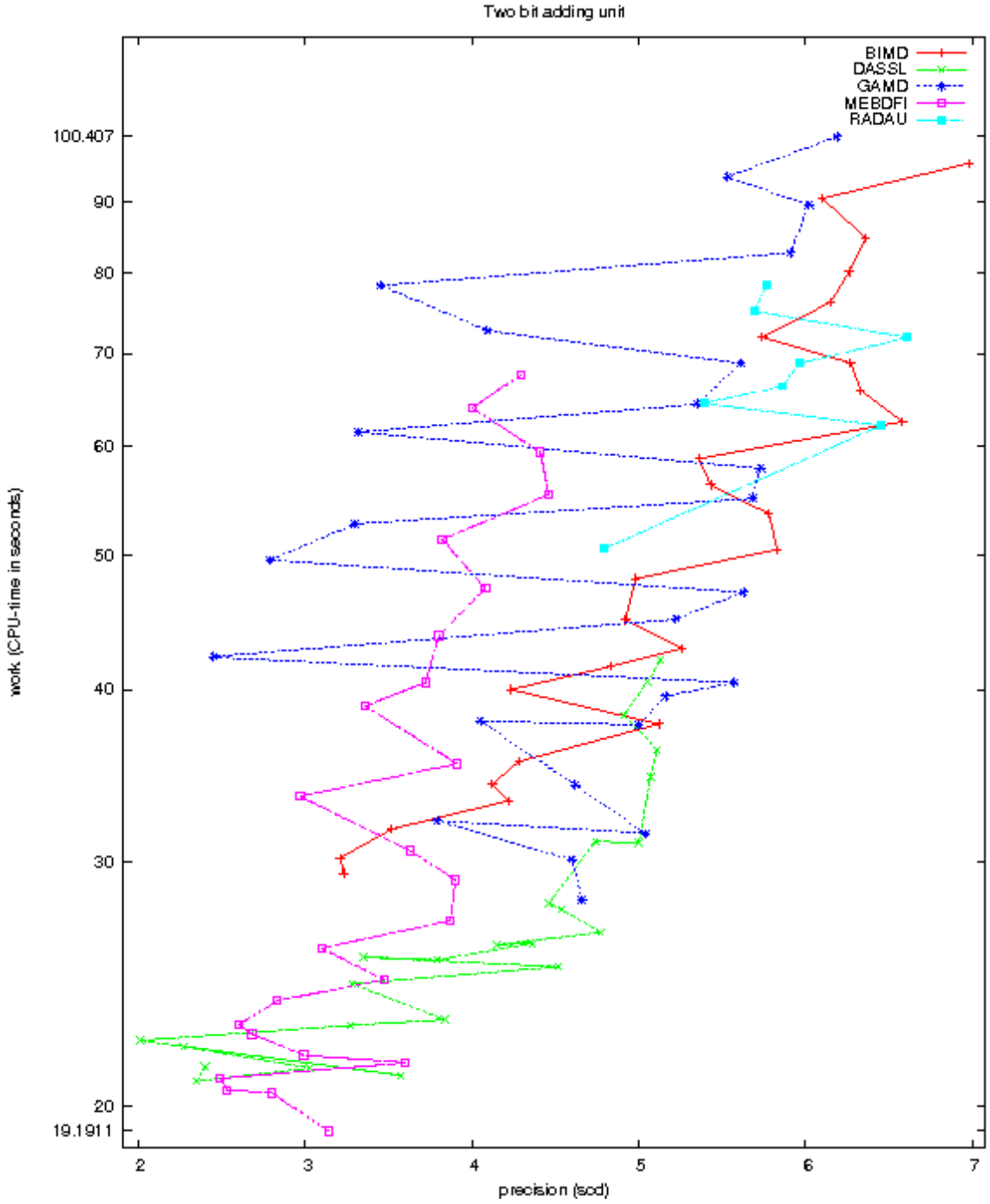

FIGURE II.16.7: *Work-precision diagram (scd versus CPU-time).*

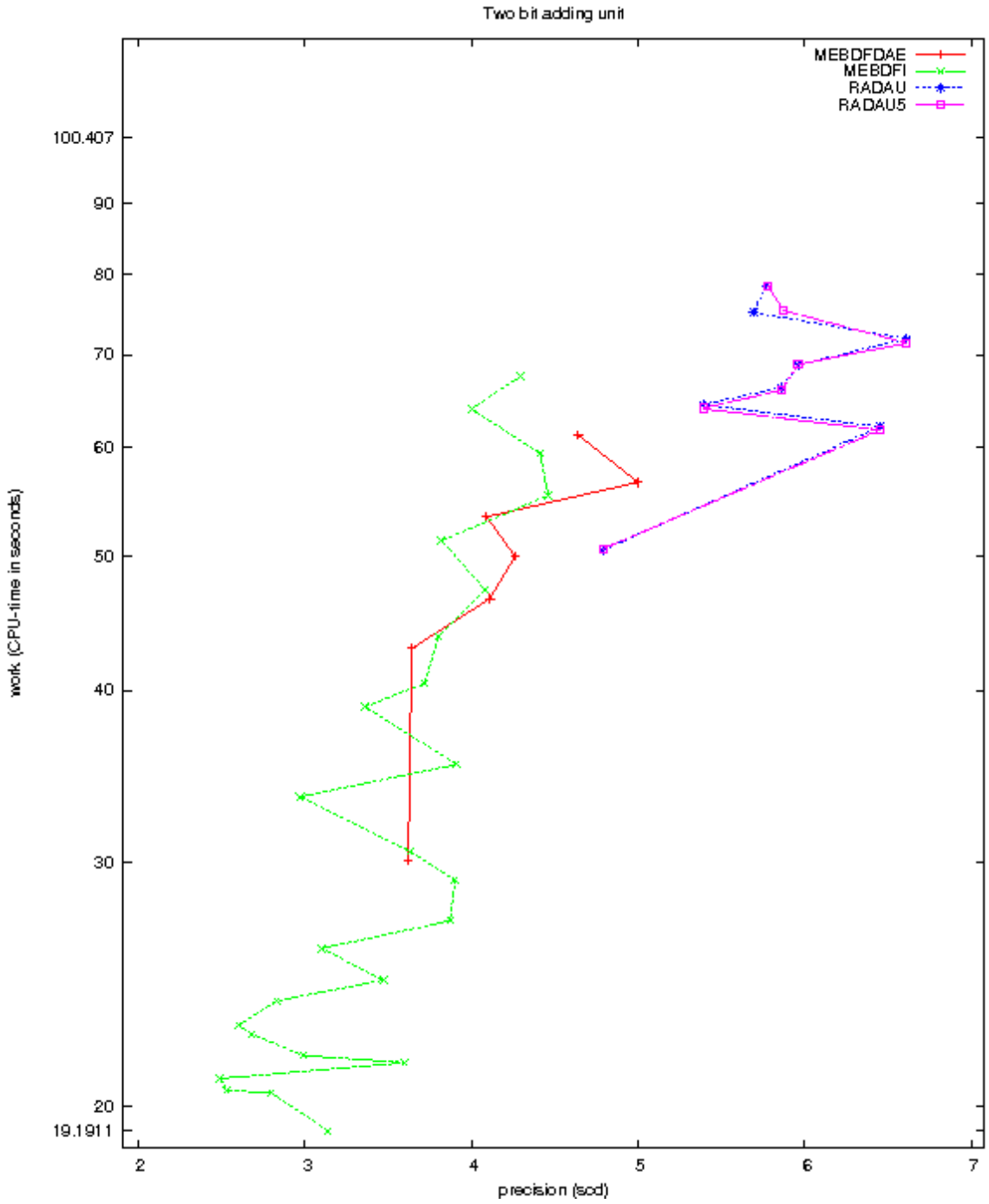

FIGURE II.16.8: *Work-precision diagram (scd versus CPU-time).*

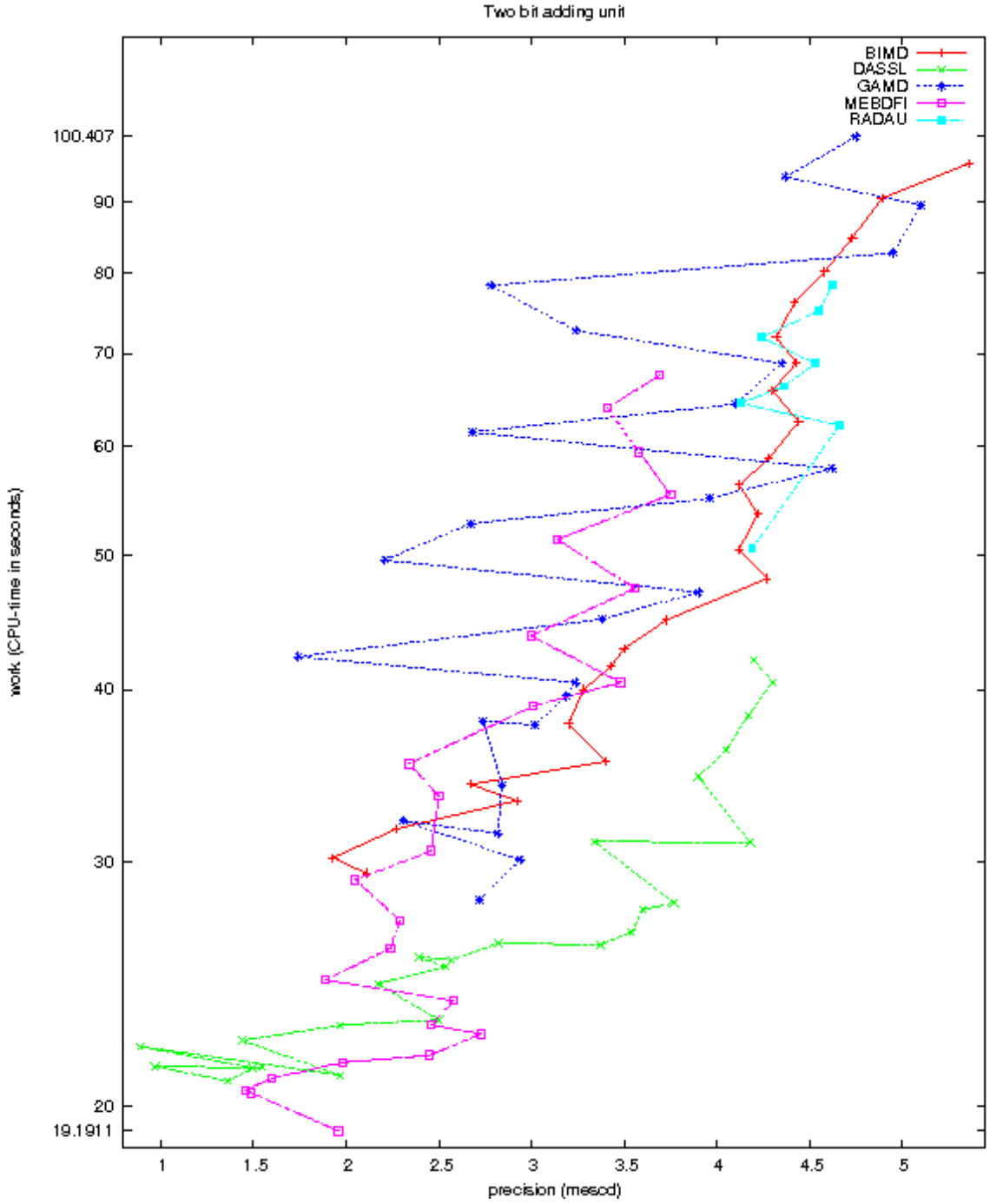

FIGURE II.16.9: *Work-precision diagram (mescd versus CPU-time).*

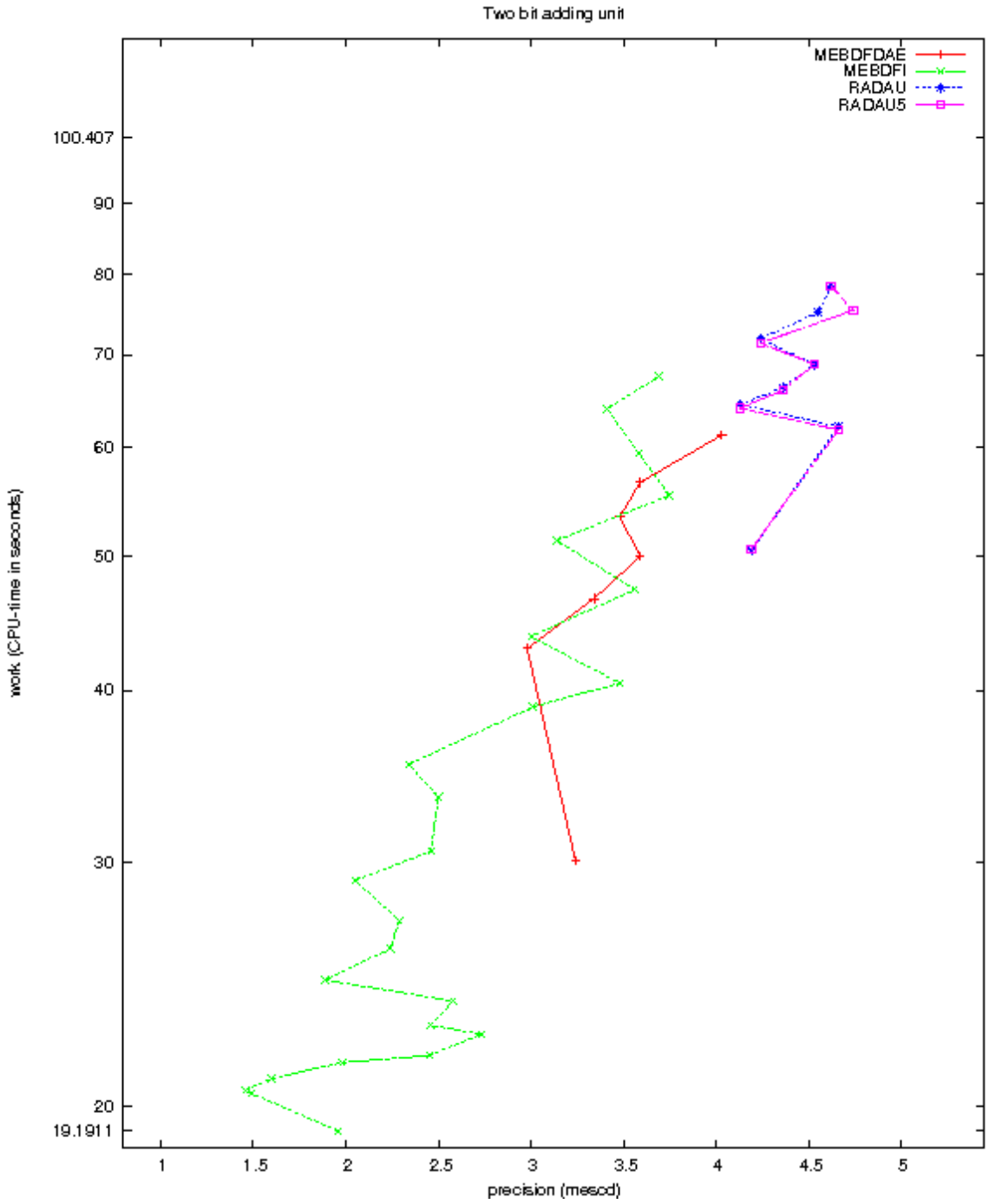

FIGURE II.16.10: *Work-precision diagram (mescd versus CPU-time).*

17 The car axis problem

17.1 General information

The problem is a stiff DAE of index 3, consisting of 8 differential and 2 algebraic equations. It has been taken from [Sch94]. Since not all initial conditions were given, we have chosen a consistent set of initial conditions. The software part of the problem is in the file caraxis.f available at [MM08].

17.2 Mathematical description of the problem

The problem is of the form

$$\begin{aligned} p' &= q, && \text{(II.17.1)}\\ Kq' &= f(t,p,\lambda), \qquad p,q \in I\!R^4, \quad \lambda \in I\!R^2, \quad 0 \le t \le 3, && \text{(II.17.2)}\\ 0 &= \phi(t,p), && \text{(II.17.3)} \end{aligned}$$

with initial conditions $p(0) = p_0$, $q(0) = q_0$, $p'(0) = q_0$, $q'(0) = q_0'$, $\lambda(0) = \lambda_0$ and $\lambda'(0) = \lambda_0'$.

The matrix K reads $\varepsilon^2 \frac{M}{2} I_4$, where I_4 is the 4×4 identity matrix. The function $f : I\!R^7 \to I\!R^4$ is given by

$$f(t,p,\lambda) = \begin{pmatrix} (L_0 - L_l)\dfrac{x_l}{L_l} & +\lambda_1 x_b + 2\lambda_2(x_l - x_r) & \\ (L_0 - L_l)\dfrac{y_l}{L_l} & +\lambda_1 y_b + 2\lambda_2(y_l - y_r) & -\varepsilon^2\dfrac{M}{2} \\ (L_0 - L_r)\dfrac{x_r - x_b}{L_r} & -2\lambda_2(x_l - x_r) & \\ (L_0 - L_r)\dfrac{y_r - y_b}{L_r} & -2\lambda_2(y_l - y_r) & -\varepsilon^2\dfrac{M}{2} \end{pmatrix}.$$

Here, $(x_l, y_l, x_r, y_r)^{\mathrm{T}} := p$, and L_l and L_r are given by

$$\sqrt{x_l^2 + y_l^2} \quad \text{and} \quad \sqrt{(x_r - x_b)^2 + (y_r - y_b)^2}.$$

Furthermore, the functions $x_b(t)$ and $y_b(t)$ are defined by

$$\begin{aligned} x_b(t) &= \sqrt{L^2 - y_b^2(t)}, && \text{(II.17.4)}\\ y_b(t) &= h\sin(\omega t). && \text{(II.17.5)} \end{aligned}$$

The function $\phi : I\!R^5 \to I\!R^2$ reads

$$\phi(t,p) = \begin{pmatrix} x_l x_b + y_l y_b \\ (x_l - x_r)^2 + (y_l - y_r)^2 - L^2 \end{pmatrix}.$$

The constants are listed below.

L	$=$	1	ϵ	$=$	10^{-2}	h	$=$	10^{-1}	ω	$=$	10
L_0	$=$	$1/2$	M	$=$	10	τ	$=$	$\pi/5$			

Consistent initial values are

$$p_0 = \begin{pmatrix} 0 \\ 1/2 \\ 1 \\ 1/2 \end{pmatrix}, \quad q_0 = \begin{pmatrix} -1/2 \\ 0 \\ -1/2 \\ 0 \end{pmatrix}, \quad q_0' = \frac{2}{M\varepsilon^2} f(0, p_0, \lambda_0), \quad p_0' = q_0, \quad \lambda_0 = \lambda_0' = (0,0)^{\mathrm{T}}.$$

The index of the variables p, q and λ is 1, 2 and 3, respectively.

17.3 Origin of the problem

The car axis problem 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.

A simplification of the car is depicted in Figure II.17.1. We model the situation that the left wheel

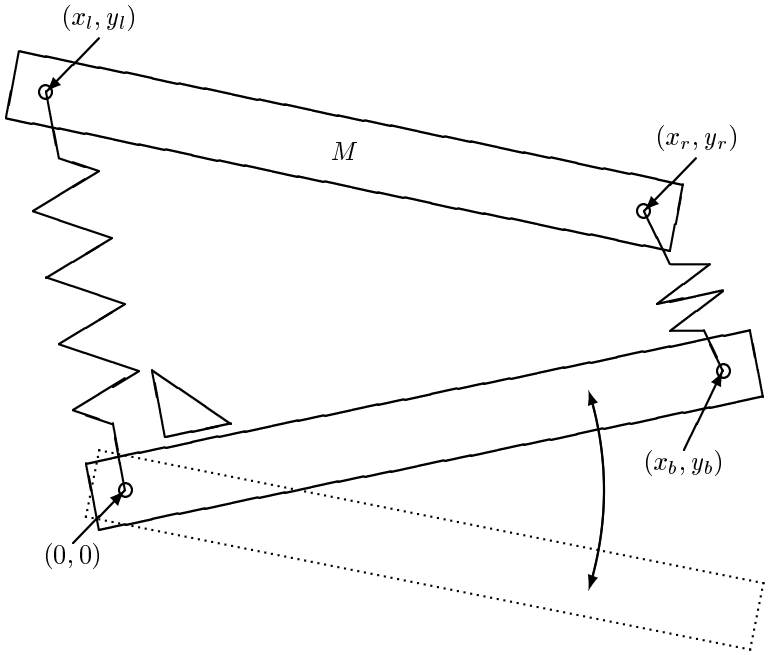

FIGURE II.17.1: *Modelnn of the car axis.*

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 (II.17.5). The length of the axis, denoted by L, remains constant over time, which means that x_b has to fulfill (II.17.4). 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.

There are two position constraints. Firstly, the distance between (x_l, y_l) and (x_r, y_r) must remain constantly L and secondly, for simplicity of the model, we assume that the left spring remains orthogonal to the axis. If we identify p with the vector $(x_l, y_l, x_r, y_r)^{\mathrm{T}}$, then we see that Equation (II.17.3) reflects these constraints.

Using Lagrangian mechanics, the equations of motions for the car axis are given by

$$\frac{M}{2}\frac{\mathrm{d}^2 p}{\mathrm{d}t^2} = F_{\mathrm{H}} + G^{\mathrm{T}}\lambda + F_{\mathrm{g}}. \tag{II.17.6}$$

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:

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

[†] in the source fortran file the variable r stands for h

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 [Sch94] sets $g = 1$.

We rewrite (II.17.6) as a system of first order differential equations by introducing the velocity vector q, so that we obtain the first order differential equations (II.17.1) and

$$\frac{M}{2} \frac{\mathrm{d}q}{\mathrm{d}t} = F_{\mathrm{H}} + G^{\mathrm{T}} \lambda + F_{\mathrm{g}}. \tag{II.17.7}$$

Setting $f = F_{\mathrm{H}} + G^{\mathrm{T}} \lambda + F_g$, it is easily checked that multiplying (II.17.7) by ε^2 yields (II.17.2).

To arrive at a consistent set of initial values p_0, q_0 and λ_0, we have to solve the system of equations consisting of the constraint

$$\phi(t_0, p_0) = 0, \tag{II.17.8}$$

and the 1 up to $k - 1$ times differentiated constraint (II.17.8), 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 $\tilde{G}$. Here, $k = 3$, yielding the additional conditions

$$\tilde{G}(\tilde{p}_0)\tilde{q}_0 = 0 \tag{II.17.9}$$

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 (II.17.6) and the fact that the first component of $\tilde{q}_0'$ vanishes, the latter condition equals

$$\phi_{\tilde{p}\tilde{p}}(\tilde{p}_0)(\tilde{q}_0, \tilde{q}_0) + \frac{2}{M} G(p_0) \left(F_{\mathrm{H}}(p_0) + G^{\mathrm{T}}(p_0)\lambda_0 + F_{\mathrm{g}}(p_0) \right) = 0. \tag{II.17.10}$$

The equations (II.17.8)–(II.17.10) are solved for

$$\begin{aligned}
x_r &= L,\\
x_l &= 0,\\
y_r &= y_l = L_0,\\
x_r' &= x_l' = -\frac{L_0}{L}\frac{2\pi}{\tau}h,\\
y_r' &= \frac{L^2\tau}{2\pi h\varepsilon^2 M}(2\lambda_1-\lambda_2),\\
y_l' &= \frac{L^2\tau}{2\pi\varepsilon^2 hM}(2\lambda_2-\lambda_1)\pm\frac{L}{\varepsilon}\sqrt{\frac{-8\lambda_2+2\lambda_1}{M}}.
\end{aligned}$$

Choosing $\lambda_1=\lambda_2=0$, we arrive at the initial conditions listed in §17.2,

TABLE II.17.1: *Reference solution at the end of the integration interval.*

y_1	$0.493455784275402809122\cdot10^{-1}$	y_6	$0.744686658723778553466\cdot10^{-2}$
y_2	0.496989460230171153861	y_7	$0.175568157537232222276\cdot10^{-1}$
y_3	$0.104174252488542151681\cdot10$	y_8	0.770341043779251976443
y_4	0.373911027265361256927	y_9	$-0.473688659084893324729\cdot10^{-2}$
y_5	$-0.770583684040972357970\cdot10^{-1}$	y_{10}	$-0.110468033125734368808\cdot10^{-2}$

17.4 Numerical solution of the problem

Tables II.17.1–II.17.2 and Figures II.17.2–II.17.3 present the reference solution at the end of the integration interval, the run characteristics, the behavior of some solution components over the integration interval and the work-precision diagrams, respectively. The reference solution was computed on using quadruple precision GAMD on an Alphaserver DS20E, with a 667 MHz EV67 processor. atol = rtol = $h_0 = 10^{-24}$. For the work-precision diagrams, we used: rtol = $10^{-(4+m/4)}$, $m = 0,1,\ldots,24$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

References

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Sch94] S. Schneider. *Intégration de systèmes d'équations différentielles raides et différentielles-algébriques par des méthodes de collocations et méthodes générales linéaires.* PhD thesis, Université de Genève, 1994.

TABLE II.17.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-4}	2.19	0.34	71	71	1693	71	71	0.0088
	10^{-7}	10^{-7}	10^{-7}	5.47	3.34	138	138	4511	138	138	0.0224
	10^{-10}	10^{-10}	10^{-10}	8.01	5.35	235	235	9669	235	235	0.0488
GAMD	10^{-4}	10^{-4}	10^{-4}	1.98	0.39	39	39	2169	39	39	0.0088
	10^{-7}	10^{-7}	10^{-7}	4.82	2.64	98	98	7167	98	98	0.0293
	10^{-10}	10^{-10}	10^{-10}	6.50	3.84	179	179	18771	179	179	0.0742
MEBDFI	10^{-4}	10^{-4}	10^{-4}	0.88	−0.23	280	278	1246	27	27	0.0059
	10^{-7}	10^{-7}	10^{-7}	4.65	3.34	650	648	2797	47	47	0.0137
	10^{-10}	10^{-10}	10^{-10}	4.21	2.08	1393	1391	5624	85	85	0.0264
PSIDE-1	10^{-4}	10^{-4}		0.83	−0.28	55	54	1403	42	220	0.0098
	10^{-7}	10^{-7}		4.41	2.27	179	172	4103	83	464	0.0273
	10^{-10}	10^{-10}		7.25	4.86	625	612	13751	115	964	0.0869
RADAU	10^{-4}	10^{-4}	10^{-4}	1.34	0.19	98	97	850	95	98	0.0039
	10^{-7}	10^{-7}	10^{-7}	3.73	2.51	289	288	2559	282	288	0.0127
	10^{-10}	10^{-10}	10^{-10}	5.99	4.22	179	178	4281	169	179	0.0166

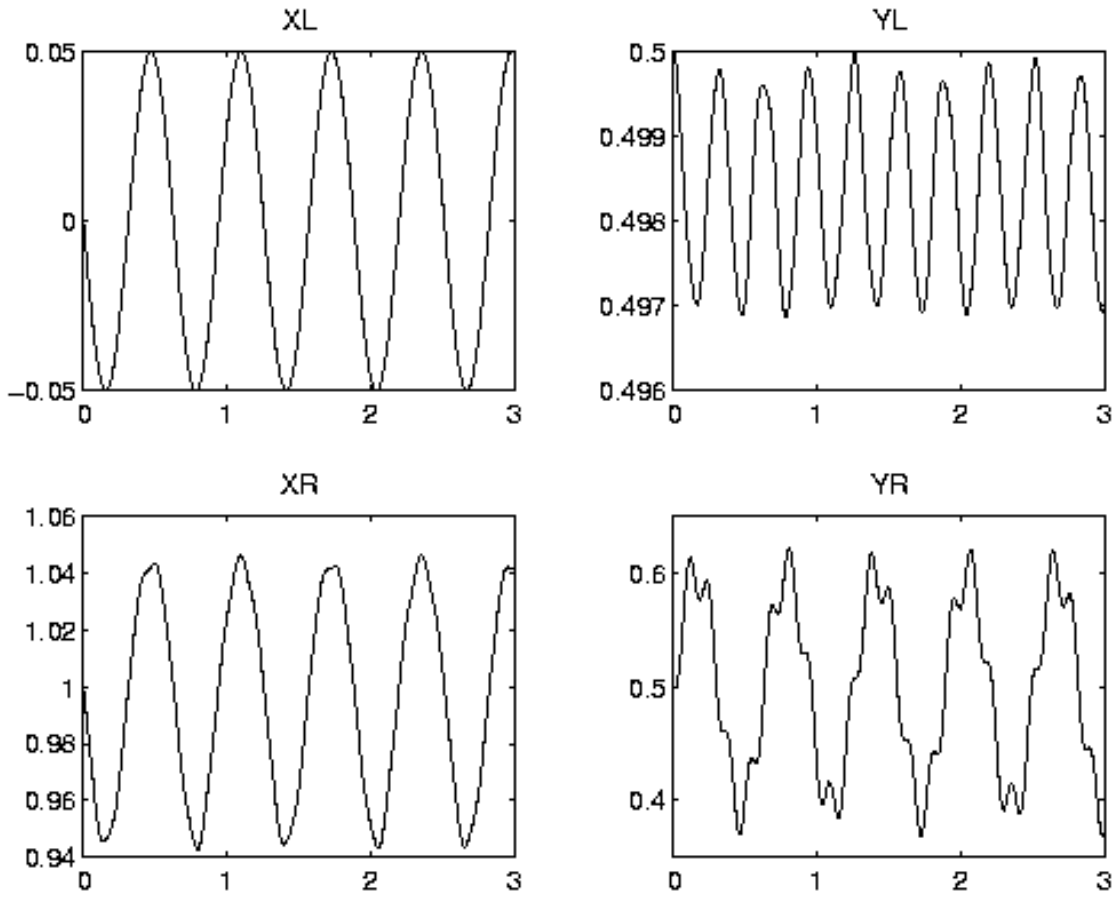

FIGURE II.17.2: *Behavior of* (x_l, y_l) *and* (x_r, y_r) *over the integration interval.*

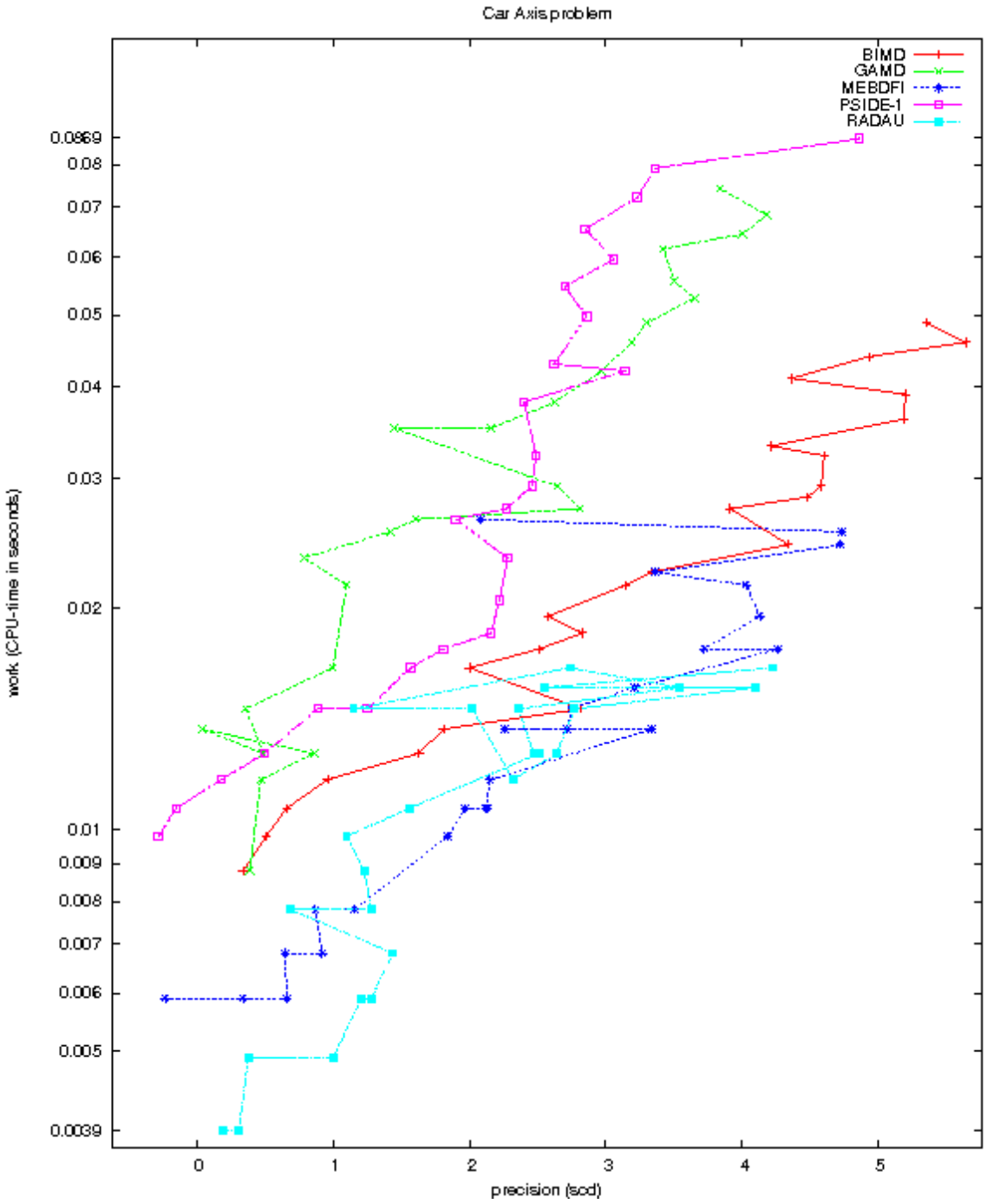

FIGURE II.17.3: *Work-precision diagram (scd versus CPU-time).*

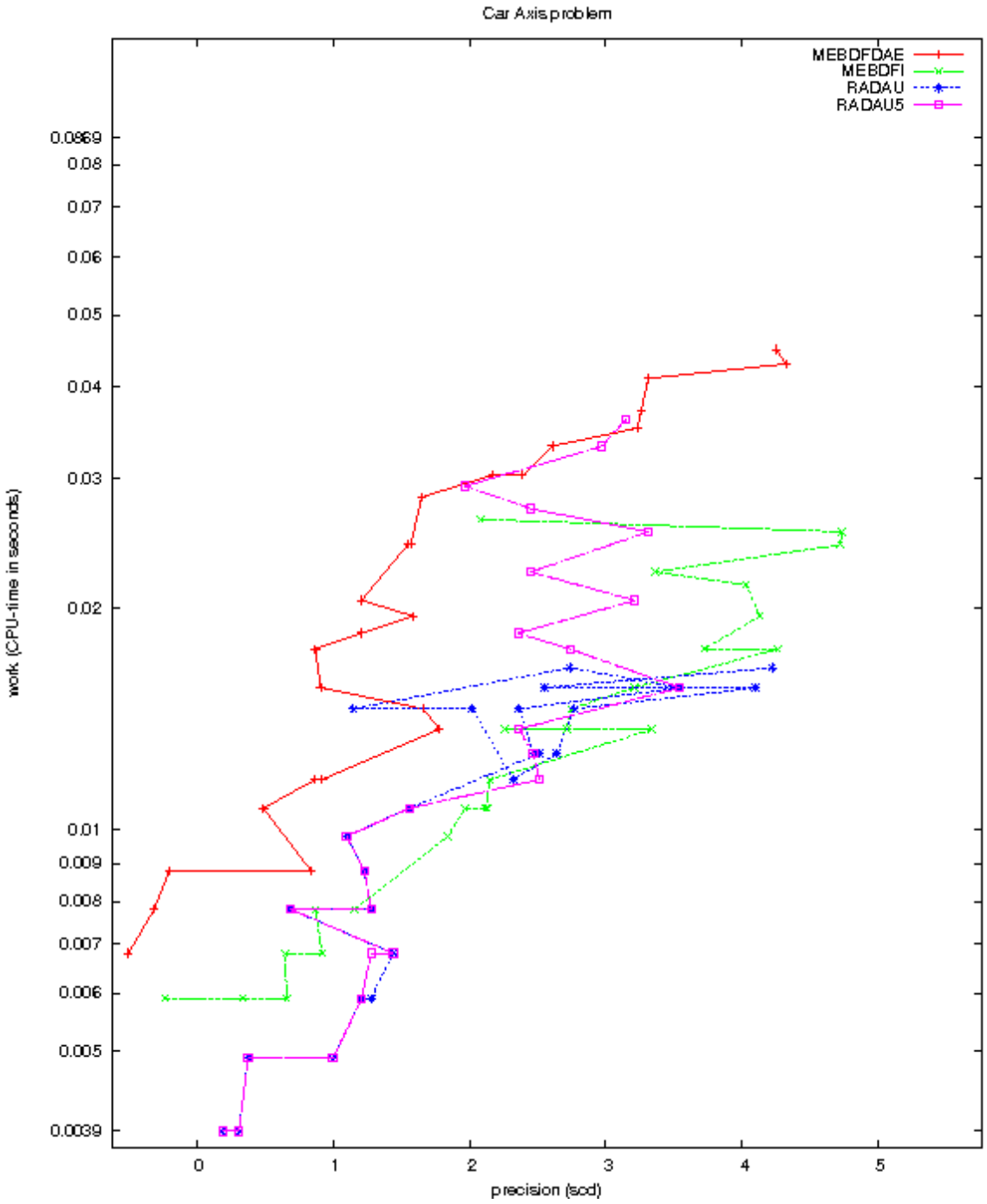

FIGURE II.17.4: *Work-precision diagram (scd versus CPU-time).*

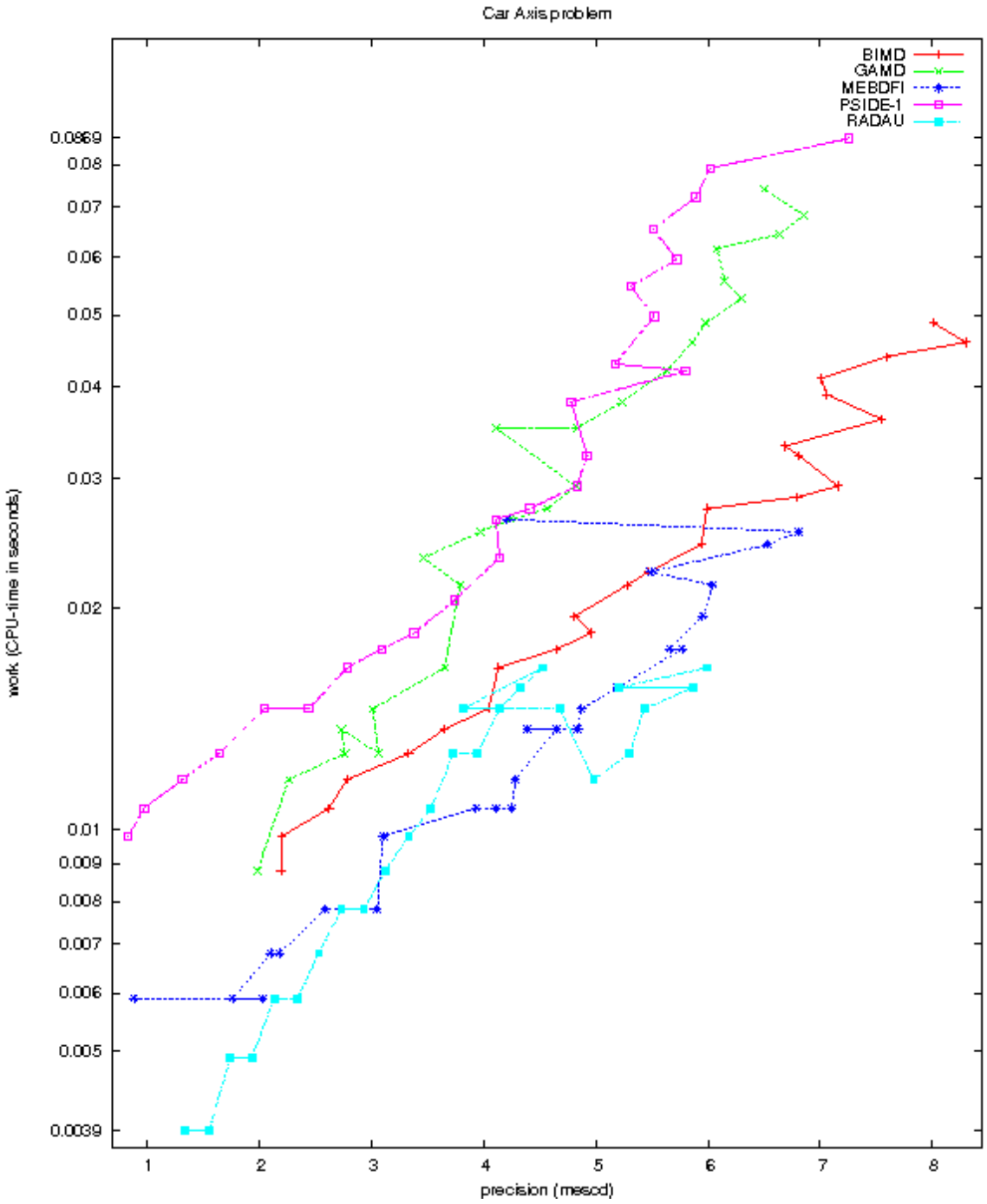

FIGURE II.17.5: *Work-precision diagram (mescd versus CPU-time).*

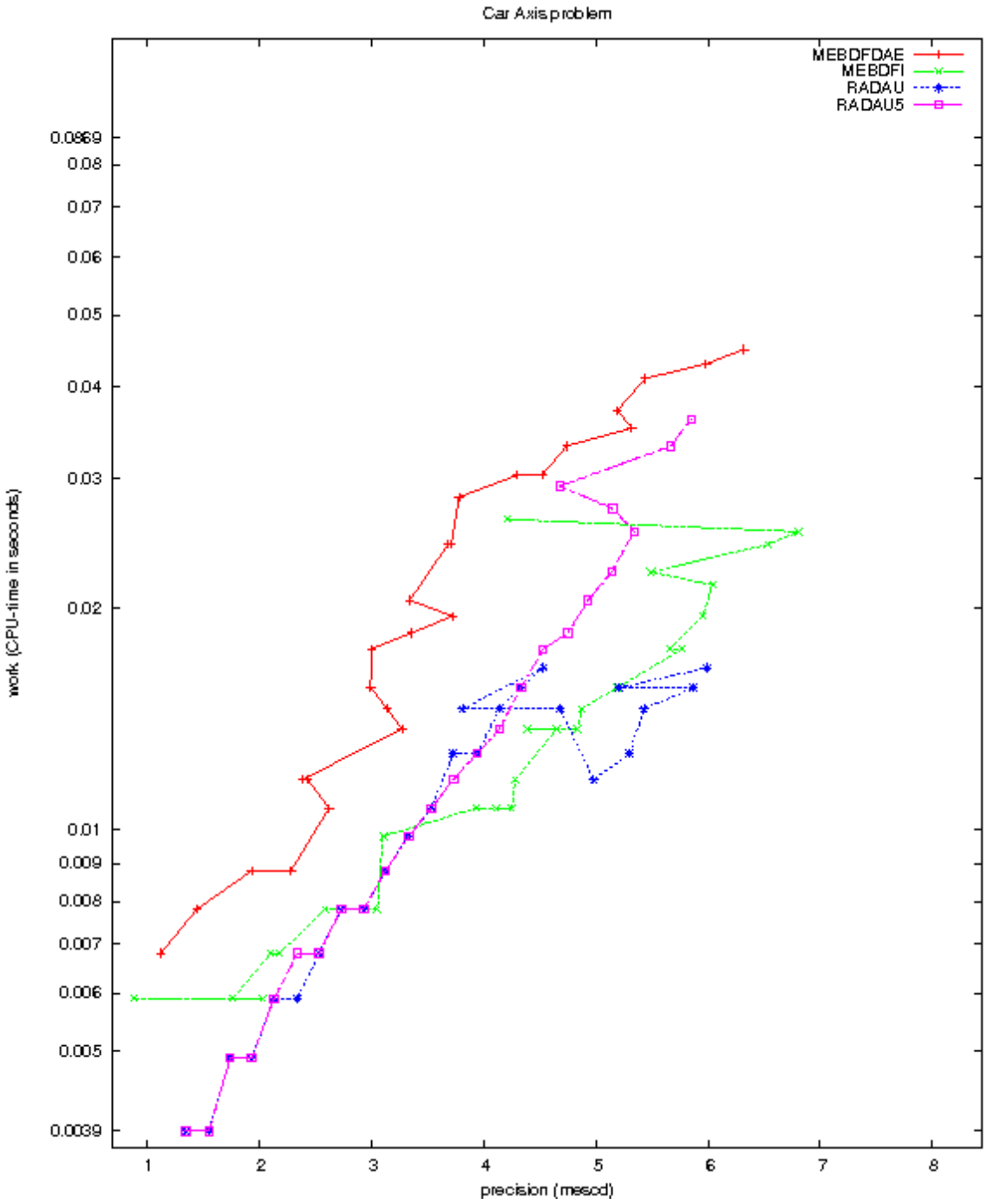

FIGURE II.17.6: *Work-precision diagram (mescd versus CPU-time).*

18 Fekete problem

18.1 General information

The problem is an index 2 DAE from mechanics. The dimension is $8N$, where N is a user supplied integer. The numerical tests shown here correspond to $N = 20$. The problem is of interest for the computation of the elliptic Fekete points [Par95]. The parallel-IVP-algorithm group of CWI contributed this problem to the test set, in collaboration with W. J. H. Stortelder. The software part of the problem is in the file fekete.f available at [MM08].

18.2 Mathematical description of the problem

The problem is of the form

$$M\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \quad y(0) = y_0, \quad y'(0) = y'_0, \tag{II.18.1}$$

with

$$y, f \in I\!R^{8N}, \quad 0 \le t \le t_{\mathrm{end}}.$$

Here, $t_{\mathrm{end}} = 1000$, $N = 20$ and M is the (constant) mass matrix given by

$$M = \begin{pmatrix} I_{6N} & 0 \\ 0 & 0 \end{pmatrix},$$

where I_{6N} is the identity matrix of dimension $6N$. For the definition of the function f, we refer to §18.3.

The components $y_{0,i}$ of of the initial vector y_0 are defined by

$$\begin{pmatrix} y_{0,3(j-1)+1} \\ y_{0,3(j-1)+2} \\ y_{0,3(j-1)+3} \end{pmatrix} = \begin{pmatrix} \cos(\omega_j)\cos(\beta_j) \\ \sin(\omega_j)\cos(\beta_j) \\ \sin(\beta_j) \end{pmatrix} \quad \text{for} \quad j = 1, \ldots, N,$$

where

$$\begin{aligned}
\beta_j &= \tfrac{3}{8}\pi \quad \text{and} \quad \omega_j = \tfrac{2j}{3}\pi + \tfrac{1}{13}\pi \quad \text{for} \quad j = 1, \ldots, 3, \\
\beta_j &= \tfrac{1}{8}\pi \quad \text{and} \quad \omega_j = \tfrac{2(j-3)}{7}\pi + \tfrac{1}{29}\pi \quad \text{for} \quad j = 4, \ldots, 10, \\
\beta_j &= -\tfrac{2}{15}\pi \quad \text{and} \quad \omega_j = \tfrac{2(j-10)}{6}\pi + \tfrac{1}{7}\pi \quad \text{for} \quad j = 11, \ldots, 16, \\
\beta_j &= -\tfrac{3}{10}\pi \quad \text{and} \quad \omega_j = \tfrac{2(j-17)}{4}\pi + \tfrac{1}{17}\pi \quad \text{for} \quad j = 17, \ldots, 20,
\end{aligned}$$

and

$$\begin{aligned}
y_{0,i} &= 0 && \text{for} \quad i = 3N+1, \ldots, 6N, \\
y_{0,6N+j} &= \tfrac{1}{2}\langle p_j(0), \widehat{f}_j \rangle && \text{for} \quad j = 1, \ldots, N, \\
y_{0,i} &= 0 && \text{for} \quad i = 7N+1, \ldots, 8N,
\end{aligned}$$

where

$$p_j = \begin{pmatrix} y_{3(j-1)+1} \\ y_{3(j-1)+2} \\ y_{3(j-1)+3} \end{pmatrix}, \quad \widehat{f}_j = \begin{pmatrix} f_{3N+3(j-1)+1}((p(0), 0, \ldots, 0)^{\mathrm{T}}) \\ f_{3N+3(j-1)+2}((p(0), 0, \ldots, 0)^{\mathrm{T}}) \\ f_{3N+3(j-1)+3}((p(0), 0, \ldots, 0)^{\mathrm{T}}) \end{pmatrix}, \tag{II.18.2}$$

and $p = (y_1, y_2, \ldots, y_{3N})^{\mathrm{T}}$. The initial derivative vector reads $y'_0 = f(y_0)$. These definitions of y_0 and y'_0 yield consistent initial values. The first $6N$ components are of index 1, the last $2N$ of index 2.

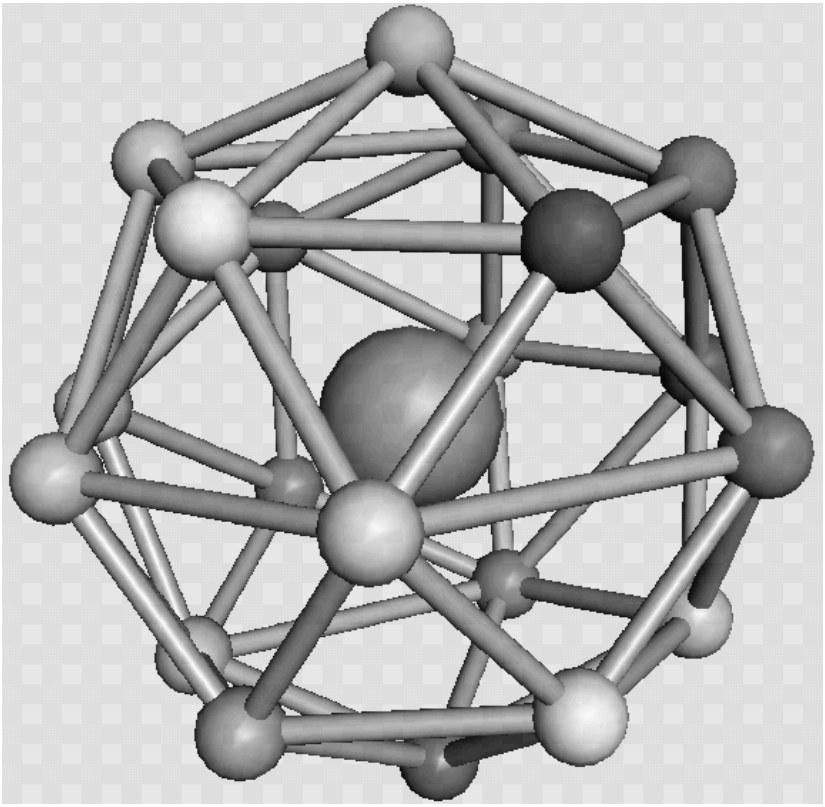

FIGURE II.18.1: *Final configuration for $N = 20$. The large ball is centered at the origin and only added to facilitate the 3-D perception. (Taken from [PSS97] by courtesy of R. van Liere.)*

18.3 Origin of the problem

This problem is of interest for the computation of the elliptic Fekete points. Let us define the unit sphere in $\mathbb{R}^3$ by $\mathcal{S}^2$ and for any configuration $x := (x_1, x_2, \ldots, x_N)^{\mathrm{T}}$ of points $x_i \in \mathcal{S}^2$, the function

$$V(x) := \prod_{i<j} \|x_i - x_j\|_2. \tag{II.18.3}$$

We denote the value of x for which V reaches its global maximum by $\widehat{x} = (\widehat{x}_1, \ldots, \widehat{x}_N)$. The points $\widehat{x}_1, \widehat{x}_2, \ldots, \widehat{x}_N$ are called the elliptic Fekete points of order N. For example, for $N = 4$, the points of the optimal solution form a tetrahedron. But, in case of 8 points, intuition fails; the elliptic Fekete points do not form a cube in this case. A cube where, for example, the upper plane is rotated over 45° with respect to the bottom plane, gives already a larger value of V. It turns out (see e.g. [Par95]) that $\widehat{x}$ is difficult to compute as solution of a global optimization problem. For reasons that will become clear later, we differentiate $\log(V)$ with respect to x_k and apply the method of Lagrange multipliers, to see that $\widehat{x}$ fulfills

$$\nabla_k \log(V(x)) \mid_{x=\widehat{x}} = \sum_{j \neq k} \frac{\widehat{x}_k - \widehat{x}_j}{\|\widehat{x}_k - \widehat{x}_j\|_2^2} = \zeta_k \widehat{x}_k, \tag{II.18.4}$$

where the ζ_k are Lagrange multipliers.

We now discuss the Fekete points from another point of view. Consider on $\mathcal{S}^2$ a number of N particles, on which two forces are invoked: a repulsive force, by which the particles will start to move away from each other, and an adhesion force, by which the particles will reach a stationary state after a certain period of time.

We denote the position in Cartesian coordinates of particle i at time t by $p_i(t)$ and the configuration of N points at time t by $p(t) = (p_1(t), \ldots, p_N(t))^{\mathrm{T}}$. The stationary configuration is assumed to be obtained at $t = t_{\mathrm{stat}}$ and will be denoted by $\widehat{p} := (\widehat{p}_1, \widehat{p}_2, \ldots, \widehat{p}_N)$, where $\widehat{p}_i := p_i(t_{\mathrm{stat}})$. The repulsive force on particle i caused by particle j is defined by

$$F_{ij} = \frac{p_i - p_j}{\|p_i - p_j\|_2^{\gamma}}.$$

Note that the choice $\gamma = 3$ can be interpreted as an electrical force working on particles with unit charge. The adhesion force working on particle i is denoted by A_i and given by

$$A_i = -\alpha q_i.$$

Here, q is the velocity vector and α is valued 0.5.

We can compute the configuration of the particles as function of time, given that the particles cannot leave the unit sphere, as solution of the DAE system

$$\begin{aligned} p' &= q, && \text{(II.18.5)}\\ q' &= g(p,q) + G^{\mathrm{T}}(p)\lambda, && \text{(II.18.6)}\\ 0 &= \phi(p), && \text{(II.18.7)} \end{aligned}$$

where $G = \partial\phi/\partial p$ and $\lambda \in I\!R^N$. The function $\phi : I\!R^{3N} \to I\!R^N$ represents the constraint, which states that the particles remain on the unit sphere:

$$\phi_i(p) = p_{i,1}^2 + p_{i,2}^2 + p_{i,3}^2 - 1.$$

The function $g : I\!R^{6N} \to I\!R^{3N}$ is given by $g = (g_i)$, $i = 1, \ldots, N$, where

$$g_i(p,q) = \sum_{j \neq i} F_{ij}(p) + A_i(q).$$

The term $G^{\mathrm{T}}(p)\lambda$ in (II.18.6) represents the normal force which keeps the particle on $\mathcal{S}^2$.

Since we know that the speed of the final configuration at $t = t_{\mathrm{stat}}$ is 0, we can substitute $q = 0$ and $p = \widehat{p}$ in formula (II.18.6), thus arriving at

$$0 = \sum_{j \neq i} F_{ij}(\widehat{p}) + G^{\mathrm{T}}(\widehat{p})\lambda \ ,$$

which is equal to

$$\sum_{i \neq j} \frac{\widehat{p}_i - \widehat{p}_j}{\|\widehat{p}_i - \widehat{p}_j\|_2^{\gamma}} = -2\lambda_i \widehat{p}_i \ . \qquad \text{(II.18.8)}$$

Comparing (II.18.4) and (II.18.8) tells us that computing $\widehat{p}$ for $\gamma = 2$ gives the local optima of the function V in (II.18.3). In [PSS97], it is showed that computing $\widehat{p}$ by solving the system (II.18.5)–(II.18.7) and then substituting $x = \widehat{p}$ in (II.18.3), results in values of V that are very competitive with those obtained by global optimization packages. For more details on elliptic Fekete points, we refer to [Par95] and [SS93].

The DAE system mentioned before is of index 3. To arrive at a more stable formulation of the problem, we stabilize the constraint (see [BCP89, p. 153]) by replacing (II.18.5) by

$$p' = q + G^{\mathrm{T}}(p)\mu, \qquad \text{(II.18.9)}$$

where $\mu \in I\!R^N$, and appending the differentiated constraint

$$0 = G(p)q. \qquad \text{(II.18.10)}$$

TABLE II.18.1: *Reference solution at the end of the integration interval.*

$y(1)$	−0.4070263380333202	$y(7)$	0.7100577833343567
$y(2)$	0.3463758772791802	$y(8)$	0.1212948055586120
$y(3)$	0.8451942450030429	$y(9)$	0.6936177005172217
$y(4)$	0.0775293475252155	$y(10)$	0.2348267744557627
$y(5)$	−0.2628662719972299	$y(11)$	0.7449277976923311
$y(6)$	0.9617122871829146	$y(12)$	0.6244509285956391

The system (II.18.9), (II.18.6), (II.18.7), (II.18.10) is now of index 2; the variables p and q are of index 1, the variables λ and μ of index 2. We cast the system in the form (II.18.1) by setting $y = (p, q, \lambda, \mu)^{\mathrm{T}}$ and $f(y) = f(p, q, \lambda, \mu) = (q + G^{\mathrm{T}}\mu, g + G^{\mathrm{T}}\lambda, \phi, Gq)^{\mathrm{T}}$, where p_i is in Cartesian coordinates.

The choice for the initial configuration as defined in §18.2 is a rough attempt to spread out the points over the sphere. To arrive at a consistent set of initial values we choose $q(0) = 0$, yielding $\mu(0) = 0$ and $\phi_i'(0) = \langle 2p_i(0), q_i(0)\rangle = 0$. Consequently,

$$\begin{aligned} \phi_i''(0) &= \langle 2p_i(0), q_i'(0)\rangle \\ &= \langle 2p_i(0), g_i(p(0), q(0)) + 2\lambda_i(0)p_i(0)\rangle. \end{aligned}$$

Requiring $\phi_i''(0) = 0$ gives

$$\lambda_i(0) = -\frac{\langle p_i(0), g_i(p(0), q(0))\rangle}{2\langle p_i(0), p_i(0)\rangle} = -\frac{1}{2}\langle p_i(0), g_i(p(0), q(0))\rangle.$$

The initial derivative vector y_0' can be chosen equal to $f(y_0)$. For $N \le 20$, $t_{\mathrm{stat}} \le 1000$, therefore we chose $t_{\mathrm{end}} = 1000$.

In Figure II.18.1 the final configuration for 20 points is plotted.

18.4 Numerical solution of the problem

All the tests concern the case with $N = 20$. Tables II.18.1–II.18.2 and Figures II.18.2–II.18.6 present the reference solution at the end of the integration interval (first 12 components), the run characteristics, the behavior of the first 6 solution components over the interval $[0, 20]$ and the work-precision diagrams, respectively. In computing the scd values, only the first sixty components were considered, since they refer to the position of the particles. The reference solution was computed using RADAU5, rtol $= 10^{-12}$, atol $= 10^{-12}$, and h0 $= 10^{-12}$. For the work-precision diagrams, we used: rtol $= 10^{-(2+m/16)}$, $m = 0, 1, \ldots, 64$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

References

[BCP89] K.E. Brenan, S.L. Campbell, and L.R. Petzold. *Numerical Solution of Initial–Value Problems in Differential–Algebraic Equations*. North–Holland, New York–Amsterdam–London, 1989.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Par95] P.M. Pardalos. An open global optimization problem on the unit sphere. *Journal of Global Optimization*, 6:213, 1995.

TABLE II.18.2: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-2}	10^{-2}	10^{-2}	4.12	2.63	30	29	415	29	30	0.2450
	10^{-3}	10^{-3}	10^{-3}	5.36	4.19	43	43	668	43	43	0.3445
	10^{-4}	10^{-4}	10^{-4}	6.69	5.33	65	65	1094	65	65	0.5124
GAMD	10^{-2}	10^{-2}	10^{-2}	4.16	2.99	26	24	526	24	26	0.2147
	10^{-3}	10^{-3}	10^{-3}	4.79	3.78	26	26	947	26	26	0.3006
	10^{-4}	10^{-4}	10^{-4}	5.76	4.45	38	38	1319	38	38	0.4119
MEBDFI	10^{-2}	10^{-2}	10^{-2}	3.56	2.10	60	57	192	15	15	0.1064
	10^{-3}	10^{-3}	10^{-3}	4.58	3.23	129	128	428	18	18	0.1513
	10^{-4}	10^{-4}	10^{-4}	5.81	4.81	218	216	707	23	23	0.2176
PSIDE-1	10^{-2}	10^{-2}		3.66	2.20	73	53	693	16	288	1.3137
	10^{-3}	10^{-3}		4.40	3.19	88	59	779	11	344	1.4357
	10^{-4}	10^{-4}		5.32	4.12	114	75	967	9	448	1.7363
RADAU	10^{-2}	10^{-2}	10^{-2}	3.43	1.97	33	30	274	27	32	0.5065
	10^{-3}	10^{-3}	10^{-3}	4.11	2.65	43	41	315	38	43	0.5993
	10^{-4}	10^{-4}	10^{-4}	5.36	4.29	61	58	442	54	61	0.7662

[PSS97] J.D. Pintér, W.J.H. Stortelder, and J.J.B. de Swart. Computation of elliptic Fekete point sets. Report MAS-R9705, CWI, Amsterdam, 1997. To appear in CWI Quarterly.

[SS93] M. Shub and S. Smale. Complexity of Bezout's theorem III. Condition number and packing. *Journal of Complexity*, 9:4–14, 1993.

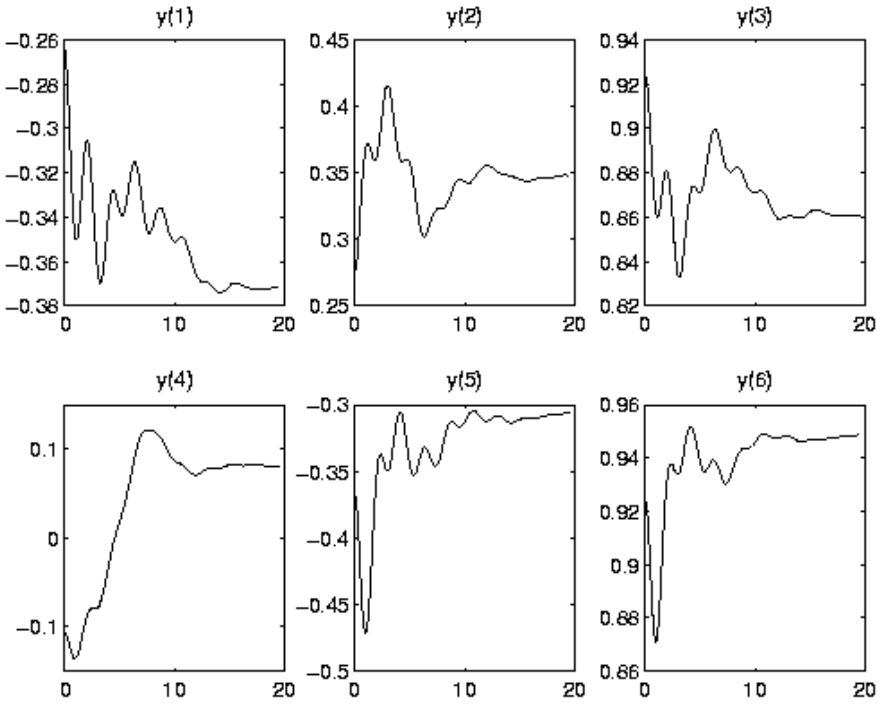

FIGURE II.18.2: *Behavior of the solution over the integration interval.*

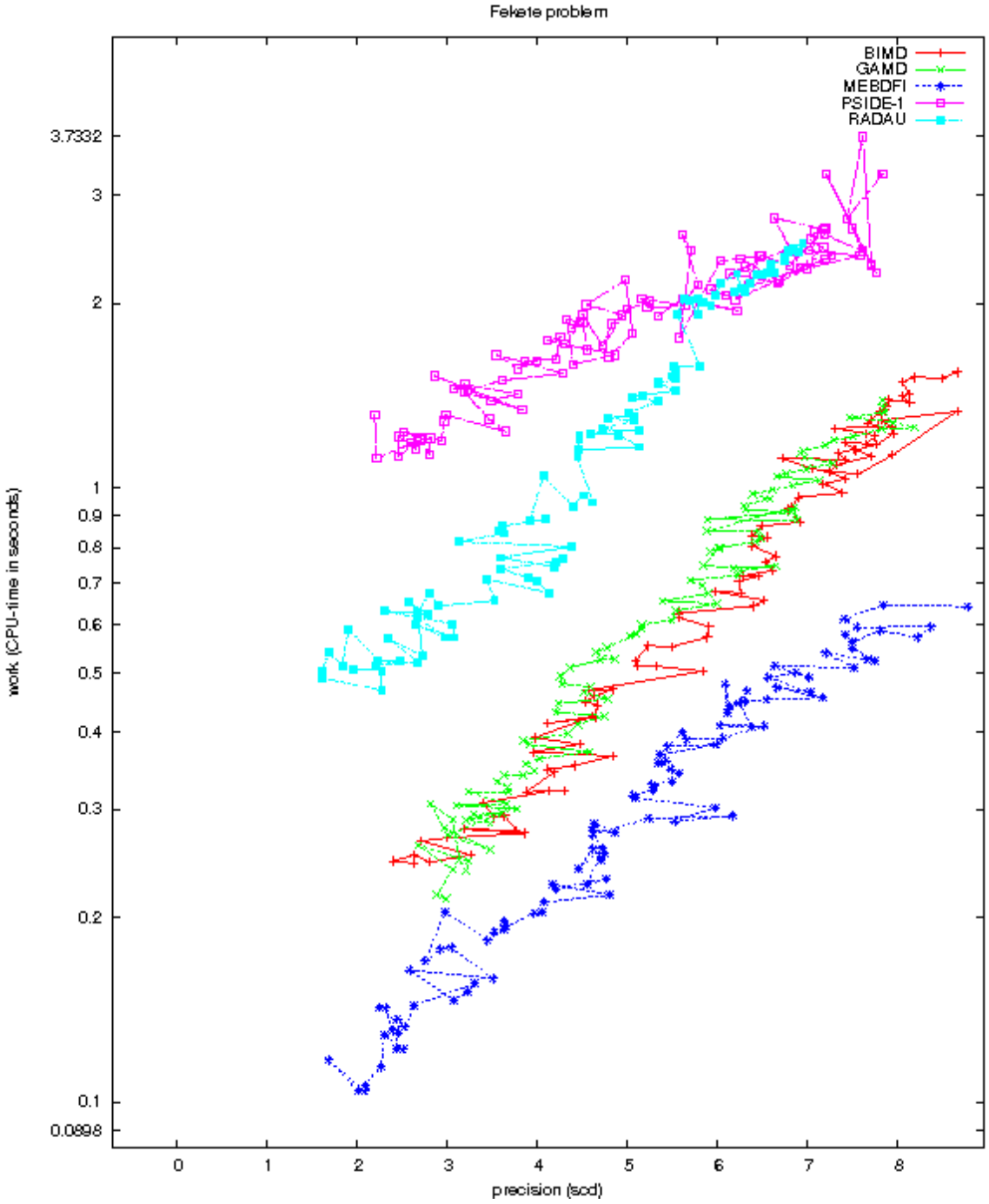

FIGURE II.18.3: *Work-precision diagram (scd versus CPU-time).*

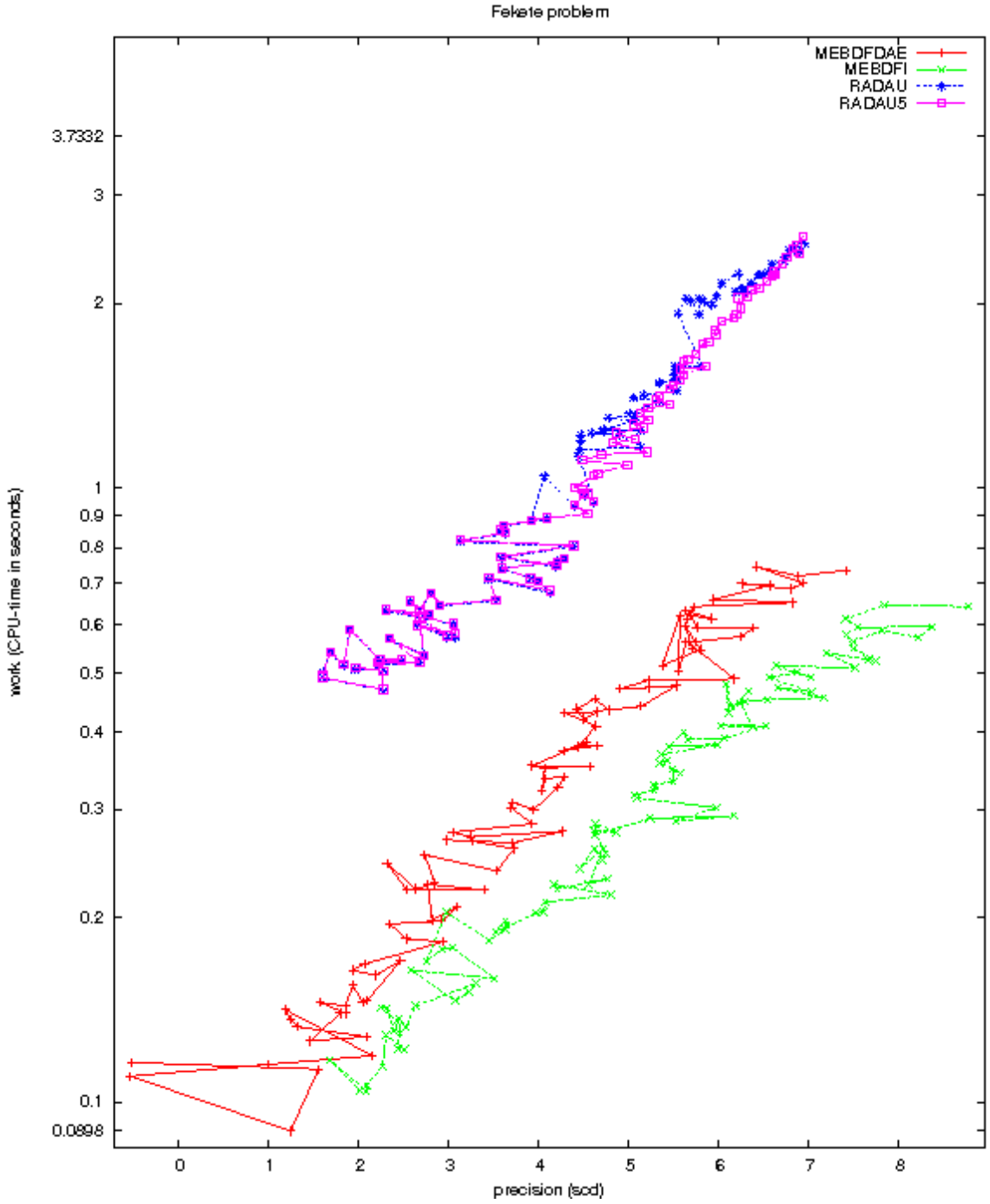

FIGURE II.18.4: *Work-precision diagram (scd versus CPU-time).*

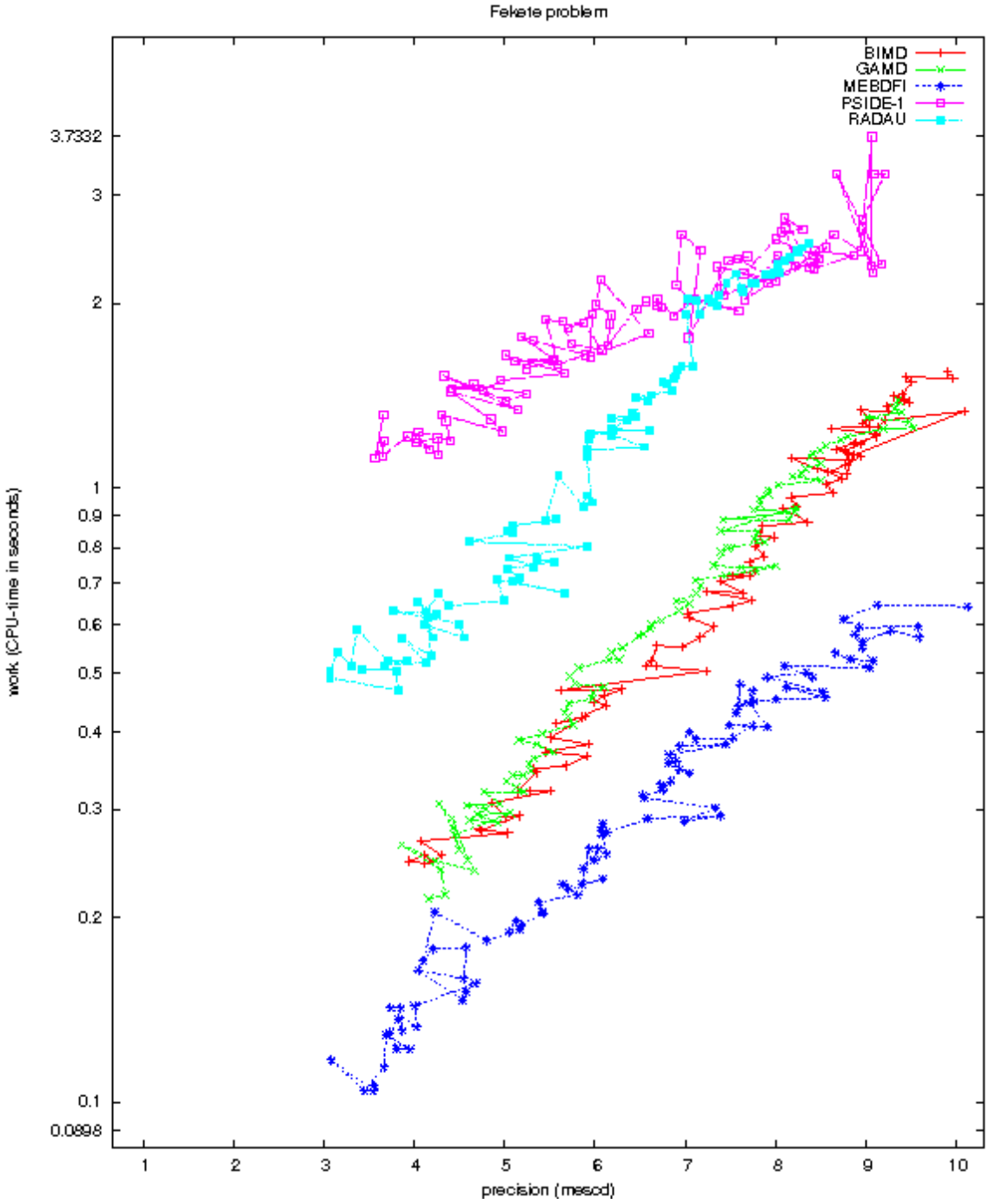

FIGURE II.18.5: *Work-precision diagram (mescd versus CPU-time).*

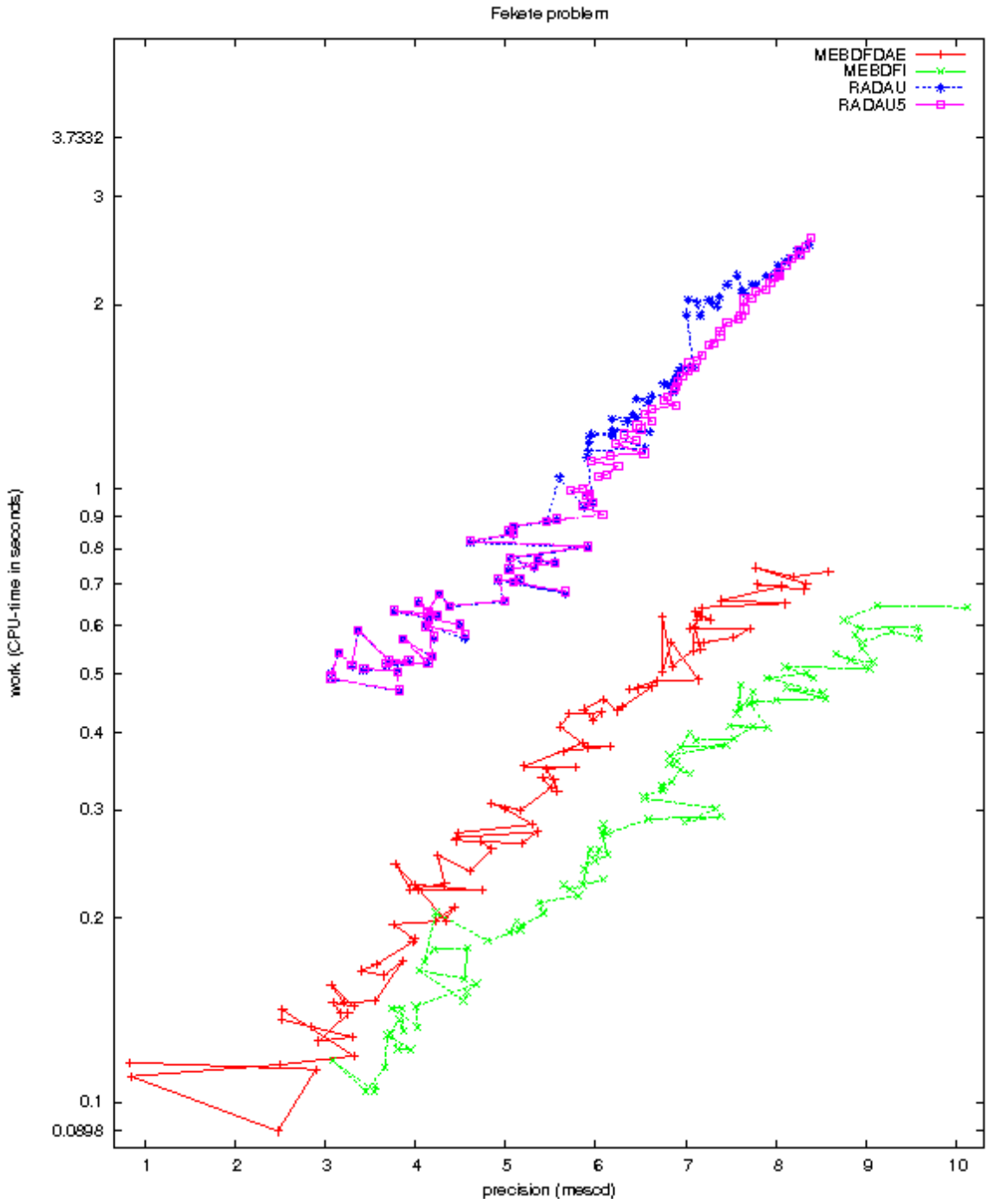

FIGURE II.18.6: *Work-precision diagram (mescd versus CPU-time).*

19 Slider Crank

19.1 General Information

This problem was contributed by Bernd Simeon, March 1998. The slider crank shows some typical properties of simulation problems in *flexible multibody systems*, i.e., constrained mechanical systems which include both rigid and elastic bodies. It is also an example of a *stiff mechanical system* since it features large stiffness terms in the right hand side. Accordingly, there are some fast variables with high frequency oscillations.

This problem is originally described by a second order system of differential-algebraic equations (DAEs), but transformed to first order and semi-explicit system of dimension 24. The index of the problem is originally 3, but an index 1 and index 2 formulation are supplied as well. By default, the subroutines provide the index 2 formulation.

Comments to `simeon@ma.tum.de`.

The software part of the problem is in the file crank.f available at [MM08].

19.2 Mathematical description of the problem

The original problem has the form

$$\begin{aligned} \boldsymbol{M}(p,q)\begin{pmatrix} \ddot{p} \\ \ddot{q} \end{pmatrix} &= \boldsymbol{f}(p,\dot{p},q,\dot{q}) - \boldsymbol{G}(p,q)^{\mathrm{T}}\lambda, \\ 0 &= \boldsymbol{g}(p,q) + \boldsymbol{r}(t), \end{aligned} \tag{II.19.1}$$

where $0 \le t \le 0.1$, $p \in I\!R^3$, $q \in I\!R^4$, $\lambda \in I\!R^3$, $\boldsymbol{M} : I\!R^7 \to I\!R^7 \times I\!R^7$, $\boldsymbol{f} : I\!R^{14} \to I\!R^7$, $\boldsymbol{g} : I\!R^7 \to I\!R^3$, $\boldsymbol{r} : I\!R \to I\!R^3$, and $\boldsymbol{G} = \partial \boldsymbol{g}/\partial(p,q)$. The matrix $\boldsymbol{M}(p,q)$ is symmetric positive semi-definite and rank $\boldsymbol{M}(p,q)$ is 3, which implies that the DAE (II.19.1) is of index 3. For the index 2 formulation, the position constraints are replaced by the velocity constraints

$$0 = \frac{\mathrm{d}}{\mathrm{d}t}\Big(\boldsymbol{g}(p,q) + \boldsymbol{r}(t)\Big) = \boldsymbol{G}(p,q)\begin{pmatrix} \dot{p} \\ \dot{q} \end{pmatrix} + \dot{\boldsymbol{r}}(t). \tag{II.19.2}$$

Additionally, the system is transformed to first order and semi explicit form

$$\begin{aligned} \begin{pmatrix} \dot{p} \\ \dot{q} \end{pmatrix} &= \begin{pmatrix} v_p \\ v_q \end{pmatrix}, \\ \begin{pmatrix} \dot{v}_p \\ \dot{v}_q \end{pmatrix} &= \begin{pmatrix} a_p \\ a_q \end{pmatrix}, \\ 0 &= \boldsymbol{M}(p,q)\begin{pmatrix} a_p \\ a_q \end{pmatrix} - \boldsymbol{f}(p,v_p,q,v_q) + \boldsymbol{G}(p,q)^{\mathrm{T}}\lambda\,, \\ 0 &= \boldsymbol{G}(p,q)\begin{pmatrix} v_p \\ v_q \end{pmatrix} + \dot{\boldsymbol{r}}(t)\,, \end{aligned} \tag{II.19.3}$$

which increases the dimension of the problem to 24. If we define $y := (p,q,v_p,v_q,a_p,a_q,\lambda)^{\mathrm{T}}$, then the consistent values are given by $y(0) := y_0$ and $y'(0) := y_0'$. The components of y_0 are zero, except for

$y_{0,3}$	$0.450016933 \cdot 10^{0}$	$y_{0,16}$	$-1.344541576709835 \cdot 10^{-3}$
$y_{0,6}$	$0.103339863 \cdot 10^{-4}$	$y_{0,17}$	$-5.062194924490193 \cdot 10^{3}$
$y_{0,7}$	$0.169327969 \cdot 10^{-4}$	$y_{0,18}$	$-6.829725665986310 \cdot 10^{-5}$
$y_{0,8}$	$0.150000000 \cdot 10^{3}$	$y_{0,19}$	$1.813207639590617 \cdot 10^{-20}$
$y_{0,9}$	$-0.7499576703969453 \cdot 10^{2}$	$y_{0,20}$	$-4.268463266810281 \cdot 10^{0}$
$y_{0,10}$	$-0.2689386719979040 \cdot 10^{-5}$	$y_{0,21}$	$2.098339029337557 \cdot 10^{-1}$
$y_{0,11}$	$0.4448961125815990 \cdot 10^{0}$	$y_{0,22}$	$-6.552727150584648 \cdot 10^{-8}$
$y_{0,12}$	$0.4634339319238670 \cdot 10^{-2}$	$y_{0,23}$	$3.824589509350831 \cdot 10^{2}$
$y_{0,13}$	$-0.1785910760000550 \cdot 10^{-5}$	$y_{0,24}$	$-4.635908708561371 \cdot 10^{-9}$
$y_{0,14}$	$-0.2689386719979040 \cdot 10^{-5}$		

The first 14 components of y_0' read $y_{0,i}' = y_{0,i+7}$, $i = 1, \ldots, 14$; the last 10 are zero.

For the index 2 formulation, the index of the variables p, q, v_p and v_q equals 1 and that of a_p, a_q and λ equals 2. The equations are given in detail in the next subsections, in which already some references to the origin of the problem, treated in §19.3, are given.

19.2.1 Equations of motion

The position or gross motion coordinates p are

$$p := \begin{pmatrix} \phi_1 \\ \phi_2 \\ x_3 \end{pmatrix} \qquad \begin{matrix} \text{crank angle} \\ \text{connecting rod angle} \\ \text{sliding block displacement} \end{matrix}$$

The deformation coordinates q (of the elastic connecting rod, see below) are

$$q := \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} \qquad \begin{matrix} \text{first lateral mode} \sin(\pi x/l_2) \\ \text{second lateral mode} \sin(2\pi x/l_2) \\ \text{longitudinal displacement midpoint} \\ \text{longitudinal displacement endpoint} \end{matrix}$$

The mass matrix $\boldsymbol{M}$ reads

$$\boldsymbol{M}(p,q) = \begin{pmatrix} M_r(p) + M_e(p,q) & C(p,q)^{\mathrm{T}} \\ C(p,q) & M_\Delta \end{pmatrix}$$

with rigid motion mass matrix

$$M_r(p) = \begin{pmatrix} J_1 + m_2 l_1^2 & 1/2\, l_1 l_2 m_2 \cos(\phi_1 - \phi_2) & 0 \\ 1/2\, l_1 l_2 m_2 \cos(\phi_1 - \phi_2) & J_2 & 0 \\ 0 & 0 & m_3 \end{pmatrix},$$

coupling blocks

$$M_e(p,q) = \begin{pmatrix} 0 & \rho l_1 (\cos(\phi_1 - \phi_2) c_1^{\mathrm{T}} + \sin(\phi_1 - \phi_2) c_2^{\mathrm{T}}) q & 0 \\ \rho l_1 (\cos(\phi_1 - \phi_2) c_1^{\mathrm{T}} + \sin(\phi_1 - \phi_2) c_2^{\mathrm{T}}) q & q^{\mathrm{T}} M_\Delta q + 2\rho c_{12}^{\mathrm{T}} q & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

and

$$C(p,q)^{\mathrm{T}} = \begin{pmatrix} \rho l_1 (-\sin(\phi_1 - \phi_2) c_1^{\mathrm{T}} + \cos(\phi_1 - \phi_2) c_2^{\mathrm{T}}) \\ \rho c_{21}^{\mathrm{T}} + \rho q^{\mathrm{T}} B \\ 0^{\mathrm{T}} \end{pmatrix},$$

and elastic body space discretization mass matrix

$$M_\Delta = \rho\, d\, h\, l_2 \begin{pmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 8 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix}.$$

The forces are given by

$$\boldsymbol{f}(p,\dot{p},q,\dot{q}) = \begin{pmatrix} f_r(p,\dot{p}) + f_e(p,\dot{p},q,\dot{q}) \\ f_\Delta(p,\dot{p},q,\dot{q}) - \text{grad } W_\Delta(q) - D_\Delta \dot{q} \end{pmatrix},$$

where the rigid motion terms are collected in

$$f_r(p,\dot{p}) = \begin{pmatrix} -1/2\, l_1(\gamma(m_1+2m_2)\cos\phi_1 + l_2 m_2 \dot{\phi}_2^2 \sin(\phi_1-\phi_2)) \\ -1/2\, l_2 \gamma m_2 \cos\phi_2 + 1/2\, l_1 l_2 m_2 \dot{\phi}_1^2 \sin(\phi_1-\phi_2) \\ 0 \end{pmatrix}.$$

For the force term $f_e(p,\dot{p},q,\dot{q})$ we have

$$\begin{pmatrix} \rho l_1 \dot{\phi}_2^2(-\sin(\phi_1-\phi_2)c_1^{\mathrm{T}} + \cos(\phi_1-\phi_2)c_2^{\mathrm{T}})q - 2\rho l_1 \dot{\phi}_2(\cos(\phi_1-\phi_2)c_1^{\mathrm{T}} + \sin(\phi_1-\phi_2)c_2^{\mathrm{T}})\dot{q} \\ \rho l_1 \dot{\phi}_1^2(\sin(\phi_1-\phi_2)c_1^{\mathrm{T}} - \cos(\phi_1-\phi_2)c_2^{\mathrm{T}})q - 2\rho\dot{\phi}_2 c_{12}^{\mathrm{T}}\dot{q} - 2\dot{\phi}_2 \dot{q}^{\mathrm{T}} M_\Delta q \\ -\rho\dot{q}^{\mathrm{T}} B \dot{q} - \rho\gamma(\cos\phi_2 c_1^{\mathrm{T}} q - \sin\phi_2 c_2^{\mathrm{T}} q) \\ 0 \end{pmatrix},$$

and for $f_\Delta(p,\dot{p},q,\dot{q})$ the expression

$$\dot{\phi}_2^2 M_\Delta q + \rho\Big(\dot{\phi}_2^2 c_{12} + l_1\dot{\phi}_1^2(\cos(\phi_1-\phi_2)c_1 + \sin(\phi_1-\phi_2)c_2) + 2\dot{\phi}_2 B\dot{q}\Big) - \rho\gamma\Big(\sin\phi_2 c_1 + \cos\phi_2 c_2\Big).$$

The gradient of the elastic potential $W_\Delta(q)$ in case of linear elasticity (which is the default) is grad $W_\Delta(q) = K_\Delta q$ with stiffness matrix

$$K_\Delta = E\, d\, h\, / l_2 \begin{pmatrix} \pi^4/24(h/l_2)^2 & 0 & 0 & 0 \\ 0 & \pi^4 2/3(h/l)^2 & 0 & 0 \\ 0 & 0 & 16/3 & -8/3 \\ 0 & 0 & -8/3 & 7/3 \end{pmatrix}.$$

Alternatively, in case of the nonlinear beam model (`IPAR(1) = 1`, see below), it holds grad $W_\Delta(q) = K_\Delta q + k_\Delta(q)$,

$$k_\Delta(q) = 1/2\,\pi^2\, E\, d\, h / l_2^2 \begin{pmatrix} q_1 q_4 - \beta q_2(-4q_3+2q_4) \\ 4q_2 q_4 - \beta q_1(-4q_3+2q_4) \\ 4\beta q_1 q_2 \\ 1/2 q_1^2 + 2q_2^2 - 2\beta q_1 q_2 \end{pmatrix}, \qquad \beta = 80/(9\pi^2).$$

The damping matrix D_Δ is by default zero. The coupling matrices and vectors arising from the space discretization read

$$B = d\, h\, l_2 \begin{pmatrix} 0 & 0 & -16/\pi^3 & 8/\pi^3 - 1/\pi \\ 0 & 0 & 0 & 1/(2\pi) \\ 16/\pi^3 & 0 & 0 & 0 \\ 1/\pi - 8/\pi^3 & -1/(2\pi) & 0 & 0 \end{pmatrix}$$

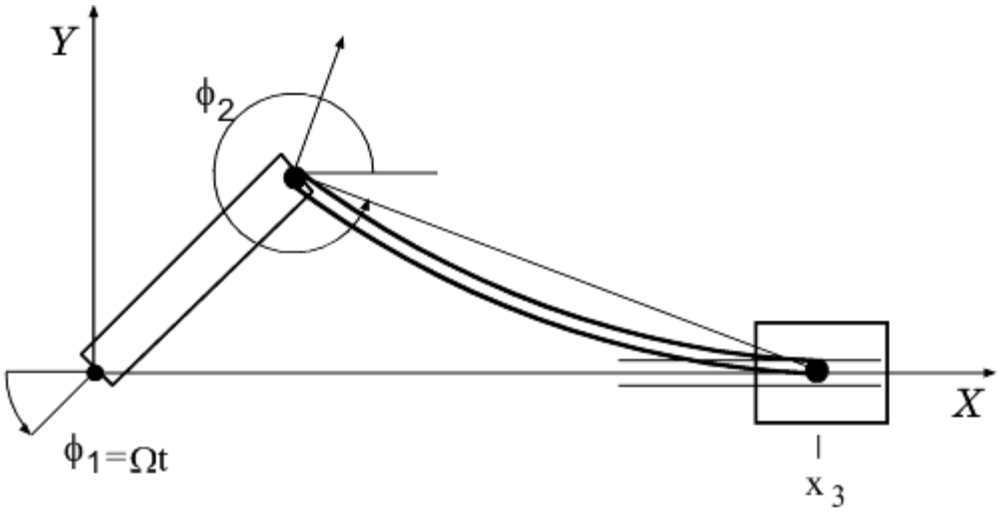

FIGURE II.19.1: The multibody system with crank, connecting rod, sliding block.

and

$$\begin{aligned} c_1 &= d\,h\,l_2(0,\,0,\,2/3,\,1/6)^{\mathrm{T}}, \\ c_2 &= d\,h\,l_2(2/\pi,\,0,\,0,\,0)^{\mathrm{T}}, \\ c_{12} &= d\,h\,l_2^2(0,\,0,\,1/3,\,1/6)^{\mathrm{T}}, \\ c_{21} &= d\,h\,l_2^2(1/\pi,\,-1/(2\pi),\,0,\,0)^{\mathrm{T}}. \end{aligned}$$

Finally, the position constraints $0 = \boldsymbol{g}(p,q) + \boldsymbol{r}(t)$ are given by

$$\begin{aligned} 0 &= l_1 \sin\phi_1 + l_2 \sin\phi_2 + q_4 \sin\phi_2, \\ 0 &= x_3 - l_1 \cos\phi_1 - l_2 \cos\phi_2 - q_4 \cos\phi_2, \\ 0 &= \phi_1 - \Omega t\,. \end{aligned}$$

19.2.2 Parameters

For the simulation, the following data are used:
The bodies have lengths $l_1 = 0.15$, $l_2 = 0.30[m]$.
The masses of the bodies are $m_1 = 0.36$, $m_2 = 0.151104$, $m_3 = 0.075552[kg]$.
The moments of inertia are $J_1 = 0.002727$, $J_2 = 0.0045339259[kg\,m^2]$.
The flexible connecting rod has height and width $h = d = 0.008[m]$.
The mass density $\rho = 7870[kg/m^3]$, and Young's modulus $E = 2.\cdot 10^{11}[N/m^2]$.
The gravity constant was set to zero since gravitation plays no role here, $\gamma = 0$.
The angular velocity of the prescribed crank motion is $\Omega = 150[rad/s]$.

19.3 Origin of the problem

The planar slider crank mechanism, see Figure II.19.1, consists of a rigid crank (body 1), an elastic connecting rod (body 2), a rigid sliding block (body 3) and two revolving and one translational joint. Koppens [Kop89] and Jahnke [JPD93] investigated this example using an ODE model in minimum coordinates. In [Sim96], an alternative DAE approach is introduced.

The mathematical model outlined above is derived in two steps. First, the elastic connecting rod is discretized in space. The geometry of the rod allows to apply an Euler-Bernoulli beam

$$\begin{aligned} u_1(x,y) &= w_1(x) - y w_2'(x), \\ u_2(x,y) &= w_2(x), \end{aligned}$$

to describe the longitudinal and lateral displacements u_1 and u_2 of material point (x, y) in the body-fixed coordinate system. For the longitudinal displacement w_1 of the neutral fiber, a simple quadratic model

$$w_1(x) \doteq \xi^2(-4q_3 + 2q_4) + \xi(4q_3 - q_4), \qquad \xi = x/l_2,$$

is sufficient to show the basic effects. The lateral displacement w_2 is approximated by the first two sinus shape functions

$$w_2(x) \doteq \sin(\pi\xi)q_1 + \sin(2\pi\xi)q_2\,.$$

These functions satisfy the boundary conditions $w_1(0) = 0$, $w_2(0) = 0$, $w_2(l_2) = 0$. Accordingly, the body-fixed coordinate system's origin is placed in $(x, y) = (0, 0)$, and its x-axis passes through the point $(l_2 + w_1(x), 0)$.

As already mentioned in §19.2, we provide two versions of the problem. The first one (default) assumes linear elasticity while the second takes the coupling of longitudinal and lateral displacements in terms of $k_\Delta(q)$ into account. Set `IPAR(1) = 1` to switch to this nonlinear beam model. See below for a comparison of the results.

In the second step, the equations of motion of the overall multibody system are assembled. Due to the choice of ϕ_2 as gross motion coordinate, there is no constraint equation necessary to express the revolving joint between crank and connecting rod. The revolving joint between sliding block and connecting rod and the translational joint lead to two constraints that depend on the deformation variable q_4. The third constraint equation defines the crank motion using $\boldsymbol{r}(t) = (0,\, 0,\, -\Omega t)^{\mathrm{T}}$. Here, other functions for the crank motion could also be prescribed.

The model described so far features no dissipation. Consequently, the solutions show a purely oscillatory behavior. We supply also a nonzero damping matrix D_Δ which can be activated by setting `IPAR(2) = 1`. Then, 0.5 percent dissipation is included in the right hand side of the elastic connecting rod.

In §19.4, we investigate the dynamic behavior of the slider crank model corresponding to the nonlinear model without damping with the initial values listed in §19.2, which were calculated such that the motion is almost smooth, using an asymptotic expansion technique [Sim97]. In Figure II.19.4 we see the behavior of the numerical solution for this setting of the model. A close look at these plots reveals that both lateral displacements q_1, q_2 as well as longitudinal displacements q_3, q_4 still show some small oscillations. The corresponding frequencies as solutions of the eigenvalue problem $\omega^2 M_\Delta q = K_\Delta q$ are

$$\omega_1 = 1277, \quad \omega_2 = 5107, \quad \omega_3 = 6841, \quad \omega_4 = 24613 \;\; [rad/s]\,.$$

In particular, q_3 and q_4 are characterized by the relatively large frequency ω_4. Any explicit discretization in time will need stepsizes smaller than the shortest period of oscillation, even for tracking a smooth solution. On the other hand, the challenge for implicit methods is to be able to take larger steps. In this simulation the gross motion coordinates p differ only slightly from the motion of a mechanism with rigid connecting rod.

The subroutines that describe the model offer several possibilities to test other variants of the model than those tested in §19.4. We now discuss some of them.

Oscillatory solution

We provide also a second set of initial values (`subroutine init2`) which lead to a strongly oscillatory solution. Here, the initial deformation as well as the corresponding velocity were set to zero, $q(0) =$

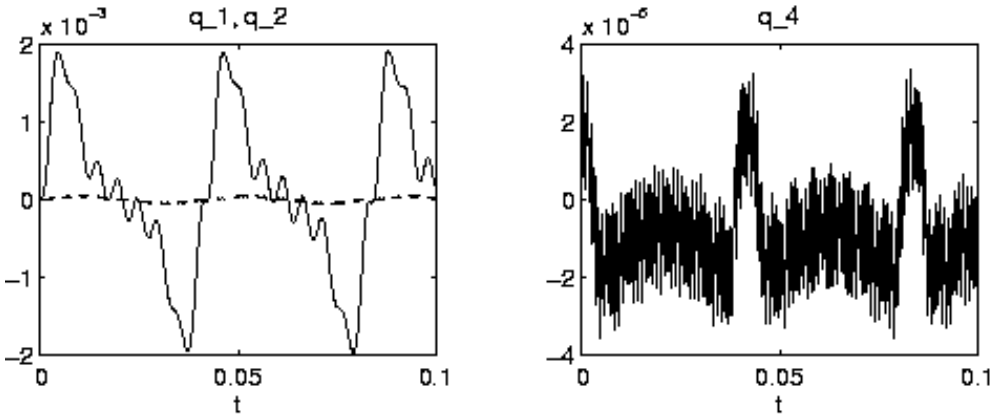

FIGURE II.19.2: Solution of slider crank for 'rigid' initial values, i.e., deformation $q(0) = v_q(0) = 0$.

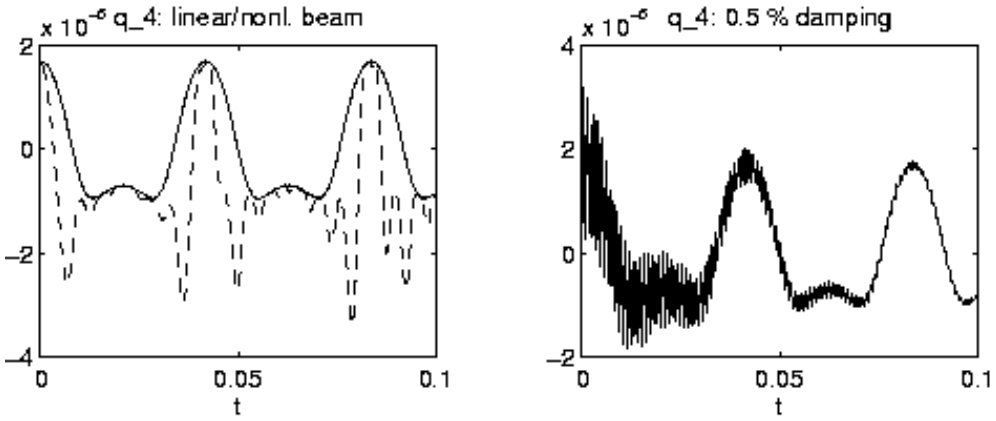

FIGURE II.19.3: Left: Comparison of linear and nonlinear beam model. Right: Oscillatory solution with physical damping.

$v_q(0) = 0$, which is equivalent to consistent initial values on a rigid motion trajectory. Figure II.19.2 plots the behavior of q_1, q_2 and q_4 for this setting. Both lateral and longitudinal modes oscillate now with different frequencies.

Nonlinear beam model and damping

The left and right plot in Figure II.19.3 show the effects of setting `IPAR(1) = 1` and `IPAR(2) = 1`, respectively. On the left, the difference between linear and nonlinear beam model is illustrated, with initial values close to the smooth motion. In particular, the components q_3 and q_4 change if the nonlinear model is employed. At points of maximum bending, the longitudinal displacement has now much smaller minima. If we increase the crank's angular velocity, the resulting forces acting on the connecting rod are much larger and we can then even observe how the sharp needles turn into a singularity, the buckling phenomenon.

On the right of Figure II.19.3, the damping was activated by `IPAR(2) = 1`, with initial values on a rigid motion trajectory (`init2`). Obviously, the oscillation shown in Figure II.19.2 on the right is now slowly damped out.

TABLE II.19.1: *Failed runs.*

solver	m	reason
MEBDFDAE	$19, \ldots, 24$	stepsize too small
MEBDFI	$21, 22, 23, 24$	stepsize too small
PSIDE-1	$17, 18, \ldots, 24$	iteration matrix singular
RADAU	24	core dump / overflow in decomposition
RADAU5	24	core dump / overflow in decomposition

TABLE II.19.2: *Reference solution at the end of the integration interval.*

y_1	$1.500000000000104 \cdot 10^{1}$	y_{13}	$4.974111734266989 \cdot 10^{-4}$
y_2	$-3.311734988256260 \cdot 10^{-1}$	y_{14}	$1.105560003626645 \cdot 10^{-3}$
y_3	$1.697373328427860 \cdot 10^{-1}$	y_{15}	0
y_4	$1.893192899613509 \cdot 10^{-4}$	y_{16}	$6.488737541276957 \cdot 10^{3}$
y_5	$2.375751249879174 \cdot 10^{-5}$	y_{17}	$2.167938629509884 \cdot 10^{3}$
y_6	$-5.323896770569702 \cdot 10^{-6}$	y_{18}	$3.391137060286523 \cdot 10^{1}$
y_7	$-8.363313279112129 \cdot 10^{-6}$	y_{19}	$1.715134772216488 \cdot 10^{-1}$
y_8	$1.500000000000000 \cdot 10^{2}$	y_{20}	$-1.422449408912512 \cdot 10^{0}$
y_9	$6.025346755138369 \cdot 10^{1}$	y_{21}	$1.003946428124810 \cdot 10^{0}$
y_{10}	$-8.753116326670527 \cdot 10^{0}$	y_{22}	$-6.232935833287916 \cdot 10^{1}$
y_{11}	$-3.005541400289738 \cdot 10^{-2}$	y_{23}	$-1.637920993367306 \cdot 10^{2}$
y_{12}	$-5.500431812571696 \cdot 10^{-3}$	y_{24}	$2.529857947066878 \cdot 10^{1}$

19.4 Numerical solution of the problem

The results presented here refer to index 2 formulation of the linear model without damping, using the initial values corresponding to a smooth solution.

Tables II.19.2–II.19.3 and Figures II.19.4–II.19.8 present the reference solution at the end of the integration interval, the run characteristics, the behavior of some of the solution components over the integration interval and the work-precision diagrams, respectively. In computing the scd values, only the first seven and the last three components were taken into account, since they refer to the physically important quantities. The reference solution was computed using MEBDFI with atol $= 10^{-14}$ and rtol $= 10^{-14}$ and h0 $= 10^{-12}$. For the work-precision diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, \ldots, 24$; atol = rtol; h0 $= 10^{-2} \cdot$ rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU5 and RADAU. The failed runs are in Table II.19.1; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

.

Remarks

- The slider crank is an example for a stiff mechanical system given in DAE form. See Lubich [Lub93] for an investigation of such systems and the implications for numerical methods in the ODE case.
- The nonlinear beam model leads to a higher computational effort but does not provoke convergence failures of Newton's method in RADAU5, as might be expected in case of nonlinear stiffness terms.

TABLE II.19.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-6}	0.23	2.50	102	102	1762	102	102	0.0420
	10^{-6}	10^{-6}	10^{-8}	0.39	3.38	1155	1155	22548	1155	1155	0.5144
	10^{-8}	10^{-8}	10^{-10}	2.50	5.49	992	992	35662	992	992	0.7086
GAMD	10^{-4}	10^{-4}	10^{-6}	0.23	2.28	60	60	1983	60	60	0.0342
	10^{-6}	10^{-6}	10^{-8}	−0.16	2.83	534	527	25206	527	534	0.4089
	10^{-8}	10^{-8}	10^{-10}	1.70	4.69	650	650	46109	650	650	0.7271
MEBDFI	10^{-4}	10^{-4}	10^{-6}	0.22	1.49	250	242	1593	28	28	0.0176
	10^{-6}	10^{-6}	10^{-8}	0.03	3.03	3328	3324	15099	170	170	0.2011
	10^{-8}	10^{-8}	10^{-10}	2.72	5.71	6316	6315	28395	313	313	0.3845
PSIDE-1	10^{-4}	10^{-4}		−0.05	0.93	45	41	858	29	180	0.0234
	10^{-6}	10^{-6}		0.16	2.43	259	235	5020	147	888	0.1298
	10^{-8}	10^{-8}		1.66	4.66	1639	1445	31526	54	2324	0.6412
RADAU	10^{-4}	10^{-4}	10^{-6}	0.20	1.90	104	92	717	89	104	0.0224
	10^{-6}	10^{-6}	10^{-8}	0.14	2.89	132	131	3367	123	131	0.0654
	10^{-8}	10^{-8}	10^{-10}	1.65	4.65	420	419	10589	397	414	0.2089

- As an alternative to stiff solvers, it is still possible to apply methods based on explicit discretizations, e.g., half-explicit or projection methods for constrained mechanical systems. The code MDOP5 [Sim95], a projection method based on DOPRI5, uses 2260 integration steps to solve this problem in the default setting, with atol = 10^{-6} and rtol = 10^{-5}, and initial values close to the smooth motion. Thus, the stiffness is no that severe in case of this carefully chosen one-dimensional elastic body model.
- There is also an extended version of the slider crank with a two-dimensional FE grid for the connecting rod. There, explicit methods do not work any longer. An animation of the system motion can be found at `http://www.mathematik.tu-darmstadt.de/~ simeon/` .

References

[JPD93] M. Jahnke, K. Popp, and B. Dirr. Approximate analysis of flexible parts in multibody systems using the finite element method. In Schiehlen W., editor, *Advanced Multibody System Dynamics*, pages 237–256, Stuttgart, 1993. Kluwer Academic Publishers.

[Kop89] W. Koppens. *The dynamics of systems of deformable bodies.* PhD thesis, Technische Universiteit Eindhoven, 1989.

[Lub93] C. Lubich. Integration of stiff mechanical systems by Runge-Kutta methods. *ZAMP*, 44:1022–1053, 1993.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at `http://www.dm.uniba.it/~testset`.

[Sim95] B. Simeon. MBSPACK - Numerical integration software for constrained mechanical motion. *Surv. on Math. in Ind.*, 5:169–202, 1995.

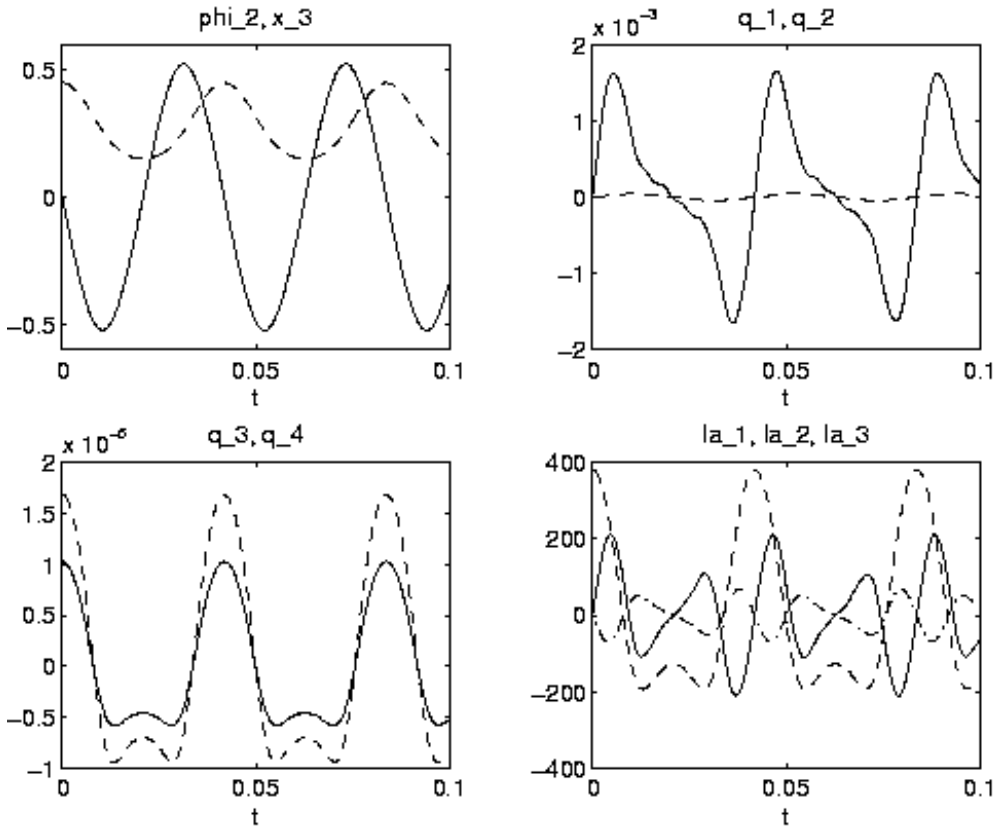

FIGURE II.19.4: *Behavior of the ith solution component;* $i \in \{2, 3, \ldots, 7, 22, 23, 24\}$.

[Sim96] B. Simeon. Modelling a flexible slider crank mechanism by a mixed system of DAEs and PDEs. *Math. Modelling of Systems*, 2:1–18, 1996.

[Sim97] B. Simeon. DAEs and PDEs in elastic multibody systems, 1997. *To appear in Numerical Algorithms.*

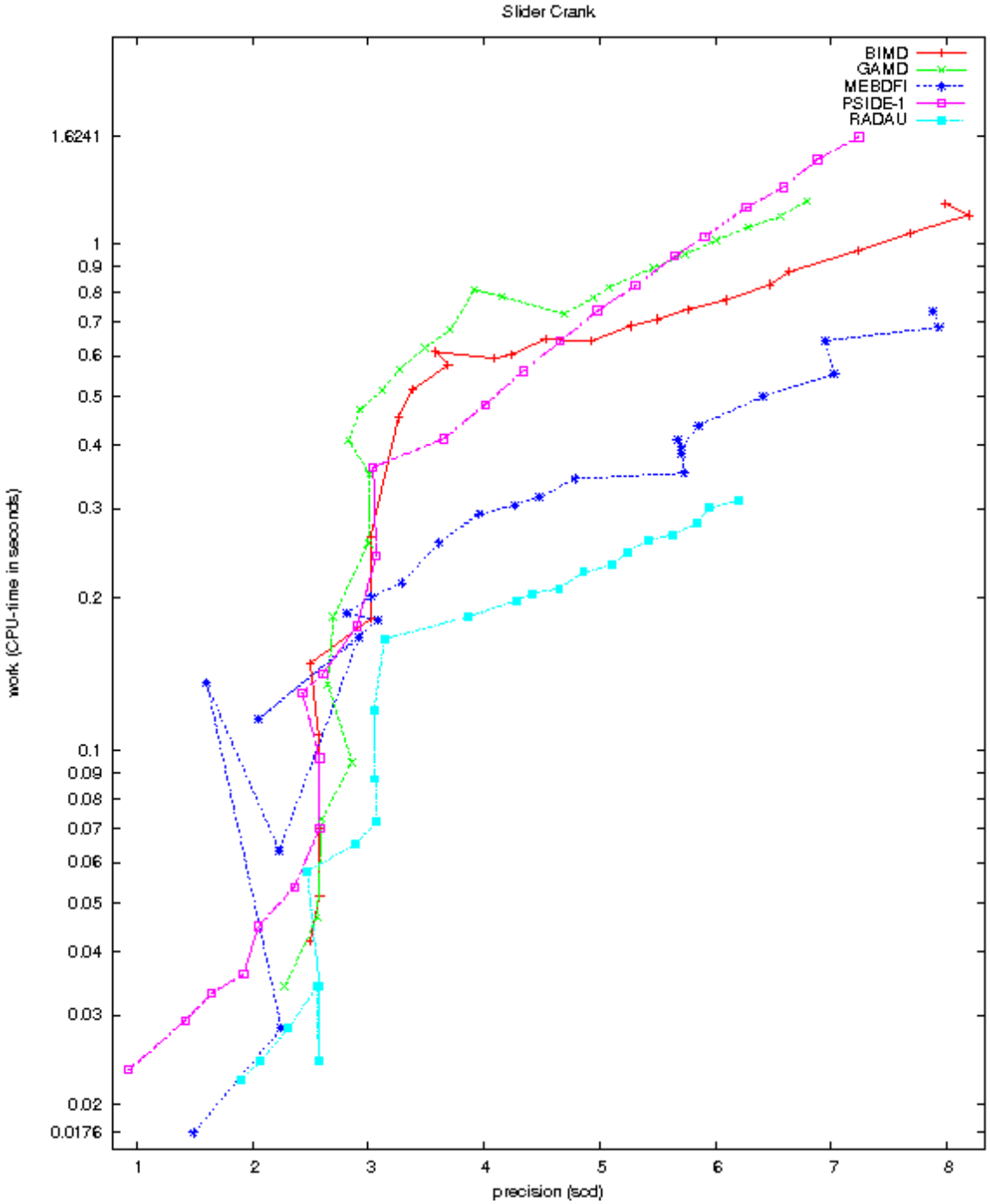

FIGURE II.19.5: *Work-precision diagram (scd versus CPU-time).*

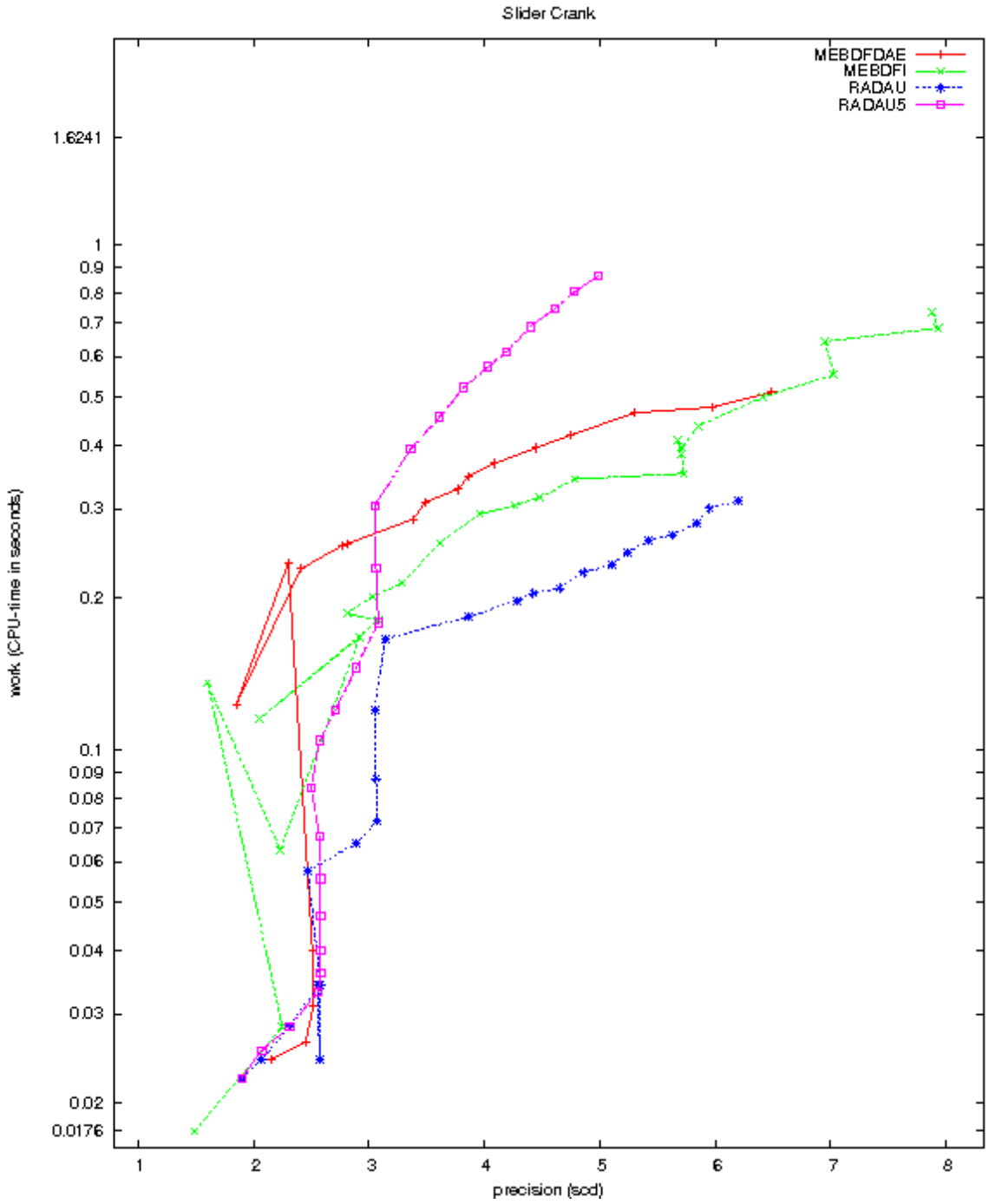

FIGURE II.19.6: *Work-precision diagram (scd versus CPU-time).*

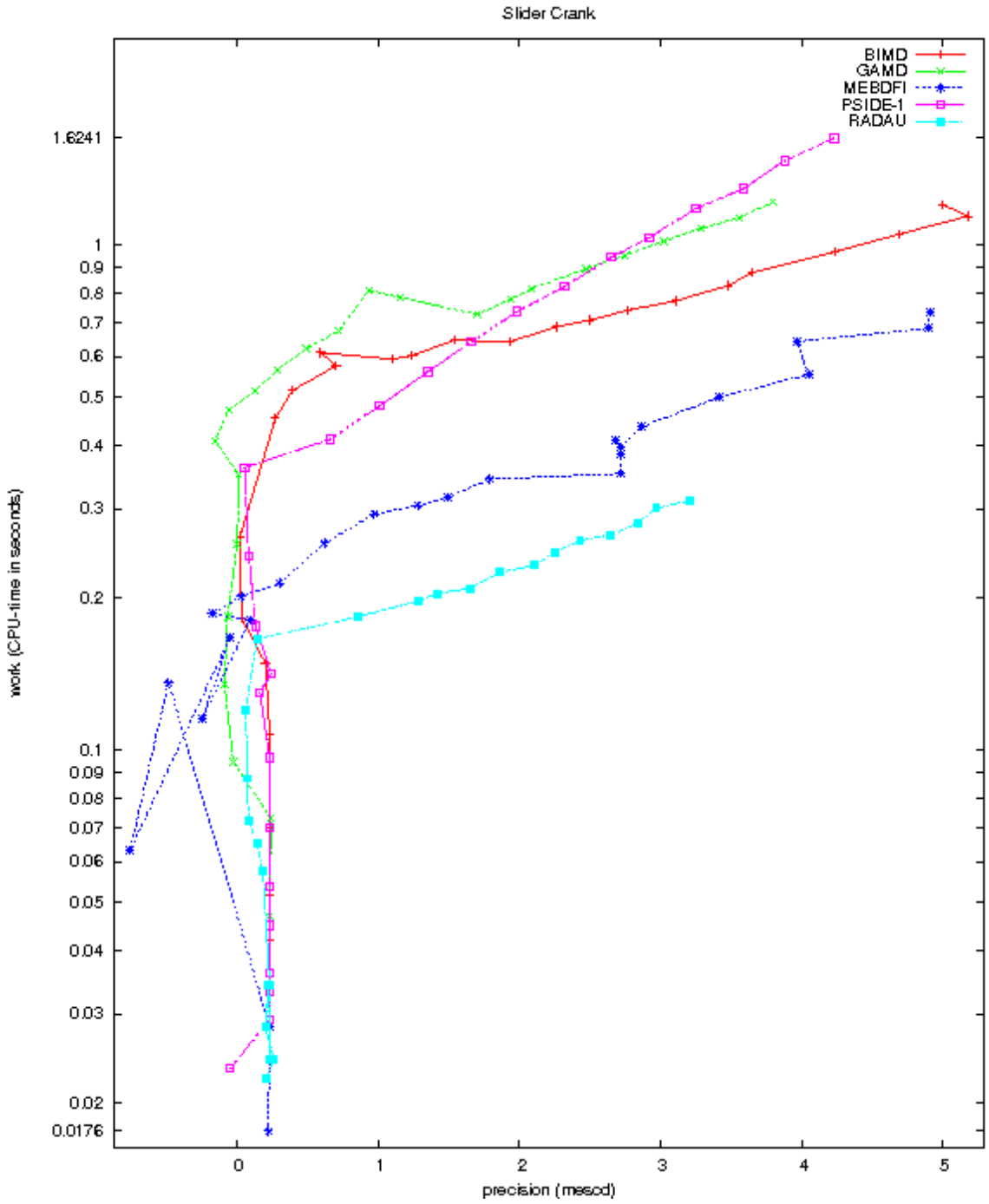

FIGURE II.19.7: *Work-precision diagram (mescd versus CPU-time).*

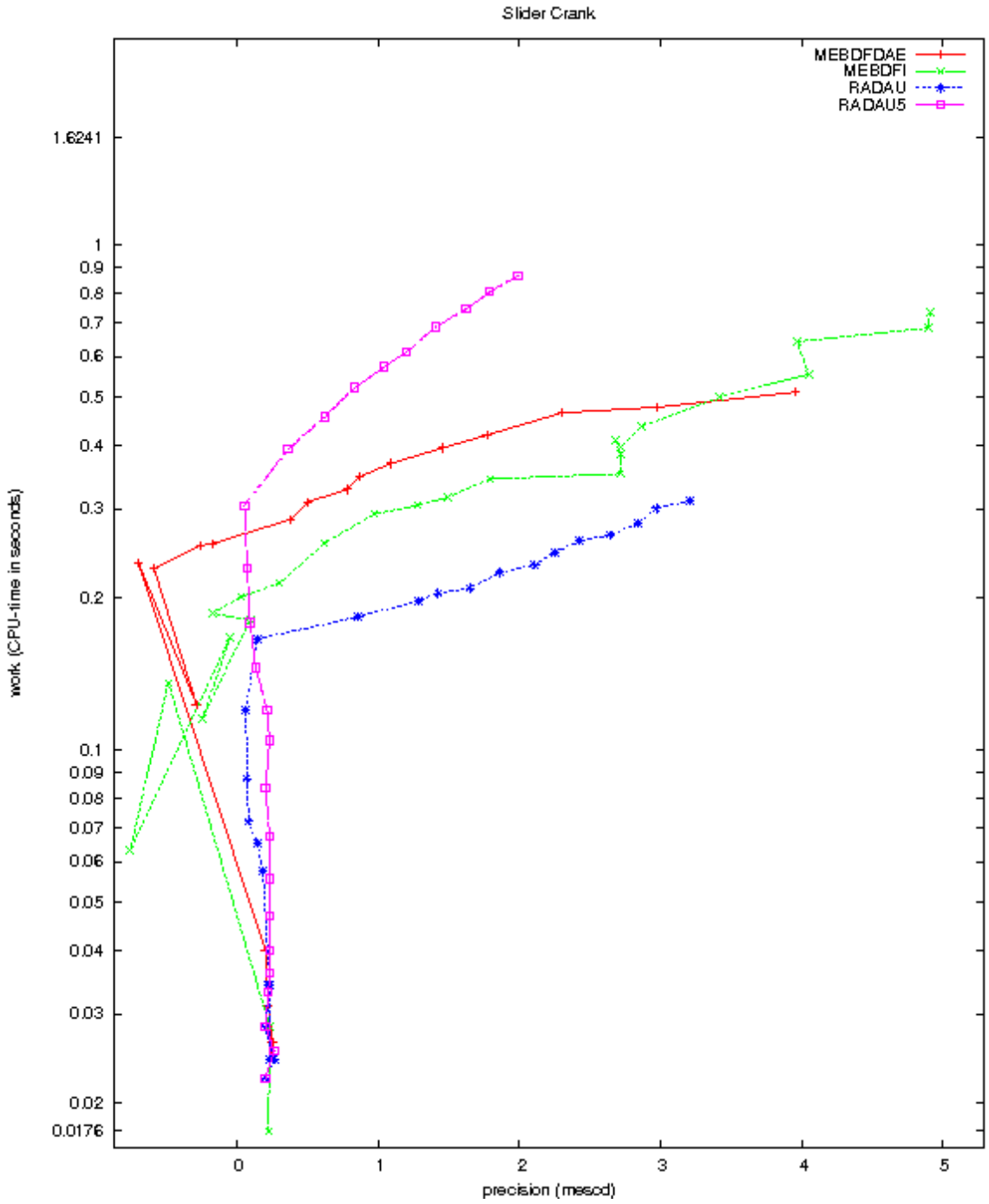

FIGURE II.19.8: *Work-precision diagram (mescd versus CPU-time).*

20 Water tube system

20.1 General information

This IVP is an index 2 system of 49 non-linear Differential-Algebraic Equations and describes the water flow through a tube system, taking into account turbulence and the roughness of the tube walls. The parallel-IVP-algorithm group of CWI contributed this problem to the test set in cooperation with B. Koren (CWI) and Paragon Decision Technology B.V.

The software part of the problem is in the file water.f available at [MM08].

20.2 Mathematical description of the problem

The problem is of the form

$$M\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \qquad y(0)=y_0, \quad y'(0)=y'_0, \tag{II.20.1}$$

where $0 \le t \le 17\cdot 3600$ and $y \in I\!R^{49}$. Furthermore,

$$M = \begin{bmatrix} M^\phi & O & O \\ O & O & O \\ O & O & M^p \end{bmatrix}, \tag{II.20.2}$$

where $M^\phi \in I\!R^{18\times 18}$ and $M^p \in I\!R^{13\times 13}$ are given by

$$M^\phi_{i,j} = \begin{cases} v_i & \text{for } i=j, \\ 0 & \text{otherwise.} \end{cases} \qquad M^p_{i,j} = \begin{cases} C_5 & \text{for } i=j=1, \\ C_8 & \text{for } i=j=2, \\ 0 & \text{otherwise,} \end{cases}$$

The first 38 components of y are of index 1, the last 11 are of index 2. For the definition of f and the values of C_5, C_8 and v we refer to §20.3.

The initial vectors y_0 and y'_0 are given by

$$y_0 = \begin{cases} 0 & \text{for} \quad i=1,2,\ldots,18 \\ 0.047519404529185289807 & \text{for} \quad i=19,20,\ldots,36 \\ 109800 & \text{for} \quad i=37,38,\ldots,49 \end{cases} \quad \text{and} \quad y'_0 = (0,\ldots,0)^{\mathrm{T}}. \tag{II.20.3}$$

The function f contains several square roots. It is clear that the function can not be evaluated if one of the arguments of one of these square roots becomes negative. To prevent this situation, we set `IERR=-1` in the Fortran subroutine that defines f if this happens. See page *IV-ix* of the the description of the software part of the test set for more details on `IERR`.

20.3 Origin of the problem

This test example describes how water flows through a water tube system. The system is represented by a set of nodes, which are connected by tubes. The structure of the water tube system is depicted in Figure II.20.1. There are two types of nodes: normal nodes and buffer nodes, to which a buffer is attached. We denote the set of all nodes by $\mathcal{N}$, and the set of buffer nodes by $\mathcal{B}$. For the system under consideration, $\mathcal{B} = \{5, 8\}$. The rectangles in Figure II.20.1 represent the buffers. The pipes are in the horizontal plane; the buffers are connected to the nodes perpendicular to this plane. The pipes from the buffer nodes to the rectangles are virtual; in reality the buffers are directly attached to the buffer nodes. In the model every node can have inflow and outflow, which are denoted by $e_i^{\mathrm{in}}(t)$ and $e_i^{\mathrm{out}}(t)$. In our example, inflow occurs only at node 1 and node 13, whereas only node 10 has outflow.

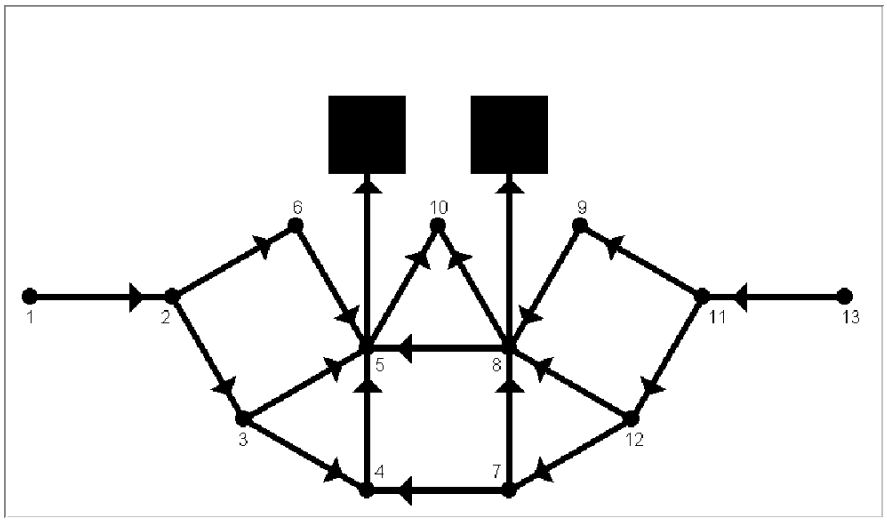

FIGURE II.20.1: *Structure of water tube system.*

The unit of time in the model is second. Defining the time in hours by $\hat{t} = t/3600$, these flows are defined by

$$\begin{aligned} e_1^{\text{in}}(t) &= (1 - \cos(\mathrm{e}^{-\hat{t}} - 1))/200, \\ e_{13}^{\text{in}}(t) &= (1 - \cos(\mathrm{e}^{-\hat{t}} - 1))/80, \\ e_{10}^{\text{out}}(t) &= \hat{t}^2(3\hat{t}^2 - 92\hat{t} + 720)/10^6. \end{aligned}$$

Figure II.20.2 shows plots of these flows as function of $\hat{t}$. Note that the outflow has a peak at 8 AM and is increasing again after 3 PM.

Although it seems that node 6 and node 9 could be omitted, we include them in the model, to leave open the possibility that these nodes have inflow or outflow. The arrows in Figure II.20.1 denote the direction in which we compute the flows. For example, if there is a flow from node 4 to node 3, then this flow will be negative.

To model the flow of the water, we introduce some symbols, which are listed in Table II.20.1. The roughness $k_{i,j} = 2 \cdot 10^{-4}$ is measured as the average height of the obstacles on the tube wall. The structure $S_{i,j}$ is defined as

$$S_{i,j} = \begin{cases} 1 & \text{if there is a tube from } i \text{ to } j, \\ 0 & \text{otherwise.} \end{cases}$$

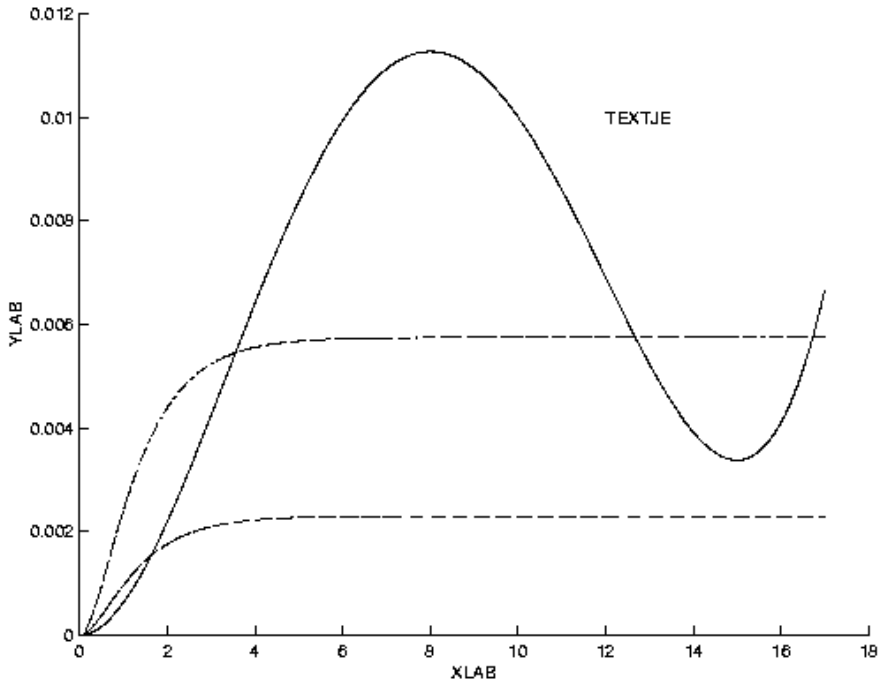

FIGURE II.20.2: *Inflows and outflow in m^3/s as function of time in hours.*

From Figure II.20.1 we see that

$$S = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Some of the quantities in Table II.20.1 can be computed directly from others:

$$\begin{aligned} \mu &= \nu \cdot \rho, \\ \phi_{i,j}(t) &= u_{i,j}(t) \cdot A_{i,j}, \\ A_{i,j} &= \pi \cdot d_{i,j}^2/4, \\ m_{i,j} &= A_{i,j} \cdot l_{i,j} \cdot \rho, \\ R_{i,j}(t) &= u_{i,j}(t) \cdot d_{i,j}/\nu. \end{aligned}$$

The definition of $R_{i,j}(t)$ was taken from [Sch78, p. 816].

TABLE II.20.1: *List of symbols for modeling flow in tubes.*

Symbol	Unit	Meaning
$\phi_{i,j}(t)$	m^3/s	flow through tube from i to j at time t
$u_{i,j}(t)$	m/s	mean velocity of flow through tube from i to j at time t
$F_{i,j}(t)$	N	total force on water in tube from i to j at time t
$F^{\mathrm{a}}_{i,j}(t)$	N	adhesion force on water in tube from i to j at time t
$\lambda_{i,j}(t)$	-	coefficient of resistance of tube from i to j at time t
$R_{i,j}(t)$	-	Reynolds number of flow through tube from i to j at time t
$p_i(t)$	N/m^2	pressure in i at time t
$S_{i,j}$	-	incidence matrix for structure of the tube system
$m_{i,j}$	kg	mass of water in tube from i to j
$d_{i,j}$	m	diameter of tube from i to j
$l_{i,j}$	m	length of tube from i to j
$A_{i,j}$	m^2	area of tube from i to j
$k_{i,j}$	m	roughness of wall of tube from i to j
$e_i^{\mathrm{in}}(t)$	m^3/s	inflow at i at time t
$e_i^{\mathrm{out}}(t)$	m^3/s	outflow at i at time t
B_i $(i \in \mathcal{B})$	m^2	area of buffer i
R^{crit}	-	critical Reynolds number
g	m/s^2	gravity constant
ρ	kg/m^3	density of water
μ	$kg/(m \cdot s)$	viscosity of water
ν	m^2/s	kinematic viscosity of water
v	kg/m^4	auxiliary vector, see (II.20.15)

We now explain how to model the flow through a tube, using Newton's second Law, which states that

$$m_{i,j}\frac{\mathrm{d}u_{i,j}(t)}{\mathrm{d}t} = F_{i,j}(t). \tag{II.20.4}$$

Assuming that gravity has no influence on the water flow in all tubes (remember that the pipes are

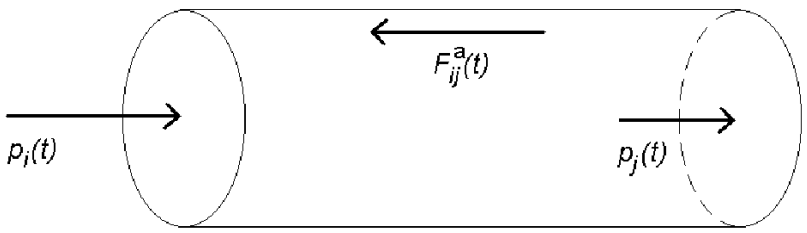

FIGURE II.20.3: *Forces on water in tube.*

in the horizontal plane), we see from Figure II.20.3 that the total force on the water in a tube equals

$$F_{i,j}(t) = A_{i,j}(p_i(t) - p_j(t)) - F^{\mathrm{a}}_{i,j}(t). \tag{II.20.5}$$

The magnitude of the adhesion force depends on the type of flow. For laminar flows ($|R_{i,j}(t)| \le R^{\text{crit}}$), we use the formula [Sch78, p. 12]

$$F^{\text{a}}_{i,j}(t)/A_{i,j} = 32\mu \cdot l_{i,j} \cdot u_{i,j}(t)/d^2_{i,j}. \tag{II.20.6}$$

For turbulent flows ($|R_{i,j}(t)| > R^{\text{crit}}$), we have [Sch78, p. 597]

$$F^{\text{a}}_{i,j}(t)/A_{i,j} = \lambda_{i,j}(t) \cdot \rho \cdot l_{i,j} \cdot u_{i,j}(t)^2/d_{i,j}, \tag{II.20.7}$$

where the resistance $\lambda_{i,j}(t)$ is computed from Colebrook and White's formula [Sch78, p. 621]:

$$0 = \frac{1}{\sqrt{\lambda_{i,j}(t)}} - 1.74 + 2\log\left(\frac{2k_{i,j}}{d_{i,j}} + \frac{18.7}{|R_{i,j}(t)|\sqrt{\lambda_{i,j}(t)}}\right). \tag{II.20.8}$$

Although for laminar flows the adhesion force does not depend on the resistance coefficient (cf. (II.20.6)), we have to choose a value for $\lambda_{i,j}$ in case of laminar flows. We compute this value by replacing $R_{i,j}$ in (II.20.8) by R^{crit}, i.e., we choose the value such that if a flow changes from laminar into turbulent, the resistance coefficient changes gradually.

For the normal nodes, Kirchoff's law holds, which states that

$$\forall\, n \in \mathcal{N} - \mathcal{B}: \qquad 0 = \sum_{i|S_{i,n}=1} \phi_{i,n}(t) + e^{\text{in}}_n(t) - \sum_{j|S_{n,j}=1} \phi_{n,j}(t) - e^{\text{out}}_n(t) \tag{II.20.9}$$

For the buffer nodes, we add a term $\psi_n(t)$ that represents the flow to the buffer:

$$\forall\, n \in \mathcal{B}: \qquad \psi_n(t) = \sum_{i|S_{i,n}=1} \phi_{i,n}(t) + e^{\text{in}}_n(t) - \sum_{j|S_{n,j}=1} \phi_{n,j}(t) - e^{\text{out}}_n(t) \tag{II.20.10}$$

We now explain how to compute $\psi_n(t)$. A buffer can be interpreted as the water column in Figure II.20.4, with ground area B_n and height h. Due to the flow $\psi_n(t)$ the height of the buffer changes at a rate $\psi_n(t)/B_n$. The difference between the pressure at the top and bottom of the column satisfies

$$p_n - p_0 = g \cdot \rho \cdot h.$$

Consequently, the pressure difference changes at a rate given by

$$\frac{\mathrm{d}(p_n - p_0)}{\mathrm{d}t} = g \cdot \rho \cdot \frac{\mathrm{d}h}{\mathrm{d}t} = g \cdot \rho \cdot \frac{\psi_n(t)}{B_n}. \tag{II.20.11}$$

Notice that the pressure p_0 is constant and therefore drops out in this formula. Substituting (II.20.11) in (II.20.10) gives

$$\forall\, n \in \mathcal{B}: \qquad C_n \frac{\mathrm{d}p_n(t)}{\mathrm{d}t} = \sum_{i|S_{i,n}=1} \phi_{i,n}(t) + e^{\text{in}}_n(t) - \sum_{j|S_{n,j}=1} \phi_{n,j}(t) - e^{\text{out}}_n(t), \tag{II.20.12}$$

where the quantity $C_n := B_n/(\rho \cdot g)$ can be interpreted as the capacity of the buffer at node n.

We arrive at the formulation in §20.2 by setting

$$\begin{aligned} y = (\quad & \phi_{1,2}(t), \phi_{2,3}(t), \phi_{2,6}(t), \phi_{3,4}(t), \phi_{3,5}(t), \phi_{4,5}(t), \phi_{5,10}(t), \phi_{6,5}(t), \phi_{7,4}(t), \\ & \phi_{7,8}(t), \phi_{8,5}(t), \phi_{8,10}(t), \phi_{9,8}(t), \phi_{11,9}(t), \phi_{11,12}(t), \phi_{12,7}(t), \phi_{12,8}(t), \phi_{13,11}(t), \\ & \lambda_{1,2}(t), \lambda_{2,3}(t), \lambda_{2,6}(t), \lambda_{3,4}(t), \lambda_{3,5}(t), \lambda_{4,5}(t), \lambda_{5,10}(t), \lambda_{6,5}(t), \lambda_{7,4}(t), \\ & \lambda_{7,8}(t), \lambda_{8,5}(t), \lambda_{8,10}(t), \lambda_{9,8}(t), \lambda_{11,9}(t), \lambda_{11,12}(t), \lambda_{12,7}(t), \lambda_{12,8}(t), \lambda_{13,11}(t), \\ & p_5(t), p_8(t), p_1(t), p_2(t), \ldots, p_4(t), p_6(t), p_7(t), p_9(t), p_{10}(t), \ldots, p_{13}(t) \qquad)^{\text{T}}. \end{aligned} \tag{II.20.13}$$

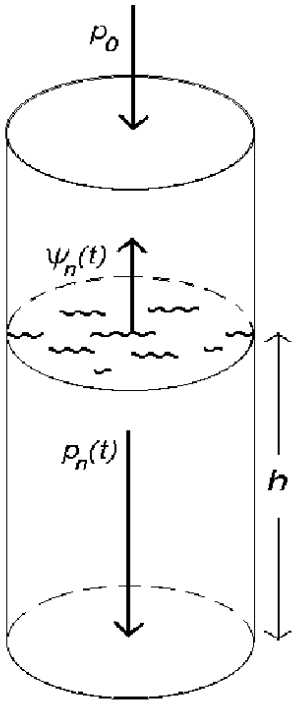

FIGURE II.20.4: *Representation of water buffer.*

All pressures are of index 2, except for those at the buffer nodes. The reordering of the pressures in (II.20.13) is such that the elements in y appear in order of increasing index, as required by RADAU, RADAU5 and MEBDFDAE.

The first 18 equations in (II.20.1) are obtained by first substituting (II.20.5) in (II.20.4). Next, we divide both sides by $A_{i,j}$, thus yielding

$$\frac{\rho \cdot l_{i,j}}{A_{i,j}} \frac{\mathrm{d}\phi_{i,j}(t)}{\mathrm{d}t} = p_i(t) - p_j(t) - F^{\mathrm{a}}_{i,j}(t)/A_{i,j}. \tag{II.20.14}$$

Finally, (II.20.6) and (II.20.7) are substituted in (II.20.14). Consequently, if we define $V_{i,j} = \rho \cdot l_{i,j}/A_{i,j}$, then the vector v in (II.20.2) is given by

$$\begin{aligned} v = (\quad & V_{1,2}, V_{2,3}, V_{2,6}, V_{3,4}, V_{3,5}, V_{4,5}, V_{5,10}, V_{6,5}, V_{7,4}, \\ & V_{7,8}, V_{8,5}, V_{8,10}, V_{9,8}, V_{11,9}, V_{11,12}, V_{12,7}, V_{12,8}, V_{13,11} \quad)^{\mathrm{T}}. \end{aligned} \tag{II.20.15}$$

The next 18 equations in (II.20.1) equal (II.20.8), whereas the last 13 equations are given by (II.20.9) and (II.20.12).

In this model, all tubes and buffers are equal with characteristics as specified in Table II.20.2. Moreover, we assume that the temperature is constant. The values for the physical constants are listed in Table II.20.3. The values for ρ and ν correspond to a temperature of 10°C. The value for R^{crit} was taken from [Sch78, p. 39].

We now discuss how we derived the initial conditions in (II.20.3). First we note that (II.20.9) is an index 2 constraint. Therefore, the initial values also have to satisfy the once differentiated constraint (the so-called *hidden* constraint)

$$\forall\, n \in \mathcal{N} - \mathcal{B}: \qquad 0 = \sum_{i|S_{i,n}=1} \phi'_{i,n}(t) + e_n^{\mathrm{in}\,\prime}(t) - \sum_{j|S_{n,j}=1} \phi'_{n,j}(t) - e_n^{\mathrm{out}\,\prime}(t). \tag{II.20.16}$$

TABLE II.20.2: *Characteristics of tubes.*

Quantity	Value
$l_{i,j}$	1000
$k_{i,j}$	0.0002
$d_{i,j}$	1
B_i	200

TABLE II.20.3: *Values of physical constants.*

Constant	Value
ν	$1.31 \cdot 10^{-6}$
g	9.8
ρ	$1.0 \cdot 10^{3}$
R^{crit}	$2.3 \cdot 10^{3}$

We are free to choose initial flows $\phi_{i,j}(0)$ as long as they satisfy (II.20.9); we chose these all equal to zero. This means that the resistance coefficients equal the value for the case of laminar flows, i.e., 0.047519... The pressures at the buffer nodes, which can be selected freely, are chosen to be $10^5 + g \cdot \rho$, which corresponds to initial heights of one meter in the water columns, assuming that p_0 in Figure II.20.4 equals one bar. From (II.20.12) it follows that $p'_n(0) = 0$, $n \in \mathcal{B}$ (note that the in- and outflows are initially zero). The initial pressures $p_n(0)$, $n \in \mathcal{N} - \mathcal{B}$, and the initial derivative flows $\phi'_{i,j}(0)$ follow from (II.20.14) and (II.20.16). Since the derivatives of the in- and outflows are initially zero, the initial values in (II.20.3) satisfy these equations. The other initial values, $\lambda'_{i,j}(0)$ and $p'_n(0)$, $n \in \mathcal{N} - \mathcal{B}$, appear neither in the system, nor in the hidden constraints, and can be chosen freely. We set these equal to 0.

Several observations can be made from the behavior of the flows, resistance coefficients and pressures, which are plotted in Figure II.20.6–II.20.8:

- The rise and fall of the outflow in node 10 cause the flows to node 10 to change from laminar to turbulent and back, as can be seen from the resistance coefficients $\lambda_{5,10}$ and $\lambda_{8,10}$, which correspond to y_{25} and y_{30}.
- At 8 AM, the pressures in the buffer nodes drop below their original level, which means that some of the water that was present in the buffers initially, is used to meet the peak demand.
- The time period in which the flows to node 10 have become laminar again (this period is indicated by the vertical dashed lines in the plots of y_{25} and y_{30}, causes an irregular behavior (indicated again by dashed lines) of the solution components y_3, y_6, y_9, y_{10} and y_{11} which correspond to the flow from node 3 to node 4 and the flows in the cycle 4–7–8–5, respectively.
- Some of the flows contain high-frequent oscillations of small amplitude. To see this more clearly, we plotted $\phi_{3,4}$ for $6878 < t \leq 17 \cdot 3600$ in Figure II.20.5.

20.4 Numerical solution of the problem

Tables II.20.4–II.20.5 and Figures II.20.6–II.20.8 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagrams, respectively.

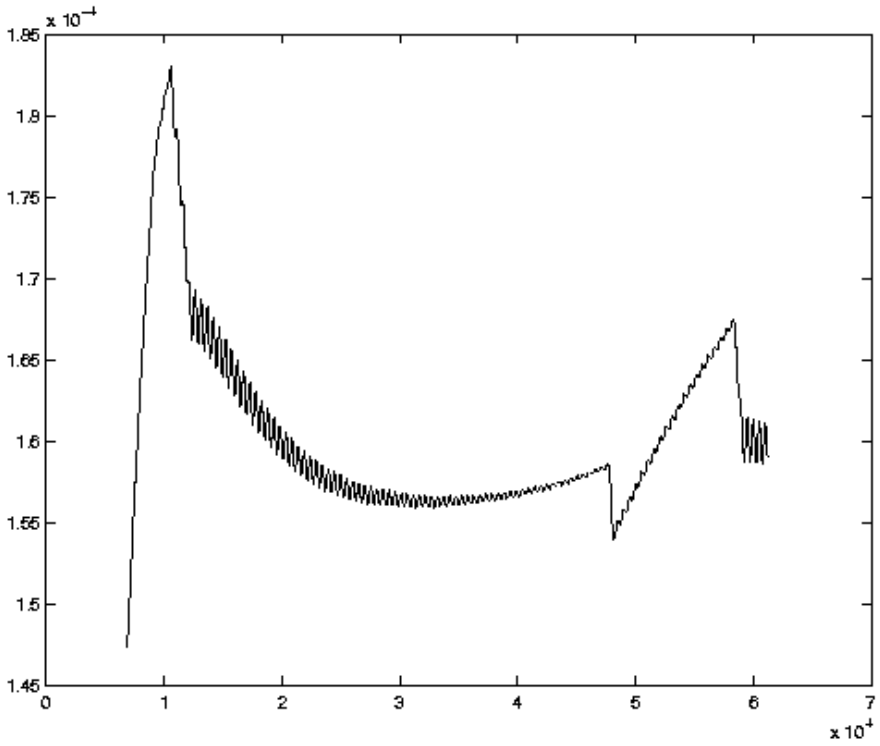

FIGURE II.20.5: *Behavior of* $\phi_{3,4}$ *for* $6878 < t \leq 17 \cdot 3600$.

Since the 13 last solution components (the pressures) are so much larger in magnitude than the other components, we used the following vector-valued input tolerances:

$$\begin{array}{ll} \text{atol}(i) = \text{atol} & \text{for } i = 1, \ldots, 36, \\ \text{atol}(i) = 10^6 \cdot \text{atol} & \text{for } i = 37, \ldots, 49, \\ \text{rtol}(i) = \text{rtol} & \text{for } i = 1, \ldots, 49. \end{array}$$

The reference solution was computed by PSIDE with `rtol = atol` $= 10^{-14}$. For the work-precision diagrams, we used: rtol $= 10^{-(4+m/4)}$, $m = 0, 1, \ldots, 24$; atol = rtol; h0 = rtol for BIMD, GAMD, MEBDFDAE, MEBDFI, RADAU and RADAU5.

The failed runs are in Table II.20.6; listed are the name of the solver that failed, for which values of m this happened, and the reason for failing.

.

References

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[Sch78] H. Schlichting. *Boundary-Layer Theory.* Series in mechanical engineering. Mc Graw-Hill, seventh edition, 1978.

TABLE II.20.4: *Reference solution at the end of the integration interval.*

y_1	$0.2298488296477430 \cdot 10^{-002}$	y_{26}	$0.4751940452918529 \cdot 10^{-001}$
y_2	$0.1188984650746585 \cdot 10^{-002}$	y_{27}	$0.4751940452918529 \cdot 10^{-001}$
y_3	$0.1109503645730845 \cdot 10^{-002}$	y_{28}	$0.4751940452918529 \cdot 10^{-001}$
y_4	$0.1589620100314825 \cdot 10^{-003}$	y_{29}	$0.4751940452918529 \cdot 10^{-001}$
y_5	$0.1030022640715102 \cdot 10^{-002}$	y_{30}	$0.4249217433601160 \cdot 10^{-001}$
y_6	$0.8710606306836165 \cdot 10^{-003}$	y_{31}	$0.4732336439609648 \cdot 10^{-001}$
y_7	$0.3243571480903489 \cdot 10^{-002}$	y_{32}	$0.4732336439609648 \cdot 10^{-001}$
y_8	$0.1109503645730845 \cdot 10^{-002}$	y_{33}	$0.4270002118868241 \cdot 10^{-001}$
y_9	$0.7120986206521341 \cdot 10^{-003}$	y_{34}	$0.4751940452918529 \cdot 10^{-001}$
y_{10}	$0.6414613963833099 \cdot 10^{-003}$	y_{35}	$0.4751940452918529 \cdot 10^{-001}$
y_{11}	$0.9416978549524347 \cdot 10^{-003}$	y_{36}	$0.3651427026675656 \cdot 10^{-001}$
y_{12}	$0.3403428519096511 \cdot 10^{-002}$	y_{37}	$0.1111268591478108 \cdot 10^{+006}$
y_{13}	$0.2397639310739395 \cdot 10^{-002}$	y_{38}	$0.1111270045592387 \cdot 10^{+006}$
y_{14}	$0.2397639310739395 \cdot 10^{-002}$	y_{39}	$0.1111271078730254 \cdot 10^{+006}$
y_{15}	$0.3348581430454180 \cdot 10^{-002}$	y_{40}	$0.1111269851929858 \cdot 10^{+006}$
y_{16}	$0.1353560017035444 \cdot 10^{-002}$	y_{41}	$0.1111269255355337 \cdot 10^{+006}$
y_{17}	$0.1995021413418736 \cdot 10^{-002}$	y_{42}	$0.1111269322658045 \cdot 10^{+006}$
y_{18}	$0.5746220741193575 \cdot 10^{-002}$	y_{43}	$0.1111269221703983 \cdot 10^{+006}$
y_{19}	$0.4751940452918529 \cdot 10^{-001}$	y_{44}	$0.1111270121140691 \cdot 10^{+006}$
y_{20}	$0.4751940452918529 \cdot 10^{-001}$	y_{45}	$0.1111274419515807 \cdot 10^{+006}$
y_{21}	$0.4751940452918529 \cdot 10^{-001}$	y_{46}	$0.1111255158881087 \cdot 10^{+006}$
y_{22}	$0.4751940452918529 \cdot 10^{-001}$	y_{47}	$0.1111278793439227 \cdot 10^{+006}$
y_{23}	$0.4751940452918529 \cdot 10^{-001}$	y_{48}	$0.1111270995171642 \cdot 10^{+006}$
y_{24}	$0.4751940452918529 \cdot 10^{-001}$	y_{49}	$0.1111298338971779 \cdot 10^{+006}$
y_{25}	$0.4311196778792902 \cdot 10^{-001}$		

TABLE II.20.5: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
BIMD	10^{-4}	10^{-4}	10^{-4}	3.55	1.23	17	16	250	16	17	0.0420
	10^{-7}	10^{-7}	10^{-7}	6.05	3.45	333	314	5830	314	333	0.8989
	10^{-10}	10^{-10}	10^{-10}	9.22	7.32	673	586	17454	586	673	2.1101
GAMD	10^{-4}	10^{-4}	10^{-4}	3.51	1.18	18	16	340	16	18	0.0439
	10^{-7}	10^{-7}	10^{-7}	5.94	3.40	233	202	8038	204	233	0.7642
	10^{-10}	10^{-10}	10^{-10}	9.32	7.18	554	458	22918	448	536	1.9744
MEBDFI	10^{-4}	10^{-4}	10^{-4}	3.85	1.83	81	77	1197	18	18	0.0488
	10^{-7}	10^{-7}	10^{-7}	6.32	3.30	1267	1171	13926	192	192	0.5846
	10^{-10}	10^{-10}	10^{-10}	9.09	7.18	3189	3037	28403	351	351	1.2561
PSIDE-1	10^{-4}	10^{-4}		4.37	2.45	64	50	799	16	244	0.1015
	10^{-7}	10^{-7}		5.80	3.09	134	104	2320	40	468	0.2723
	10^{-10}	10^{-10}		7.86	5.45	827	719	14105	39	1292	1.2102

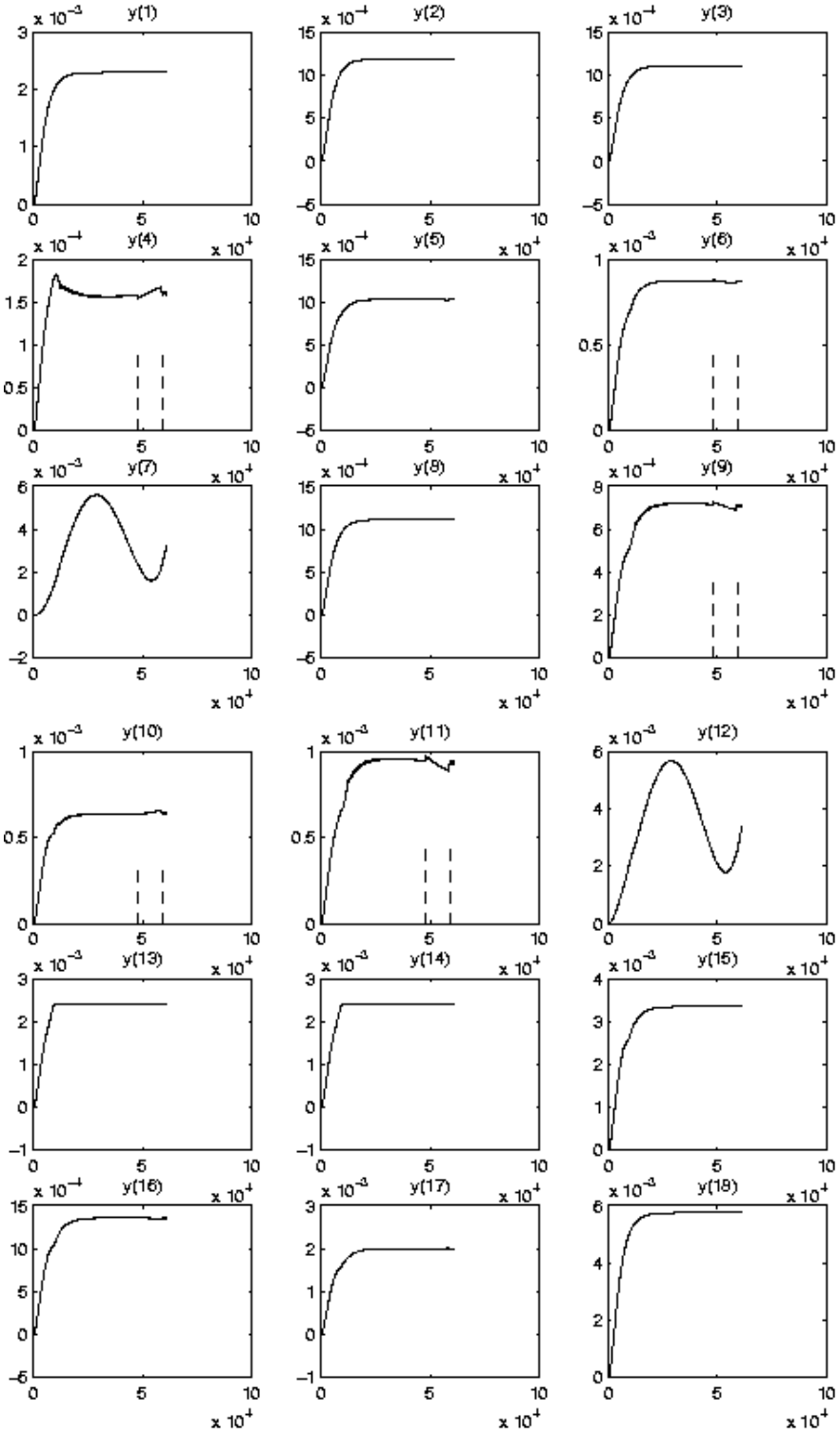

FIGURE II.20.6: *Behavior of flows over the integration interval.*

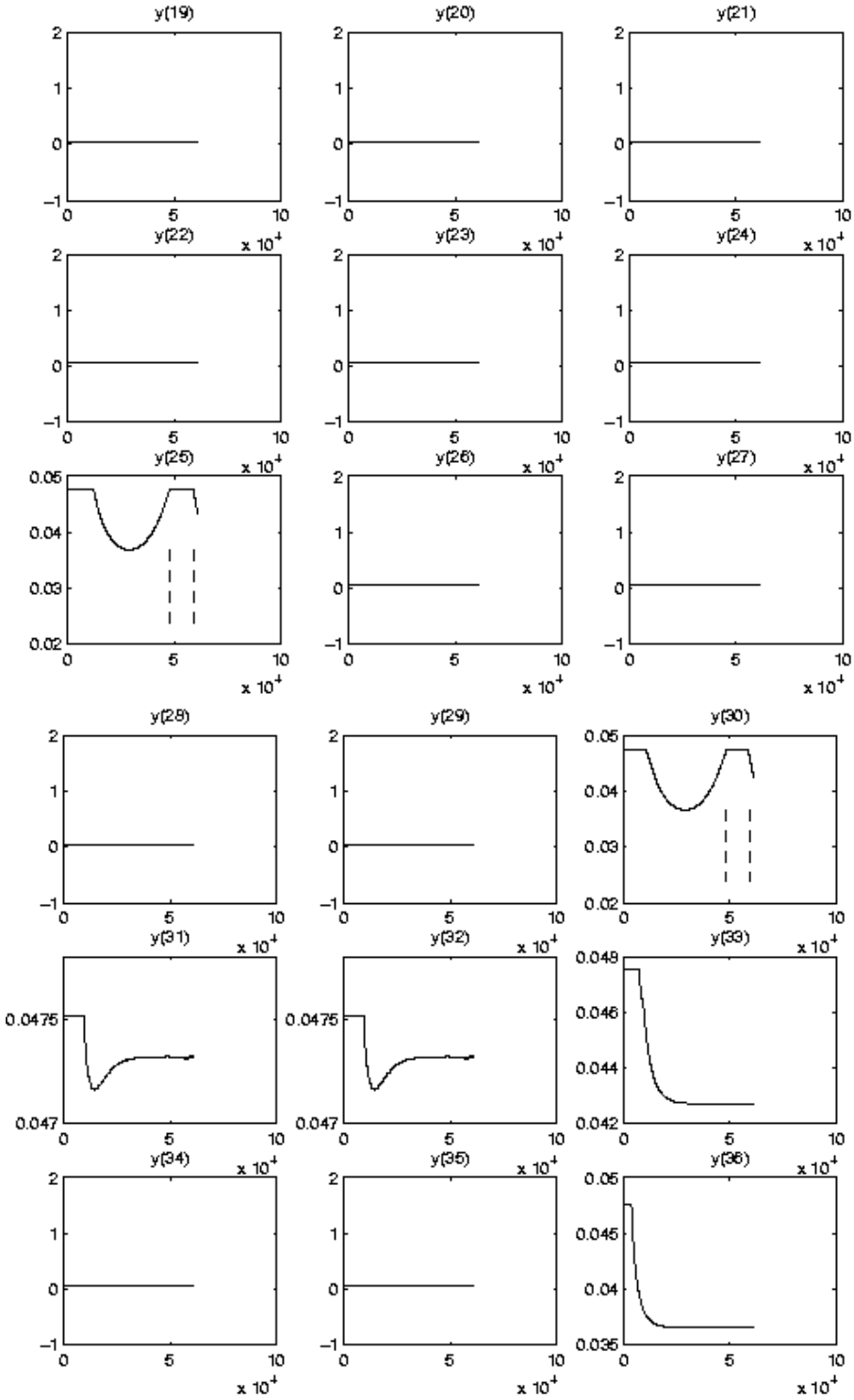

FIGURE II.20.7: *Behavior of resistance coefficients over the integration interval.*

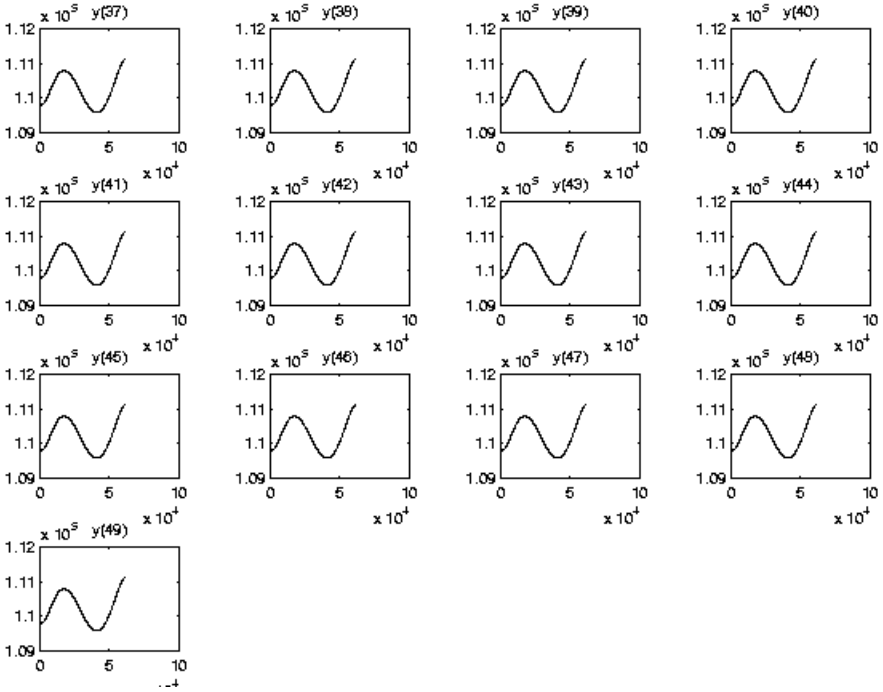

FIGURE II.20.8: *Behavior of pressures over the integration interval.*

TABLE II.20.6: *Failed runs.*

solver	m	reason
RADAU	$0, \ldots 6, 8, 9, 11, 12, 13, 14, 16, \ldots, 20, 24$	solver cannot handle `IERR=-1`.
RADAU5	6	stepsize too small

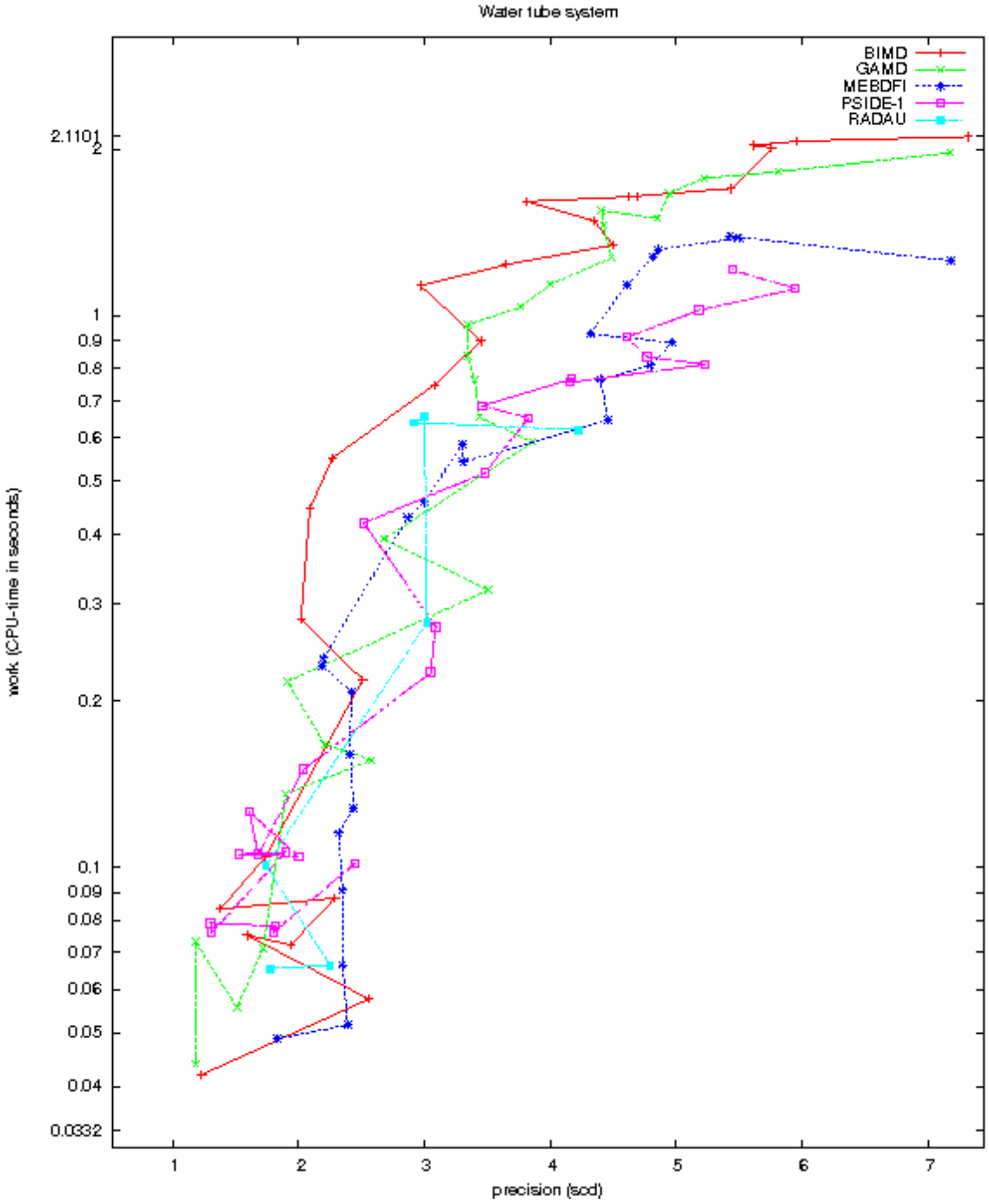

FIGURE II.20.9: *Work-precision diagram (scd versus CPU-time).*

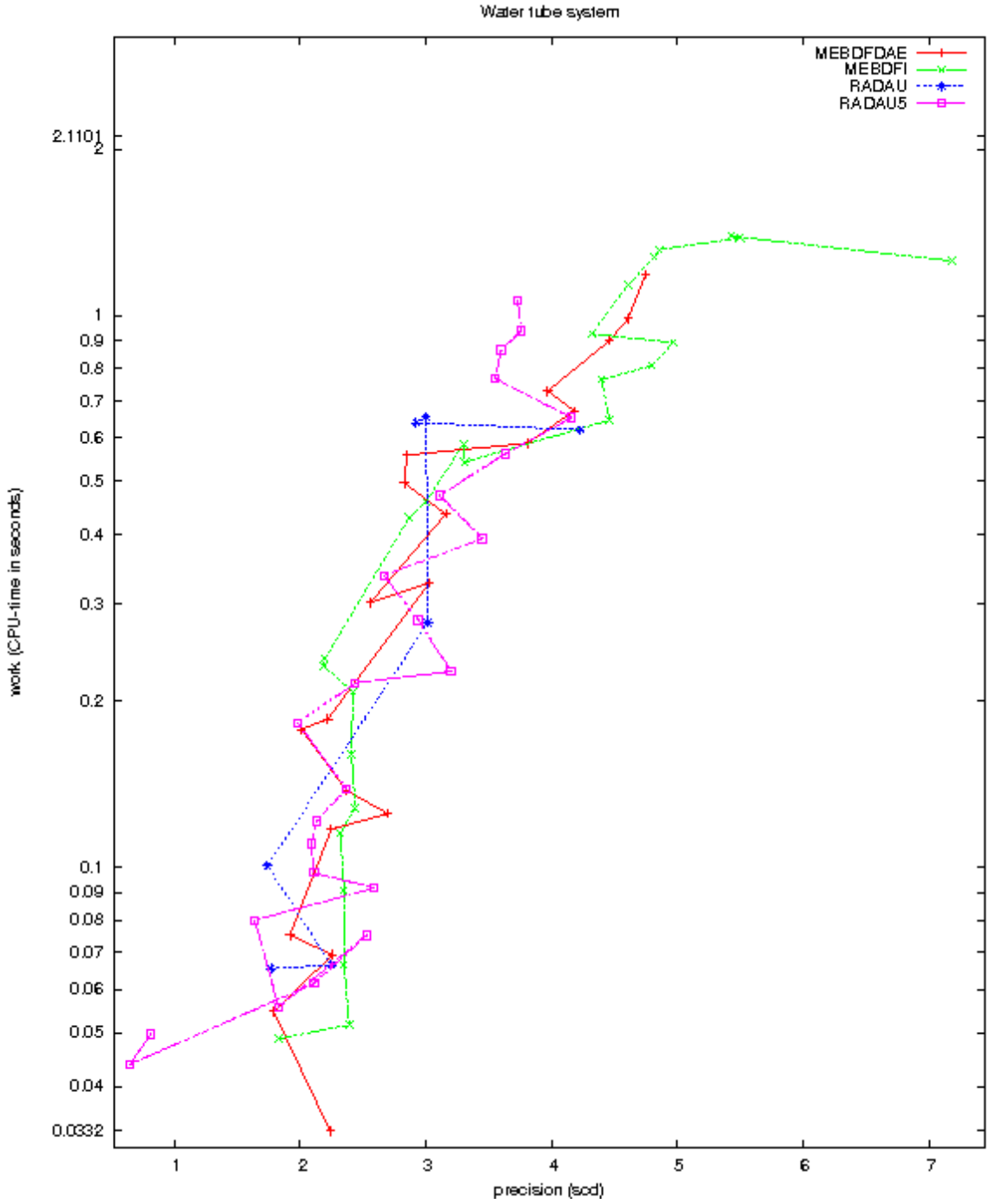

FIGURE II.20.10: *Work-precision diagram (scd versus CPU-time).*

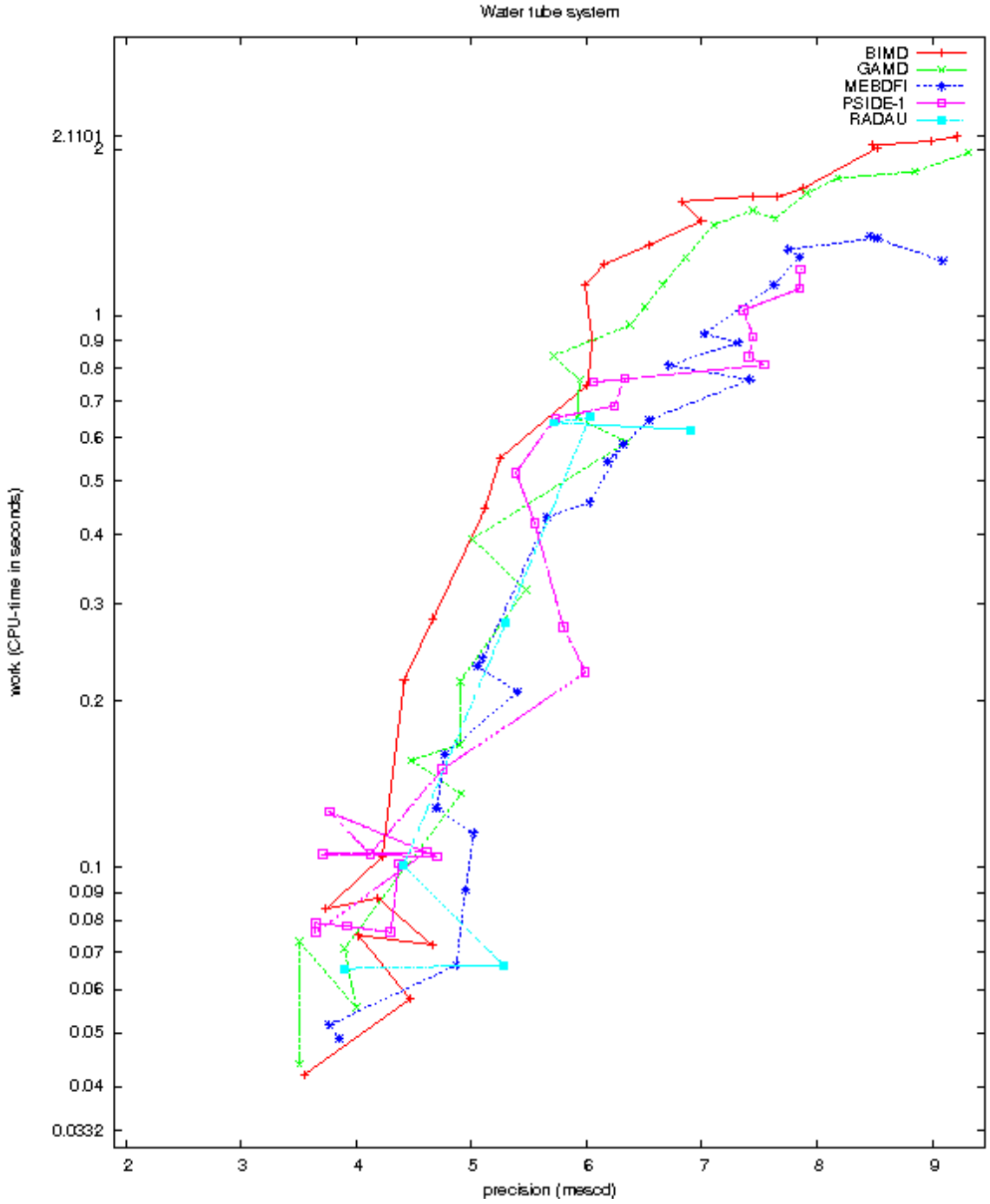

FIGURE II.20.11: *Work-precision diagram (mescd versus CPU-time).*

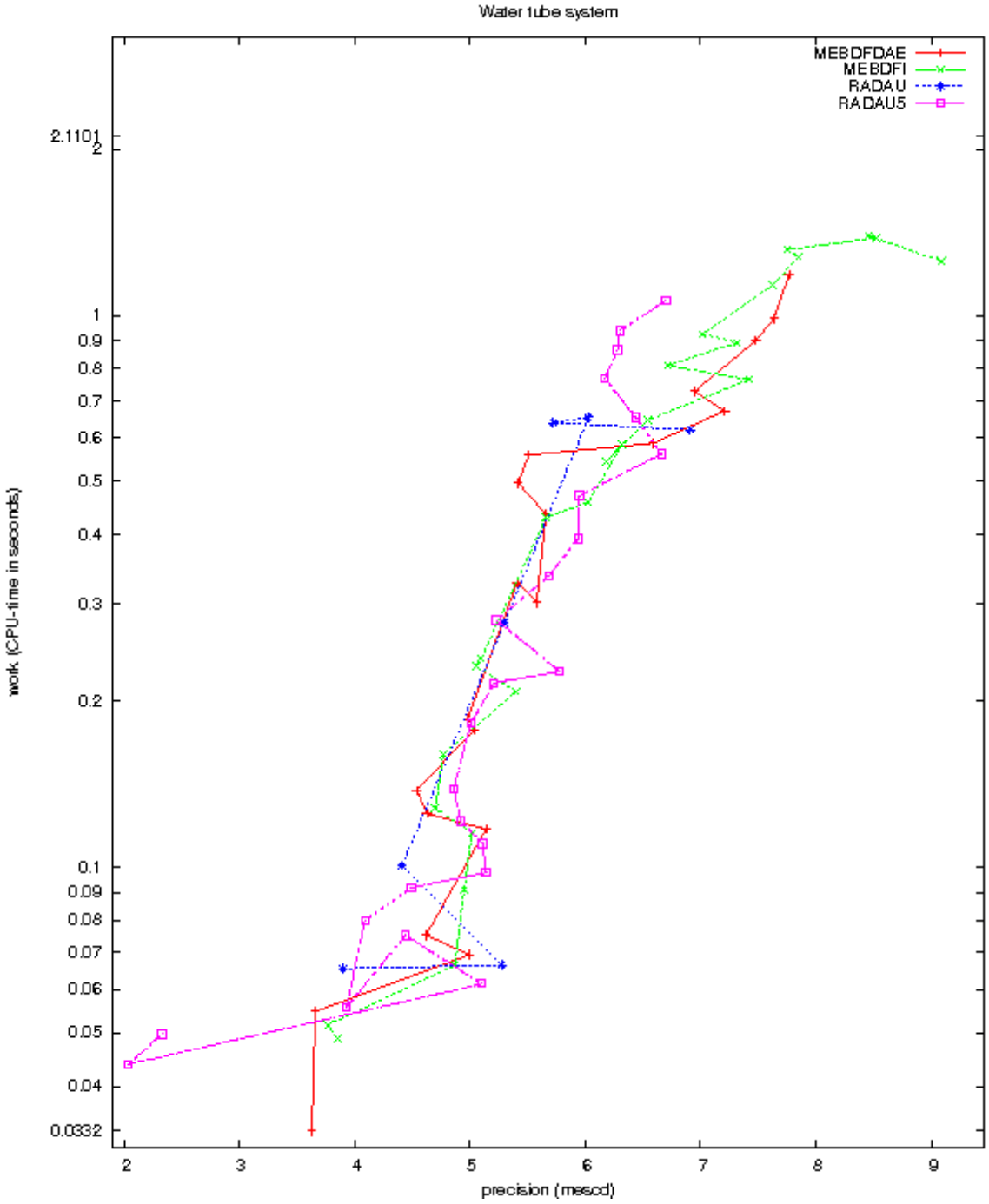

FIGURE II.20.12: *Work-precision diagram (mescd versus CPU-time).*

21 NAND gate

21.1 General information

The problem is a system of 14 stiff IDEs of index 1. It has been contributed by Michael Günther and Peter Rentrop [GR96].

The software part of the problem is in the file nand.f available at [MM08].

21.2 Mathematical description of the problem

The problem is of the form:

$$C(y(t))\,\frac{\mathrm{d}y}{\mathrm{d}t} \quad = \quad f(t,y(t)), \qquad y(0)=y_0, \quad y'(0)=y_0' \tag{II.21.1}$$

with

$$y \in I\!R^{14}, \qquad 0 \le t \le 80.$$

The equations are given by:

$$C_{GS}\cdot(\dot y_5-\dot y_1) = i^D_{DS}(y_2-y_1,y_5-y_1,y_3-y_5,y_5-y_2,y_4-V_{DD}) + \frac{y_1-y_5}{R_{GS}} \tag{II.21.2}$$

$$C_{GD}\cdot(\dot y_5-\dot y_2) = -i^D_{DS}(y_2-y_1,y_5-y_1,y_3-y_5,y_5-y_2,y_4-V_{DD}) + \frac{y_2-V_{DD}}{R_{GD}}, \tag{II.21.3}$$

$$C_{BS}(y_3-y_5)\cdot(\dot y_5-\dot y_3) = \frac{y_3-V_{BB}}{R_{BS}} - i^D_{BS}(y_3-y_5), \tag{II.21.4}$$

$$C_{BD}(y_4-V_{DD})\cdot(-\dot y_4) = \frac{y_4-V_{BB}}{R_{BD}} - i^D_{BD}(y_4-V_{DD}), \tag{II.21.5}$$

$$\begin{aligned} &C_{GS}\cdot\dot y_1 + C_{GD}\cdot\dot y_2 + C_{BS}(y_3-y_5)\cdot\dot y_3 - (C_{GS}+C_{GD}+C_{BS}(y_3-y_5)+C_5)\cdot\dot y_5 \\ &-C_{BD}(y_9-y_5)\cdot(\dot y_5-\dot y_9) \quad = \quad \tfrac{y_5-y_1}{R_{GS}} + i^D_{BS}(y_3-y_5) + \tfrac{y_5-y_7}{R_{GD}} + i^E_{BD}(y_9-y_5), \end{aligned} \tag{II.21.6}$$

$$C_{GS}\cdot\dot y_6 = -i^E_{DS}(y_7-y_6,V_1(t)-y_6,y_8-y_10,V_1(t)-y_7,y_9-y_5) + C_{GS}\cdot\dot V_1(t) - \frac{y_6-y_{10}}{R_{GS}}, \tag{II.21.7}$$

$$C_{GD}\cdot\dot y_7 = i^E_{DS}(y_7-y_6,V_1(t)-y_6,y_8-y_{10},V_1(t)-y_7,y_9-y_5) + C_{GD}\cdot\dot V_1(t) - \frac{y_7-y_5}{R_{GD}}, \tag{II.21.8}$$

$$C_{BS}(y_8-y_{10})\cdot(\dot y_8-\dot y_{10}) = -\frac{y_8-V_{BB}}{R_{BS}} + i^E_{BS}(y_8-y_{10}), \tag{II.21.9}$$

$$C_{BD}(y_9-y_5)\cdot(\dot y_9-\dot y_5) = -\frac{y_9-V_{BB}}{R_{BD}} + i^E_{BD}(y_9-y_5), \tag{II.21.10}$$

$$\begin{aligned} &C_{BS}(y_8-y_{10})\cdot(\dot y_8-\dot y_{10}) - C_{BD}(y_{14}-y_{10})\cdot(\dot y_{10}-\dot y_{14}) + C_{10}\cdot\dot y_{10} \\ &\quad = \tfrac{y_{10}-y_6}{R_{GS}} + i^E_{BS}(y_8-y_{10}) + \tfrac{y_{10}-y_{12}}{R_{GD}} + i^E_{BD}(y_{14}-y_{10}), \end{aligned} \tag{II.21.11}$$

$$C_{GS}\cdot\dot y_{11} = -i^E_{DS}(y_{12}-y_{11},V_2(t)-y_{11},y_{13},V_2(t)-y_{12},y_{14}-y_{10}) + C_{GS}\cdot\dot V_2(t) - \frac{y_{11}}{R_{GS}}, \tag{II.21.12}$$

$$C_{GD}\cdot\dot{y}_{12}=i^E_{DS}(y_{12}-y_{11},V_2(t)-y_{11},y_{13},V_2(t)-y_{12},y_{14}-y_{10})+C_{GD}\cdot\dot{V}_2(t)-\frac{y_{12}-y_{10}}{R_{GD}},\quad\text{(II.21.13)}$$

$$C_{BS}(y_{13})\cdot\dot{y}_{13}=-\frac{y_{13}-V_{BB}}{R_{BS}}+i^E_{BS}(y_{13}),\quad\text{(II.21.14)}$$

$$C_{BD}(y_{14}-y_{10})\cdot(\dot{y}_{14}-\dot{y}_{10})=-\frac{y_{14}-V_{BB}}{R_{BS}}+i^E_{BD}(y_{14}-y_{10}).\quad\text{(II.21.15)}$$

The functions C_{BD} and C_{BS} read

$$C_{BD}(U)=C_{BS}(U)=\begin{cases}C_0\cdot\left(1-\frac{U}{\phi_B}\right)^{-\frac{1}{2}} & \text{for}\quad U\le 0,\\ C_0\cdot\left(1+\frac{U}{2\cdot\phi_B}\right) & \text{for}\quad U>0\end{cases}$$

with $C_0=0.24\cdot10^{-4}$ and $\phi_B=0.87$.

The functions i^D_{BS} and i^E_{BS} have the same form denoted by i_{BS}. The only difference between them is that the constants used in i_{BS} depend on the superscript D and E. The same holds for the functions $i^{D/E}_{BD}$ and $i^{D/E}_{DS}$. The functions i_{BS}, i_{BD} and i_{DS} are defined by

$$i_{BS}(U_{BS}) = \begin{cases}-i_S\cdot\left(\exp(\frac{U_{BS}}{U_T})-1\right) & \text{for}\quad U_{BS}\le 0,\\ 0 & \text{for}\quad U_{BS}>0,\end{cases}$$

$$i_{BD}(U_{BD}) = \begin{cases}-i_S\cdot\left(\exp(\frac{U_{BD}}{U_T})-1\right) & \text{for}\quad U_{BD}\le 0,\\ 0 & \text{for}\quad U_{BD}>0,\end{cases}$$

$$i_{DS}(U_{DS},U_{GS},U_{BS},U_{GD},U_{BD})=\begin{cases}GDS_+(U_{DS},U_{GS},U_{BS}) & \text{for}\quad U_{DS}>0,\\ 0 & \text{for}\quad U_{DS}=0,\\ GDS_-(U_{DS},U_{GD},U_{BD}) & \text{for}\quad U_{DS}<0,\end{cases}$$

where

$$GDS_+(U_{DS},U_{GS},U_{BS})=$$

$$\begin{cases}0 & \text{for}\quad U_{GS}-U_{TE}\le 0,\\ -\beta\cdot(1+\delta\cdot U_{DS})\cdot(U_{GS}-U_{TE})^2 & \text{for}\quad 0<U_{GS}-U_{TE}\le U_{DS},\\ -\beta\cdot U_{DS}\cdot(1+\delta\cdot U_{DS})\cdot(2\cdot(U_{GS}-U_{TE})-U_{DS}) & \text{for}\quad 0<U_{DS}<U_{GS}-U_{TE},\end{cases}$$

with

$$U_{TE}=U_{T0}+\gamma\cdot\left(\sqrt{\Phi-U_{BS}}-\sqrt{\Phi}\right),\quad\text{(II.21.16)}$$

and

$$GDS_-(U_{DS},U_{GD},U_{BD})=$$

$$\begin{cases}0 & \text{for}\quad U_{GD}-U_{TE}\le 0,\\ \beta\cdot(1-\delta\cdot U_{DS})\cdot(U_{GD}-U_{TE})^2 & \text{for}\quad 0<U_{GD}-U_{TE}\le -U_{DS},\\ -\beta\cdot U_{DS}\cdot(1-\delta\cdot U_{DS})\cdot(2\cdot(U_{GD}-U_{TE})+U_{DS}) & \text{for}\quad 0<-U_{DS}<U_{GD}-U_{TE},\end{cases}$$

with

$$U_{TE}=U_{T0}+\gamma\cdot\left(\sqrt{\Phi-U_{BD}}-\sqrt{\Phi}\right).\quad\text{(II.21.17)}$$

The constants used in the definition of i_{BS}, i_{BD} and i_{DS} carry a superscript D or E. Using for example the constants with superscript E in the functions i_{BS} yields the function i^E_{BS}. These constants are shown in Table II.21.1. The other constants are given by

TABLE II.21.1: *Dependence of constants on D and E for i_{BS}, i_{BD} and i_{DS}.*

	E	D
i_S	10^{-14}	10^{-14}
U_T	25.85	25.85
U_{T0}	0.2	-2.43

	E	D
β	$1.748 \cdot 10^{-3}$	$5.35 \cdot 10^{-4}$
γ	0.035	0.2
δ	0.02	0.02
Φ	1.01	1.28

$$\begin{aligned} V_{BB} &= -2.5, \\ V_{DD} &= 5, \\ C_5 &= C_{10} = 0.5 \cdot 10^{-4}, \\ R_{GS} &= R_{GD} = 4, \\ R_{BS} &= R_{BD} = 10, \\ C_{GS} &= C_{GD} = 0.6 \cdot 10^{-4}. \end{aligned}$$

The functions $V_1(t)$ and $V_2(t)$ are

$$V_1(t) = \begin{cases} 20 - tm & \text{if} \quad 15 < tm \le 20, \\ 5 & \text{if} \quad 10 < tm \le 15, \\ tm - 5 & \text{if} \quad 5 < tm \le 10, \\ 0 & \text{if} \quad tm \le 5, \end{cases}$$

with $tm = t \bmod 20$ and

$$V_2(t) = \begin{cases} 40 - tm & \text{if} \quad 35 < tm \le 40, \\ 5 & \text{if} \quad 20 < tm \le 35, \\ tm - 15 & \text{if} \quad 15 < tm \le 20, \\ 0 & \text{if} \quad tm \le 15, \end{cases}$$

with $tm = t \bmod 40$. From these definitions for $V_1(t)$ and $V_2(t)$ we see that the function f in (II.21.1) has discontinuities in its derivative at $tm = 5, 10, 15, 20$. Therefore, we restart the solvers at $t = 5, 10, \ldots, 75$.

Consistent initial values are given by $y_0' = 0$ and

$$\begin{aligned} &y_1 = y_2 = y_5 = y_7 = 5.0, \\ &y_3 = y_4 = y_8 = y_9 = y_{13} = y_{14} = V_{BB} = -2.5, \\ &y_6 = y_{10} = y_{12} = 3.62385, \\ &y_{11} = 0. \end{aligned}$$

All components of y are of index 1.

It is clear from Formulas (II.21.16) and (II.21.17) that the function f can not be evaluated if one of the values $\Phi - U_{BS}$, $\Phi - U_{BD}$ or Φ becomes negative. To prevent this situation, we set `IERR=-1` in the Fortran subroutine that defines f if this happens. See page *IV-ix* of the the description of the software part of the test set for more details on `IERR`.

21.3 Origin of the problem

The NAND gate in Figure II.21.1 consists of two n-channel enhancement MOSFETs (ME), one n-channel depletion MOSFET (MD) and two load capacitances C_5 and C_{10}. MOSFETs are special transistors, which have four terminals: the drain, the bulk, the source and the gate, see also Figure II.21.3. The drain voltage of MD is constant at $V_{DD} = 5$[V]. The bulk voltages are constantly $V_{BB} = -2.5$[V]. The gate voltages of both enhancement transistors are controlled by two voltage

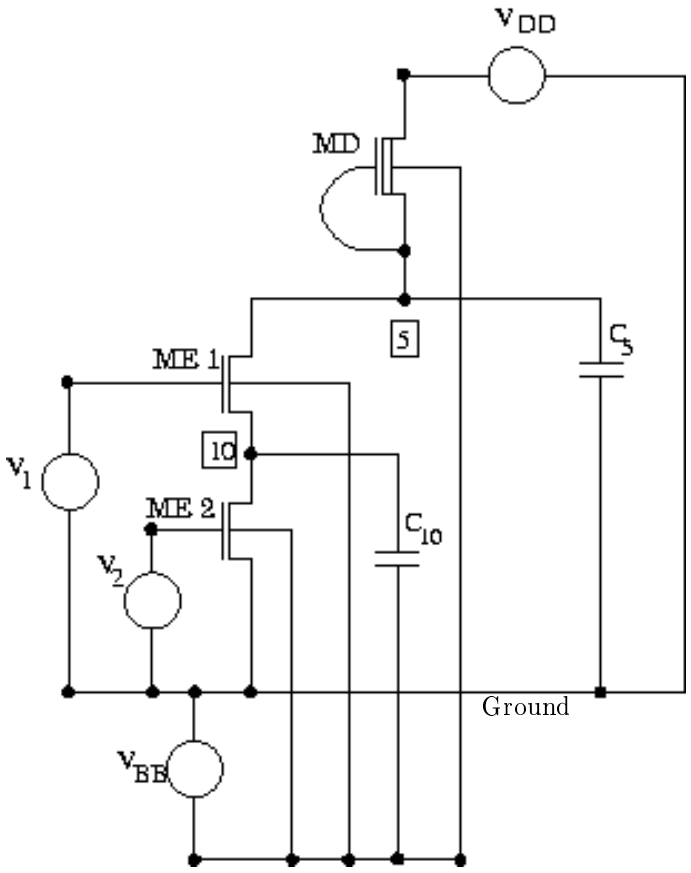

FIGURE II.21.1: *Circuit diagram of the NAND gate (taken from [GR96])*

		V2	
		LOW	HIGH
V1	LOW	HIGH	HIGH
	HIGH	HIGH	LOW

FIGURE II.21.2: *Response of the NAND gate*

sources V_1 and V_2. Depending on the input voltages, the NAND gate generates a response at node 5 as shown in Figure II.21.2. If we represent the logical values 1 and 0 by high respectively low voltage levels, we see that the NAND gate executes the *Not AND* operation. This behavior can be explained from Figure II.21.1 as follows. Roughly speaking, a transistor acts as a switch between drain and source; it closes if the voltage between gate and source drops below a certain threshold value. The circuit is constructed such that the voltage at node 10 drops to zero unless V_1 is high and V_2 is low, in which case it is approximately 5[V]. This means that as soon either V_1 or V_2 is low, then the corresponding enhancement transistors lock; the voltage at node 5 is high at $V_{DD} = 5$[V] due to MD.

If both V_1 and V_2 exceed a given threshold voltage, then a drain current through both enhancement transistors occurs. The MOSFETs open and the voltage at node 5 breaks down. The response is low. In the circuit analysis the three MOSFETs are replaced by the circuit shown in Figure II.21.3. Here,

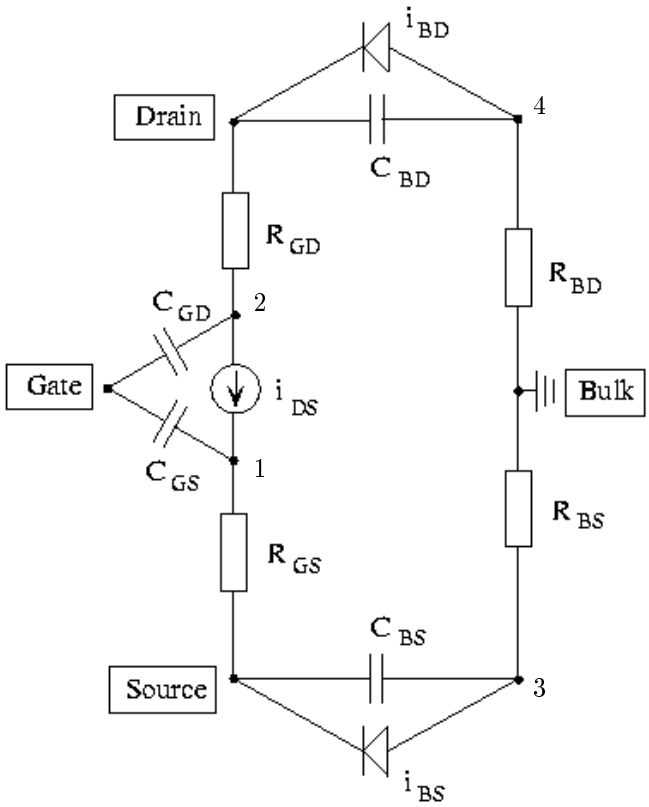

FIGURE II.21.3: *Companion model of a MOSFET (taken from [GR96])*

the well-known companion model of Shichmann and Hodges [SH68] is used. The characteristics of the circuit elements can differ depending on the MD or ME case. This circuit has four internal nodes indicated by 1, 2, 3 and 4. The static behavior of the transistor is described by the drain current i_{DS}. To include secondary effects, load capacitances like R_{GS}, R_{GD}, R_{BS}, and R_{BD} are introduced. The so-called *pn*-junction between source and bulk is modeled by the diode i_{BS} and the non-linear capacitance C_{BS}. Analogously, i_{BD} and C_{BD} model the *pn*-junction between bulk and drain. Linear gate capacitances C_{GS} and C_{GD} are used to describe the intrinsic charge flow effects roughly.

To formulate the circuit equations, we note that the circuit consists of 14 nodes. These 14 nodes are the nodes 5 and 10 and the 12 internal nodes of the three transistors. For every node a variable is introduced that represents the voltage in that node. Table II.21.2 shows the variable–node correspondence. In terms of these voltages the circuit equations are formulated by using the Kirchoff Current Law (KCL) along with the transistor model shown in Figure II.21.3. In Figure II.21.4, we check the behavior of the NAND gate by plotting V_1 and V_2 together with the numerical value for the voltage at node 5, which is obtained as y_{10} in §21.4. The picture confirms that the NAND gate produces a high signal in the intervals $[0, 5]$, $[10, 15]$, $[20, 25]$, $[40, 45]$, $[50, 55]$ and $[60, 65]$, whereas the output signal

TABLE II.21.2: *Correspondence between variables and nodes*

variables	nodes
1–4	internal nodes MD-transistor
5	node 5
6–9	internal nodes ME1-transistor
10	node 10
11–14	internal nodes ME2-transistor

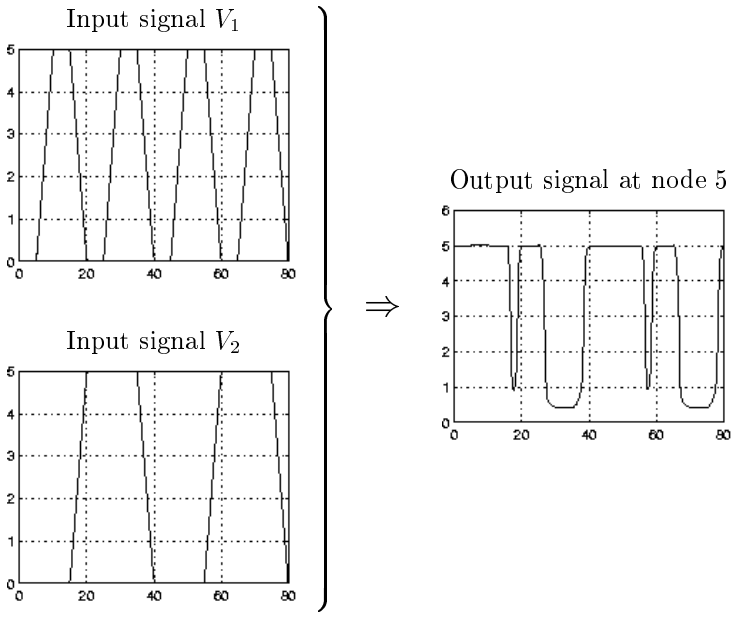

FIGURE II.21.4: *Plots of V_1, V_2 and the output of the NAND gate.*

on $[30, 35]$ and $[70, 75]$ is low.

We remark that in this description the unit of time is the nanosecond, while in the report [GR96] the unit of time is the second.

21.4 Numerical solution of the problem

Tables II.21.3–II.21.4 and Figures II.21.5–II.21.7 present the reference solution at the end of the integration interval, the run characteristics, the behavior of the solution over the integration interval and the work-precision diagram, respectively. In computing the scd values, only y_5, the response of the gate at node 5, was considered. The reference solution was computed on the Cray C90, using PSIDE with Cray double precision and atol = rtol = 10^{-16}. For the work-precision diagram, we used: rtol = $10^{-(4+m/8)}$, $m = 0, 1, \ldots, 64$; atol = rtol, h0 = rtol for MEBDFI. .

.

TABLE II.21.3: *Reference solution at the end of the integration interval.*

y_1	$0.4971088699385777 \cdot 10$	y_8	$-0.2500077409198803 \cdot 10$
y_2	$0.4999752103929311 \cdot 10$	y_9	$-0.2499998781491227 \cdot 10$
y_3	$-0.2499998781491227 \cdot 10$	y_{10}	-0.2090289583878100
y_4	$-0.2499999999999975 \cdot 10$	y_{11}	$-0.2399999999966269 \cdot 10^{-3}$
y_5	$0.4970837023296724 \cdot 10$	y_{12}	-0.2091214032073855
y_6	-0.2091214032073855	y_{13}	$-0.2499999999999991 \cdot 10$
y_7	$0.4970593243278363 \cdot 10$	y_{14}	$-0.2500077409198803 \cdot 10$

TABLE II.21.4: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
DDASSL	10^{-4}	10^{-4}		3.69	5.25	1037	951	1639	246		0.0459
	10^{-7}	10^{-7}		6.22	8.81	3825	3604	5207	638		0.1376
MEBDFI	10^{-4}	10^{-4}	10^{-4}	3.76	4.57	1120	1006	7693	249	249	0.0683
	10^{-7}	10^{-7}	10^{-7}	6.24	7.50	3786	3429	24487	755	755	0.2255
PSIDE-1	10^{-4}	10^{-4}		2.39	3.33	464	411	6574	109	1796	0.0927
	10^{-7}	10^{-7}		5.28	8.48	773	643	13134	222	2760	0.1796

References

[GR96] M. Günther and P. Rentrop. The NAND-gate – a benchmark for the numerical simulation of digital circuits. In W. Mathis and P. Noll, editors, *2.ITG-Diskussionssitzung "Neue Anwendungen Theoretischer Konzepte in der Elektrotechnik" - mit Gedenksitzung zum 50. Todestag von Wilhelm Cauer*, pages 27–33, Berlin, 1996. VDE-Verlag.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4.* Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[SH68] H. Shichman and D.A. Hodges. Insulated-gate field-effect transistor switching circuits. *IEEE J. Solid State Circuits*, SC-3:285–289, 1968.

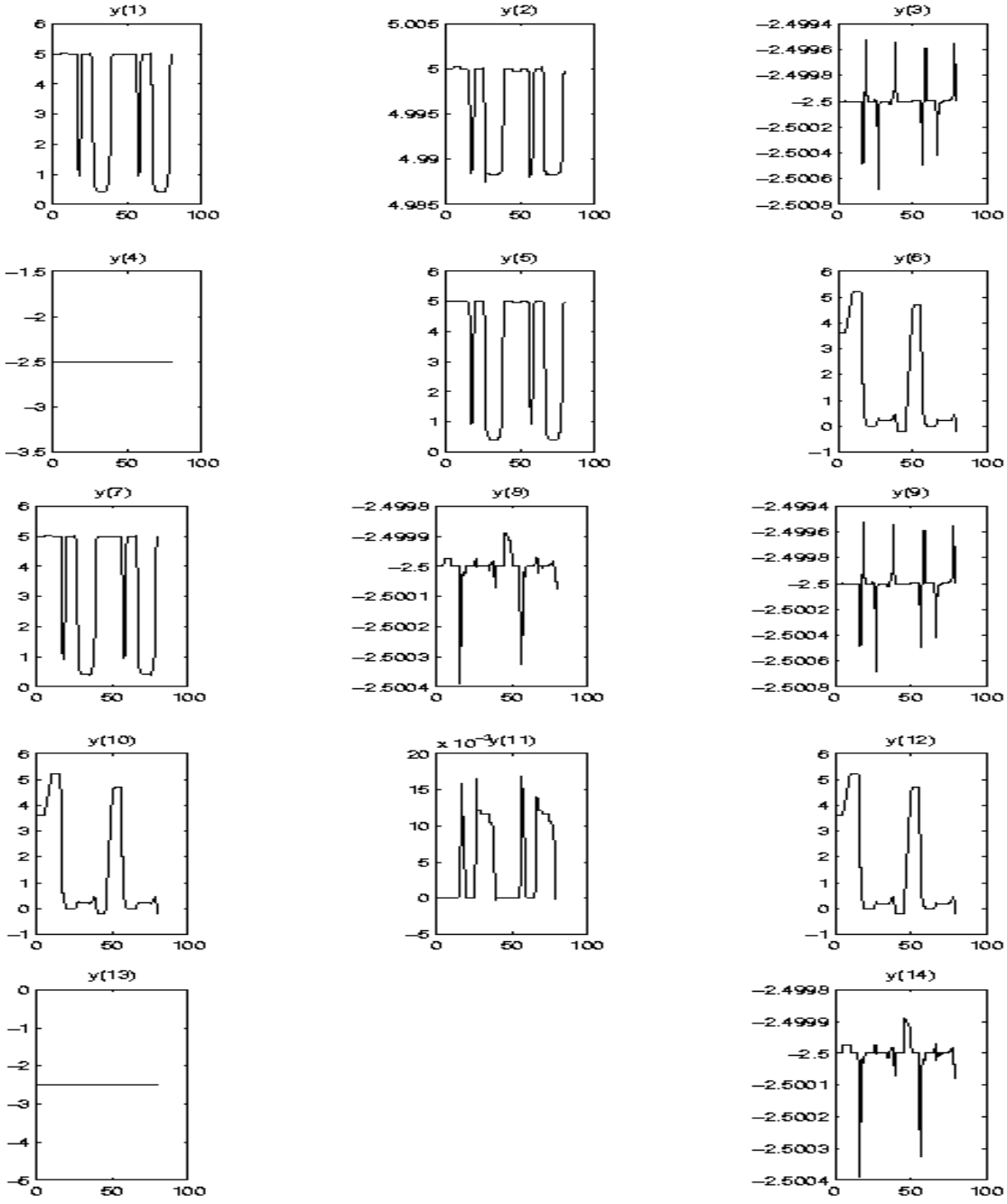

FIGURE II.21.5: *Behavior of the solution over the integration interval.*

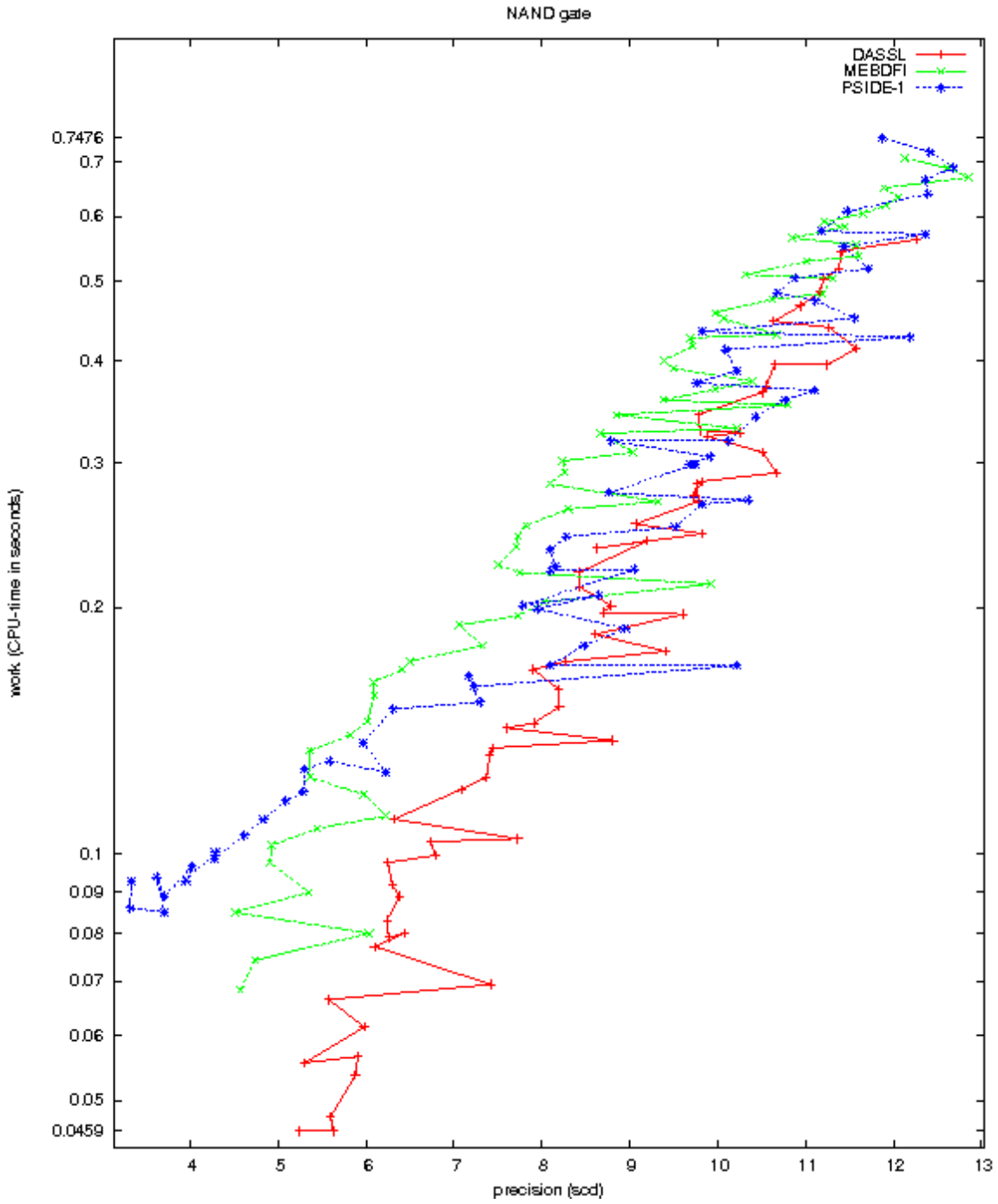

FIGURE II.21.6: *Work-precision diagram (scd versus CPU-time).*

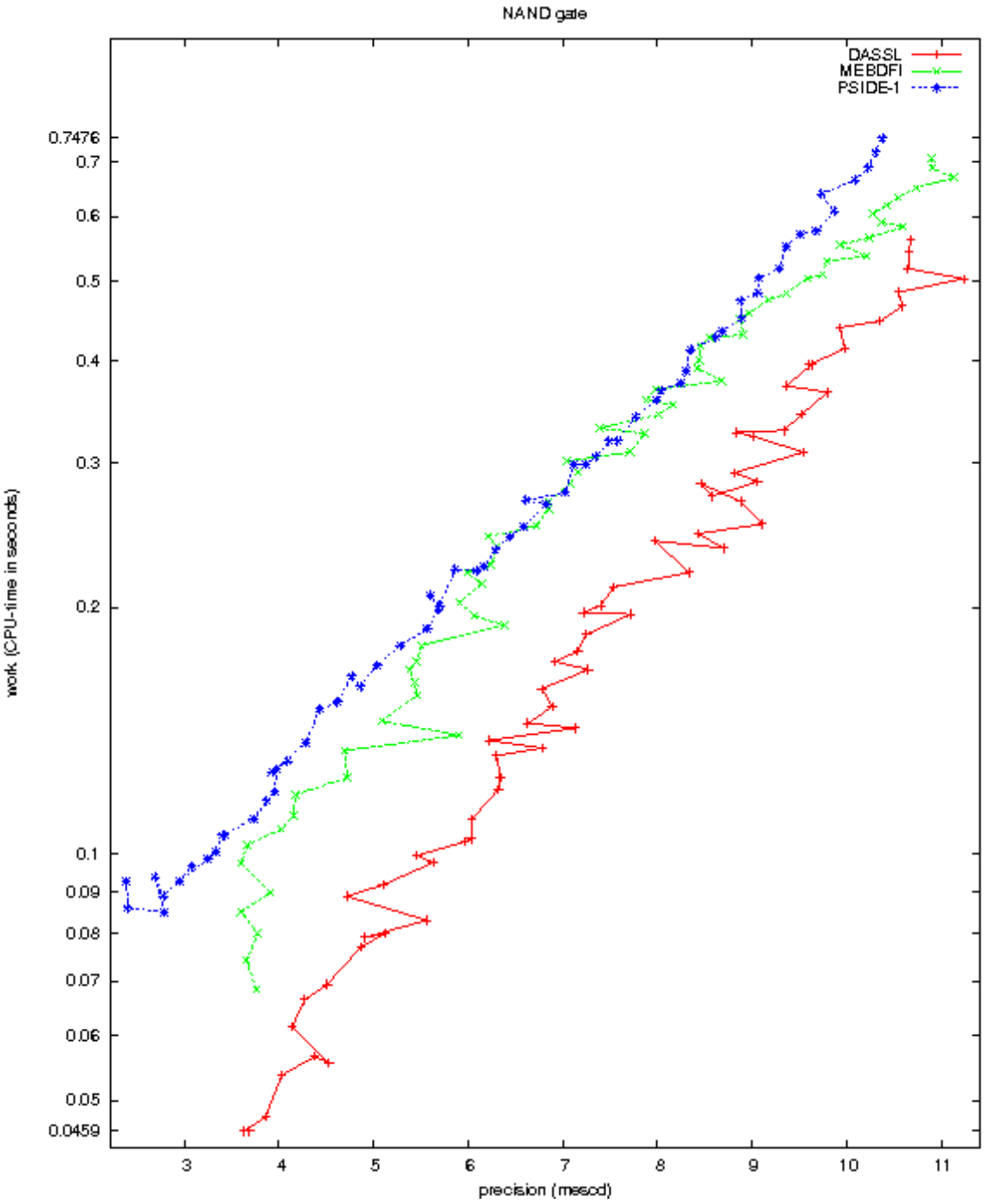

FIGURE II.21.7: *Work-precision diagram (mescd versus CPU-time).*

22 Wheelset

22.1 General Information

The wheelset is an IDE of dimension 17 which shows some typical properties of simulation problems in contact mechanics, i.e., friction, contact conditions, stiffness, etc.. This problem is originally described by an index 3 IDE with additional index 1 equations, but can be reduced to index 2. Test results are based on the index-2 formulation. This problem was contributed by Bernd Simeon, Claus Führer, Peter Rentrop, Nov. 1995. Comments to `simeon@ma.tum.de` or `Claus.Fuhrer@na.lu.se`. See also [SFR91].

The software part of the problem is in the file wheel.f available at [MM08].

22.2 Mathematical description of the problem

The index 3 formulation of the wheelset problem reads

$$\dot{p} = v, \tag{II.22.1}$$

$$M(p)\begin{pmatrix} \dot{v} \\ \dot{\beta} \end{pmatrix} = \begin{pmatrix} f(u) - (\partial g_1(p,q)/\partial p)^{\mathrm{T}} C\lambda \\ d(u) \end{pmatrix}, \tag{II.22.2}$$

$$0 = g_1(p,q), \tag{II.22.3}$$

$$0 = g_2(p,q), \tag{II.22.4}$$

where $u := (p, v, \beta, q, \lambda)^{\mathrm{T}} \in I\!R^{17}$, $p, v \in I\!R^5$, $\beta \in I\!R$, $q \in I\!R^4$, $\lambda \in I\!R^2$ and C is a scalar constant. Furthermore, $M : I\!R^5 \to I\!R^6 \times I\!R^6$, $f : I\!R^{17} \to I\!R^5$, $d : I\!R^{17} \to I\!R$, $g_1 : I\!R^9 \to I\!R^2$ and $g_2 : I\!R^9 \to I\!R^4$. The integration interval is from 0 to 10 [s].

For the index 2 formulation of the problem (II.22.3) is replaced by

$$0 = (\partial g_1(p,q)/\partial p)\, v. \tag{II.22.5}$$

The non-zero components of the consistent initial values $u(0) := u_0$ and $u'(0) := u_0'$ are given by

$u_{0,1}$	$0.1494100000000000 \cdot 10^{-2}$	$u_{0,12}$	$7.4122380357667139 \cdot 10^{-6}$
$u_{0,2}$	$0.4008900000000000 \cdot 10^{-6}$	$u_{0,13}$	0.1521364296121248
$u_{0,3}$	$0.1124100000000000 \cdot 10^{-5}$	$u_{0,14}$	$7.5634406395172940 \cdot 10^{-6}$
$u_{0,4}$	$-0.2857300000000000 \cdot 10^{-3}$	$u_{0,15}$	0.1490635714733819
$u_{0,5}$	$0.2645900000000000 \cdot 10^{-3}$	$u_{0,16}$	$-0.8359300000000000 \cdot 10^{-2}$
		$u_{0,17}$	$-0.7414400000000000 \cdot 10^{-2}$

$u_{0,6}'$	-1.9752588940112850	$u_{0,9}'$	-5.5333628217315490
$u_{0,7}'$	$-1.0898297102811276 \cdot 10^{-3}$	$u_{0,10}'$	-0.3487021489546511
$u_{0,8}'$	$7.8855083626142589 \cdot 10^{-2}$	$u_{0,11}'$	-2.1329687243809270

The other components of u_0 and u_0' are zero. For the index 3 formulation, the index of variables p, v, β, q and λ equals 1, 2, 2, 1 and 3. For the index 2 problem, these numbers read 1, 1, 1, 1 and 2.

The equations are given in detail in the next subsections, in which some references to the origin of the problem, treated in §22.3, are already given. Table II.22.1 lists all problem parameters.

22.2.1 Differential equations

The position coordinates p are defined as

$$p := \begin{pmatrix} x \\ y \\ z \\ \theta \\ \varphi \end{pmatrix} \qquad \begin{array}{l} \text{lateral displacement} \\ \text{vertical displacement} \\ \text{longitudinal displacement} \\ \text{yaw angle} \\ \text{roll angle} \end{array}$$

and the contact variables as $q^{\mathrm{T}} := (\ \psi_L \quad \xi_L \quad \psi_R \quad \xi_R\ )$ with

$$\begin{aligned} \xi_{L|R} &:= \text{coordinate of the contact point left/right,} \\ \psi_{L|R} &:= \text{shift angle left/right.} \end{aligned}$$

The first three equations in (II.22.2) yield the momentum equations:

$$\begin{aligned} m_R\,\ddot{x} &= m_R\left(2\,v_0\,\kappa\,\cos\alpha\,\dot{z} + v_0^2\,\kappa\,\cos\alpha\,(1+\kappa\,(x\cos\alpha - y\sin\alpha))\right) \\ &\quad + T_{L_1} + T_{R_1} + Q_1 - m_R\,\tilde{g}\,\sin\alpha - b_{1,1}\,\lambda_1 - b_{1,2}\,\lambda_2 - 2\,c_x\,x\,, \\ m_R\,\ddot{y} &= -m_R\left(2\,v_0\,\kappa\,\sin\alpha\,\dot{z} + v_0^2\,\kappa\,\sin\alpha\,(1+\kappa\,(x\cos\alpha - y\sin\alpha))\right) \\ &\quad + T_{L_2} + T_{R_2} + Q_2 - m_R\,\tilde{g}\,\cos\alpha - b_{2,1}\,\lambda_1 - b_{2,2}\,\lambda_2\,, \\ m_R\,\ddot{z} &= m_R\left(-2\,v_0\,\kappa\,(\dot{x}\cos\alpha - \dot{y}\sin\alpha) + v_0^2\,\kappa^2\,z\right) \\ &\quad + T_{L_3} + T_{R_3} + Q_3 + F_A - b_{3,1}\,\lambda_1 - b_{3,2}\,\lambda_2\,, \end{aligned}$$

where $b_{i,j}$ denotes the (i,j) element of the constraint Jacobian $\partial g_1(p,q)/\partial p$. The next three equations yield the spin equations:

$$\begin{aligned} I_2\,\ddot{\theta}\,\cos\varphi &= -\dot{\theta}\,\dot{\varphi}\sin\varphi + v_0\,\kappa\Big(\,\dot{\varphi}(\sin\alpha\cos\theta\cos\varphi + \cos\alpha\sin\varphi) - \dot{\theta}\sin\alpha\sin\theta\sin\varphi\Big) \\ &\quad -I_1\,(\omega_0+\beta)\,(\dot{\varphi} - v_0\kappa\sin\theta\sin\alpha) \\ &\quad -(I_1 - I_2)\left(\dot{\theta}\,\sin\varphi - v_0\,\kappa\,(\cos\theta\cos\varphi\sin\alpha + \sin\varphi\cos\alpha)\right) \\ &\qquad\qquad \left(\dot{\varphi} - v_0\,\kappa\,\sin\alpha\sin\theta\right) \\ &\quad +\Big[\,-(\xi_L\sin\theta + R(\xi_L)\sin\psi_L\cos\theta\cos\varphi)\,T_{L_1} \\ &\qquad -R(\xi_L)\sin\psi_L\sin\varphi\,T_{L_2} \\ &\qquad +(-\xi_L\cos\theta + R(\xi_L)\sin\psi_L\sin\theta\cos\varphi)\,T_{L_3}\Big] \\ &\quad +\Big[\quad \text{corresponding terms of the right side} \quad\Big] \\ &\quad -\cos\theta\,\sin\varphi\,M_1 + \cos\varphi\,M_2 + \sin\theta\,\sin\varphi\,M_3 - b_{4,1}\,\lambda_1 - b_{4,2}\,\lambda_2\,, \\ I_2\,\ddot{\varphi} &= I_2\,\dot{\theta}\,v_0\,\kappa\,\sin\alpha\,\cos\theta \\ &\quad +I_1\,(\omega_0+\beta)\left(\dot{\theta}\cos\varphi + v_0\,\kappa\,(\cos\theta\sin\varphi\sin\alpha - \cos\varphi\cos\alpha)\right) \\ &\quad +(I_1 - I_2)\Big(\dot{\theta}\,\sin\varphi - v_0\,\kappa(\cos\theta\cos\varphi\sin\alpha + \sin\varphi\cos\alpha)\Big) \\ &\qquad\qquad \Big(\dot{\theta}\,\cos\varphi + v_0\,\kappa(\cos\theta\sin\varphi\sin\alpha - \cos\varphi\cos\alpha)\Big) \\ &\quad +\Big[\,-(\xi_L\cos\theta\sin\varphi - R(\xi_L)\cos\psi_L\cos\theta\cos\varphi)\,T_{L_1} \\ &\qquad +(\xi_L\cos\varphi + R(\xi_L)\cos\psi_L\sin\varphi)\,T_{L_2} \\ &\qquad +(\xi_L\sin\theta\sin\varphi - R(\xi_L)\cos\psi_L\sin\theta\cos\varphi)\,T_{L_3}\Big] \\ &\quad +\Big[\quad \text{corresponding terms of the right side} \quad\Big] \\ &\quad +\sin\theta\,M_1 + \cos\theta\,M_3 - b_{5,1}\,\lambda_1 - b_{5,2}\,\lambda_2\,, \end{aligned}$$

$$
\begin{aligned}
I_1\,(\dot\beta\,+\,\ddot\theta\,\sin\varphi) \;=\;& \dot\theta\,\dot\varphi\cos\varphi - v_0\,\kappa\,(\dot\varphi(\cos\alpha\cos\varphi - \sin\alpha\cos\theta\sin\varphi) - \dot\theta\sin\alpha\sin\theta\cos\varphi)\\
&+\Big[-R(\xi_L)\,(\cos\psi_L\sin\theta + \sin\psi_L\cos\theta\sin\varphi)\,T_{L_1}\\
&\quad +R(\xi_L)\,\sin\psi_L\,\cos\varphi\,T_{L_2}\\
&\quad -R(\xi_L)\,(\cos\psi_L\cos\theta - \sin\psi_L\sin\theta\sin\varphi)\,T_{L_3}\Big]\\
&+\Big[\quad\text{corresponding terms of the right side}\quad\Big]\\
&+\cos\theta\,\cos\varphi\,M_1 + \sin\varphi\,M_2 - \sin\theta\,\cos\varphi\,M_3 + L_A\,.
\end{aligned}
$$

The forces Q and moments M of the wagon body satisfy the following equations:

$$
\begin{aligned}
Q_1 &= \tfrac{m_A\,\bar g}{\cos\alpha}\left(\tfrac{v_0^2\,\kappa}{\bar g} - \tan\alpha\right) && \text{(lateral force)},\\
Q_2 &= -m_A\,\tilde g\,\cos\alpha\,\left(\tfrac{v_0^2\,\kappa}{\bar g}\,\tan\alpha + 1\right) && \text{(vertical force)},\\
Q_3 &= -2\,c_z\,z && \text{(longitudinal force)},\\
M_1 &= 0\\
M_2 &= Q_3\,x_l && \text{(yaw moment)},\\
M_3 &= -h_A\,Q_1 && \text{(roll moment)},\\
0 &= \cos\theta\,M_1 - \sin\theta\,M_3 && \text{(no pitch moment)}.
\end{aligned}
$$

The creep forces $T_{L_{1,2,3}}$ and $T_{R_{1,2,3}}$ of the left and right contact point are obtained via the transformation

$$
\begin{pmatrix} T_{L|R_1}\\ T_{L|R_2}\\ T_{L|R_3}\end{pmatrix} = \begin{pmatrix} \sin\theta & \cos\theta\cos\Delta_{L|R} & \mp\cos\theta\sin\Delta_{L|R}\\ 0 & \pm\sin\Delta_{L|R} & \cos\Delta_{L|R}\\ \cos\theta & -\sin\theta\cos\Delta_{L|R} & \pm\sin\theta\sin\Delta_{L|R}\end{pmatrix}\begin{pmatrix} T_{1_{L|R}}\\ T_{2_{L|R}}\\ 0\end{pmatrix},
$$

where $T_{1_{L|R}}$ and $T_{2_{L|R}}$ denote the creep forces with respect to the local reference frame of the contact point and $\pm$ stands for the left and right side, respectively. The creep forces are approximated by

$$
\begin{aligned}
T_{1_{L|R}} &:= -\mu N_{L|R}\tanh\left(\frac{GC_{11}c^2}{\mu N_{L|R}}\nu_1\right),\\
T_{2_{L|R}} &:= -\mu N_{L|R}\tanh\left(\frac{GC_{22}c^2}{\mu N_{L|R}}\nu_2 + \frac{GC_{23}c^3}{\mu N_{L|R}}\varphi_3\right),
\end{aligned}
$$

and corrected by

$$
\begin{aligned}
&\text{if}\quad T_1^2 + T_2^2 > (\mu N)^2\,,\quad \text{then}\\
&\tilde T_1 := \frac{T_1}{\sqrt{T_1^2+T_2^2}}\,\mu N \quad\text{and}\quad \tilde T_2 := \frac{T_2}{\sqrt{T_1^2+T_2^2}}\,\mu N.
\end{aligned}
$$

The constant parameters

$$
\mu, G, C_{11}, C_{22}, C_{23}
$$

(friction coefficient, glide module, Kalker coefficients) are listed in Table II.22.1. For the computation of c, the size of contact ellipse, which uses the parameters σ, $\widehat{G}$ and ϵ, we refer to [Jas87]. For alternative creep force models see also [Jas87].

The normal forces N are given by

$$
\begin{pmatrix} N_L\\ N_R\end{pmatrix} = \gamma\begin{pmatrix} \cos\Delta_R & -\sin\Delta_R\\ -\cos\Delta_L & -\sin\Delta_L\end{pmatrix}\begin{pmatrix} b_{1,1} & b_{1,2}\\ b_{2,1} & b_{2,2}\end{pmatrix}\begin{pmatrix}\lambda_1\\ \lambda_2\end{pmatrix},
$$

where

$$\gamma := \frac{1}{\sin\Delta_L\cos\Delta_R + \sin\Delta_R\cos\Delta_L}\,.$$

Here, $\Delta_{L|R}$ denotes the contact angles and is defined as

$$\begin{aligned}
\tan\Delta_L &= \frac{(R'(\xi_L)\cos\varphi - \sin\varphi\cos\psi_L)\cos\theta + \sin\psi_L\sin\theta}{-R'(\xi_L)\sin\varphi - \cos\psi_L\cos\varphi}\,;\\
\tan\Delta_R &= \frac{(R'(\xi_R)\cos\varphi - \sin\varphi\cos\psi_R)\cos\theta + \sin\psi_R\sin\theta}{+R'(\xi_R)\sin\varphi + \cos\psi_R\cos\varphi}\,.
\end{aligned}$$

For the creepages we have the relations

$$\begin{aligned}
\nu_1 &= \frac{1}{v_{roll}}\left(\sin\theta v_{r1} + \cos\theta v_{r3}\right)\\
\nu_2 &= \frac{1}{v_{roll}}\left(\cos\theta\cos\Delta_{L|R}v_{r1} \pm \sin\Delta_{L|R}v_{r2} - \sin\theta\cos\Delta_{L|R}v_{r3}\right)\\
\varphi_3 &= \frac{1}{v_{roll}}\Big(\mp\sin\Delta_{L|R}(\omega+\beta-v_0\,\kappa\,\sin\alpha) + \cos\Delta_{L|R}(\dot\theta - v_0\,\kappa\cos\alpha)\Big)
\end{aligned}$$

where $v_{r1,2,3}$ (relative velocity at the contact point) and v_{roll} (rolling velocity) are given by (correspondingly for the right side)

$$\begin{aligned}
v_{r1} &= \dot x - \dot\theta(R(\xi_L)(\sin\theta\sin\varphi\cos\psi_L + \cos\theta\sin\psi_L) + \xi_L\sin\theta\cos\varphi)\\
&\quad -\dot\varphi\cos\theta(\xi_L\sin\varphi - R(\xi_L)\cos\varphi\cos\psi_L)\\
&\quad +(\omega_0+\beta)R(\xi_L)(-\sin\theta\cos\psi_L - \sin\varphi\cos\theta\sin\psi_L)\\
&\quad +v_0\kappa\cos\alpha(R(\xi_L)(\sin\theta\sin\varphi\cos\psi_L + \cos\theta\sin\psi_L) + \xi_L\sin\theta\cos\varphi - z),\\
v_{r2} &= \dot y + \dot\varphi(\xi_L\cos\varphi + R(\xi_L)\sin\varphi\cos\psi_L) + (\omega_0+\beta)R(\xi_L)\cos\varphi\sin\psi_L\\
&\quad +v_0\kappa\sin\alpha(z - \xi_L\sin\theta\cos\varphi - R(\xi_L)(\sin\theta\sin\varphi\cos\psi_L + \cos\theta\sin\psi_L)),\\
v_{r3} &= \dot z + v_0 + v_0\kappa(x\cos\alpha - y\sin\alpha)\\
&\quad -\dot\theta(\xi_L\cos\theta\cos\varphi + R(\xi_L)(\cos\theta\sin\varphi\cos\psi_L - \sin\theta\sin\psi_L))\\
&\quad +\dot\varphi\sin\theta(\xi_L\sin\varphi - R(\xi_L)\cos\varphi\cos\psi_L)\\
&\quad +(\omega+\beta)R(\xi_L)(\sin\theta\sin\varphi\sin\psi_L - \cos\theta\cos\psi_L)\\
&\quad -v_0\kappa\sin\alpha(\xi_L\sin\varphi - R(\xi_L)\cos\varphi\cos\psi_L)\\
&\quad +v_0\cos\alpha(\xi_L\cos\theta\cos\varphi + R(\xi_L)(\cos\theta\sin\varphi\cos\psi_L - \sin\theta\sin\psi_L)),
\end{aligned}$$

and

$$v_{roll} = \frac{1}{2}\left\|\begin{pmatrix} -2\dot x + 2v_0\kappa z\cos\alpha\\ -2\dot y - 2v_0\kappa z\sin\alpha\\ -2\dot z - 2v_0 - 2v_0\kappa(x\cos\alpha - y\sin\alpha)\end{pmatrix} + \begin{pmatrix} v_{r1}\\ v_{r2}\\ v_{r3}\end{pmatrix}\right\|_2\,.$$

22.2.2 Constraints

The constraints (II.22.3) read

$$\begin{pmatrix} G(\hat\xi_L) - y - \xi_L\sin\varphi + R(\xi_L)\cos\varphi\cos\psi_L\\ G(\hat\xi_R) - y - \xi_R\sin\varphi + R(\xi_R)\cos\varphi\cos\psi_R\end{pmatrix} = 0$$

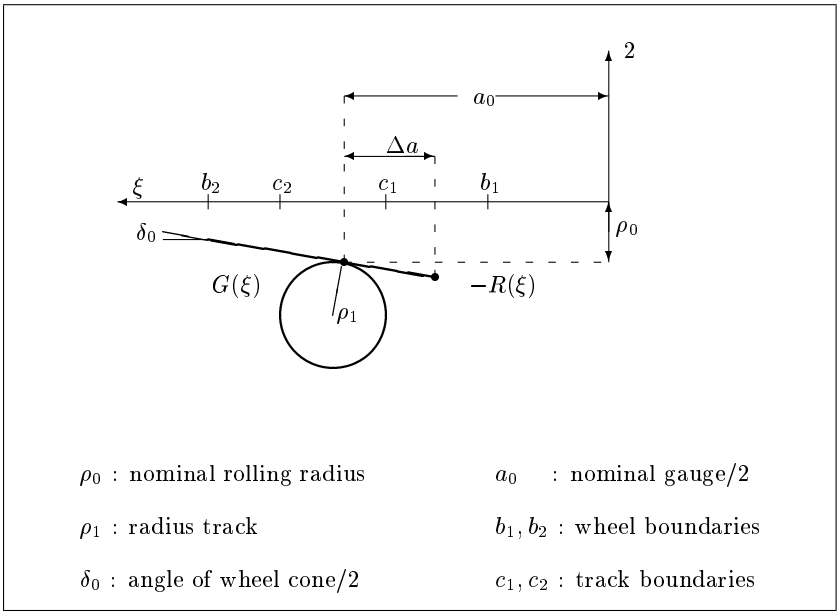

FIGURE II.22.1: *Profile functions (left side).*

with profile functions R (wheel) and G (rail), see Figure II.22.1,

$$R(\xi) \quad = \quad \rho_0 + \tan\delta_0\,(a_0 - |\xi|) \qquad \text{for } a_0 - \Delta a < |\xi| < b_2\,;$$

$$G(\hat{\xi}) \quad = \quad \sqrt{\rho_1^2 - \left(|\hat{\xi}| - a_0 - \rho_1 \sin\delta_0\right)^2} - \rho_0 - \cos\delta_0\,\rho_1 \qquad \text{for } c_1 < |\hat{\xi}| < c_2\,.$$

Here, ξ stands for the left or right coordinate $\xi_{L/R}$, respectively, and $\hat{\xi}$ is defined by

$$\hat{\xi}_{L|R} := x + \xi_{L|R}\cos\theta\cos\varphi + R(\xi_{L|R})\left(\cos\theta\sin\varphi\cos\psi_{L|R} - \sin\theta\sin\psi_{L|R}\right).$$

The constraints (II.22.4) read

$$\begin{aligned} G'(\hat{\xi}_L)\,(R'(\xi_L)\sin\varphi + \cos\varphi\cos\psi_L) + R'(\xi_L)\cos\theta\cos\varphi & \\ -\cos\theta\sin\varphi\cos\psi_L + \sin\theta\sin\psi_L \quad &= \quad 0, \\ R'(\xi_L)\sin\theta\cos\varphi - \sin\theta\sin\varphi\cos\psi_L - \cos\theta\sin\psi_L \quad &= \quad 0, \\ G'(\hat{\xi}_R)\,(R'(\xi_R)\sin\varphi + \cos\varphi\cos\psi_R) + R'(\xi_R)\cos\theta\cos\varphi & \\ -\cos\theta\sin\varphi\cos\psi_R + \sin\theta\sin\psi_R \quad &= \quad 0, \\ R'(\xi_R)\sin\theta\cos\varphi - \sin\theta\sin\varphi\cos\psi_R - \cos\theta\sin\psi_R \quad &= \quad 0, \end{aligned}$$

where $G'(\hat{\xi}_{L|R}) := \frac{\mathrm{d}}{\mathrm{d}\hat{\xi}_{L|R}} G(\hat{\xi}_{L|R})\,, R'(\xi_{L|R}) := \frac{\mathrm{d}}{\mathrm{d}\xi_{L|R}} R(\xi_{L|R})$.

22.3 Origin of the problem

The motion of a simple wheelset on a rail track exhibits a lot of the difficulties which occur in the simulation of contact problems in mechanics. The state space form approach for this class of problems

TABLE II.22.1: *Parameter values according to [Jas90], where a hardware bogie model, scaled 1:4, is investigated.*

Parameter	Meaning	Unit	Value
m_R	mass wheelset	kg	16.08
$\tilde{g}$	gravity constant	m/s^2	9.81
v_0	nominal velocity	m/s	30.0
F_A	propulsion force	N	0
L_A	propulsion moment	kg m^2	0
κ	describes track geometry		0
α	describes track geometry	rad	0
ω_0	nominal angular velocity	1/s	v_0/ρ_0
I_1	lateral moment of inertia	kg m^2	0.0605
I_2	vertical moment of inertia	kg m^2	0.366
m_A	mass of wagon body	kg	0.0
h_A	height of wagon body	m	0.2
c_x	spring constant	N/m	6400.0
c_z	spring constant	N/m	6400.0
x_l	width of wheelset/2	m	0.19
δ_0	cone angle/2	rad	0.0262
ρ_0	nominal radius	m	0.1
a_0	gauge/2	m	0.1506
ρ_1	radius track	m	0.06
μ	friction coefficient		0.12
G	glide module	N/m^2	$7.92 \cdot 10^{10}$
C_{11}	Kalker coefficient		4.72772197
C_{22}	Kalker coefficient		4.27526987
C_{23}	Kalker coefficient		1.97203505
$\widehat{G}$	parameter for computation of contact ellipse		0.7115218
ϵ	parameter for computation of contact ellipse		1.3537956
σ	parameter for computation of contact ellipse		0.28
C	scaling factor for Lagrange multipliers		10^4

requires simplifications and table look ups in order to eliminate the nonlinear constraints. The above example provides thus an alternative by using the IDE approach.

Figure II.22.2 shows the mechanical model. The coordinates p denote the displacements and rotations of the wheelset with respect to the reference frame which is centered in the middle of the track. The wheelset is subjected to

- the gravity and centrifugal forces;
- creep forces in the contact points of wheel and rail;
- forces of the wagon body, which is represented by a frame connected to the wheelset via springs and dampers and proceeding with constant speed v_0;
- constraint forces which enforce the contact of wheel and rail on both sides.

We are particularly interested in a complete and correct formulation of the nonlinear constraint equations. An elimination of the constraints without severe simplifications or the introduction of tables for the dependent variables is impossible. In this example thus a reduction to state space form involves various obstacles, whereas the IDE formulation is straightforward.

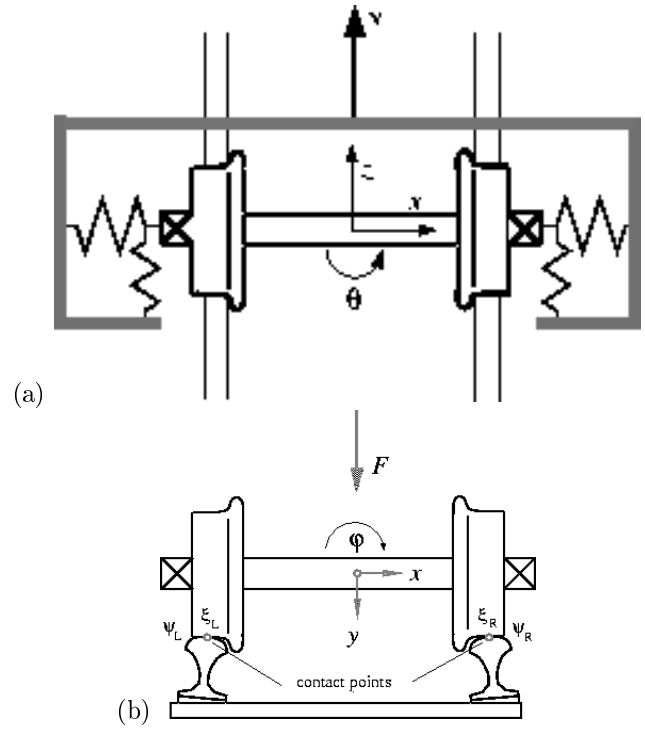

FIGURE II.22.2: *The wheelset and the track. (a) View from above, (b) lateral cross section.*

Equations (II.22.1)–(II.22.2) stand for the kinematic and dynamic equations with positive definite mass matrix $M(p)$. By means of the profile functions R and G which describe the cross sections of wheel and rail depending on the contact points we first express the constraint equations as $0 = g_1$, see Figure II.22.3. These constraints are of index 3 and enforce that the contact points of wheel and rail coincide on both sides. Additionally, we have to guarantee that wheel and rail do not intersect, which is accomplished by the conditions $0 = g_2$. Note that $\partial g_2/\partial q$ is regular, which means that we can apply formally the implicit function theorem to eliminate the additional contact variables q and that these constraints are of index 1. The equations of motion of the wheelset are then derived by applying the formalism of Newton and Euler. Here we used the property that this class of contact problems $(\partial g_1/\partial) q\, \dot{q} \equiv 0$. This also implies that if we, in order to get the index 2 formulation, differentiate the constraint (II.22.3) with respect to t, then we get

$$0 = \frac{\mathrm{d}g_1}{\mathrm{d}t}(p,q) = \frac{\partial g_1}{\partial p}\dot{p} + \frac{\partial g_1}{\partial q}\dot{q} = \frac{\partial g_1}{\partial p}\dot{p} - \frac{\partial g_1}{\partial q}\left(\frac{\partial g_2}{\partial q}\right)^{-1}\frac{\partial g_2}{\partial p}\dot{p},$$

which simplifies to (II.22.5).

Remarks

- $N(p,q,\lambda) \in \mathbb{R}^2$ denotes the normal forces which act in the contact points. They are necessary to evaluate the creep forces.
- The variable $\beta \in \mathbb{R}$ denotes the deviation of the angular velocity and is given by an additional differential equation.
- The parameters κ and α describe the track geometry. The setting $\kappa = \alpha = 0$ refers to a straight track.

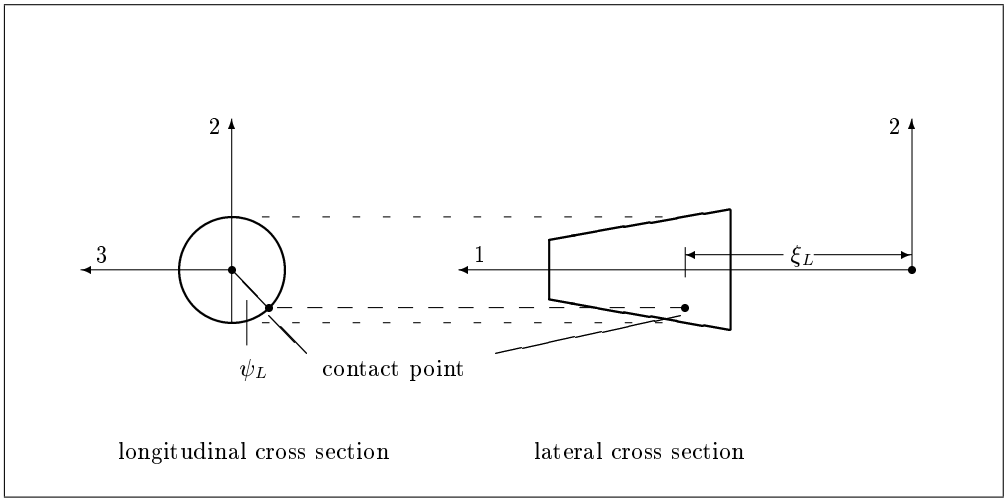

FIGURE II.22.3: *Shift angle and coordinate of contact point on the left side.*

TABLE II.22.2: *Reference solution at the end of the integration interval.*

u_1	$0.86355386965811 \cdot 10^{-2}$	u_{10}	$-0.13633468454173 \cdot 10^{-1}$
u_2	$0.13038281022727 \cdot 10^{-4}$	u_{11}	-0.24421377661131
u_3	$-0.93635784016818 \cdot 10^{-4}$	u_{12}	$-0.33666751972196 \cdot 10^{-3}$
u_4	$-0.13642299804033 \cdot 10^{-1}$	u_{13}	-0.15949425684022
u_5	$0.15292895005422 \cdot 10^{-2}$	u_{14}	$0.37839614386969 \cdot 10^{-3}$
u_6	$-0.76985374142666 \cdot 10^{-1}$	u_{15}	0.14173214964613
u_7	$-0.25151106429207 \cdot 10^{-3}$	u_{16}	$-0.10124044903201 \cdot 10^{-1}$
u_8	$0.20541188079539 \cdot 10^{-2}$	u_{17}	$-0.56285630573753 \cdot 10^{-2}$
u_9	-0.23904837703692		

- The constant C in (II.22.2) means that we internally scaled the Lagrange multipliers.

The initial values correspond to a setting in which the dynamic behavior of the wheelset model is investigated when the wheelset starts with an initial deflection in lateral direction (x-direction) of 0.14941 [cm]. In [Jas90], a limit cycle was observed for this problem and the model data given above. This type of limit cycle, the so-called hunting motion, is a well known phenomenon in railway vehicle dynamics. In Figure II.22.4 we see this limit cycle as computed by DASSL applied to the index-2 formulation of the problem. The results are in good agreement with those given in [Jas90], which were obtained by a state space form approach and with measurements on a hardware model.

22.4 Numerical solution of the problem

Tables II.22.2–II.22.3 present the reference solution at the end of the integration interval, and the run characteristics, respectively. Figure II.22.5 shows the the behavior of the components of p and the angular velocity β over the integration interval. Figures II.22.6- II.22.7 contain the work-precision diagrams. For this diagrams, we used: rtol $= 10^{-(4+m/8)}$, $m = 0, 1, \ldots, 48$; atol = rtol, h0 = rtol for MEBDFI. .

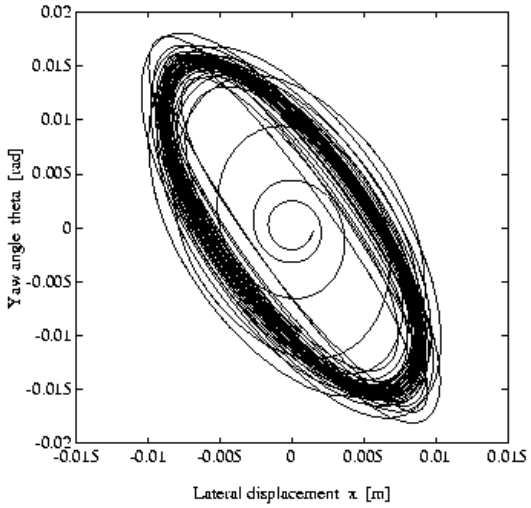

FIGURE II.22.4: *Limit cycle or 'hunting motion' of wheelset.*

Remarks

- The Jacobian was computed internally by the solvers.
- For the runs with DASSL, we excluded the Lagrange multipliers from the error control by setting `atol(16)=atol(17)=rtol(16)=atol(17)=`10^{10}.
- The reference solution was computed using DASSL with atol = rtol = 10^{-9} for p, v and q, and atol = rtol = 10^{10} for λ.

TABLE II.22.3: *Run characteristics.*

solver	rtol	atol	h0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU
DDASSL	10^{-4}	10^{-4}		1.35	0.15	5949	5117	10304	1407		0.3250
	10^{-5}	10^{-5}		2.78	1.40	9888	8667	16150	1815		0.4782
	10^{-6}	10^{-6}		3.67	2.32	16010	14298	25256	2577		0.7213
MEBDFI	10^{-4}	10^{-4}	10^{-4}	1.32	0.12	5758	5188	42694	1185	1185	0.4031
	10^{-5}	10^{-5}	10^{-5}	3.93	2.59	9317	8485	64945	1765	1765	0.6266
	10^{-6}	10^{-6}	10^{-6}	4.89	3.22	13240	12255	86260	2248	2248	0.8560
PSIDE-1	10^{-4}	10^{-4}		1.53	0.42	1276	945	22090	547	4920	0.5134
	10^{-5}	10^{-5}		2.81	1.67	2335	1507	39204	608	8752	0.8384
	10^{-6}	10^{-6}		4.52	3.34	3070	2068	54074	571	10736	1.0775

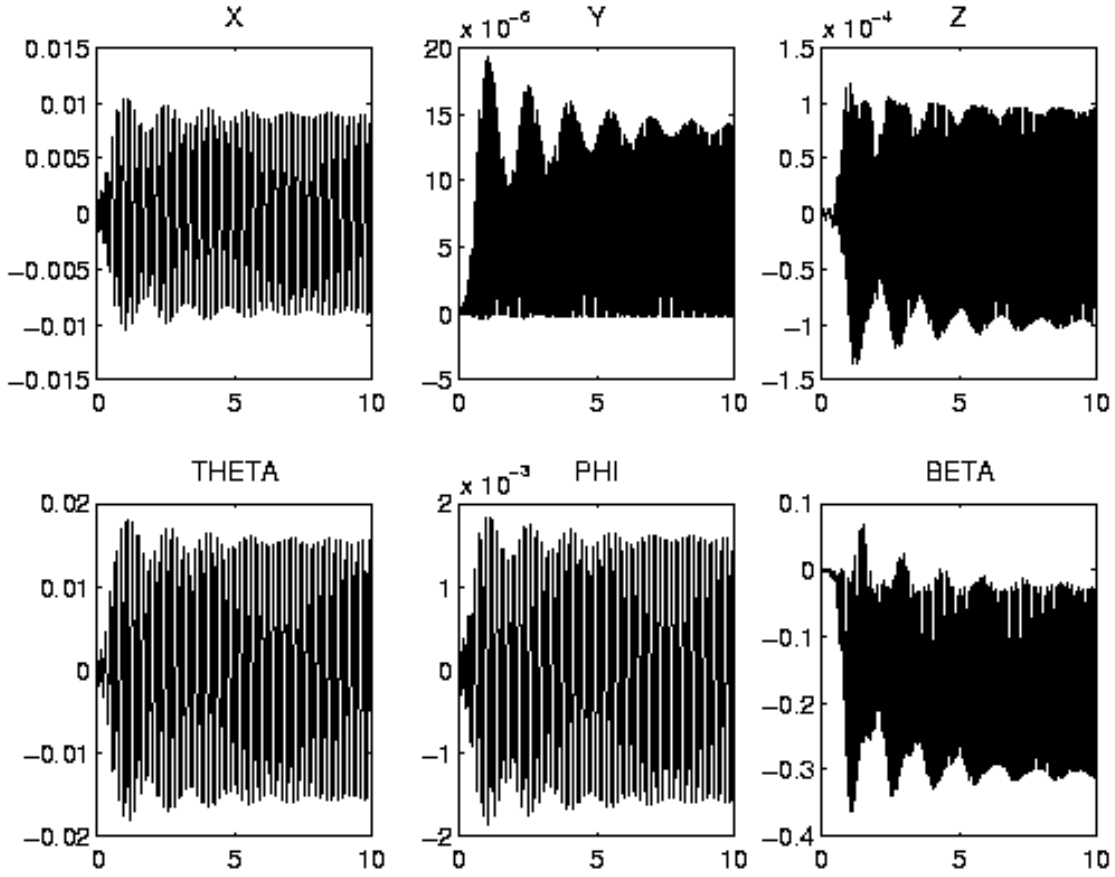

FIGURE II.22.5: *Behavior of some solution components over the integration interval.*

References

[Jas87] A. Jaschinski. Anwendung der Kalkerschen Rollreibungstheorie zur dynamischen Simulation von Schienenfahrzeugen. Technical Report DFVLR 87-07, DFVLR Deutsche Forschungs-

und Versuchsanstalt für Luft- und Raumfahrt, D-8031 Oberpfaffenhofen, 1987.

[Jas90] A. Jaschinski. *On the Application of Similarity Laws to a Scaled Railway Bogie Model*. PhD thesis, Technische Universiteit Delft, 1990.

[MM08] F. Mazzia and C. Magherini. *Test Set for Initial Value Problem Solvers, release 2.4*. Department of Mathematics, University of Bari and INdAM, Research Unit of Bari, February 2008. Available at http://www.dm.uniba.it/~testset.

[SFR91] B. Simeon, C. Führer, and P. Rentrop. Differential-algebraic equations in vehicle system dynamics. *Surv. Math. Ind.*, 1:1–37, 1991.

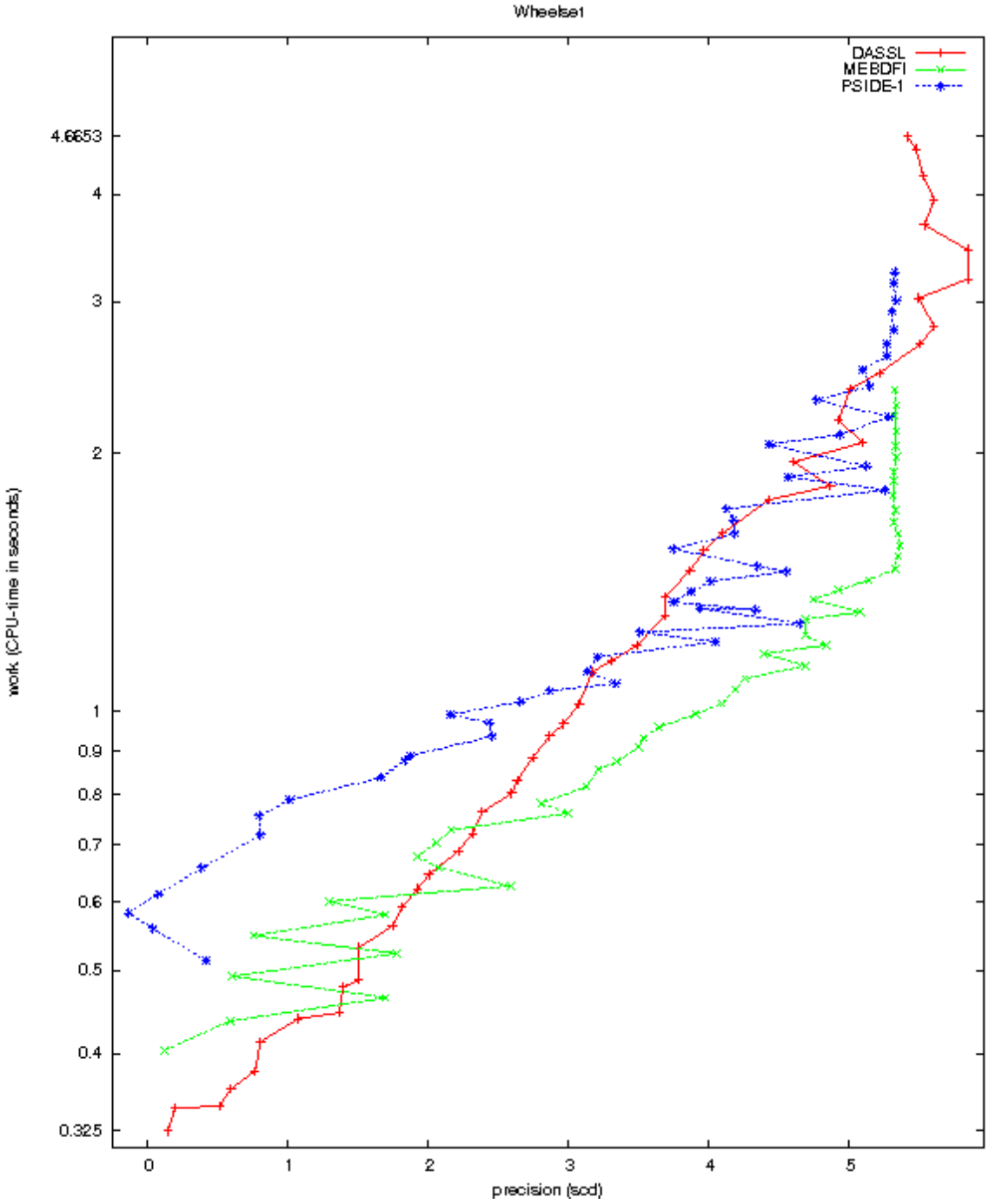

FIGURE II.22.6: *Work-precision diagram (scd versus CPU-time).*

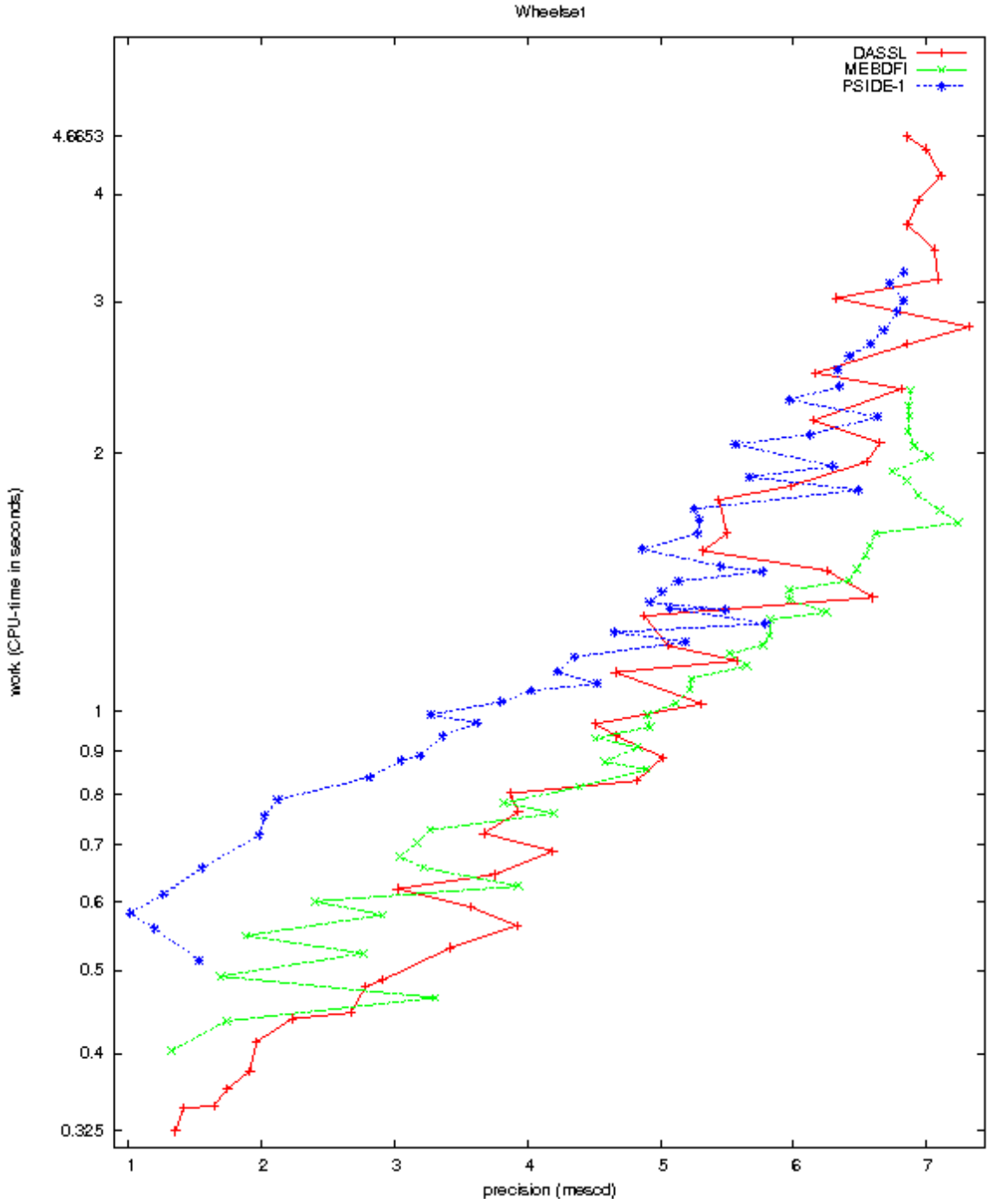

FIGURE II.22.7: *Work-precision diagram (mescd versus CPU-time).*