CHEMSIMUL - A Program Package for Numerical Simulation of Chemical Reaction Systems.

Rasmussen, O Lang; Bjergbakke, Erling

Publication date:
1984

Document Version
Publisher's PDF, also known as Version of record

Risø-R-395

RISØ

Risø-R-395

CHEMSIMUL - A Program Package for Numerical Simulation of Chemical Reaction Systems

Ole Lang Rasmussen og Erling Bjergbakke

Risø National Laboratory, DK-4000 Roskilde, Denmark
January 1984

Risø-R-395

CHEMSIMUL - A PROGRAM PACKAGE FOR NUMERICAL SIMULATION OF CHEMICAL REACTION SYSTEMS

Ole Lang Rasmussen and Erling Bjergbakke

Computer Installation and Accelerator Departments

Abstract. A description is given of a program package, CHEMSIMUL, for numerical simulation of chemical reaction systems. The main components in the package are a translator of chemical equations to differential equations, a balance equation program, a differential equation solver, EPISODE, and an input/output program. The performance of the program is demonstrated by four examples. A manual for the input file and the complete program text with comments are given in Appendices I and II.

Inis-descriptors: C CODES; CHEMICAL REACTION KINETICS; CHEMICAL REACTIONS; REACTIONS; DIFFERENTIAL EQUATIONS; NUMMERICAL SOLUTION; SIMULATION

UDC 541.124 : 681.3.06

January 1984

Risø National Laboratory, DK 4000 Roskilde, Denmark

CONTENTS

1. INTRODUCTION

Chemical systems in nature or in industrial processes are usually very complex. Even in experimental systems it is often difficult to obtain simple kinetics. The classical way of interpreting kinetics - analytical solution of the differential equations and/or steady-state approximation - already fails in fairly simple chemical systems. During the last twenty years it has become evident that numerical integration of complex differential equation systems by means of modern computers will be the dominant method in chemical kinetics.

At RISØ NATIONAL LABORATORY (RNL) the use of computers in chemical kinetics was started in 1966 when H. Fricke was engaged as a consultant in the Accelerator Department. The work on the development of the kinetic program has been continued since then in close cooperation between the mathematician (programmer) and the chemist (user).

The first computations were performed on very simple chemical systems. The first differential equation solver utilized the well-known fifth-order Runge-Kutta method. This method is suited for non-stiff differential equation systems, but we soon realized that the differential equations governing a chemical reaction are of the stiff type. A method for handling stiff systems was published by C.W.Gear in 1971 [1] and shortly after that we started to use this method. In the following years other implementations of Gear's method have been made, e.g. by Hindmarsh and Byrne[2], and in the present version of our simulation program we use their EPISODE package instead of the original DIFSUB by C.W.Gear.

In the first numerical integration program we constructed the differential equations manually. This was a feasible method as long as the chemical systems was simple, but as the systems became larger, the method was no longer practical. This led to the development of a subroutine that translates a set of chemical equations into the corresponding set of differential equations, which then can be processed by a differential equation solver.

CHEMSIMUL has been in use for several years at RNL, and several modifications were implemented during the years. We believe that the present version will be the basal program for many years. Future improvements and additional features in the program will be published as appendixes to this report.

2. THE DIFFERENTIAL EQUATIONS

We start with a demonstration of how we can derive the differential equation system describing the chemical process from a simple reaction equation.

Let the chemical reaction equation be written symbolically as:

$$R1 + R2 \rightarrow R3 + R4$$

and let the rate constant be denoted by k. Furthermore, let the chemical medium containing the 4 reactants R1, R2, R3 and R4 be irradiated either by an electronic beam or by gamma-rays. Let us assume that R2 and R3 are produced by the radiation and let G(R2) and G(R3) be the radiation yield measured in molecules/100 eV. Then the differential equations are:

$$\begin{aligned} d[R1]/dt &= -k\times[R1]\times[R2] \\ d[R2]/dt &= -k\times[R1]\times[R2] + const\times G(R2)\times D \\ d[R3]/dt &= \ k\times[R1]\times[R2] + const\times G(R3)\times D \\ d[R4]/dt &= \ k\times[R1]\times[R2] \end{aligned}$$

where D is dose rate in krd/sec; const is given by
const = $1.036\ mol\times dm^{-3}\times 100\ eV\times molecules\times krd^{-1}$
[R1] is concentration of R1 in $mol\times dm^{-3}$.

Notice that the sign of k is negative in the two first equations because the concentration of R1 and R2 are decreasing but positive in the last two since the concentrations R3 and R4 are increasing. We see that a single chemical reaction equation can be described by a system of ordinary nonlinear differential equations. Starting with the time $t = t_o$ and with the initial values of the reactants $[R1]_o$, $[R2]_o$, $[R3]_o$ and $[R4]_o$ we can integrate the system up to the time $t = t_{end}$.

After this introductory simple example we will generalize the problem: A general reaction system can thus be described by a set of reaction equations, each equation having the form:

$$n_i\times R_i + n_j\times R_j \rightarrow \sum_{k=1}^{N} n_k\times R_k$$

where R_i and R_j are initial reactants, R_k, $(k=1,\ldots i-1,i+1,\ldots j-1,j+1,\ldots N)$

are products; n_i, n_j and n_k are integers on which we put the restrictions $n_i>=0$, $n_j>=0$ and $1<=n_i + n_j<=2$ since only first- and second-order reactions are allowed.

The general system of differential equations describing the changes of reactants concentration can formally be written as:

$$dR_i/dt = \sum_{\substack{j,k=1 \\ j,k\neq i}}^{N} n_{jk} \times M_{ijk} \times R_j \times R_k + \sum_{\substack{j=1 \\ j\neq i}}^{N1} n_{j1} \times M_{ij1} \times R_j \times 1 + const \times G(R_i) \qquad (1)$$

$i = 1, 2, \ldots\ldots\ldots\ldots$; $n_{jk} = 1$ if $j\neq k$, $n_{jk} = 2$ if $j=k$.

N and N1 are numbers of M_{ijk} respective M_{ij1} which are different from zero. M_{ijk} can be considered as a 3-dimensional matrix whose elements are the reaction constants. For a given i, M_{ijk} is the reaction constants belonging to the reaction equations in which R_i appears either on the left or on the right side. Since the products $R_j \times R_k$ do not exist for all sets (j,k) then some M_{ijk} are zero. Only those sets (j,k) exist for which the reactants (R_j, R_k) appear on the left side of the reaction equation system. If some of the contributions to dR_i/dt are due to a first-order reaction, i.e. there is only one element on the left side, then one of the concentrations for a reactant is arbitrarily set equal to 1. Thus the equation (1) can be used to describe zeroth-, first-, and second-order reactions. The first sum describes second order, the second sum first-order and the last term the zeroth-order reactions. The integer $n_{jk}(n_{j1})$ is positive when R_i appears on the right side of the reaction equation, otherwise negative.

It is not difficult to see that a reaction system need not be very large before it becomes a cumbersome task to write down the corresponding system of differential equations. The translation of the chemical reaction system should therefore be done automatically. In the next chapter we shall outline the principle for doing this. For details the reader is recommended to study the program description in Appendix II.

3. TRANSLATION FROM REACTION EQUATIONS TO DIFFERENTIAL EQUATIONS

CHEMSIMUL is mainly written in Burroughs Extended Algol language, only the differential equation solver and the subroutines for computing the right-hand

side of the differential equation system and its Jacobian matrix are written in FORTRAN-66. The reason for this is that the differential equation solver used in this program is the EPISODE - package coded in FORTRAN-66 by A.C. Hindmarsh[2]. Since it is possible to call a Fortran subroutine from a program written in Burroughs Extended Algol we have avoided translating EPISODE to Algol.

We have chosen Burroughs Extended Algol as our programming language for two reasons: First of all, this language is the basic programming language for the Burroughs computer system at Risø, and secondly, it is unique with respect to language concepts. One of its strong features is its capability of handling character strings, a feature which is used in the translation of the chemical reaction system to the corresponding system of ordinary differential equations.

The chemical reaction equations are expressed in a nomenclature very similar to that used by chemists which facilitates the use of CHEMSIMUL. The program contains a translation part that, by analyzing the reaction equations, identifies and stores the names of the reactants and stores the reaction constants belonging to the reaction equations. Then a table, a 2-dimensional matrix, is constructed, which contains the information necessary for computing the right-hand side of the differential system. Each line in the matrix consists of 5 elements containing the address of the reactant R_i for which the derivative is to be computed, the stoichiometric constant n_{jk}, the address for a rate constant M_{ijk} and two addresses for reactants R_j and R_k on the left side of the reaction equation. For each reactant in the reaction system there corresponds a line with 5 elements in the matrix. This matrix is used by the subroutine computing the right-hand side of the differential equation system and the Jacobian matrix for the system.

4. THE BALANCE EQUATION

The accurate solution of the differential equation system describing the chemical reactions requires an overall conservation of the mass balance[3]. Two main factors are involved in the conservation of mass namely the reaction mechanism and the integration process. The integration method employed (EPISODE) is a linear multistep method. Rosenbaum[4,5] has proved that this inte-

gration method has the implicit property of mass preservation. We have included a mass balance control in our program, and the integration is performed only if this control is positive. The balance equation checks if the reaction mechanism formally obeys the demand for stoichiometric balance. But it must be pointed out that the formal stoichiometric balance need not be the same as an atom-to-atom balance. The program recognizes each species as a name and the individual letters and numbers in the name is of no significance in the balance equation. It is unnecessary to check the mass balance continuously, it is sufficient to do that at the start of the integration, i.e. at time t=0. The control of the mass balance is done as follows: Let the chemical system be specified by M reaction equations with N reactants, each reactants being represented by an alphanumerical symbol (or its true chemical symbol). Then the system can be represented by a matrix equation

$$A \times V = Z \qquad (2)$$

The elements of matrix A are the stoichiometric constants in the reaction equations and the elements of vector V are the symbols representing the reactants while Z on the right side is the zero vector. Each row in A is derived from a reaction equation by collecting all the terms on the same side, e.g. the left side. Therefore, each row must have both positive and negative stoichiometric constants together with zero elements. Since equation (2) represents a real process it must be possible to find a nonzero solution for V whose elements are strictly positive. If N=M then the rank of A must be smaller than N and if N is different from M then the rank of A must be smaller or equal to the smallest value of N or M. By Gaussian elimination in integer arithmetic (the coefficients in A are all integers since stoichiometric constants are by definition integers) we end up with a matrix whose rows either contains zeros exclusively or have both positive and negative elements (and zeros) if the reaction system has a correct mass balance. If during the Gaussian elimination it happens that the nonzero elements in a row are exclusively positive or negative then we can conclude that the system is not in mass balance.

5. USERS REQUIREMENTS

CHEMSIMUL is constructed to simulate homogeneous kinetics in monophase. Only zeroth-, first-, and second-order reactions are allowed. Reactions of higher order must be constructed from a suitable number of first- and second-order reactions. The philosophy behind this design is that the user will be forced to match his reaction system to physical reality as closely as possible. The reactions are written in normal chemical notation, e.g.

$A = B + C;$ k_1 (first-order reaction)
$A + D = E + F;$ k_2 (second-order reaction)

k_1 and k_2 are the rate constants for the reactions. Zeroth-order reactions are executed by means of a G-value.

G-value is a term used in radiation chemistry and defined as the number of molecules produced per 100 eV radiation energy absorbed. The zeroth-order reaction has the following differential equation:

$$d[A]/dt = G(A) \times doserate \times k \ mol \times dm^{-3} \times sec^{-1}$$

where the doserate is in krd per second and the conversion factor k is 1.036×10^{-6}. This means that if G(A) = 1 and the doserate is 1 krd/sec then

$$d[A]/dt = 1 \times 1 \times 1.036 \times 10^{-6} \ mol \times dm^{-3} \times sec^{-1}$$

The monophase system can be extended to include simple solid-liquid and liquid-gas equilibria. If the problem is to maintain equilibrium between dissolved oxygen in liquid phase and oxygen in gas phase then the tool is to use first-order reactions

$O2 = Dummy1 ; k_1$
$Dummy1 = O2 ; k_2$

If the problem is to maintain a saturated solution, the two reactions

$Fe(solid) = Fe[++]; k_1$
$Fe[++] = Fe(solid) ; k_2$

must be combined with a large initial concentration of Fe(solid) so that the

consumption of Fe[++] through other reactions does not alter the equilibrium. The same applies for liquid-gas equilibrium. Precipitation is treated in a similar way. The program is designed to calculate irradiation as well either during the whole reaction period, or during a fraction(s) of the period. It can also be used without irradiation and for the addition of reagents during reaction.

The standard version of the program uses first- or second-order rate constants, but a special version uses activation energies (E_a) and frequency factor (A) combined with a temperature-time function so that the rate constants are changed with the changes in temperature according to the Arrhenius equation

$$k = A \times \exp(-E_a/RT)$$

The number of reactions have a preset maximum number of 100 but it is changed automatically by the program when necessary.

6. OUTPUT POSSIBILITIES

The standard-output form is a table with concentrations of all the reacting species as function of time. Plots of selected species concentration as function of time are optional. The scales of the plots can be chosen either as linear-linear, linear-logarithmic or logarithmic-logarithmic. The variable in a plot can be the concentration of species (combination of concentrations of a number of species) or simple arithmetic expressions (addition, subtraction, and multiplication - not division) containing species concentrations. The purpose is to make it possible to plot a curve identical to the experimental measurements, e.g. extinction $E = (\varepsilon_A \times [A] + \varepsilon_B[B]) \times \ell$, where ε_A and ε_B are the extinction coefficients for species A and B and ℓ is the optical pathlength. For comparison it is possible to include a number of experimental results in the plot. Maximum of experimental points has a preset value of 20, but can be extended to the users demand.

7. THE INPUT FILE

In the design of the data input file it was the intention to make it as close to the standard chemical notation as possible. (For a complete description of the input file see Appendix I).

A reaction equation for reaction no.1

$$H^+ + OH^- \rightarrow H_2O$$

with the rate constant 1.43×10^{11} $dm^3 mol^{-1} sec^{-1}$ is entered in the following form:

```
RE1: H[+] + OH[-] = H2O; 1.43E11
```

The concentration of $H^+ = 10^{-7}$ M is entered as

```
CON(H[+]) = E-7
```

Radiation yield $G(H^+) = 3.32$ is entered as

```
G(H[+]) = 3.32
```

Irradiation dose 7 krd as

```
DOSE = 7
```

Irradiation duration 10^{-6} sec as

```
RADTIME = E-6
```

Integration time (=reaction time) 10 sec as

```
TEND = 10
```

Line numbers in the output table 20 as

```
PRINTS = 20
```

Five of these are printouts during irradiation

```
RADPRS = 5
```

A plot of H^+ concentration as

```
PE1: H[+];
ENDPE
```

A plot of H^+ concentration and experimental values of H^+ at time t_n as

```
PE1: H[+];
t1 , [H+]t1
t2 , [H+]t2
.
.
.
tn , [H+]tn
ENDPE
```

8. TEST EXAMPLES

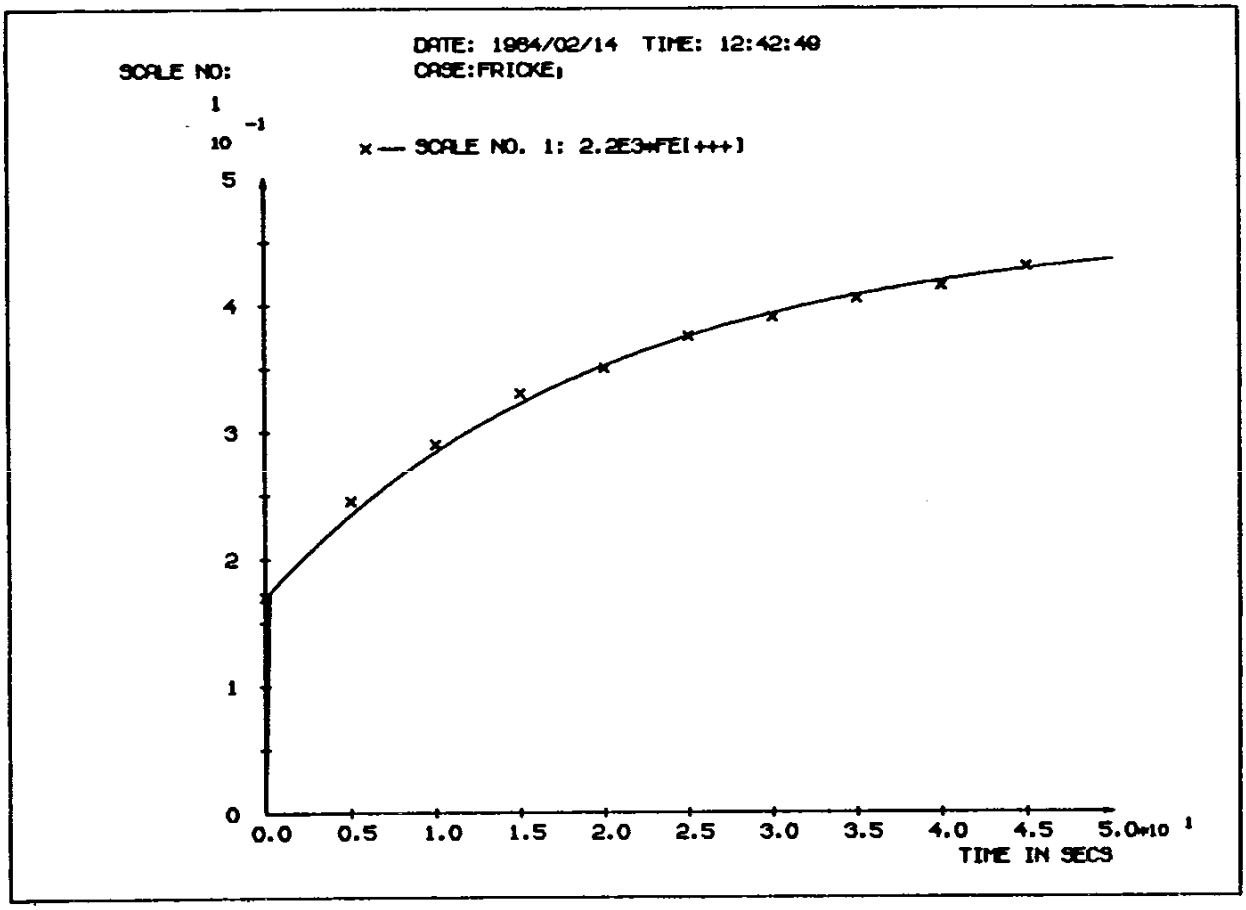
For documentation of the performance of the program we have calculated a pulse radiolysis experiment on the Fricke dosimeter[6] and three well documented systems, the SMOG problem[7], the H_2-O_2 combustion[8], and the modeling of the Belusov reaction, Oregonator[9].

8.1 Fricke dosimeter

The Fricke dosimeter is 1 mM ferrous sulfate in aerated 0.4 M sulfuric acid. The ferrous ions are oxydized to ferric ions by the irradiation. The experimental points on the curve are measured buildups of ferric ion extinction following a 15.5 krd electron pulse of 1 μsec duration. The extinction was measured in a 1-cm cell; the molar extinction coefficient for ferric ions is 2200.

```
NEWSYS
RE1:H+O2=HO2;2E10
RE2:2*H=H2;1.1E10
RE3:H+OH=H2O; 1.5E10
RE4:H+HO2=H2O2;1.3E10
RE5:OH+HO2=H2O3;1.18E10
RE6:2*OH=H2O2;6E9
RE7:OH+FE[++]=FE[+++]+OH[-];2.5E8
RE8:2*HO2=H2O2+O2;1.05E6
RE9:HO2+FE[++]=FE[+++]+HO2[-];1.6E6
RE10:HO2[-]+H[+]=H2O2;E5
RE11:H2O2+FE[++]=FE[+++]+OH+OH[-];55
RE12:H2O3+FE[++]=FE[+++]+HO2+OH[-];6E4
RE13:H+FE[++]=HFE[++];1.6E7
RE14:HFE[++]+H[+]=FE[+++]+H2;E5
RE15:H+H2O2=H2O+OH;6E7
RE16:H+FE[+++]=FE[++]+H[+];1.0E8
RE17:H2O3=H2O+O2;2.1
G(H)=3.7
G(OH)=2.9
G(H2O2)=0.8
G(H2)=0.4
CON(FE[++])=E-3
CON(O2)= 2.4E-4
CON(H[+])=.5
DOSE=15.5
RADTIME=E-6
NRR=1
PRINTS=200
RADPRS=1
TEND=50
HMAX=.2
EPS=E-5
FSTSTP=1E-8
CASE:FRICKE;
PE1:2.2E3*FE[+++];
E-3,0.17
5,0.245
10,0.29
15,0.33
```

```
20,0.35
25,0.375
30,0.39
35,0.405
40,0.415
45,0.43
ENDPE
DEVNAME=BENSON1133
DEVCLASS=PLOTTER
RESCLASS=1
ENDDATA
```


DATE: 1984/02/14 TIME: 12:42:49
CASE:FRICKE1
SCALE NO:
1
10 -1
x — SCALE NO. 1: 2.2E3*FE(+++)
5
4
3
2
1
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0*10 1
TIME IN SECS

8.2 SMOG

Farrow and Edelson's reaction scheme[7] describes the atmospheric chemistry of a NO_2, NO, O_2, H_2O, C_3H_6 mixture. The original scheme consists of 81 equations, the concentrations are given in ppm, the time in min and the rate constants in min^{-1}, $ppm^{-1} \times min^{-1}$ for first- and second-order reactions. Higherorder reactions were reduced to second-order reactions as follows:

11. $NO + NO_2 + H_2O \rightarrow 2HNO_2$, $k=10^{-3}$ $ppm^{-1}min^{-1}$

was changed to

11. $NO + NO_2 \rightarrow N_2O_3$; $k = 10^{-3}$ $ppm^{-1}min^{-1}$ and
12. $N_2O_3 + H_2O \rightarrow 2HNO_3$; $k = 10^{3}$ $ppm^{-1}min^{-1}$

14. $CO + OH + O_2 \rightarrow CO_2 + HO_2$; $k = 9.8 \times 10^{-4}$ $ppm^{-2} \times min^{-1}$

was changed to

14. $CO + OH \rightarrow COOH$; $k = 1.96 \times 10^{-2}$ $ppm^{-1} \times min^{-1}$ and
15. $COOH + O_2 \rightarrow CO_2 + HO_2$; $k = 1$ $ppm^{-1} \times min^{-1}$

Reactions 22, 39, and 46 were changed similarly. The pseudo-first-order reaction

7. $N_2O_5 + H_2O \rightarrow 2HNO_3$; $k = 0.75$ min^{-1}

was changed to a second-order reaction

7. $N_2O_5 + H_2O \rightarrow 2HNO_3$; $k = 7.5 \times 10^{-4}$ $ppm^{-1}min^{-1}$

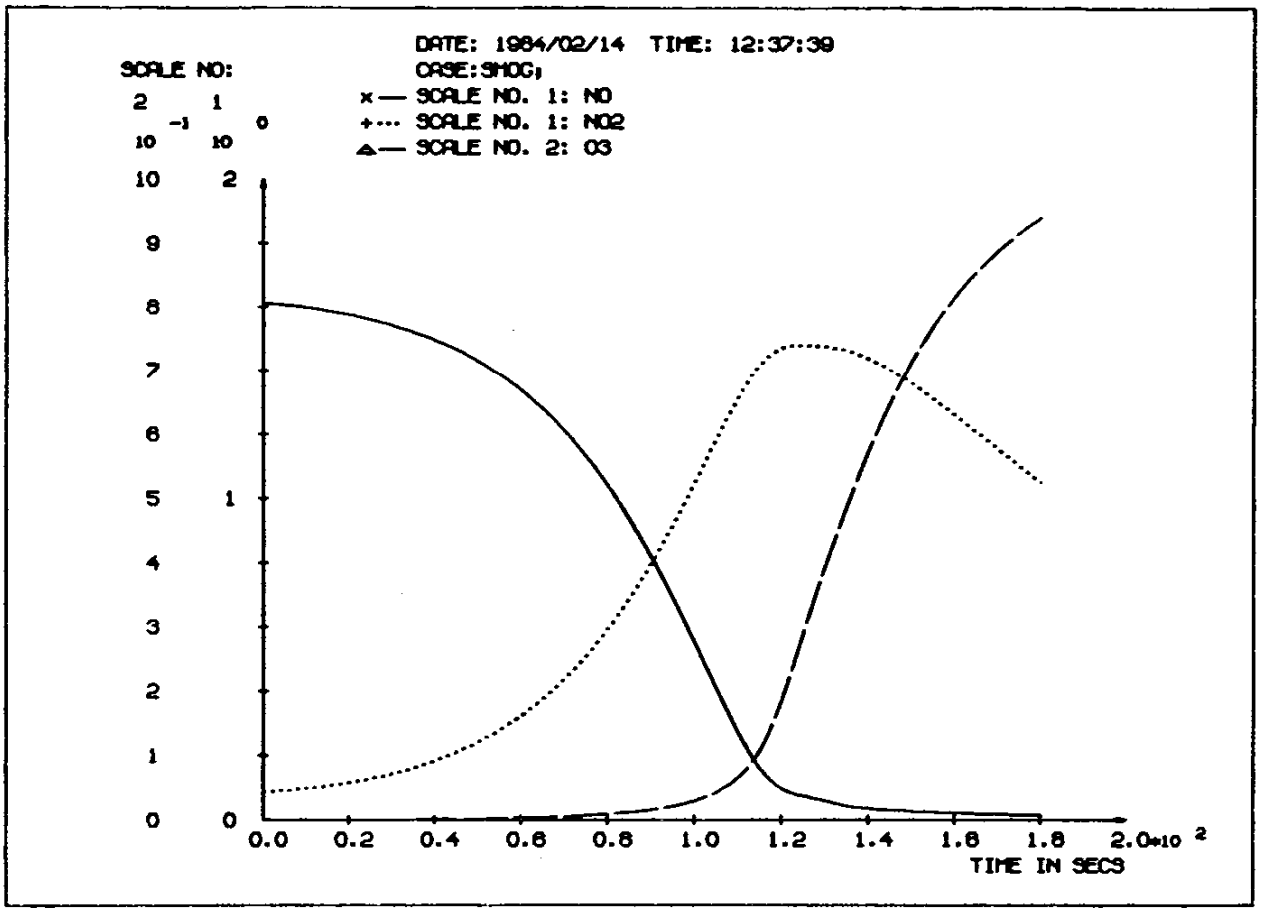
In our program, time is entered in sec and concentration in $mol \times dm^{-3}$. In this example we kept the original units (min and ppm) but uses 180 sec instead of 180 min for reaction time. The complete input file is listed below. The resulting curves for NO, NO_2 ,O_3 and C_3H_6 are identical with the curves in Fig. 1 in Farrow and Edelson's paper. The complete computing time was 10.7 sec.

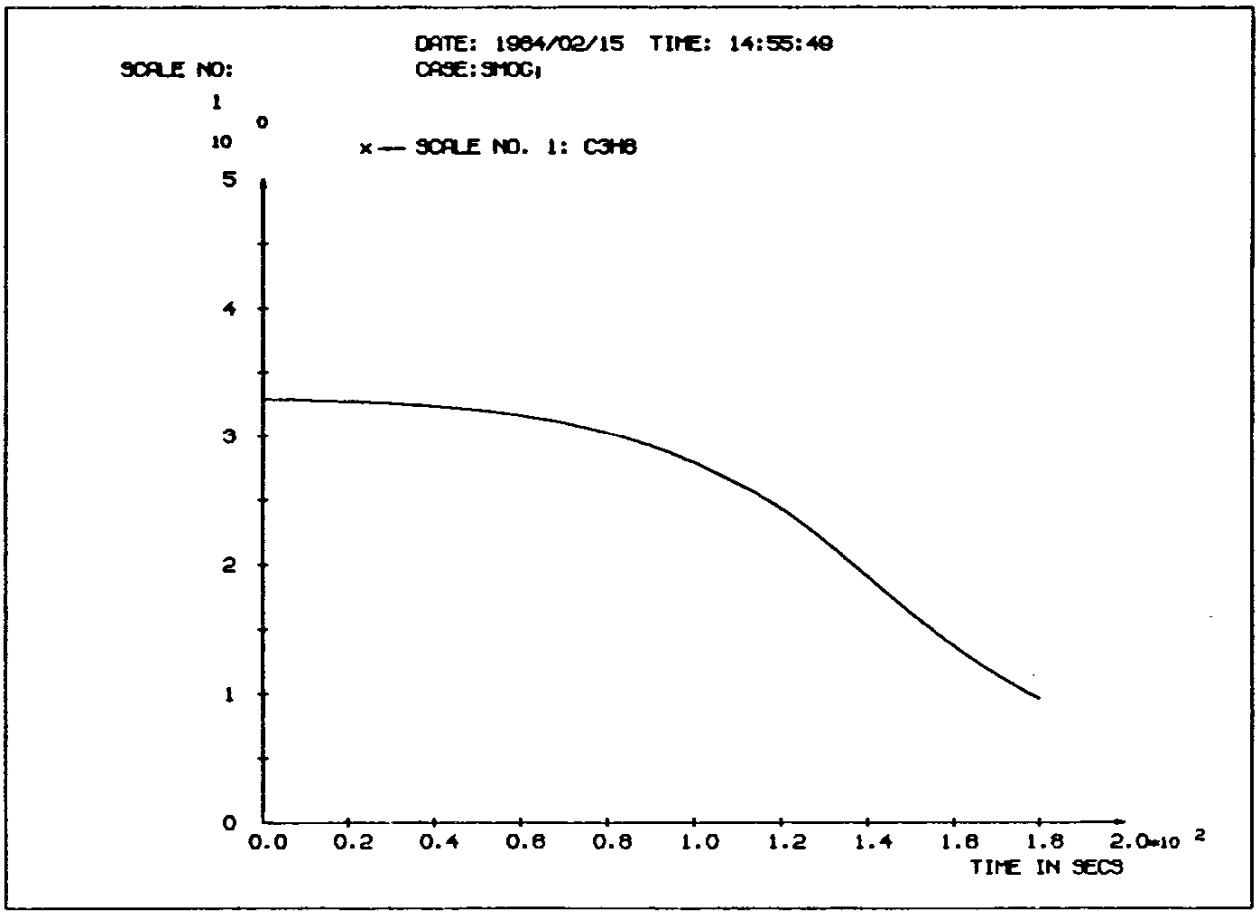
```
NEWSYS
RE1:NO2=NO+O;.37
RE2:O+O2=O3;22
RE3:O3+NO=NO2+O2;22.5
RE4:O3+NO2=NO3+O2;.049
RE5:NO3+NO2=N2O5;858
RE6:N2O5=NO3+NO2;2.76
```

```
RE7:N2O5+H2O=2*HNO3;7.5E-4
RE8:NO3+NO2=NO+O2+NO2;.49
RE9:O+NO2=O2+NO;5.15E3
RE10:NO3+NO=2*NO2;1.4E5
RE11:NO+NO2=N2O3;E-3
RE12:N2O3+H2O=2*HNO2;E3
RE13:HNO2=NO+OH;E-2
RE14:CO+OH=COOH;1.96E2
RE15:COOH+O2=CO2+HO2;1
RE16:HO2+NO=NO2+OH;30
RE17:HO2+NO2=HNO2+O2;E-3
RE18:OH+NO2=HNO3;1.47E3
RE19:OH+NO=HNO2;E4
RE20:HO2+HO2=H2O2+O2;70
RE21:C3H6+O=C2H5+CHO;2.925E3
RE22:C3H6+O=CH3+CH3CO;5E2
RE23:C3H6+O=C3H6O;2.5E2
RE24:C3H6O+O2=C2H4O2+HCHO;1
RE25:C3H6+O3=C2H4O2+HCHO;7E-3
RE26:C3H6+O3=CH3CHO+CH2O2;7E-3
RE27:C2H4O2=CH3O+CHO;.3
RE28:C2H4O2+NO=CH3CHO+NO2;3
RE29:C2H4O2+NO2=CH3CHO+NO3;.1
RE30:CH2O2=OH+CHO;.3
RE31:CH2O2+NO=HCHO+NO2;3
RE32:CH2O2+NO2=HCHO+NO3;.1
RE33:CH3+O2=CH3O2;15
RE34:CH3O2+NO=CH3O+NO2;6
RE35:CH3O+O2=HCHO+HO2;.26
RE36:CH3O+NO2=CH3NO3;16.2
RE37:C2H5+O2=C2H5O2;15
RE38:C2H5O2+NO=NO2+C2H5O;6
RE39:C2H5O+O2=CH3CHO+HO2;.26
RE40:C2H5O+NO2=C2H5NO3;16.2
RE41:C3H6+OH=C3H6OH;1.5E4
RE42:C3H6OH+O2=C3H6OHO2;1
RE43:C3H6OHO2=C2H5CHO+HO2;E-3
RE44:C3H6OHO2+NO=C3H6OHO+NO2;6
RE45:C3H6OHO+O2=CH3CHO+OHCH2O2;.015
RE46:OHCH2O2+NO=OHCH2O+NO2;4
```

```
RE47:OHCH2O2=HCHO+HO2;E-3
RE48:C3H6+OH=OHC3H6;E4
RE49:OHC3H6+O2=OHC3H6O2;1
RE50:OHC3H6O2=C2H6CO+HO2;E-3
RE51:OHC3H6O2+NO=OHC3H6O+NO2;6
RE52:OHC3H6O+O2=HCHO+CH3COHHO2;.015
RE53:CH3COHHO2=CH3CHO+HO2;E-3
RE54:CH3COHHO2+NO=CH3COHHO+NO2;6
RE55:CH3COHHO+O2=CH3COOH+HO2;5E-3
RE56:CH3COHHO=CH3CHO+OH;2E3
RE57:C3H6+OH=C3H5+H2O;5E3
RE58:CH3CO+O2=CH3COO2;15
RE59:CH3COO2+NO=CH3COO+NO2;8
RE60:CH3COO+NO=CH3CO+NO2;.1
RE61:CH3COO2+NO2=CH3COO2NO2;2
RE62:CH3COO=CH3+CO2;E-4
RE63:CH3COO2NO2+NO=CH3COO+2*NO2;.16
RE64:CH3COO2NO2=CH3COO+NO3;E-2
RE65:CH3O2+NO2=CH3O2NO2;E-3
RE66:CH3O2NO2+NO=CH3O+2*NO2;.1
RE67:CH3O2NO2 =CH3O+NO3;5E-2
RE68:C2H5O2+NO2=C2H5O2NO2;E-3
RE69:C2H5O2NO2+NO=C2H5O+2*NO2;.1
RE70:C2H5O2NO2=C2H5O+NO3;5E-2
RE71:HCHO=H+CHO;3.3E-5
RE72:HCHO+O=OH+CHO;2.2E2
RE73:HCHO+O3=OH+CHO+O2;2.45E-5
RE74:HCHO+OH=CHO+H2O;2.5E4
RE75:CH3CHO=CH3+CHO;3.6E-3
RE76:CH3CHO+O=CH3CO+OH;2.94E2
RE77:CH3CHO+O3=CH3CO+OH+O2;5E-4
RE78:CH3CHO+OH=CH3CO+H2O;2.5E4
RE79:CHO+O2=CO+HO2;2.6
RE80:CHO+OH=CO+H2O;3.6E4
RE81:CH3O+CH3O=HCHO+CH3OH;3.6E5
RE82:C2H5O+C2H5O=CH3CHO+C2H5OH;3.6E5
RE83:CH3COO+CH3O=HCHO+CH3COOH;3.6E5
RE84:CH3COO+C2H5O=CH3CHO+CH3COOH;3.6E5
RE85:OHCH2O=HCHO+OH;3E3
CON(NO)=1.612
```

```
CON(NO2)=.088
CON(C3H6)=3.29
CON(O2)=2E5
CON(H2O)=E3
PRINTS=20
TEND=200
FSTSTP=E-2
EPS=E-3
PE1:NO;
ENDPE
PE2:NO2;
ENDPE
PE3:O3;
ENDPE
PE4:C3H6;
ENDPE
CASE:SMOG;
DEVCLASS=PLOTTER
DEVNAME=BENSON1133
RESCLASS=1
MF=21
ENDDATA
```


DATE: 1984/02/14 TIME: 12:37:39
CASE:SMOG1
SCALE NO:
x — SCALE NO. 1: NO
+··· SCALE NO. 1: NO2
△— SCALE NO. 2: O3
2 1
10 -1 10 0
10 2
9
8
7
6
5 1
4
3
2
1
0 0
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0*10 2
TIME IN SECS


DATE: 1984/02/15 TIME: 14:55:48
CASE:SMOG1
SCALE NO:
1
10 0
x — SCALE NO. 1: C3H8
5
4
3
2
1
0
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0*10 2
TIME IN SECS

8.3 H_2 - O_2 combustion.

In the reaction scheme of Hoppensteadt, Alfeld and Aiken[8] an explosion is described in simplified form. We changed the pseudo-first-order termination reactions to first order-reactions. The complete computations with input data are listed below. The curves for H_2, O_2 and H_2O are identical to those of Fig. 4.1 in the paper by Deuflhard, Bader and Nowak[10]. This example demonstrates a case where a species approaches zero concentration very rapidly, and this has a tendency to create instability and negative values of concentration with our integration method. The reason for this instability is that the integration step at the critical point is too large requiring us to solve the problem by restricting the maximum integration step. In this case we use HMAX = E-4 sec. It is also possible to solve the problem by using higher relative accuracy in the computation but this is a more time-consuming method.

```
RE1:H2+O2=H+HO2;60
RE2:H2+OH=H+H2O;2.3E11
RE3:O2+H=OH+O;4.02E9
RE4:H2+O=H+OH;2.82E12
RE5:OH=P1;920
RE6:H=P2;80
RE7:O=P3;920
CON(H2)=E-7
CON(O2)=5E-8
PRINTS=10
TEND=E-1
EPS=E-2
FSTSTP=1E-6
CASE:KGAS;
PE1:H2;
ENDPE
PE2:O2;
ENDPE
PE3:H2O;
ENDPE
DEVCLASS=PLOTTER
DEVNAME=BENSON1133
RESCLASS=1
HMAX=1E-4
ENDDATA
```

TIME	H2	O2	H	HO2
0.	1.000E-07	5.000E-08	0. 0.	
5.00000E-03	1.000E-07	5.000E-08	3.526E-15	1.500E-15
1.00000E-02	1.000E-07	5.000E-08	1.970E-14	3.000E-15
1.50000E-02	1.000E-07	5.000E-08	9.390E-14	4.500E-15
2.00000E-02	1.000E-07	5.000E-08	4.343E-13	6.000E-15
2.50000E-02	1.000E-07	5.000E-08	1.967E-12	7.500E-15
3.00000E-02	9.998E-08	4.999E-08	8.823E-12	9.000E-15
3.50000E-02	9.992E-08	4.997E-08	3.953E-11	1.050E-14
4.00000E-02	9.966E-08	4.988E-08	1.768E-10	1.199E-14
4.50000E-02	9.848E-08	4.947E-08	7.838E-10	1.347E-14
5.00000E-02	9.347E-18	4.774E-08	3.348E-09	1.489E-14
5.50000E-02	7.537E-08	4.144E-08	1.234E-08	1.606E-14
6.00000E-02	3.556E-08	2.729E-08	2.969E-08	1.667E-14
6.50000E-02	2.767E-09	1.363E-08	3.501E-08	1.678E-14
7.00000E-02	1.162E-14	7.788E-09	2.079E-08	1.678E-14
7.50000E-02	1.847E-18	5.639E-09	1.221E-08	1.678E-14
8.00000E-02	3.734E-20	4.651E-09	7.390E-09	1.678E-14
8.50000E-02	5.159E-21	4.136E-09	4.537E-09	1.678E-14
9.00000E-02	1.731E-21	3.847E-09	2.808E-09	1.678E-14
9.50000E-02	9.197E-22	3.678E-09	1.745E-09	1.678E-14
1.00000E-01	6.305E-22	3.577E-09	1.088E-09	1.678E-14

TIME	OH	H2O	O	W
0.	0.	0.	0.	1.000E+00
5.00000E-03	5.819E-17	2.580E-15	2.502E-18	1.000E+00
1.00000E-02	3.261E-16	2.122E-14	1.398E-17	1.000E+00
1.50000E-02	1.555E-15	1.135E-13	6.664E-17	1.000E+00
2.00000E-02	7.192E-15	5.437E-13	3.082E-16	1.000E+00
2.50000E-02	3.259E-14	2.488E-12	1.396E-15	1.000E+00
3.00000E-02	1.461E-13	1.119E-11	6.262E-15	1.000E+00
3.50000E-02	6.549E-13	5.018E-11	2.806E-14	1.000E+00
4.00000E-02	2.931E-12	2.245E-10	1.256E-13	1.000E+00
4.50000E-02	1.304E-11	9.966E-10	5.589E-13	1.000E+00
5.00000E-02	5.646E-11	4.279E-09	2.426E-12	1.000E+00
5.50000E-02	2.215E-10	1.612E-08	9.617E-12	1.000E+00
6.00000E-02	6.959E-10	4.189E-08	3.211E-11	1.000E+00
6.50000E-02	2.041E-09	6.155E-08	2.045E-10	1.000E+00
7.00000E-02	9.347E-10	6.252E-08	8.468E-10	1.000E+00

7.50000E-02	3.652E-10	6.252E-08	3.643E-10	1.000E+00
8.00000E-02	1.757E-10	6.252E-08	1.757E-10	1.000E+00
8.50000E-02	9.404E-11	6.252E-08	9.404E-11	1.000E+00
9.00000E-02	5.351E-11	6.252E-08	5.351E-11	1.000E+00
9.50000E-02	3.158E-11	6.252E-08	3.158E-11	1.000E+00
1.00000E-01	1.906E-11	6.252E-08	1.906E-11	1.000E+00

TIME	P1	P2	P3
0.	0.	0.	0.
5.00000E-03	1.032E-16	5.470E-16	4.461E-18
1.00000E-02	8.487E-16	4.466E-15	3.645E-17
1.50000E-02	4.541E-15	2.386E-14	1.947E-16
2.00000E-02	2.175E-14	1.142E-13	9.321E-16
2.50000E-02	9.953E-14	5.226E-13	4.265E-15
3.00000E-02	4.476E-13	2.350E-12	1.918E-14
3.50000E-02	2.008E-12	1.054E-11	8.604E-14
4.00000E-02	8.996E-12	4.721E-11	3.855E-13
4.50000E-02	4.017E-11	2.105E-10	1.722E-12
5.00000E-02	1.770E-10	9.209E-10	7.595E-12
5.50000E-02	7.400E-10	3.736E-09	3.191E-11
6.00000E-02	2.683E-09	1.205E-08	1.185E-10
6.50000E-02	8.454E-09	2.586E-08	4.809E-10
7.00000E-02	1.624E-08	3.700E-08	3.883E-09
7.50000E-02	1.896E-08	4.343E-08	6.515E-09
8.00000E-02	2.014E-08	4.726E-08	7.691E-09
8.50000E-02	2.073E-08	4.960E-08	8.288E-09
9.00000E-02	2.106E-08	5.104E-08	8.618E-09
9.50000E-02	2.125E-08	5.193E-08	8.809E-09
1.00000E-01	2.137E-08	5.249E-08	8.923E-09

NO. OF INTEGRATIONSTEPS = 1001

MAXIMUMSTEP = 1.000E-04

MINIMUMSTEP = 1.000E-08

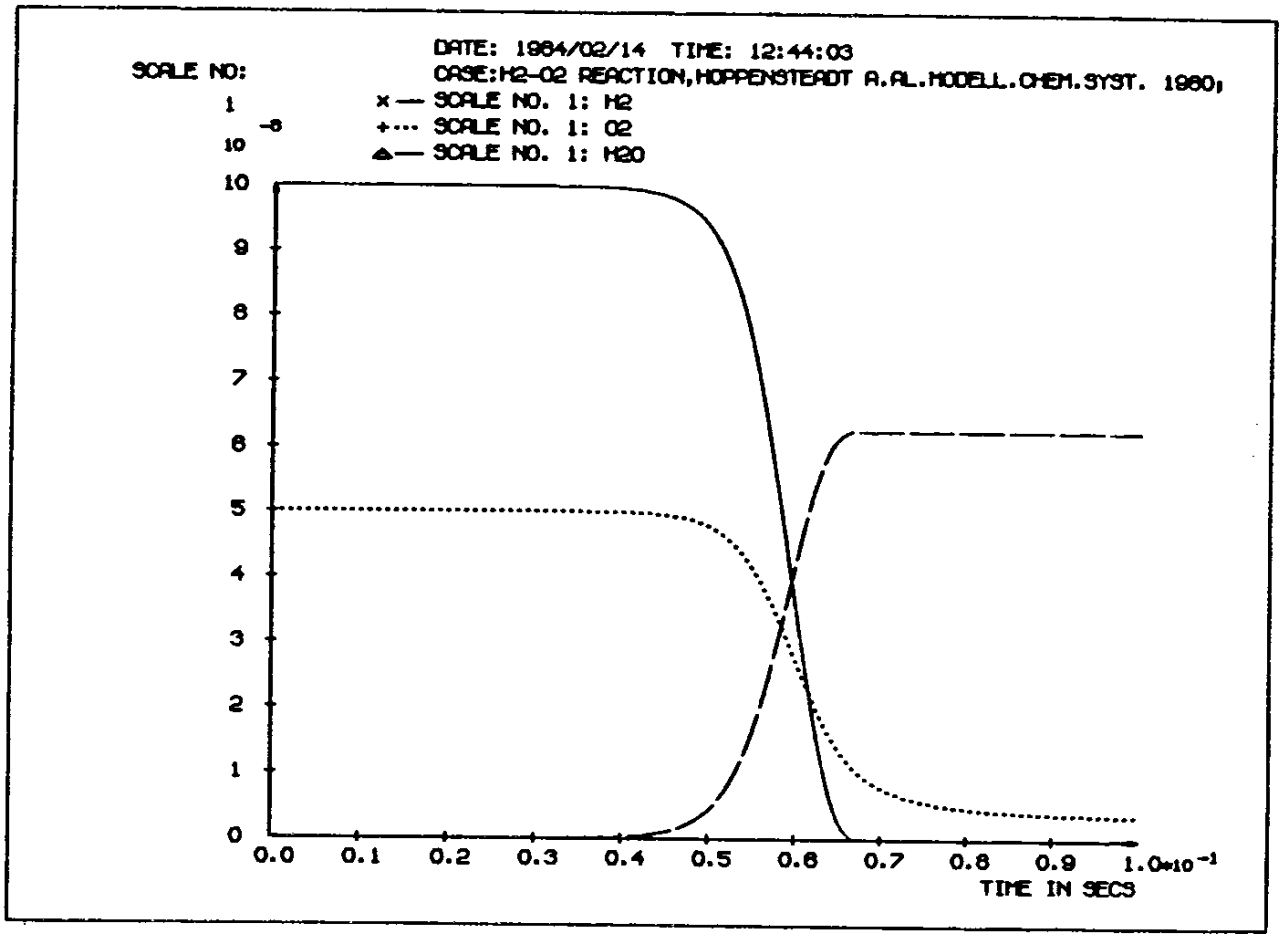
LAST STEP USED = 1.000E-04

THE LAST ORDER USED = 5

NO. OF FUNCTION EVALUATIONS = 1002

NO. OF DERIVATIVE EVALUATIONS = 65

COMP. TIME= 7.23 SEC.

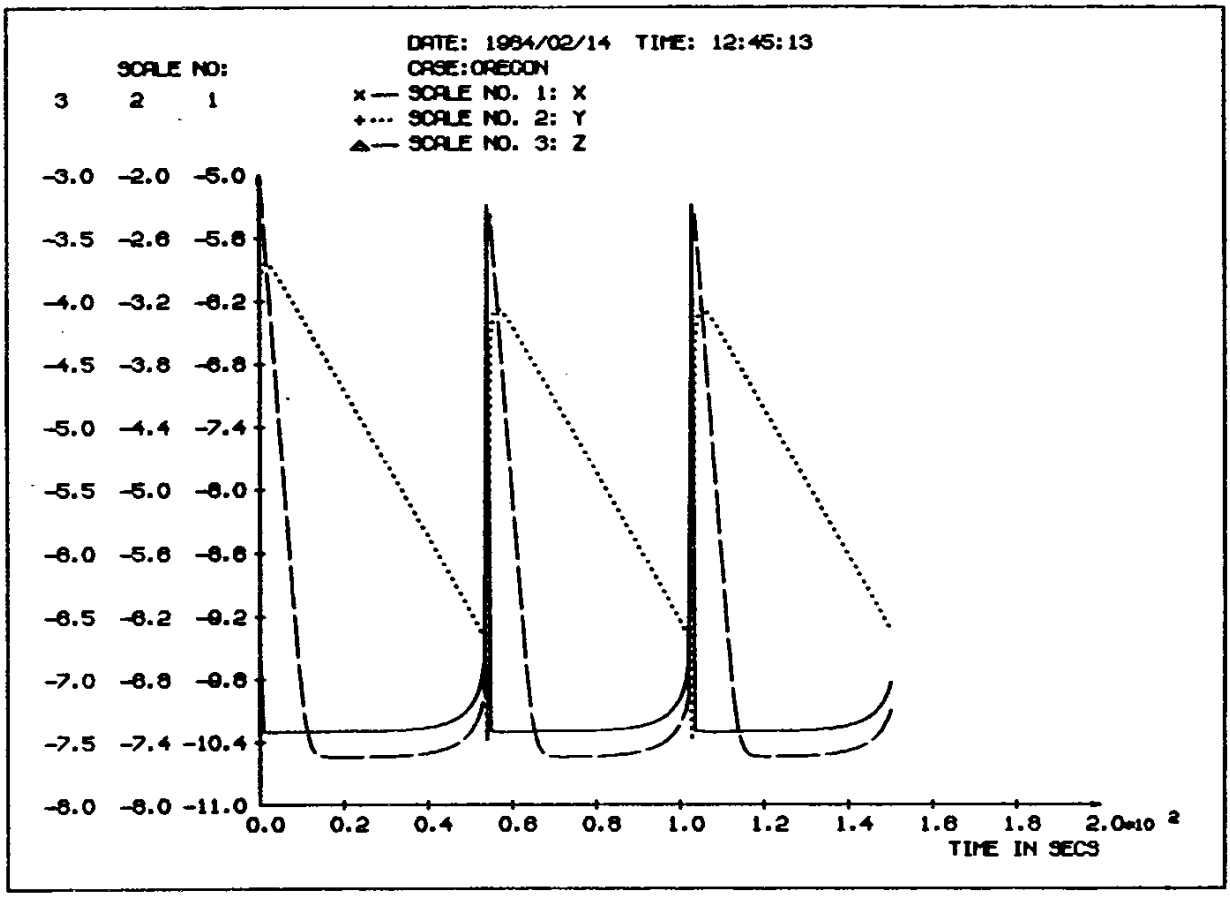

DATE: 1984/02/14 TIME: 12:44:03
CASE:H2-02 REACTION,HOPPENSTEADT A.AL.MODELL.CHEM.SYST. 1980;
SCALE NO:
x — SCALE NO. 1: H2
+··· SCALE NO. 1: O2
▲— SCALE NO. 1: H2O
1
10 -6
10
9
8
7
6
5
4
3
2
1
0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0*10 -1
TIME IN SECS

8.4 Oregonator.

Field and coworkers have proposed a ten-step mechanism[9] to simulate the oscillations in the Belusov - Zhabotinskii reaction[11]. We found it more convenient, however, to extract the input data from the work of Blandamer and Morris[12]. The input file is listed below. Our calculated curves for X, Y and Z are identical to the curves shown in Fig. 6 in the Blandamer and Morris work. The computation time was 13.9 sec. In this case it is also necessary to place a limit on the maximum integration step size, and a high relative integration accuracy must be applied.

```
NEWSYS
RE1:A+Y=X;1.34
RE2:X+Y=P;1.6E9
RE3:B+X=2*X+Z;8E3
RE4:X+X=Q;4E7
RE5:Z=Y;1
RE6:DUMMY1=A;.12
RE7:A=DUMMY1;200
RE8:DUMMY2=B;.12
RE9:B=DUMMY2;200
CON(A)=6E-2
CON(B)=6E-2
CON(Z)=E-3
CON(X)=3E-10
CON(Y)=E-3
CON(DUMMY1)=100
CON(DUMMY2)=100
PRINTS=200
TEND=150
FSTSTP=E-2
EPS=E-7
PE1:X;
ENDPE
PE2:Y;
ENDPE
PE3:Z;
ENDPE
CASE:OREGON;
DEVCLASS=PLOTTER
```

```
DEVNAME=BENSON1133
RESCLASS=2
HMAX=1
PLONPA=1
MF=21
ENDDATA
```


DATE: 1984/02/14 TIME: 12:45:13
CASE:OREGON
SCALE NO:
3 2 1
x— SCALE NO. 1: X
+··· SCALE NO. 2: Y
▲— SCALE NO. 3: Z
-3.0 -2.0 -5.0
-3.5 -2.6 -5.6
-4.0 -3.2 -6.2
-4.5 -3.8 -6.8
-5.0 -4.4 -7.4
-5.5 -5.0 -8.0
-6.0 -5.6 -8.6
-6.5 -6.2 -9.2
-7.0 -6.8 -9.8
-7.5 -7.4 -10.4
-8.0 -8.0 -11.0
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0x10 2
TIME IN SECS

9. SPECIAL PROBLEMS

In our work on the modeling of radiation chemistry in connection with high-level waste disposal[13] we have simulated thousands of years of reaction time. In such cases computation time becomes a critical parameter and therefore we have searched for methods to simplify the problems. We found that in many cases we could cut the computation time appreciably by lowering the forward and backward rate constants for acid-base equilibria (maintaining the equilibria constants). If this method is used it is necessary to check that all the equilibria are kept. When simulating discontinuous reactions like applying a short pulse of radiation, a short period of computation instability is often experienced. The instability expresses itself as negative concentrations of the fast-decaying transient species. Negative concentrations can be suppressed by increasing the relative computational accuracy or by decreasing the maximum integration step. Normally however, a few negative concentrations of relatively small numerical value can be ignored since they have no influence on the net result.

It is possible to mark a single reaction in order to check the contribution of the reaction to the whole mechanism. In order to be sure that the balance equation can be satisfied it is necessary to make a double marking as follows:

$$OH + H_2 \rightarrow H + H_2O;\ k_1$$

is changed to

$$OH + H_2 \rightarrow OHH_2\ ;\ k_1$$
$$OHH_2 + A \rightarrow H + H_2O + B;\ k_2$$

where the rate of reaction in the second reaction is much faster than in the first ($k_2 \times [A] \times [OHH_2] \gg k_1 \times [OH] \times [H_2]$). The initial concentration of A is chosen to be so high that the relative change during the cause of the reactions is negligible. The initial concentration of B is zero.

REFERENCES

1) C.W. Gear (1971). Algorithm 407, DIFSUB for solution of ordinary differential equations. Commun. ACM 14, 185.

2) A.C. Hindmarsh and G.D. Byrne (1975). EPISODE: An experimental package for the integration of systems of ordinary differential equations. Lawrence Livermore Laboratory Report UCID-30112.

3) G.M. Ridler, P.F. Ridler and J.G. Sheppard (1977). A systematic method of checking of systems of chemical equations for mass balance. J. Phys. Chem. 81, 2435.

4) J.S. Rosenbaum (1976). Conservation properties of numerical integration methods for systems of ordinary differential equations. J. Comp. Phys. 20, 259.

5) J.S. Rosenbaum (1977). Conservation properties for numerical integration methods for systems of differential equations. 2. J. Phys. Chem. 81, 2362.

6) K. Sehested, E. Bjergbakke, N.W. Holm and H. Fricke (1973). The reaction mechanism of the ferrous-sulfate dosimeter at high dose rates. Dosimetry in Agriculture, Industry, Biology and Medicine. IAEA-SM-160/30, 397, IAEA, Vienna.

7) L.A. Farrow and D. Edelson (1974). The steady-state approximation: Fact and Fiction. Int. J. Chem. Kinet. VI, 787.

8) F.C. Hoppensteadt, P. Alfeld and R. Aiken (1981). Numerical treatment of rapid chemical kinetics by pertubation and projection methods. Chap. in Modelling of Chemical Reaction Systems, Eds. K.H. Ebert, P. Deuflhard and W. Jäger. Springer-Verlag Berlin, Heidelberg, New York.

9) R.J. Field and R.M. Noyes (1974). Oscillations in chemical systems. IV. Limit cycle behaviour in a model of a real chemical reactions. J. Phys. Chem. 60, 1877.

10) P. Deuflhard, G. Bader and U. Nowak (1981). Larkin - A software package for the numerical simulation of LARge systems arising in chemical reaction KINetics. Chap. in Modelling of Chemical Reaction systems, Eds. K.H. Ebert, P. Deuflhard and W. Jäger. Springer-Verlag Berlin, Heidelberg, New York.

11) G. Nicolis and J. Portnow (1973). Chem. Rev. 73, 365.

12) M.J. Blandamer and S.H. Morris (1975). Investigation into the effect of temperature and added t-butyl alcohol on the dynamic properties of the Belusov reaction. J.C.S. Faraday I 71, 2319.

13) H. Christensen and E. Bjergbakke (1982). Radiolysis of ground water from spent fuel. Swedish Nuclear Fuel Safety Project. Studsvik/NW-82/364.

APPENDIX I

Description of the data file to program CHEMSIMUL

The input file to the program CHEMSIMUL starts with the command NEWSYS and ends with the command ENDDATA. Within the frame of these commands all other data can be entered in arbitrary sequence.

The command NEWSYS announces that data for a new reaction system will follow. The command is necessary whenever a new set of chemical reactions is employed in a series of simulations. It is not necessary for the first data-set.

The command ENDDATA tells the program that the translation of the chemical reaction system can start. If all input data are correct the program continues with computation; otherwise it stops with self-explanatory error messages. A data file can contain several sets of data, separated with the command ENDDATA. The first data-set must contain all the information relevant for a simulation. The following data-sets needs only contain the information which is changed relative to the previous data-set.

The input data instructions can be divided in five types:

1. The reaction equation system.

2. Numerical values of initial concentrations, rate constants and G-values.

3. Integration parameters.

4. Plot instructions.

5. Instructions for special actions.

In the following these types will be described with examples:

1. Reaction equations.

   Each reaction equation is prefixed with an identification: RE nnn: where RE means Reaction Equation and nnn is the equation number.

ex: RE27:H[+] + OH[-] = H2O; 1.0E11

RE123:H[+] + S[-] = HS; 1.0E10

The reaction equation terminates with a semicolon (;) followed by the value of the rate constants for that reaction. (E11 means 10 to power 11)

2. Numerical values of concentrations, the G-values and rate constants

Numerical values of start concentrations are written

ex: CON(OH[-]) = E-7

CON(H[+]) = E-7

Default value is zero (0), i.e. those chemical species appearing in the reaction equations for which no start concentrations are given are automatically put to zero. The concentration unit is mol×dm^{-3}.

If the reaction system is irradiated e.g. by pulses of electrons or by gamma radiation, the G-values must be supplied. This is done by the data

ex: G(OH) = 2.7

G(H[+]) = 2.8

Default value is zero. The unit for G is molecules/100eV.

If we wish to change the rate constants that are written in a reaction equation this is done by

ex: K123 = E11

which changes the value 1.0E10 to E11.

3. Integration parameters

Every parameter must be followed by a = and its value (for example, DOSE = 2)

Parameters marked $ are obligatory.

DOSE The total input of radiation (pulse) in krd during the radiation time.

EPS$ Relative accuracy in the integration routine EPISODE.

FSTSTP$ The first integration step in seconds. The program has automatic step control.

HMAX$ The maximum length in seconds of an integration step.

NNR The number of repeated radiations, normally 1. If NNR>1 it means that several radiations are performed. (This is a feature designed specifically for accelerator experiments.)

PRINTS$ Number of lines printed in the result table. Each line contains the value of time and the values of concentrations at that time. The the table is equidistant in time by default.

RADPRS Number of lines printed in the result table during the radiation. See description for PRINTS and notice that RADPRS <= PRINTS.

RADTIME The total radiation time (pulse-length) in seconds.

TEND$ The end time in seconds for the integration.
Start value of time is zero.

4. Plot instruction

If plotting is wanted for a chemical species we write

```
ex:     PE1: H2;
        t1, H2(t1)
        t2, H2 (t2)
        .
        .
        .
        tn, H2 (tn)
        ENDPE
        PE2: OH[-];
        ENDPE
```

The program will plot the concentration of H_2 and OH^- on the plotter device chosen. Notice that the plot-statement must be termi-

nated by a semicolon (;). After each plot-statement the command ENDPE (=end of plot-expression) must also be written. Between a plot-expression and the following ENDPE one can write a table of experimental data, where the first item is the time and the second the corresponding concentration computed by the plot-expression (which can be an arithmetical expression with +, -, *, but not division).

5. Special instructions

CASE: The text following this instruction is used as an identification of the simulated problem.

DATACON This instruction will initiate a control of the data file and suppress computation. If the data file contains errors the program sends a self-explanatory message. If the data file is correct it sends the message INPUT FILE CORRECT.

DEC= Define the number of decimals in the resulting concentrations. Default is 3.

DEVCLASS= Define the deviceclass for a plotter. For further information see Risø-R-493 (1984), S. Rahbek and E. Hansen, RIGS Risø Interactive Graphics System.

DEVNAME= Define the device name of a plotter within the deviceclass.

DIFFEQ This instruction initiates an output of the differential equation system. Default is no output.

LINELE= Define the number of characters per line of the result table. Default is 132 (= the line width of a lineprinter).

NOTABLE This instruction suppresses output of the result table. It can be used if plots are wanted, but no output of numerical results.

PLONPA= Define the maximum number of plots on a plotter sheet. Possible values are 1, 2, and 3. Default value is 3.

RESCLASS= Define whether the plotscales are given in linear or logarithmic form.

RESCLASS = 1. Time and concentration both linear.

RESCLASS = 2. Time linear, concentration logarithmic.

RESCLASS = 3. Time logarithmic, concentration linear.

RESCLASS = 4. Time and concentration both logarithmic.

SUPINP This instruction suppresses output of input-variables.

APPENDIX II.
The program CHEMSIMUL.

```
%*****************************************************************
%                      C H E M S I M U L                         *
%                                                                *
%                      A PROGRAM PACKAGE                         *
%                                                                *
%                              FOR                               *
%                                                                *
%             SIMULATION OF CHEMICAL REACTION SYSTEMS.           *
%                                                                *
%                              BY                                *
%                                                                *
%               O. LANG  RASMUSSEN AND E. BJERGBAKKE.            *
%                                                                *
%                     RISOE NATIONAL LABORATORY.                 *
%                                                                *
%*****************************************************************

%*****************************************************************
%                                                                *
% CHEMSIMUL, WHICH IS PROGRAMMED IN BURROUGHS EXTENDED ALGOL     *
% MAKE USE OF TWO OTHER PROGRAM PACKAGES:                        *
%                                                                *
% 1: 'EPISODE' FOR INTEGRATING THE SYSTEM OF ORDINARY DIFFE-     *
%    RENTIAL EQUATIONS SYSTEM, PROGRAMMED IN FORTRAN-66 [1].     *
%                                                                *
% 2: 'RIGS' FOR PLOTS OF THE INTEGRATION RESULTS, PROGRAMMED     *
%    IN BURROUGHS EXTENDED ALGOL [2].                            *
%                                                                *
% REFERENCES:                                                    *
% [1]. HINDMARSH, A.C. AND BYRNE, G.D.,                          *
%      'EPISODE': AN EXPERIMENTAL PACKAGE FOR THE INTEGRATION    *
%      OF SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS.            *
%      (LAWRENCE LIVERMORE LABORATORY REPORT UCID-30112 (1975))  *
%                                                                *
% [2]. STEEN RAHBEK AND ERIK HANSEN,                             *
%      'RIGS': RISOE INTERACTIVE GRAPHICS SYSTEM.                *
%      (RISOE NATIONAL LABORATORY REPORT RISOE-R-493 (1983))     *
%                                                                *
%*****************************************************************
$SET INSTALLATION 128 SET LINEINFO RESET LIST
BEGIN
%*************************************************
% DECLARATION  OF:                               *
%                 FILES                          *
%                 VARIABLES                      *
%                 PROCEDURES AND                 *
%                 LIBRARY.                       *
%*************************************************

FILE          % DECLARATION
  INP(KIND=DISK,FILETYPE=7),
  INTMSG(KIND=DISK,TITLE="INTEGRATIONMSG.",
       FILETYPE=7,NEWFILE=FALSE),
  OUT(KIND=PRINTER),
  ERRFIL(KIND=REMOTE,UNITS=CHARACTERS,MAXRECSIZE=80);

INTEGER       % DECLARATION
  YEAR  ,     % THE YEAR
  MONTH ,     %     MONTH
  DAY   ,     %     DAY
```

```
HOUR   ,      %     HOUR
MINUTE,       %     MINUTE
SECOND,       % AND SECOND FOR DATING THE COMPUTATION

MAXRE  ,      % USED FOR DIMENSIONING THE
              % STRING ARRAYS:  RE, RC
              % AND THE
              % REAL   ARRAY:  RK.
              % DEFAULT VALUE IS 100 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXGV  ,      % USED FOR DIMENSIONING THE
              % STRING ARRAY: GV.
              % DEFAULT VALUE IS 100 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXNR  ,      % USED FOR DIMENSIONING THE
              % STRING ARRAYS: RNA, CON
              % AND THE
              % REAL   ARRAYS:  RCON, RG, RGC.
              % DEFAULT VALUE IS 100 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXPE  ,      % USED FOR DIMENSIONING THE
              % STRING ARRAYS: PE, XPDATA
              % AND THE
              % REAL   ARRAYS: XPD, ADDCON.
              % DEFAULT VALUE IS 20 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXPT  ,      % USED FOR DIMENSIONING THE
              % INTEGER ARRAYS: PEN, PRN,
              % AND THE
              % REAL    ARRAY : PFAC.
              % DEFAULT VALUE IS 100 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXNT  ,      % USED FOR DIMENSIONING THE
              % INTEGER ARRAYS: ETERM, ERK.
              % DEFAULT VALUE IS 100 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

MAXSET,       % USED FOR DIMENSIONING THE
              % REAL ARRAY: XPD.
              % DEFAULT VALUE IS 20 BUT CAN BE INCREASED
              % BY THE PROGRAM IF NECESSARY.
              % (SEE BELOW).

LINELE,       % THE LENGTH OF AN OUTPUT LINE USED
              % BY THE OUTPUT PROCEDURES:
              %      'PRINTDIFFEQ' AND 'RESULTS'.
              % DEFAULT VALUE IS 132 (= THE MAXIMAL WIDTH
              % OF THE LINE-PRINTER) BUT CAN BE SET TO ANY
              % VALUE SMALLER THAN 132 BY THE USER.

DEC    ,      % THE NUMBER OF DECIMALS IN THE OUTPUT
              % VALUES FOR CONCENTRATIONS OF REACTANTS.
              % DEFAULT VALUE IS 3 BUT CAN BE SET TO ANY
```

```
                % REASONABLE VALUE BY THE USER. THE MAXIMUM
                % NUMBER OF DECIMALS IN SINGLE PRECISION FOR
                % B7800 IS 11.

MF      ,       % THE METHOD FLAG INDICATE THE INTEGRATION
                % METHOD USED IN THE INTEGRATION PACKAGE
                % 'EPISODE' CALLED VIA THE PROCEDURE 'DRIVE'.
                % THE DEFAULT VALUE IS 21 CORRESPONDING TO
                % THE METHOD BY GEAR FOR STIFF DIFFERENTIAL
                % EQUATIONS WITH ANALYTICAL COMPUTATION OF
                % THE JACOBIAN MATRIX. SEE REF [1].

PLONPA,         % THE NUMBER OF PLOTS ON A PLOTTER-SHEET.
                % THE DEFAULT VALUE IS 3 BUT CAN BE CHANGED
                % ACCORDING TO THE USERS DEMAND.

NRR     ,       % THE NUMBER OF REPEATED RADIATIONS.
                % DEFAULT VALUE IS 1 CORRESPONDING TO NORMAL
                % USE IN CASE OF RADIATION BUT CAN BE CHANGED
                % TO ANY VALUE GREATER OR EQUAL THAN 0 BY THE
                % USER.

RADPRS,         % NUMBER OF PRINTOUT LINES FOR CONCENTRATION
                % OF REACTANTS DURING THE RARIATION AS FUNC-
                % TION OF TIME. DEFAULT VALUE IS 0, AND IT
                % MUST BE SET BY THE USER.

PRINTS,         % NUMBER OF TOTAL PRINTOUTS (INCLUDING RADPRS)
                % FOR CONCENTRATION OF REACTANTS AS FUNCTION OF
                % TIME.
                % DEFAULT VALUE IS 0 AND IT MUST  BE SET
                % BY THE USER.

RESCLASS,       % A FLAG INDICATING WHETHER THE TIME AND THE
                % RESULTING CONCENTRATION VALUES FOR THE REAC-
                % TANTS SHALL BE GIVEN IN LINEAR OR IN LOGA-
                % RITHMIC FORM. IF THE RESULTS OF THE INTEGRA-
                % TION ARE PLOTTED, THE TIME IS ON THE X-SCALE
                % AND THE CONCENTRATION ON THE Y-SCALE.
                % RESCLASS CAN TAKE THE VALUES:
                % 1: X-SCALE IS LINEAR, Y-SCALE IS LINEAR.
                % 2: X-SCALE IS LINEAR, Y-SCALE IS LOGARITHMIC.
                % 3: X-SCALE IS LOGARITHMIC, Y-SCALE IS LINEAR.
                % 4: X-SCALE AND Y-SCALE ARE BOTH LOGARITHMIC.

NRE     ,       % THE NUMBER OF REACTION EQUATIONS STORED BY
                % THE INPUT PROCEDURE IN STRING ARRAY RE.

NRC     ,       % THE NUMBER OF RATE CONSTANTS STORED BY
                % THE INPUT PROCEDURE IN STRING ARRAY RC.

NGV     ,       % THE NUMBER OF G-VALUES STORED BY THE INPUT
                % PROCEDURE IN STRING ARRAY GV.

NCON    ,       % THE NUMBER OF CONCENTRATION VALUES STORED
                % BY THE INPUT PROCEDURE IN STRING ARRAY CON.

NPE     ,       % THE NUMBER OF PLOTTING EXPRESSIONS STORED
                % BY THE INPUT PROCEDURE IN STRING ARRAY PE.

MAXRL   ,       % THE MAXIMUM LENGTH IN CHARACTERS OF THE
                % REACTANT NAMES FOUND DURING THE TRANSLATION
                % PROCESS OF THE REACTION EQUATIONS. IT IS
                % SET BY THE INPUT PROCEDURE AND IS USED INTER-
```

```
                % NALLY FOR RESERVING PLACE FOR STORING THE NA-
                % MES OF THE REACTANTS.

  NR        ,   % THE ACTUAL NUMBER OF REACTANT NAMES FOUND
                % DURING THE TRANSLATION OF REACTION EQUATIONS.
                % NR IS LESSER OR EQUAL MAXNR.

  NT        ,   % THE ACTUAL NUMBER OF TERMS ON THE RIGHT
                % HAND SIDE OF THE DIFFERENTIAL EQUATIONS. IT
                % IS SET BY THE INPUT PROCEDURE AND MUST SATIS-
                % FY THE RELATION: NT LEQ MAXNT.

  PT        ;   % THE ACTUAL NUMBER OF TERMS IN THE PLOTTING
                % EXPRESSIONS. IT IS SET BY THE INPUT PROCEDURE
                % AND MUST SATISFY THE RELATION: PT LEQ MAXPT.

REAL            % DECLARATION
  DOSE      ,   % THE TOTAL DOSE OF RADIATION.
                % THE DEFAULT VALUE IS 0 AND IN CASE OF
                % RADIATION IT MUST BE SET BY THE USER.

  RADTIME   ,   % THE RADIATION TIME IN SECONDS FOR DOSE.
                % DEFAULT VALUE IS 0 AND IN CASE OF RADIATION
                % IT MUST BE SET BY THE USER.

  FSTSTP    ,   % THE FIRST INTEGRATION STEP IN SECONDS. IT
                % IS USED BY THE INTEGRATION PACKAGE 'EPISODE'
                % AND IS THUS A PARAMETER IN THE CALL AF 'DRIVE'.

  TEND      ,   % THE END TIME FOR THE INTEGRATION.
                % DEFAULT VALUE IS 0 AND IT MUST BE SET BY USER.

  EPS       ,   % THE RELATIVE ACCURACY IN THE INTEGRATION OF
                % THE DIFFERENTIAL EQUATIONS. IT IS USED BY THE
                % PACKAGE 'EPISODE' AND IS THUS A PARAMETER
                % IN THE CALL OF 'DRIVE'. SEE REF [1].
                % DEFAULT VALUE IS 0 AND IT MUST BE SET BY USER.

  HMIN      ,   % THE SMALLEST INTEGRATION STEP FOUND BY THE
                % INTEGRATION SUBROUTINE IN PACKAGE 'EPISODE'.

  HMAX      ,   % THE LARGEST INTEGRATION STEP DONE BY THE
                % INTEGRATION SUBROUTINE IN 'EPISODE'.

  HMAXO     ,   % THE LARGEST INTEGRATION STEP ALLOWED BY
                % THE INTEGRATION SUBROUTINE IN 'EPISODE'.
                % DEFAULT VALUE IS 0 AND IT MUST BE SET BY USER.

  MAXTIME   ,   % THE MAXIMUM TIME ALLOWED FOR THE COMPUTATION
                % IT IS SET BY THE PROGRAM SUCH THAT IT IS SMAL-
                % LER THAN THE TIME ALLOCATED FOR THE JOB CLASS
                % IN WHICH THE COMPUTATION IS DONE. THE PURPOSE
                % IN DOING THIS IS TO OBTAIN RESULTS EVEN IF THE
                % ALLOCATED COMPUTING TIME IS INSUFFICIENT FOR
                % DOING THE WHOLE COMPUTATION.

  D7        ;   % WORKING VARIABLE USED FOR DETERMINING THE
                % DATE OF COMPUTATION.

BOOLEAN         % DECLARATION
  INPUTERROR ,  % DEFAULT VALUE IS FALSE, BUT IS SET TO
                % TRUE WHEN ERRORS ARE FOUND DURING INPUT,
                % TRANSLATION OF REACTION EQUATIONS ETC.
                % WHEN IT IS SET TO TRUE THE PROGRAM WRITES
```

```
                % ERROR MESSAGES AND GO TO END (OR CONTI-
                % NUES WITH A NEW COMPUTATION).

INTMES,         % IF TRUE THEN THE INTEGRATION MESSAGES
                % FROM EPISODE IS PRINTED.
                % WHEN THE CHARACTER STRING 'INTMES' IS
                % WRITTEN IN THE INPUT FILE THEN INTMES
                % IS SET TO TRUE, AND WHEN THE STRING
                % 'NOINTMES' IS WRITTEN THE INTMES IS
                % SET TO FALSE.
                % DEFAULT VALUE IS FALSE.

NEWRE           ,% DEFAULT VALUE IS FALSE, BUT IS SET TO
                 % TRUE WHEN A REACTION EQUATION IS READ.

NEWRC           ,% DEFAULT VALUE IS FALSE, BUT IS SET TO
                 % TRUE WHEN A RATE CONSTANT IS READ.

NEWSYS          ,% THIS VARIABLE DETERMINES WHETHER INPUT
                 % OF REACTION SYSTEM WILL TAKE PLACE OR
                 % NOT. IF TRUE A NEW REACTION SYSTEM WITH
                 % ALL RELEVANT DATA SHALL BE READ, IF FALSE
                 % NO NEW REACTION SYSTEM SHALL BE READ, BUT
                 % CORRECTION OF RATE CONSTANTS, START
                 % CONCENTRATIONS ETC. CAN BE READ.
                 % NEWSYS IS AUTOMATICALLY SET TO FALSE WHEN
                 % THE INPUT DATA 'ENDDATA' WHICH ENDS THE
                 % DATA FOR A COMPUTATION IS READ. THAT FEA-
                 % TURE MAKES IT EASY FOR THE USER TO DO COM-
                 % PUTATIONS OF THE SAME REACTION SYSTEM BUT
                 % WITH DIFFERENT VALUES OF THE OTHER INPUT
                 % DATA.
                 % DEFAULT VALUE IS TRUE.

NEWGV           ,% DEFAULT VALUE IS FALSE, BUT IS SET TO
                 % TRUE WHEN A G-VALUE IS READ.

NEWCON          ,% DEFAULT VALUE IS FALSE, BUT IS SET TO
                 % TRUE WHEN A START CONCENTRATION IS
                 % READ.

NEWPE           ,% DEFAULT VALUE IS FALSE, BUT IS SET TO
                 % TRUE WHEN A PLOT EXPRESSION IS READ.

TABLE           ,% WHEN TRUE THE TABLE WITH CONCENTRATION
                 % OF REACTANTS AS FUNCTION OF TIME IS
                 % PRINTED, WHEN FALSE NO PRINTING TAKES
                 % PLACE.
                 % DEFAULT VALUE IS TRUE.
                 % WHEN THE CHARACTER STRING 'TABLE' IS WRIT-
                 % TEN IN THE INPUT-FILE THE TABLE IS SET TO
                 % TRUE. IF THE CHARACTER STRING 'NOTABLE'
                 % IN WRITTEN THEN THE TABLE IS SET TO FALSE.

DATACON         ,% WHEN DATACON IS TRUE THE PROGRAM MAKES A
                 % A CONTROL OF INPUT DATA ONLY, WHEN FALSE,
                 % BOTH CONTROL AND COMPUTATION IS DONE (IF
                 % NO ERRORS ARE DETECTED OF COURSE).
                 % WHEN THE CHARATER STRING 'DATACON' IS
                 % WRITTEN IN THE DATA INPUT FILE THEN
                 % DATACON IS SET TO TRUE, WHEN THE CHARAC-
                 % TER STRING 'NODATACON' IS WRITTEN THEN
                 % DATACON IS SET TO FALSE.
                 % DEFAULT VALUE IS FALSE.
```

```
   DIFFEQ      ,% WHEN DIFFEQ IS TRUE THEN THE DIFFERENTIAL
                % EQUATION SYSTEM IS PRINTED, WHEN FALSE,
                % NO PRINTOUT IS DONE.
                % WHEN THE CHARACTER STRING 'DIFFEQ' IS WRIT-
                % TEN IN THE DATA INPUT FILE THE DIFFEQ IS
                % SET TO TRUE, WHEN THE CHARACTER STRING
                % 'NODIFFEQ' IS WRITTEN THEN DIFFEQ IS SET TO
                % FALSE.
                % DEFAULT VALUE IS FALSE.

   OUTINP      ,% WHEN TRUE THEN CONTROL OUTPUT OF INPUT
                % DATA IS DONE, WHEN FALSE NO OUTPUT IS
                % DONE.
                % DEFAULT VALUE IS TRUE.

   TERMINAL    ,% WHEN TRUE THEN THE PROGRAM IS RUN IN TER-
                % MINAL MODE AND IF PLOT OF THE RESULTS IS
                % WANTED THE USER CAN COMMUNICATE WITH THE
                % 'RIGS' PACKAGE FROM THE TERMINAL.
                % IF FALSE THE PROGRAM IS RUN IN BATCH MODE
                % AND THE COMMUNICATION WITH 'RIGS' IS DONE
                % AUTOMATICALLY FROM THE PROGRAM ACCORDING
                % TO THE COMMANDS WRITTEN IN THE PROGRAM.
                % THE VALUE OF TERMINAL IS SET AUTOMATICAL-
                % LY BY THE PROGRAM. (SEE BELOW IN THE PRO:
                % GRAM LISTING WHERE THE VARIABLES ARE INI-
                % TIALIZED).

   PLOTTING    ,% IF TRUE THE RESULTS ARE PLOTTED IN ACCOR-
                % DANCE WITH THE INPUT PLOT EXPRESSIONS,
                % IF FALSE NO PLOT TAKES PLACE.
                % THE VALUE OF PLOTTING IS SET TO THE VALUE
                % TRUE IF PLOT EXPRESSIONS ARE READ.
                % DEFAULT VALUE IS FALSE.

   DATAERR     ,% IF TRUE NO COMPUTAION IS DONE. IT IS SET
                % TO TRUE IF INPUTERROR IS TRUE. THE REASON
                % FOR TWO BOOLEANS FOR INPUT ERRORS IS THAT
                % IT SHALL BE POSSIBLE FOR THE USER TO CON-
                % TROL SEVERAL SETS OF INPUT DATA IN THE
                % SAME RUN OF THE PROGRAM. FOR TEST OF
                % EACH DATASET (WHICH END WITH THE DATA
                % 'ENDDATA') THE INPUT PROCEDURE STARTS
                % WITH INITIALIZE INPUTERROR TO FALSE,
                % AND IF ONE OF THE DATA SETS IS WRONG
                % THAT WILL BE DETECTED BY THE VALUE OF
                % DATAERR.

   DEVCLASSOK ,% USED IN THE PROGRAM FOR CHECK OF DEVI-
                % CECLASS IS CORRECT.

   XVAL        ,% SET TO TRUE IF EXPERIMENTAL VALUES
                % ARE READ.
                % DEFAULT VALUE IS FALSE.

   INPE        ;% SET TO TRUE IF A PLOTEXPRESSION IS READ.
                % DEFAULT VALUE IS FALSE.

STRING          % DECLARATION
  TEXT         ,% CONTAINS A TEXT PRECEDED BY AN IDENTIFICATION
                % MARKED: CASE: . THIS TEXT IS A DATA IN THE
                % DATA INPUT FILE AND THUS GIVEN BY THE USER.
                % DEFAULT VALUE OF TEXT IS EMPTY.
```

```
  DEVCLASS ,% CONTAINS THE DEVICECLASS FOR THE OUTPUT MEDIUM
           % FOR PLOTS. FOR THE NAMES OF THE DIFFERENT DEVI-
           % CE CLASSES SEE THE DESCRIPTION OF 'RIGS' [2].
           % DEFAULT VALUE IS "PLOTTER".

  DEVNAME  ,% CONTAINS THE DEVICENAME OF THE OUTPUT MEDIUM
           % FOR PLOTS. SEE DESCRIPTION OF 'RIGS' FOR THE
           % DIFFERENT DEVICENAME.
           % DEFAULT VALUE IS "BENSON1133".

  DCLASS   ,% DCLASS AND
  DNAME    ,% DNAME ARE WORKING VARIABLES USED FOR CONTROL
           % OF CORRECT DEVCLASS AND DEVNAME.

  DATE     ;% WORKING VARIABLE USED FOR COMPUTING
           % THE DATE FOR THE COMPUTATION.

LABEL       % DECLARATION
  FIN      ;%

LIBRARY     % DECLARATION.
  FLIB(TITLE="CHEMSIMUL/FORLIB.");


% INCLUSION OF THE PLOTTER PACKAGE 'RISOE/RIGS/A'.
$INCLUDE "RISOE/RIGS/A"

%*******************************************************
%                                                      *
% THE FOLLOWING DECLARATIONS REFER TO B 7800 LIBRARY   *
% CONCEPT. 'FLIB' (AN ABREVIATION FOR FORTRAN LIBRARY  *
% IS THE INTERNAL NAME OF THE LIBRARY WHICH CONTAINS   *
% THE SUBROUTINES: 'DRIVE', 'INFORM', 'INITCM' AND     *
% 'LOCKF7' TO WHICH CALLS ARE DONE FROM THE ALGOL      *
% MAIN PROGRAM.                                        *
%                                                      *
% THE PROCEDURE DECLARATIONS ARE THE SOCALLED          *
% 'SKELETON'- DECLARATIONS FOR THE CORRESPONDING FOR-  *
% TRAN SUBROUTINES IN THE LIBRARY 'FLIB'. THE DECLA-   *
% RATIONS ARE NECESSARY IN THE BURROUGHS SOFTWARE SY-  *
% STEM.                                                *
%*******************************************************

PROCEDURE DRIVE(N,TO,HO,HMAXO,YO,TOUT,EPS,IERROR,MF,INDEX);
%*******************************************************
% 'DRIVE' IS THE DRIVER FOR THE FOR FORTRAN PACKAGE    *
% 'EPISODE' CODED BY A.C.HINDMARSH (REF [1]) WHICH     *
% DOES THE NUMERICAL INTEGRATION OF A SYSTEM OF        *
% ORDINARY NON-LINEAR DIFFERENTIAL EQUATIONS.          *
% FOR THE PRESENT TIME IT IS NOT POSSIBLE TO USE       *
% SUBROUTINE OR FUNCTION PARAMETERS IN SUBROUTINES     *
% IN FORTRAN LIBRARIES AND THEREFORE IT HAS BEEN       *
% NECESSARY TO DO SOME MINOR CHANGES IN THE CALL       *
% OF DRIVE COMPARED TO THE ORIGINAL VERSION BY HIND-   *
% MARSH.                                               *
%*******************************************************

% SPECIFICATION OF THE PROCEDURE PARAMETERS.

INTEGER
  N      , % THE ORDER OF THE DIFFERENTIAL EQUATION SYSTEM.

  IERROR, % FLAG FOR THE ERROR CONTROL.
```

```
    MF     , % METHOD FLAG.

    INDEX ; % FLAG INDICATING THE TYPE OF CALL.

  REAL
    TO     , % START VALUE FOR THE INDENPENDENT VARIABLE T.

    HO     , % THE FIRST INTEGRATION STEP.

    HMAXO , % THE MAXMIMUM INTEGRATION STEP ALLOWED (THIS
            % PARAMETER IS NEW COMPARED TO THE ORIGINAL
            % 'EPISODE' AND IS INTRODUCED FOR CONTROL OF
            % THE INTEGRATION STEPS).

    TOUT   , % END OF INTEGRATION.

    EPS    ; % RELATIVE ACCURACY IN 'EPISODE'.

  REAL ARRAY

    YO[*] ; % THE SOLUTION VECTOR.

  LIBRARY
    FLIB   ; % THIS SPECIFICATION INDICATES THAT THE CODE
            % OF THE PROCEDURE WILL BE FOUND IN THE
            % LIBRARY FLIB.

  % FOR MORE DETAILS, SEE REF [1].

  PROCEDURE INFORM(HMIN,HMAX,HUSED,NQUSED,NSTEP,NFE,NJE);
  %*****************************************************
  % 'INFORM' TRANSFERS TO THE ALGOL MAIN PROGRAM THE   *
  % CONTENTS OF SOME VARIABLES STORED IN A COMMON      *
  % AREA IN 'EPISODE'.                                 *
  %*****************************************************

  % SPECIFICATIONS OF PARAMETERS.

  INTEGER
    NQUSED  , % THE LAST INTEGRATION ORDER USED.

    NSTEP   , % NUMBER OF INTEGRATION STEPS.

    NFE     , % NUMBER OF FUNCTION EVALUATIONS.

    NJE     ; % NUMBER OF JACOBIAN EVALUATIONS.

  REAL
    HMIN    , % SMALLEST STEP USED IN EPISODE

    HMAX    , % LARGEST STEP USED IN EPISODE.

    HUSED   ; % LAST STEP USED IN EPISODE.

  LIBRARY
    FLIB    ; % SEE DESCRIPTION OF PROCEDURE DRIVE.


  PROCEDURE INITCM(NR,NT,NT5,ETERM,ERK,RGC,RAD,INITOK);
  %*****************************************************
  %                                                    *
  % 'INITCM' TRANSFER TO THE ALGOL MAIN PROGRAM THE CON-*
  % TENTS OF SOME VARIABLES STORED IN A COMMON AREA IN  *
```

```
% THE FORTRAN LIBRARY 'FLIB'.                                    *
%*****************************************************

% SPECIFICATION OF PARAMETERS.

INTEGER
  NR,        % NUMBER OF REACTANTS.
  NT5  ,
  NT;        % NUMBER OF TERMS IN DIFF.EQ.SYSTEM.

INTEGER ARRAY
  ETERM[*] ; % INTEGER ARRAY OF LENGTH 5*NT+1.

REAL ARRAY
  ERK[*]   , % REAL ARRAY OF LENGTH NT+1.

  RGC[*]   ; % REAL ARRAY OF LENGTH NR+1.

BOOLEAN
  RAD      , % % TRUE IF REACTANT IS EXPOSED TO
             % RADIATION, ELSE FALSE.
  INITOK;    % TRUE IF THE INITIATION OF COMMON
             % VARIABLES IN 'FLIB' IS OK,ELSE FALSE.
LIBRARY FLIB;

PROCEDURE LOCKF7;
%*****************************************************
% CLOSES AN OUTPUT FILE IN 'FLIB' CONTAINING MESSAGES *
% FROM THE INTEGRATION PROCESS IN 'EPISODE'.          *
%*****************************************************
LIBRARY   % SPECIFICATION.
  FLIB;

  %*******************************************************
  % INITIALIZE VARIABLES FOR ARRAY DECLARATIONS          *
  %*******************************************************
  MAXRE := 100;
  MAXNT := 400;
  MAXNR := 100;
  MAXGV := 50;
  MAXPE := 20;
  MAXPT := 100;
  MAXSET:= 20;

  BEGIN

  STRING ARRAY          % DECLARATION
    RE[1:MAXRE]      ,% CONTAINS THE REACTION EQUATIONS READ
                      % AND STORED BY THE PROCEDURE CHEMINPUT.
    RC[1:MAXRE]      ,% CONTAINS THE INPUT DATA FOR RATE
                      % CONSTANTS STORED BY THE PROCEDURE
                      % CHEMINPUT.
    GV[1:MAXGV]      ,% CONTAINS THE INPUT DATA FOR G-VALUES
                      % READ AND STORED BY PROCEDURE CHEMINPUT.
    RNA[1:MAXNR]     ,% CONTAINS THE REACTANT NAMES FOUND AND
                      % STORED BY THE PROCEDURE CHEMTRANS.
    CON[1:MAXNR]     ,% CONTAINS THE INPUT DATA FOR THE START
                      % CONCENTRATIONS OF THE REACTANTS.
    PE[1:MAXPE]      ,% CONTAINS THE PLOT EXPRESSIONS READ AND
                      % STORED BY THE PROCEDURE CHEMINPUT.
    XPDATA[1:MAXPE] ;% CONTAINS INPUT DATA FOR THE EXPERI-
                      % MENTAL VALUES TO BE USED IN PLOTS.

  LONG INTEGER ARRAY % DECLARATION
```

```
  ETERM[0:5*MAXNT];% CONTAINS THE INFORMATION FOR USE BY
                   % PROCEDURES DIFFUN, PEDERV STORED BY
                   % THE TRANSLATION PROCEDURE CHEMTRANS.
                   % FOR THE DETAILED DISCRIPTION SEE
                   % THE DESCRIPTION OF PROCEDURE DIFFUN.

INTEGER ARRAY      % DECLARATION
  PEN[1:MAXPT]    ,% CONTAINS ADRESSES FOR PLOTEXPRESSION
                   % NUMBERS STORED BY THE TRANSLATION PRO-
                   % CEDURE TRANSPLOT AND USED BY THE
                   % PLOT PROCEDURE CHEMPLOT.
  PRN[1:MAXPT]    ;% CONTAINS ADRESSES FOR REACTANT NUMBERS
                   % FOR THE REACTANTS IN THE PLOTEXPRES-
                   % SIONS. USED BY THE PLOT PROCEDURE
                   % CHEMPLOT.

REAL ARRAY         % DECLARATION
  PFAC[1:MAXPT]   ,% CONTAINS THE FACTORS TO THE REACTANTS
                   % IN THE PLOTEXPRESSIONS STORED BY THE
                   % PROCEDURE CHEMTRANS.
  RK[1:2,1:MAXRE] ,% CONTAINS THE REACTION EQUATION NUMBER
                   % AND THE ADDRESS FOR THE CORRESPONDING
                   % RATE CONSTANT.
  RCON[1:MAXNR]   ,% CONTAINS THE VALUES OF THE RATE
                   % CONSTANTS.
  RG[1:MAXNR]     ,% CONTAINS THE G-VALUES.
  ADDCON[1:MAXPE] ;% CONTAINS THE ADDITIVE CONSTANT IN THE
                   % PLOTEXPRESSIONS.

LABEL              % DECLARATION
  PROGRAMSTART    ;

PROCEDURE CHEMINPUT;
%************************************************
% CHEMINPUT IS THE CENTRAL INPUT PROCEDURE FOR   *
% ALL RELEVANT DATA TO BE USED IN 'CHEMSIMUL'.   *
%                                                *
% THE FOLLOWING VARIABLES ARE GLOBAL             *
% AND ARE DESCRIBED IN THE START OF THE PROGRAM. *
% INTEGER MAXRE,MAXGV,MAXNR,MAXPE,MAXPT,MAXNT,   *
%         LINELE,DEC,MF,NRR,RADPRS,PRINTS,       *
%         RESCLASS,NRE,NRC,NGV,NCON,NPE,MAXRL,   *
%         NR,NT,PT.                              *
% STRING  TEXT,DEVCLASS,DEVNAME.                 *
% REAL    DOSE,RADTIME,FSTSTP,TEND,EPS,HMAXO.    *
% LABEL   FIN.                                   *
% BOOLEAN NEWSYS,NEWRE,NEWRC,NEWGV,NEWCON,NEWPE, *
%         TABLE,PLOTTING,DATACON,DATAERR,DIFFEQ, *
%         TERMINAL, OUTINP.                      *
% STRING ARRAY  RE, RC, GV, CON, PE, XPDATA.     *
%                                                *
%************************************************

BEGIN
%************************************************
% DECLARATION OF LOCAL VARIABLES.                *
%************************************************

TRUTHSET           % DECLARATION
  NUMBER("0123456789");

STRING             % DECLARATION
  INPS, S, S1;
```

```
BOOLEAN                 % DECLARATION
  FOUND,
  STORED,
  B,
  NEWCARD;

INTEGER                 % DECLARATION
  I,J,K,L,DSET;         % COUNTING VARIABLES.

LABEL                   % DECLARATION
  START;                % START POINT IN THE PROCEDURE.


%************************************************
% THE FOLLOWING FORMATS ARE ERROR MESSAGES WHICH *
% WILL BE PRINTED IF INPUT ERRORS ARE DETECTED.  *
%************************************************
FORMAT
MESS1("ERROR: WRONG IDENTIFICATION IN REACTION EQUATION:"/A*),
MESS2("ERROR: WRONG IDENTIFICATION FOR K VALUE:"/A*),
MESS3("ERROR: NO ')=' IN INPUTRECORD FOR G VALUE:"/A*),
MESS4("ERROR: NO ')=' IN INPUTRECORD FOR CON VALUE:"/A*),
MESS5("ERROR: WRONG IDENTIFICATION IN PLOT EXPRESSION:"/A*),
MESS6("ERROR: YOU HAVE USED 'TERMINAL' OR 'HARDCOPY' AS "/
         "       DEVCLASS IN THE INPUT FILE."/
         "       DEVCLASS='PLOTTER' IS STILL IN ACTION"),
MESS7("ERROR: MITCHMATCH IN DEVCLASS/DEVNAME"),
MESS8("WRONG IDENTIFICATION IN:"/A*),
MESS9("INPUTERRORS: COMPUTATION FOR THIS PROBLEM DELETED"),
MESS10("THE FIRST INTEGRATION STEP 'FSTSTP' IS WANTED"),
MESS11("THE MAX. INTEGRATION STEP 'HMAX' IS WANTED"),
MESS12("THE END OF INTEGRATION 'TEND' IS WANTED"),
MESS13("THE ACCURACY OF INTEGRATION 'EPS' IS WANTED"),
MESS14("THE NUMBER OF RESULTS 'PRINTS' IS WANTED");

STRING PROCEDURE IDENT(S,C);
STRING S; INTEGER C;
COMMENT
  THE PROCEDURE IS USED FOR CHECK OF
  IDENTIFICATION OF AN INPUT STRING;
IDENT:=TAKE(S,MIN(C,LENGTH(S)));

REAL PROCEDURE NUM(S,R);
STRING S; REAL R;
COMMENT
  THE PROCEDURE IS USED FOR STORING
  AN INPUT NUMBER;
NUM:=DECIMAL(DROP(S,R));

  START:
  INPUTERROR:=NEWRE:=NEWRC:=NEWGV:=NEWCON:=NEWPE:=FALSE;
  NEWCARD:=TRUE;
  NRC:=0;
  IF NEWSYS THEN
  BEGIN
    COMMENT
      INITIALIZATION OF VARIABLES FOR A NEW SYSTEM;
    LINELE := 132;
    DEC    := 3;
    MF     := 21;
    PLONPA := 3;
    DEVCLASS := "PLOTTER"; DEVNAME := "BENSON1133";
    SETDEVICE(DEVCLASS,DEVNAME);
    RADPRS:=PRINTS:=0;
```

```
  NRE:=NRC:=NGV:=NCON:=NPE:=0;
  MAXRL:=NR:=NT:=PT:=0;
  DOSE:=RADTIME:=FSTSTP:=TEND:=EPS:=HMAX0:=0;
  RESCLASS:=NRR:=1;
  TABLE:=OUTINP:=TRUE;
  NEWSYS:=PLOTTING:=DATACON:=DATAERR:=DIFFEQ:=FALSE;
  INTMES:=INPE:=XVAL:=FALSE;
END OF INITIALIZATION;

WHILE NEWCARD DO
BEGIN
  READ(INP,<A80>,INPS)[FIN];
  INPS:=TAIL(INPS," ");
  S1:=INPS; S:=EMPTY;

  COMMENT REMOVE ALL SPACES IN S1 TO FACILITATE
          THE ANALYSIS;
  WHILE LENGTH(S1)>0 DO
  BEGIN
    S:=S CAT HEAD(S1,NOT " ");
    S1:=DROP(S1,LENGTH(HEAD(S1,NOT " ")));
    S1:=DROP(S1,LENGTH(HEAD(S1," ")));
  END;

  COMMENT
    STRING S CONTAINS THE INPUT RECORD WITH
    SPACES REMOVED AND IS THUS READY FOR
    EXAMINATION AND STORING IF THE CONTENT
    IS CORRECT;

  L:=LENGTH(S);
  IF          S = EMPTY       THEN NEWCARD  :=TRUE  ELSE
  IF IDENT(S,5)="ENDPE"       THEN INPE:=FALSE      ELSE
  IF IDENT(S,5) NEQ "ENDPE" AND INPE THEN
  BEGIN
    COMMENT
      STORE EXPERIMENTAL DATA IN STRING
      ARRAY EXPDATA. EVERY SET OF DATA
      BELONGING TO A PLOTTING EXPRESSION
      ARE STORED IN A STRING ELEMENT SE-
      PARATED BY SEMICOLONS ;

    DSET:=DSET+1; MAXSET:=MAX(MAXSET,DSET);
    XVAL:=TRUE;
    XPDATA[NPE]:=(XPDATA[NPE] CAT S) CAT ";";
  END ELSE
  IF IDENT(S,5)="CASE:"     THEN TEXT:=INPS         ELSE
  IF IDENT(S,3)="MF="       THEN MF:=NUM(S,3)       ELSE
  IF IDENT(S,4)="DEC="      THEN DEC:=NUM(S,4)      ELSE
  IF IDENT(S,4)="NRR="      THEN NRR:=NUM(S,4)      ELSE
  IF IDENT(S,4)="EPS="      THEN EPS:=NUM(S,4)      ELSE
  IF IDENT(S,5)="DOSE="     THEN DOSE:=NUM(S,5)     ELSE
  IF IDENT(S,5)="TEND="     THEN TEND:=NUM(S,5)     ELSE
  IF IDENT(S,5)="HMAX="     THEN HMAX0:=NUM(S,5)    ELSE
  IF IDENT(S,5)="TABLE"     THEN TABLE:=TRUE        ELSE
  IF IDENT(S,6)="DIFFEQ"    THEN DIFFEQ:=TRUE       ELSE
  IF IDENT(S,6)="OUTINP"    THEN OUTINP:=TRUE       ELSE
  IF IDENT(S,6)="INTMES"    THEN INTMES:=TRUE       ELSE
  IF IDENT(S,8)="NOINTMES"  THEN INTMES:=FALSE      ELSE
  IF IDENT(S,6)="SUPINP"    THEN OUTINP:=FALSE      ELSE
  IF IDENT(S,8)="RADTIME="  THEN RADTIME:=NUM(S,8)  ELSE
  IF IDENT(S,7)="FSTSTP="   THEN FSTSTP:=NUM(S,7)   ELSE
  IF IDENT(S,7)="RADPRS="   THEN RADPRS:=NUM(S,7)   ELSE
  IF IDENT(S,7)="PRINTS="   THEN PRINTS:=NUM(S,7)   ELSE
```

```
IF IDENT(S,7)="LINELE="  THEN LINELE:=NUM(S,7)  ELSE
IF IDENT(S,7)="PLONPA="  THEN PLONPA:=NUM(S,7)  ELSE
IF IDENT(S,7)="DATACON"  THEN DATACON:=TRUE     ELSE
IF IDENT(S,7)="NOTABLE"  THEN TABLE:=FALSE      ELSE
IF IDENT(S,7)="ENDDATA"  THEN NEWCARD:=FALSE    ELSE
IF IDENT(S,8)="NODIFFEQ" THEN DIFFEQ:=FALSE     ELSE
IF IDENT(S,9)="RESCLASS=" THEN
                           RESCLASS:=NUM(S,9)   ELSE
IF IDENT(S,9)="NODATACON" THEN DATACON:=FALSE   ELSE
IF IDENT(S,6)="NEWSYS" THEN
BEGIN
  NEWSYS:=TRUE;
  GO TO START;
END ELSE
IF IDENT(S,9)="DEVCLASS=" THEN
BEGIN
  DCLASS:=HEAD(DROP(S,9),NOT " ") ;
  DEVCLASSOK:= TRUE;
  IF NOT TERMINAL THEN
  IF DCLASS="TERMINAL" OR DCLASS="HARDCOPY" THEN
  BEGIN
    WRITE(ERRFIL,MESS6);
    DEVCLASSOK:=FALSE;
  END;
END ELSE
IF IDENT(S,8)="DEVNAME=" THEN
BEGIN
  DNAME:=HEAD(DROP(S,8),NOT " ") ;
  IF DEVCLASSOK THEN
  BEGIN
    SETDEVICE(DCLASS,DNAME);
    IF NOT GETERRORBIT THEN
    BEGIN
      DEVCLASS:=DCLASS; DEVNAME:=DNAME;
    END ELSE
    BEGIN
      WRITE(ERRFIL,MESS7);
      INPUTERROR:=TRUE;
    END;
  END;
END ELSE

IF IDENT(S,2)="RE" THEN
BEGIN
COMMENT
  ANALYSE THE IDENTIFICATION OF INPUT STRING
  CONRAINING THE REACTION EQUATION, AND
  IF CORRECT STORE IT IN THE STRING ARRAY RE;

  NEWRE:=TRUE;
  S1:=DROP(S,2);
  IF HEAD(S1,NUMBER)=HEAD(S1,NOT":") THEN
  BEGIN
    FOUND:=FALSE;

    COMMENT
      EXCHANGE AN EXISTING REACTION EQUATION
      IN STRING ARRAY RE BY A NEW ONE WITH
      SAME IDENTIFICATION;
    FOR I:=1 STEP 1 WHILE I<=NRE AND NOT FOUND DO
    IF HEAD(S,NOT":")=HEAD(RE[I],NOT ":") THEN
    BEGIN
      FOUND:=TRUE; RE[I]:=S;
    END;
```

```
    IF NOT FOUND THEN
    BEGIN
      COMMENT THE INPUT REACTION EQUATION HAS
              A NEW IDENTIFICATION;
      STORED:=FALSE;

      COMMENT
        STORE THE NEW REACTION EQUATION IN
        THE FIRST EMPTY STRING ELEMENT OF STRING
        ARRAY RE;
      FOR J:=1 STEP 1 WHILE J<=NRE AND NOT STORED DO
      IF RE[J]=EMPTY THEN
      BEGIN
        RE[J]:=S; STORED:=TRUE;
      END;

      IF NOT STORED THEN
      BEGIN
        COMMENT
          NO EMPTY ELEMENT EXISTS IN RE FOR
          STORING THE NEW INPUT REACTION
          EQUATION, INCREASE LENGTH OF RE AND
          RC WITH 5 STRING ELEMENTS AND STORE
          THE INPUT REACTION EQUATION;

        NRE:=NRE+1;
        IF NRE>MAXRE THEN
        BEGIN
          COMMENT INCREASE STRING ARRAYS RE AND RC;
          MAXRE:=MAXRE + 5;
          RESIZE(RE,MAXRE,RETAIN);
          RESIZE(RC,MAXRE,RETAIN);
        END;
        RE[NRE]:=S;
      END NOT STORED;
    END NOT FOUND;
  END ELSE
  BEGIN
    COMMENT WRONG IDENTIFICATION FOR REACTION
            EQUATION, PRINT ERROR MESSAGE;

    INPUTERROR:=TRUE;
    WRITE(ERRFIL,MESS1,LENGTH(INPS),INPS);
  END;
END ELSE

IF IDENT(S,1)="K" THEN
BEGIN
  COMMENT
    ANALYSE THE IDENTIFICATION OF INPUT STRING
    CONTAINING THE RATE CONSTANT BELONGING
    TO A REACTION EQUATION AND IF CORRECT,THEN
    STORE IT IN THE STRING ARRAY RC;

  NEWRC:=TRUE;
  S1:=DROP(S,1);
  IF HEAD(S1,NUMBER)=HEAD(S1,NOT"=") THEN
  BEGIN
    FOUND:=FALSE;

    COMMENT
      EXCHANGE AN EXISTING RATE CONSTANT
      IN STRING ARRAY RC BY A NEW ONE WITH
```

```
    SAME IDENTIFICATION;
   FOR I:=1 STEP 1 WHILE I<=NRC AND NOT FOUND DO
   IF HEAD(S,NOT "=")=HEAD(RC[I],NOT "=") THEN
   BEGIN
     FOUND:=TRUE; RC[I]:=S;
   END;

   IF NOT FOUND THEN
   BEGIN
     COMMENT THE INPUT STRING WITH RATE CONSTANT
             HAS A NEW IDENTIFICATION;
     STORED:=FALSE;

     COMMENT
       STORE THE NEW RATE CONSTANT IN THE FIRST
       EMPTY STRING ELEMENT OF STRING ARRAY RC;
     FOR J:=1 STEP 1 WHILE J<=NRC AND NOT STORED DO
     IF RC[J]=EMPTY THEN
     BEGIN
      RC[J]:=S; STORED:=TRUE;
     END;

     IF NOT STORED THEN
     BEGIN
       COMMENT
         NO EMPTY ELEMENT EXISTS IN RC FOR
         STORING THE NEW INPUT VALUE FOR
         RATE CONSTANT, INCREASE THE
         LENGTH OF RC AND RE WITH 5 STRING
         ELEMENTS AND STORE THE INPUT RE-
         ACTION CONSTANT;

       NRC:=NRC+1;
       IF NRC>MAXRE THEN
       BEGIN
         COMMENT
           INCREASE STRING ARRAYS RE AND RC;
         MAXRE:=MAXRE + 5;
         RESIZE(RE,MAXRE,RETAIN);
         RESIZE(RC,MAXRE,RETAIN);
       END;
       RC[NRC]:=S;
     END;
   END;
  END ELSE
  BEGIN
    COMMENT WRONG IDENTIFICATION FOR A RATE
            CONSTANT, PRINT ERROR MESSAGE;

    INPUTERROR:=TRUE;
    WRITE(ERRFIL,MESS2,LENGTH(INPS),INPS);
  END;
END ELSE

IF IDENT(S,2)="G(" THEN
BEGIN
  COMMENT
    ANALYSE THE IDENTIFICATION OF INPUT STRING
    CONTAINING THE G-VALUE BELONGING TO A STOI-
    CHIOMETRIC EQUATION AND IF CORRECT, THEN
    STORE IT IN STRING ARRAY GV;

  NEWGV:=TRUE;
  S1:=TAIL(S,NOT")=");
```

```
  IF TAKE(S1,MIN(2,LENGTH(S1)))=")=" THEN
  BEGIN
    FOUND:=FALSE;
    COMMENT
      EXCHANGE AN EXISTING G-VALUE IN
      STRING ARRAY GV BY A NEW ONE WITH
      SAME IDENTIFICATION;
    FOR I:=1 STEP 1 WHILE I<=NGV  AND NOT FOUND DO
    IF HEAD(S,NOT"=")=HEAD(GV[I],NOT"=") THEN
    BEGIN
      FOUND:=TRUE; GV[I]:=S;
    END;

    IF NOT FOUND THEN
    BEGIN
      COMMENT
        THE INPUT STRING WITH NEW G-VALUE HAS A
        NEW IDENTIFICATION;

      STORED:=FALSE;
      COMMENT
        STORE THE NEW INPUT G-VALUE IN THE FIRST
        EMPTY STRING ELEMENT IN STRING ARRAY GV;
      FOR J:=1 STEP 1 WHILE J<=NGV AND NOT STORED DO
      IF GV[J]=EMPTY THEN
      BEGIN
        GV[J]:=S; STORED:=TRUE;
      END;

      IF NOT STORED THEN
      BEGIN
        COMMENT
          NO EMPTY ELEMENT EXISTS IN GV FOR
          STORING THE NEW INPUT G-VALUE, IN-
          CREASE THE LENGTH OF GV WITH 5
          STRING ELEMENTS AND STORE THE
          INPUT G-VALUE;
        NGV:=NGV+1;
        IF NGV>MAXGV THEN
        BEGIN
          COMMENT INCREASE STRING ARRAY GV ;
          MAXGV:=MAXGV + 5;
          RESIZE(GV,MAXGV,RETAIN);
        END;
        GV[NGV]:=S;
      END;
    END;
  END ELSE
  BEGIN
    COMMENT ERROR IN INPUT RECORD FOR G-VALUE,
            PRINT ERROR MESSAGE;

    INPUTERROR:=TRUE;
    WRITE(ERRFIL,MESS3,LENGTH(INPS),INPS);
  END;
END ELSE

IF IDENT(S,4)="CON(" THEN
BEGIN
  COMMENT
    ANALYSE THE IDENTIFICATION OF INPUT STRING
    CONTAINING A NEW START CONCENTRATION FOR A
    REACTANT AND IF CORRECT, THEN STORE IT IN
    STRING ARRAY CON;
```

```
    NEWCON:=TRUE;
    S1:=TAIL(S,NOT")=");
    IF TAKE(S1,MIN(2,LENGTH(S1)))=")=" THEN
    BEGIN
      COMMENT
        EXCHANGE EXISTING CONCENTRATION VALUE IN
        STRING ARRAY CON BY A NEW ONE WITH SAME
        IDENTIFICATION;

      FOUND:=FALSE;
      FOR I:=1 STEP 1 WHILE I<=NCON AND NOT FOUND DO
      IF HEAD(S,NOT"=")=HEAD(CON[I],NOT"=") THEN
      BEGIN
        FOUND:=TRUE; CON[I]:=S;
      END;

      IF NOT FOUND THEN
      BEGIN
       COMMENT
          INPUT STRING WITH REACTANT CONCENTRA-
          TION HAS A NEW IDENTIFICATION;

        STORED:=FALSE;
        COMMENT STORE THE NEW INPUT CONCENTRATION
                VALUE IN THE FIRST EMPTY STRING
                ELEMENT OF STRING ARRAY CON;

        FOR J:=1 STEP 1 WHILE J<=NCON AND NOT FOUND DO
        IF CON[J]=EMPTY THEN
        BEGIN
          CON[J]:=S; STORED:=TRUE;
        END;

        IF NOT STORED THEN
        BEGIN
          COMMENT
            NO EMPTY ELEMENT EXISTS IN CON FOR
            STORING THE NEW INPUT CONCNETRATION
            VALUE, INCREASE LENGTH OF CON AND
            RNA WITH 5 STRING ELEMENTS AND STO-
            RE THE INPUT CONCENTRATION VALUE;

          NCON:=NCON+1;
          IF NCON>MAXNR THEN
          BEGIN
            COMMENT
              INCREASE STRING ARRAY RNA AND CON;
            MAXNR:=MAXNR + 5;
            RESIZE(RNA,MAXNR,RETAIN);
            RESIZE(CON,MAXNR,RETAIN);
          END;
          CON[NCON]:=S;
        END;
      END;
    END ELSE
    BEGIN
      COMMENT
        ERROR IN INPUT RECORD FOR CONCNETRATION;

      INPUTERROR:=TRUE;
      WRITE(ERRFIL,MESS4,LENGTH(INPS),INPS);
    END;
  END ELSE
```

```
IF IDENT(S,2)="PE" THEN
BEGIN
  COMMENT
    ANALYSE THE IDENTIFICATION OF INPUT STRING
    CONTAINING A NEW PLOT EXPRESSION AND IF
    CORRECT, THEN STORE IT IN THE STRING ARRAY
    PE.
    INITIATE THE VARIABLES INPE AND DSET FOR
    INPUT OF EXPERIMENTAL DATA IF THEY ARE
    PLACED IN THE INPUT FILE AFTER THE PLOT
    EXPRESSION;

  NEWPE:=PLOTTING:=INPE:=TRUE;
  S1:=DROP(S,2);
  IF HEAD(S1,NUMBER)=HEAD(S1,NOT":") THEN
  BEGIN
    FOUND:=FALSE;
    COMMENT CHANGE EXISTING PLOT EXPRESSION WITH
            SAME IDENTIFICATION;
    FOR I:=1 STEP 1 WHILE I<=NPE AND NOT FOUND DO
    IF HEAD(S,NOT":")=HEAD(PE[I],NOT":") THEN
    BEGIN
     FOUND:=TRUE; PE[I]:=S;
    END;

    IF NOT FOUND THEN
    BEGIN
      STORED:=FALSE;

      COMMENT
        STORE THE PLOT EXPRESSION IN THE FIRST
        EMPTY STRING ELEMENT OF STRING ARRAY PE;
      FOR J:=1 STEP 1 WHILE J<=NPE AND NOT STORED DO
      IF PE[J]=EMPTY THEN
      BEGIN
       PE[J]:=S; STORED:=TRUE;
      END;

      IF NOT STORED THEN
      BEGIN
        COMMENT
          NO EMPTY ELEMENT EXISTS IN PE FOR
          STORING THE NEW INPUT PLOT EXPRES-
          SION, INCREASE THE LENGTH OF PE
          WITH 5 STRING ELEMENTS AND STORE
          THE INPUT PLOTTING EXPRESSION.
          INITIATE XPDATA AND DSET;

        NPE:=NPE+1;
        IF NPE>MAXPE THEN
        BEGIN
          COMMENT INCREASE STRING ARRAY PE;
          MAXPE:=MAXPE + 5;
          RESIZE(PE,MAXPE,RETAIN);
          RESIZE(XPDATA,MAXPE,RETAIN);
        END;
        PE[NPE]:=S; XPDATA[NPE]:=EMPTY; DSET:=0;
      END
    END;
  END ELSE
  BEGIN
    COMMENT
      WRONG IDENTIFICATION IN PLOT EXPRESSION;
```

```
      INPUTERROR:=TRUE;
      WRITE(ERRFIL,MESS5,LENGTH(INPS),INPS);
    END;

  END ELSE
  BEGIN
    COMMENT
      WRONG IDENTIFICATION IN THE INPUT RECORD,
      PRINT ERROR MESSAGE;
    INPUTERROR:=TRUE;
    WRITE(ERRFIL,MESS8,LENGTH(INPS),INPS);
  END;
END NEWCARD;

COMMENT
  IF NO ERROR HAS BEEN DETECTED DURING THE
  INPUT PROCESS AND CORRECTION TO RATE
  CONSTANTS HAS BEEN READ THEN THESE COR-
  RECTIONS WILL BE STORED IN STRING ARRAY
  RE IN PLACE OF THE ORIGINAL ONES;

IF NOT INPUTERROR AND NEWRC THEN
BEGIN
  FOR I:=1 STEP 1 UNTIL NRC DO
  BEGIN
    COMMENT STORE CORRECTION OF RATE CONSTANT K;
    B:=FALSE;
    FOR J:=1 STEP 1 WHILE J<=NRE AND NOT B DO
    BEGIN
      COMMENT
        CHECK THE IDENTIFICATION OF A RATE
        CONSTANT CORRESPONDING TO THE IDENTI-
        FICATION OF A CHEMICAL REACTION EQUATION.
        IF TRUE, REPLACE THE OLD CONSTANT WITH
        THE NEW ONE;

      IF DROP(HEAD(RC[I],NOT "="),1) =
         DROP(HEAD(RE[J],NOT ":"),2) THEN
      BEGIN
        RE[J]:=HEAD(RE[J],NOT ";")
               CAT ";"
               CAT TAIL(TAIL(RC[I],NOT "="),"=");
        B:=TRUE;
      END;
    END J;
    IF NOT B THEN
    BEGIN
      COMMENT WRONG IDENTIFICATION FOR THE NEW
              RATE CONSTANT;
      WRITE(ERRFIL,MESS2,LENGTH(RC[I]),RC[I]);
      INPUTERROR:=TRUE;
    END;
  END I;
END;

COMMENT
  CONTROL THAT NECESSARY INPUT DATA ARE STORED
  IF NOT THE PROGRAM IS ABORTED;

IF FSTSTP<=0.0 THEN
BEGIN
  WRITE(ERRFIL,MESS10); INPUTERROR:=TRUE;
END;
IF HMAX0<=0.0 THEN
```

```
    BEGIN
      WRITE(ERRFIL,MESS11); INPUTERROR:=TRUE;
    END;
    IF TEND<=0.0 THEN
    BEGIN
      WRITE(ERRFIL,MESS12); INPUTERROR:=TRUE;
    END;
    IF EPS<=0.0 THEN
    BEGIN
      WRITE(ERRFIL,MESS13); INPUTERROR:=TRUE;
    END;
    IF PRINTS<=0 THEN
    BEGIN
      WRITE(ERRFIL,MESS14); INPUTERROR:=TRUE;
    END;

    IF INPUTERROR THEN WRITE(ERRFIL,MESS9);

    COMMENT
      IF NO ERRORS AT ALL ARE DETECTED AND IF
      OUTPUT OF INPUT DATAS ARE WANTED, THESE
      ARE PRINTED;

    IF OUTINP AND NOT INPUTERROR THEN
    BEGIN
      IF NEWRE THEN
      BEGIN
        WRITE(OUT,<"NEW REACTION EQUATION SYSTEM">);
        FOR I:=1 STEP 1 UNTIL NRE DO
            WRITE(OUT,<A*>,LENGTH(RE[I]),RE[I]);
      END;

      IF NEWRC THEN
      BEGIN
        WRITE(OUT,<"NEW RATE CONSTANTS">);
        FOR I:=1 STEP 1 UNTIL NRC DO
            WRITE(OUT,<A*>,LENGTH(RC[I]),RC[I]);
      END;

      IF NEWGV THEN
      BEGIN
        WRITE(OUT,<"NEW G - VALUES">);
        FOR I:=1 STEP 1 UNTIL NGV DO
            WRITE(OUT,<A*>,LENGTH(GV[I]),GV[I]);
      END;

      IF NEWCON THEN
      BEGIN
        WRITE(OUT,<"NEW START CONCENTRATIONS">);
        FOR I:=1 STEP 1 UNTIL NCON DO
            WRITE(OUT,<A*>,LENGTH(CON[I]),CON[I]);
      END;

      IF NEWPE THEN
      BEGIN
        WRITE(OUT,<"NEW PLOT EXPRESSIONS">);
        FOR I:=1 STEP 1 UNTIL NPE DO
            IF TAIL(TAIL(PE[I],NOT ":"),":") NEQ EMPTY THEN
            WRITE(OUT,<A*>,LENGTH(PE[I]),PE[I]);
      END;
    END OUTINP AND NOT INPUTERROR;
  END CHEMINPUT;

  PROCEDURE CHEMTRANS;
```

```
%*************************************************
%                                                *
% THE PROCEDURE CHEMTRANS:                       *
%                                                *
% 1: TRANSLATES THE SYSTEM OF REACTION EQUATIONS  *
%    TO A SYSTEM OF NON-LINEAR ORDINARY DIFFEREN- *
%    TIAL EQUATIONS,                             *
%                                                *
% 2: DO A TEST FOR THE MATERIAL MASS-BALANCE.     *
%                                                *
% EACH REACTION EQUATION IS CONSIDERED AS A SYM-  *
% BOL STRING WHICH CONTAINS LETTERS, DIGITS AND   *
% SPECIAL SYMBOLS SUCH AS [,],+,=,;, ECT.         *
% THE CHEMICAL REACTION EQUATIONS USED AS INPUT TO*
% THE PROGRAM ARE SIMILAR IN THE NOMENCLATURE USED*
% BY THE CHEMISTS.                                *
% A REACTANT IN THE REACTION EQUATION IS A STRING *
% OF SYMBOLS BEGINNING WITH A LETTER, FOLLOWED BY *
% LETTERS, DIGITS AND OTHER SYMBOLS. THE LAST MUST*
% BE INCLUDED IN BRACKETS, [ AND ]. THERE IS NO   *
% RESTRICTIONS WITH RESPECT TO THE SYMBOLS USED   *
% IN THE THE BRACKETS.                            *
% THE LEFT AND RIGHT SIDE OF THE REACTIONS EQUA-  *
% TION IS SEPARATED BY THE SIGN '='. THE REACTANTS*
% ARE SEPARATED BY THE SIGN '+' AND THE EQUATION  *
% IS TERMINATED BY THE SEMICOLON (;) FOLLOWED BY  *
% THE RATE CONSTANT FOR THE REACTION EQUATION.    *
%                                                 *
% SOME OF THE REACTANTS ARE MULTIPLIED WITH AN    *
% INTEGER, THE STOICHIOMETRIC CONSTANT.           *
% ON THE LEFT SIDE THERE MUST APPEAR ONLY TWO RE- *
% ACTANTS SINCE ONLY FIRST AND SECOND ORDER ORDER *
% REACTIONS ARE ALLOWED, BUT THERE IS NO RESTRIC- *
% TIONS WITH RESPECT TO THE NUMBER OF REACTANTS   *
% ON THE RIGHT SIDE.                              *
% BY SCANNING A REACTION EQUATION THE PROCEDURE   *
% IDENTIFIES: STOICHIOMETRIC CONSTANTS, REACTANTS *
% SEPARATORS AND RATE CONSTANT AND STORE THEM     *
% IN ARRAYS.                                      *
%                                                 *
% REACTANTS ARE STORED IN ARRAY RNA IN THE ORDER  *
% THEY ARE FOUND.                                 *
% THE STOICHIOMETRIC CONSTANTS, WHICH APPEARS AS  *
% FACTORS PRECEDING THE REACTANTS, ARE STORED IN  *
% ARRAY CHEMAT SUCH THAT THE STOICHIOMETRIC CON-  *
% STANT BELONGING TO REACTANT NO. RN IN REACTION  *
% EQUATION NO. N IS STORED IN ARRAY ELEMENT       *
% CHEMAT[(RN-1)*NRE+N] WHERE NRE IS THE TOTAL NUM-*
% BER OF REACTION EQUATIONS FOUND BY THE INPUT    *
% PROCEDURE CHEMINPUT.                            *
% THE STOICHIOMETRIC CONSTANTS ARE ALSO STORED IN *
% THE ARRAY ETERM (SEE BELOW).                    *
%                                                 *
% THE RATE CONSTANTS ARE STORED IN ARRAY RK.      *
%                                                 *
% THE RESULT OF THE TRANSLATION IS A 2-DIMENSIONAL*
% INTEGER ARRAY ETERM WITH 5 COLUMNS AND AT LEAST *
% NT ROWS. NT, THE TOTAL NUMBER OF TERMS ON THE   *
% RIGHT SIDE OF THE DIFFERENTIAL EQUATION SYSTEM, *
% MUST BE EQUAL TO OR SMALLER THAN MAXNT WHICH DE-*
% TERMINES THE SIZE OF ARRAYS ETERM AND ERK (SEE  *
% DESCRIPTION OF VARIABLES IN THE BEGINNING OF THE*
% PROGRAM).                                       *
%                                                 *
```

```
% A TERM ON THE RIGHT-HAND SIDE OF A DIFFERENTIAL *
% EQUATION CONTRIBUTING TO THE TIME DERIVATIVE OF *
% THE CONCENTRATION OF A REACTANT IS A PRODUCT OF *
% 4 ELEMENTS.  THEY ARE:                          *
% 1: A STOICHIOMETRIC CONSTANT                    *
% 2: A RATE CONSTANT                              *
% 3: A CONCENTRATION OF A REACTANT                *
% 4: A CONCENTRATION OF A REACTANT.               *
%                                                 *
% REMARK: ONE OF THE TWO CONCENTRATIONS HAS THE   *
% VALUE 1, IF THE TERM IS THE CONTRIBUTION OF A   *
% FIRST ORDER REACTION.                           *
%                                                 *
% TO EVERY TERM ON THE RIGHT HAND SIDE OF THE DIF-*
% FERENTIAL SYSTEM THERE CORRESPONDS A ROW IN     *
% ETERM WHICH CONTAINS THE NECESSARY INFORMATION  *
% FOR COMPUTING THE VALUE OF THE TERM.            *
%                                                 *
% EACH ROW IN ETERM CONTAINS 5 ELEMENTS.          *
%                                                 *
% 1. ELEMENT HOLDS THE ADDRESS IN ARRAY RNA OF    *
%    THAT REACTANT FOR WHICH THE TIME-DERIVATIVE  *
%    OF THE CONCENTRATION IS TO BE COMPUTED.      *
%                                                 *
% 2. ELEMENT HOLDS THE VALUE OF THE STOICHIOME-   *
%    TRIC CONSTANT FOR THAT TERM.                 *
%                                                 *
% 3. ELEMENT HOLDS THE ADDRESS IN ARRAY RK CON-   *
%    TAINING THE RATE CONSTANT FOR THAT TERM.     *
%                                                 *
% 4. ELEMENT HOLDS THE ADDRESS IN ARRAY RCON CON- *
%    TAINING THE CONCENTRATION OF THE FIRST REAC- *
%    TANT IN THE PRODUCT.                         *
%                                                 *
% 5. ELEMENT HOLDS THE ADDRESS IN ARRAY RCON CON- *
%    TAINING THE CONCENTRATION IF THE SECOND REAC-*
%    TANT IN THE PRODUCT.                         *
%                                                 *
% THE ARRAY ETERM IS USED IN THE SUBROUTINES COM- *
% PUTING THE RIGHT HAND SIDE OF THE DIFFERENTIAL  *
% EQUATION SYSTEM AND THE JACOBIAN MATRIX FOR THAT*
% SYSTEM. THESE ROUTINES ARE CALLED BY THE INTE-  *
% GRATING ROUTINE CALLED VIA THE PROCEDURE DRIVE. *
%                                                 *
% THE INTEGER ARRAY CHEMAT IS USED IN THE TEST OF *
% THE MATERIAL MASS-BALANCE.                      *
%                                                 *
% THE FOLLOWING VARIABLES ARE GLOBAL FOR THIS     *
% PROCEDURE:                                      *
% INTEGER       NRE, MAXNR, MAXNT, MAXRL, NR, NT. *
% INTEGER ARRAY ETERM, ERK.                       *
% STRING  ARRAY RE, RNA.                          *
% REAL    ARRAY RK.                               *
% BOOLEAN       INPUTERROR.                       *
% FILE          ERRFIL, OUT.                      *
%                                                 *
% FOR THE DESCRIPTION OF THESE VARIABLES SEE THE  *
% BEGINNING OF THE PROGRAM                        *
%                                                 *
%*************************************************

BEGIN
%*************************************
% DECLARATION OF LOCAL VARIABLES.     *
```

```
%*****************************************
INTEGER                 % DECLARATION
  I,J,N,
  EN,LR,RN,
  LEV,IFAC,ESIZE;       % COUNTING AND WORKING
                        % VARIABLES

INTEGER ARRAY           % DECLARATION
  W[1:5];               % WORKING ARRAY

LONG INTEGER ARRAY      % DECLARATION
  CHEMAT[1:NRE*MAXNR];% USED FOR MASS BALANCE.

STRING                  % DECLARATION
  S,S1,S2,S3;           % USED FOR ANALYZATION
                        % OF REACTION EQUATION.

REAL                    % DECLARATION
  RFAC;                 % WORKING VARIABLE

BOOLEAN                 % DECLARATION
  SEM, EQSIGN,
  ERR, RFOUND;          % WORKING VARIABLES.

TRUTHSET                % DECLARATION
  NUM("0123456789"),
  SPECHAR("*[]+=;");    % USED FOR IDENTIFICATION
                        % OF DIGITS AND SPECIAL
                        % CHARACTERS.

  FORMAT
  MESS1("ERROR: NO '=' OR ';' OR FACTOR "
        "PRCEEDING '*'"/
        "OR MORE THAN 2 REACTANTS ON LEFT SIDE IN:"
        /A*),
  MESS2("REACTION SYSTEM IS NOT BALANCED - "
        "PROGRAM ABORTED");

  COMMENT IN ORDER TO BE SURE THAT THE REAL ARRAY RK
          IS OF SAME SIZE AS RE, IT IS RESIZED;
  RESIZE(RK[1,*],NRE,RETAIN);
  RESIZE(RK[2,*],NRE,RETAIN);

  COMMENT INITIALIZE VARIABLES;
  J:=MAXNR*NRE; MAXRL:=NR:=NT:=0;
  FOR I:=1 STEP 1 UNTIL J DO CHEMAT[I]:=0;

  FOR N:=1 STEP 1 UNTIL NRE DO
  BEGIN
    COMMENT
      ANALYSIS OF REACTION EQUATION NO: N.
      THE INTEGER N IS STORED IN VARIABLE
      EN (EQUATION NUMBER) AND THE STRING
      RE[N] IS COPIED TO STRING S WITH WITH
      THE IDENFICATION REMOVED. IF RE[N] IS
      EMPTY THE PROCEDURE CONTINUES BY ANA-
      LYZING THE NEXT REACTION EQUATION
      I.E. THE CONTENT OF STRING RE[N+1];

    EN:=DECIMAL(HEAD(DROP(RE[N],2),NOT ":"));
    S:=TAIL(TAIL(RE[N],NOT ":"),":");
    IF S NEQ EMPTY THEN
    BEGIN
      COMMENT THE STRING S IS NON-EMPTY,
```

```
        INITIALIZE SOME VARIABLES;

ERR:=SEM:=EQSIGN:=FALSE;
W[3]:=N;W[4]:=W[5]:=LR:=LEV:=0; IFAC:=1;;
S:=HEAD(S,NOT" "); S3:=EMPTY;
WHILE LENGTH(S)>0 DO
BEGIN
  COMMENT
    WHEN STRING S IS NON-EMPTY THE
    WHILE-STATEMENT IS CONTINUED;

  COMMENT
    SCAN THE STRING S FROM LEFT TO
    RIGHT, TAKE CHARACTERS DIFFERENT
    FROM THE CHARACTERS IN TRUTHSET
    SPECHAR AND STORE THEM IN S1.
    THEN CONCATENATE STRING S3 WITH
    S1. REMOVE THE CHARACTERS FOUND
    INCLUSIVE THE SPECIAL CHARACTERS
    FROM S. STORE THE SPECIAL CHARAC-
    TERS IN STRING S2;

  S1:=HEAD(S,NOT SPECHAR); S3:=S3 CAT S1;

  S:=DROP(S,LENGTH(S1));
  S2:=TAKE(S,1); S:=DROP(S,1);

  IF S2 = "[" THEN
  BEGIN
   LEV:=LEV+1; S3:=S3 CAT S2;
  END ELSE
  IF S2 = "]" THEN
  BEGIN
   LEV:=LEV-1; S3:=S3 CAT S2;
  END ELSE
  IF LEV = 0 THEN
  BEGIN
    IF S2="*" THEN
    BEGIN
      IF LENGTH(HEAD(S3,NUM))>0
         THEN IFAC:=DECIMAL(S3)
         ELSE ERR :=TRUE;
      IF IFAC>2 THEN ERR:=TRUE;
      S3:=EMPTY;
    END
    ELSE
    IF (S2="+" OR S2="=" OR S2=";") THEN
    BEGIN
      RFOUND:=FALSE;

      COMMENT
        TEST WHETHER THE STRING S3 ALREADY
        EXISTS IN STRING ARRAY RNA, I.E.
        THE REACTANT IN S3 IS ALREADY
        FOUND AND STORED IN RNA;
      FOR I:=1 STEP 1 WHILE I<=NR AND NOT RFOUND DO
      IF S3=RNA[I] THEN
      BEGIN
        RFOUND:=TRUE; RN:=I; S3:=EMPTY;
      END;

      IF NOT RFOUND THEN
      BEGIN
        COMMENT STORE NEW REACTANT;
```

```
  RN:=NR:=NR+1;
  IF NR>MAXNR THEN
  BEGIN
    COMMENT
      THE LENGTH OF ARRAYS RNA AND CHEMAT
      ARE TOO SMALL, RESIZE THEM;

    MAXNR:=MAXNR+5;
    RESIZE(RNA,MAXNR,RETAIN);
    RESIZE(CHEMAT,NRE*MAXNR,RETAIN);
  END;

  RNA[RN]:=S3; MAXRL:=MAX(MAXRL,LENGTH(S3));
  S3:=EMPTY;
END NOT RFOUND;

IF NOT EQSIGN THEN
BEGIN
  COMMENT
    LEFT SIDE OF A REACTION EQUATION.
    STORE REACTANT NO.: RN IN THE
    W-ARRAY AND CORRESPONDING STOI-
    CHIOMETRIC CONSTANT IN CHEMAT;

  I:=(RN-1)*NRE+N; W[2]:=-IFAC;
  CHEMAT[I]:=CHEMAT[I] - IFAC;
  IF IFAC=1 THEN
  BEGIN
    LR:=LR+1;
    IF LR<3 THEN
    BEGIN
      W[3+LR]:=RN;
      IF LR=2 AND W[4] = W[5] THEN
      BEGIN
        W[2]:= -2; LR:=1;
      END;
    END ELSE INPUTERROR:=TRUE;
  END
  ELSE
  IF IFAC=2 THEN
  BEGIN
    LR:=LR+1;
    IF LR<2 THEN W[4]:=W[5]:=RN
            ELSE INPUTERROR:=TRUE;
    IFAC:=1;
  END;
  IF S2="=" THEN
  BEGIN
    COMMENT THE SEPARATOR '=' IS FOUND,
            STORE ARRAY W;

    EQSIGN:=TRUE;
    IF LR<3 THEN
    FOR I:=1 STEP 1 UNTIL LR DO
    BEGIN
      W[1]:=W[3+I]; NT:=NT+1;
      IF NT>MAXNT THEN
      BEGIN
        COMMENT
          RESIZE ARRAY ETERM BY ADDING 5
          TO MAXNT FOR EACH OF THE NRE+1-N
          REMAINING REACTION EQUATIONS TO
          BE TRANSLATED, I.E. WE ASSUME
          THAT THE NUMBER OF TERMS ON THE
```

```
                    RIGHT SIDE OF THE DIFFERENTIAL
                    EQUATION SYSTEM ARE INCREASED
                    WITH 5 FOR EACH OF THE REMAIN-
                    ING REACTION EQUATIONS;

                  MAXNT:=MAXNT+5*(NRE+1-N);
                  ESIZE:=5*MAXNT+1;
                  RESIZE(ETERM,ESIZE,RETAIN);
                END;
                WRITE(ETERM[1+(NT-1)*5],5,W);
              END I;
            END S2
          END NOT EQSIGN
          ELSE
          IF EQSIGN THEN
          BEGIN
            COMMENT
            RIGHT SIDE OF A CHEMICAL REACTION EQUATION.
            STORE THE W-ARRAY AND THE STOICHIOMETRIC
            CONSTANT CORRESPONDING TO REACTANT NO: RN;

            I:=(RN-1)*NRE+N; CHEMAT[I]:=CHEMAT[I]+IFAC;
            W[1]:=RN; W[2]:=IFAC; IFAC:=1;
            NT:=NT+1;
            IF NT>MAXNT THEN
            BEGIN
              COMMENT RESIZE ARRAYS ETERM;
              MAXNT:=MAXNT+5*(NRE+1-N);
              ESIZE:=5*MAXNT+1;
              RESIZE(ETERM,ESIZE,RETAIN);
            END;
            WRITE(ETERM[1+(NT-1)*5],5,W);
            IF S2=";" THEN
            BEGIN
              COMMENT
                SEPARATOR 'SEMICOLON' IS FOUND,
                AND THE ANALYSIS OF REACTION
                EQUATION IS FINISHED. STORE
                THE REACTION EQUATION NUMBER
                AND THE RATE CONSTANT;
              SEM:=TRUE;
              IF LENGTH(S)>0 THEN
              BEGIN
                RK[1,N]:=EN; RK[2,N]:=DECIMAL(S);
                S:=DROP(S,LENGTH(S));
              END;
            END;
          END EQSIGN;
        END SEPARATOR ;
      END LEV EQ ZERO ELSE S3:=S3 CAT S2;
    END WHILESTATEMENT;

    IF NOT SEM OR  NOT EQSIGN OR ERR OR LEV NEQ 0 THEN
    BEGIN
      WRITE(ERRFIL,MESS1,LENGTH(RE[N]),RE[N]);
      INPUTERROR:=TRUE;
    END;
  END S NEQ EMPTY;
END N;

BEGIN
%*************************************************
% TEST FOR MASS BALANCE.                         *
% THE MATRIX OF STOICHIOMETRIC COEFFICIENTS FOR THE *
```

```
% REACTION SYSTEM ARE STORED IN THE ONE-DIMENSIONAL *
% ARRAY CHEMAT[1:MAXRN*NRE]. THE COEFFICIENTS (INTE-*
% GERS) FOR REACTANT NO: RN IS STORED IN ARRAY ELEM-*
% ENT: EN+(RN-1)*NRE, FOR EN = 1, 2, 3,.....NRE. *
% THE STOICHIOMETRIC COEFFICIENTS ARE NEGATIVE FOR  *
% REACTANTS ON THE LEFT SIDE OF A REACTION EQUATION *
% BUT POSITIVE FOR REACTANTS ON THE RIGHT SIDE I.E. *
% FOR THE RESULT OF THE REACTION. THE METHOD USED   *
% IS BASED ON AN ELEMENTARY GAUSSIAN ELIMINATION IN *
% INTEGER ARITHMETIC COMBINED WITH A STRAIGHTFORWARD*
% DIVISION WITH COMMON INTEGER DIVISORS IN ORDER TO *
% AVOID INTEGER OVERFLOWS.                          *
%***************************************************

%*************************************
%  DECLARATION OF LOCAL VARIABLES     *
%*************************************
INTEGER         % DECLARATION
  I,J,K,L,N,    % COUNTING VARIABLES
  J1,J2,I1,I2,  %
  M,M1,         % WORKING VARIABLES
  CMAX,         % HOLDS THE GREATEST VALUE
                % IN A ROW
  CDIV;         % COMMON DIVISOR.

BOOLEAN         % DECLARATION
  BALANCE,
  BFAC;         % WORKING VARIABLES.

  BALANCE:=TRUE;
  FOR N:=1 STEP 1 WHILE N LEQ MIN(NRE,NR) AND BALANCE DO
  BEGIN
    COMMENT
      THIS DO-STATEMENT IS INTERRUPTED WITH AN
      ERROR MESSAGE IF THE BOOLEAN 'BALANCE'
      SET TO 'FALSE' DURING THIS TEST, I.E. IF
      THE ROW NUMBER N DO'NT HAVE AT LEAST TWO
      NON-ZERO ELEMENTS OF OPPOSITE SIGNS;

    COMMENT
      FIND THE GREATEST ELEMENT IN COLUMN N;
    M:=0;  L:=(N-1)*NRE;
    FOR I:=N STEP 1 UNTIL NRE DO
    IF ABS(CHEMAT[I+L])>M THEN
    BEGIN
      M:=ABS(CHEMAT[I+L]); K:=I;
    END;
    IF M>0 THEN
    BEGIN
      IF K NEQ N THEN
      BEGIN
        COMMENT EXCHANGE ROWS IN CHEMAT;
        FOR J:=N STEP 1 UNTIL NR DO
        BEGIN
          L:=(J-1)*NRE;
          I:=CHEMAT[N+L];
          CHEMAT[N+L]:=CHEMAT[K+L];
          CHEMAT[K+L]:=I;
        END;
      END K NEQ N;

      J1:=J2:=0;
      FOR J:=1 STEP 1 WHILE (J1=0 OR J2=0) AND J<=NR DO
      BEGIN
```

```
COMMENT
  TWO ELEMENTS DIFFERENT FROM ZERO AND OF
  OPPOSITE SIGN MUST BE FOUND IN ROW NUM-
  BER N IN THE MATRIX 'CHEMAT' I.E. A RE-
  ACTION EQUATION MUST HAVE REACTANTS ON
  BOTH SIDE;

M:=CHEMAT[N+(J-1)*NRE];
IF J1=0 AND M NEQ 0 THEN
BEGIN
  COMMENT
    FIRST ELEMENT DIFFERENT FROM ZERO IS
    FOUND AND STORED IN VARIABLE M. ITS
    NUMBER IS K AND ITS SIGN IS J1;

  K:=J; J1:=SIGN(M);
END ELSE
IF J1 NEQ 0 AND J2=0 AND SIGN(M)=-J1 THEN
BEGIN
  COMMENT
    THE NEXT ELEMENT DIFFERENT FROM ZERO
    AND OF OPPOSITE SIGN IS FOUND. ITS
    SIGN IS J2;
  J2:=SIGN(M);

  I2:=CHEMAT[N+(K-1)*NRE];
  FOR I:=N+1 STEP 1 UNTIL NRE DO
  BEGIN
    COMMENT
      I2 IS ELEMENT NUMBER K IN ROW N,
      I1 IS ELEMENT NUMBER K IN ROW I
      (=N+1,...NRE).
      CMAX IS INITIALIZED TO ZERO AND
      IS LATER UPDATED TO THE LARGEST
      ABSOLUTE VALUE FOUND IN ROW I
      DURING THE GAUSSIAN ELIMINATION
      IN INTEGER ARITHMETIC;

    I1:=CHEMAT[I+(K-1)*NRE];  CMAX:=0;
    FOR L:=1 STEP 1 UNTIL NR DO
    BEGIN
      COMMENT
        ELEMENTS IN ROW I ARE CHANGED.
        ALL ELEMENT NUMBERS FROM L=1
        TO L=K ARE SET TO ZERO. M AND
        M1 ARE USED AS TEMPORARY VAR-
        ABLES. CMAX IS UPDATED;

      M:=(L-1)*NRE; M1:=CHEMAT[I+M];
      M1:=M1*I2 - CHEMAT[N+M]*I1;
      CHEMAT[I+M]:=M1;
      CMAX:=MAX(CMAX,ABS(M1));
    END;

    CDIV:=1;
    FOR CDIV:=CDIV+1 WHILE CDIV <= CMAX DO
    BEGIN
      COMMENT
        AFTER CHANGING ALL ELEMENTS IN
        ROW I THESE ARE REDUCED BY DI-
        VISION WITH COMMON DIVISORS IN
        ORDER TO AVOID INTEGER OVERFLOW
        IN GAUSSIAN ELIMINATION. BEFORE
        DIVISION A MODULO OPERATION IS
```

```
                DONE TO BE SURE THAT THE DIVISION
                IS POSSIBLE. THE COMMON DIVISOR
                'CDIV' IS INCREASED WITH 1 AFTER
                EVERY SUCCESSFUL DIVISION. THE
                FIRST VALUE USED FOR CDIV IS 2;

              BFAC:=TRUE;
              WHILE BFAC DO
              BEGIN
                FOR L:=1 STEP 1 UNTIL NR DO
                BEGIN
                  COMMENT CHECK FOR DIVISION;
                  M:=CHEMAT[I+(L-1)*NRE];
                  IF ABS(M) MOD CDIV NEQ 0 THEN BFAC:=FALSE;
                END L;
                IF BFAC THEN
                BEGIN
                  COMMENT DO THE DIVISION;
                  FOR L:=1 STEP 1 UNTIL NR DO
                  BEGIN
                    M:=(L-1)*NRE;
                    CHEMAT[I+M]:=CHEMAT[I+M] DIV CDIV;
                  END L;
                  CMAX:=CMAX DIV CDIV;
                END BFAC;
              END WHILE BFAC;
            END CDIV;
          END I;
        END J1 NEQ 0 ETC;
      END J;
      IF J1 NEQ 0 AND J2=0 THEN
      BEGIN
        BALANCE:=FALSE; INPUTERROR:=TRUE;
        WRITE(ERRFIL,MESS2);
      END;
    END M GT 0;
  END N;
END MASS BALANCE;
END CHEMTRANS;

PROCEDURE TRANSPLOT;
%*************************************************
%                                                *
% PROCEDURE TRANSPLOT DO A TRANSLATION OF THE    *
% PLOT EXPRESSIONS WHICH ARE ARITHMETIC EXPRES-  *
% SIONS, THE ONLY OPERATION NOT ALLOWED IS DIVI- *
% SION. THE TRANSLATION IS DONE IN A SIMILAR WAY *
% AS THE TRANSLATION OF THE CHEMICAL REACTION E- *
% QUATIONS.                                      *
% A PLOT EXPRESSION IS CONSIDERED AS A SYMBOL    *
% STRING CONTAINING LETTERS, DIGITS AND SPECIAL  *
% SYMBOLS. BY SCANNING THE PLOT EXPRESION FROM   *
% LEFT TO RIGHT THE PROCEDURE IDENTIFIES:        *
% REACTANTS, FACTORS TO REACTANTS AND ADDITIVE   *
% CONSTANT.                                      *
% THE FORM OF A PLOT EXPRESSION IS:              *
% PE NN: C1*R1 + C2*R2 + .... + A;               *
%                                                *
% WHERE PE NN: IS THE IDENTIFICATION OF THE PLOT *
% EXPRESSION (PE = PLOT EXPRESSION, NN IS A NUM- *
% BER), C1,C2,.. ARE REAL CONSTANTS, R1, R2,...  *
% ARE THE REACTANTS. A IS AN ADDITIVE CONSTANT.  *
%                                                *
% THE TOTAL NUMBER OF TERMS (APART FROM THE AD-  *
```

```
% DITIVE CONSTANTS) IN THE PLOTEXPRESSIONS IS PT *
% AND IT MUST BE EQUAL TO OR SMALLER THAN MAXPT  *
% WHICH DETERMINES THE SIZE OF THE ARRAYS: PEN,  *
% PRN AND PFAC. (SEE THE DECLARATIONS OF ARRAYS) *
%                                                *
% FOR EACH TERM APPEARING IN THE SYSTEM OF PLOT  *
% EXPRESSIONS THE FOLLOWING INFORMATION ARE STO- *
% RED IN THE ORDER THEY ARE FOUND:               *
%                                                *
% 1: THE PLOT EXPRESSION NUMBER FOR THE TERM IS  *
%    STORED IN INTEGER ARRAY PEN.                *
% 2: THE ADDRESS FOR THE REACTANT (IN ARRAY RNA) *
%    IS STORED IN INTEGER ARRAY PRN.             *
% 3: THE FACTOR TO THE REACTANT IN TERM IS STO-  *
%    RED IN REAL ARRAY PFAC.                     *
%                                                *
% THE INFORMATIONS STORED IN THESE ARRAYS ARE    *
% USED IN THE PROCEDURE CHEMPLOT FOR PLOTTING    *
% THE RESULTS OF THE COMPUTATION.                *
%                                                *
%                                                *
% THE FOLLOWING VARABLES ARE GLOBAL TO THE       *
% PROCEDURE:                                     *
% INTEGER       NPE, NR, PT, MAXPT.              *
% STRING  ARRAY PE, RNA.                         *
% INTEGER ARRAY PEN, PRN.                        *
% REAL    ARRAY PFAC, ADDCON.                    *
% BOOLEAN       INPUTERROR.                      *
% FILE          ERRFIL                           *
%                                                *
%************************************************

BEGIN

%************************************************
% DECLARATION  OF LOCAL VARIABLES                *
%************************************************

INTEGER                 % DECLARATION
  EN,LEV,
  K ,RN ;               % COUNTING AND WORKING VARIABLES.

REAL                    % DECLARATION
  RFAC;                 % WORKING VARIABLE.

STRING                  % DECLARATION
  S ,S0,S1,S2,
  S3,S4,S5,S6,
  S7,S8,NU;             % WORKING VARIABLES USED IN THE
                        % TRANSLATION OF PLOT EXPRESSION.

BOOLEAN                 % DECLARATION
 SEM,RFOUND,
 PROD;                  % WORKING VARIABLES.

TRUTHSET                % DECLARATION
  DIGIT("0123456789"),
  SPEC1("[]+-;"),
  SPEC2("+-*;"),
  SIGN("+-"),
  EXPART("@E");          % USED IN THE TRANSLATION
                         % PROCESS.

%************************************************
```

```
% THE FOLLOWING FORMATS ARE ERROR MESSAGES      *
% WHICH ARE PRINTED IF INPUT ERRORS ARE DE-     *
% TECTED IN THE PLOT EXPRESION DURING THE       *
% TRANSLATION.                                  *
%***********************************************

FORMAT
MESS1("ERROR: PLOTEXPR. WRONG RADIX:"/
      "-->",A*),
MESS2("ERROR: PLOTEXPR. NO EXPONENT NUMBER:"/
      "-->",A*),
MESS3("ERROR: PLOTEXPR. NO '+','-','*' OR ';' :"/
      "-->",A*),
MESS4("ERROR: PLOTEXPR. WRONG REACTANT"/
      "-->",A*),
MESS5("ERROR: PLOTEXPR. NO MATCHING BRACKETS"/
      "-->",A*);

  COMMENT
    RESIZE REAL ARRAY ADDCON TO ADDCON[1:NPE] BEFORE
    START OF TRANSLATION OF PLOT EXPRESSION;
  RESIZE(ADDCON,NPE,RETAIN);

  COMMENT INITIALIZE ARRAY ADDCON TO ZERO;
  FOR EN:=1 STEP 1 UNTIL NPE DO ADDCON[EN]:=0;

  FOR EN:=1 STEP 1 UNTIL NPE DO
  BEGIN
    S:=TAIL(TAIL(PE[EN],NOT ":"),":");
    IF S NEQ EMPTY THEN
    BEGIN
      COMMENT SKIP THE PLOT EXPRESSION IF IT IS EMPTY;
      LEV:=0; SEM:=PROD:=FALSE; SO:=EMPTY;
      WHILE LENGTH(S)>0 DO
      BEGIN
        IF LEV = 0 AND NOT PROD THEN
        BEGIN
          COMMENT
            SCAN STRING S WHENCE THE CHARACTERS
            CAN BE ACCEPTED AS PART OF A REAL
            (INTEGER) NUMBER;
          S1:=HEAD(S,SIGN);    S:=DROP(S,LENGTH(S1));
          S2:=HEAD(S,DIGIT);   S:=DROP(S,LENGTH(S2));
          S3:=HEAD(S,".");     S:=DROP(S,LENGTH(S3));
          S4:=HEAD(S,DIGIT);   S:=DROP(S,LENGTH(S4));
          S5:=HEAD(S,EXPART);  S:=DROP(S,LENGTH(S5));
          IF LENGTH(S5)>0 THEN
          BEGIN
            COMMENT
              THE RADIX (@ OR E) FOR THE EXPONEN-
              TIAL PART OF A REAL NUMBER IS FOUND;
            IF LENGTH(S5)>1 THEN
            BEGIN
              INPUTERROR:=TRUE;
              WRITE(ERRFIL,MESS1,LENGTH(PE[EN]),PE[EN]);
            END;

            COMMENT FIND EXPONENT;
            S6:=HEAD(S,SIGN);  S:=DROP(S,LENGTH(S6));
            S7:=HEAD(S,DIGIT); S:=DROP(S,LENGTH(S7));
            IF LENGTH(S7)=0 THEN
            BEGIN
              COMMENT NO EXPONENT NUMBER IS FOUND,
                      PRINT ERROR MESSAGE;
```

```
      INPUTERROR:=TRUE;
      WRITE(ERRFIL,MESS2,LENGTH(PE[EN]),PE[EN]);
    END
  END ELSE BEGIN S6:=S7:=EMPTY END;

  COMMENT
    CONCATENATE THE STRINGS S2,....S7 TO A NUMBER;
  NU:=S2 CAT S3 CAT S4 CAT S5 CAT S6 CAT S7;
  IF LENGTH(NU)>0 THEN
  BEGIN
    COMMENT
      ANALYZE THE NUMBER WHETHER IS IS A FAC-
      TOR TO A REACTANT OR AN ADDITIVE CON-
      STANT IN THE PLOT EXPRESSION;

    S8:=HEAD(S,SPEC2);
    IF LENGTH(S8) > 0 THEN
    BEGIN
      IF S8="*" THEN
      BEGIN
        RFAC:=DECIMAL(S1 CAT NU);
        PROD:=TRUE;
        S:=DROP(S,LENGTH(S8));
      END ELSE ADDCON[EN]:=DECIMAL(S1 CAT NU);

      IF S8 = ";" THEN
      BEGIN
        SEM:=TRUE; S:=DROP(S,LENGTH(S8));
      END;
    END ELSE
    BEGIN
      INPUTERROR:=TRUE;
      WRITE(ERRFIL,MESS3,LENGTH(PE[EN]),PE[EN]);
    END;
  END ELSE
  BEGIN
    COMMENT
     NO NUMBER FOUND. IF THE FOLLOWING SEPA-
     RATOR IS A SEMICOLON THE PLOT EXPRES-
     SION IS FINISHED, IF IT IS A PLUS, OR A
     MINUS, THEN RFAC IS SET TO 1. THE BOO-
     LEAN 'PROD' IS SET TO TRUE INDICATING
     THAT A NUMBER (FACTOR OR CONSTANT) IS
     FOUND;

    IF S1 = ";" THEN
    BEGIN
      SEM:=TRUE; S:=DROP(S,LENGTH(S1));
    END ELSE
    BEGIN
      RFAC:= IF S1="-" THEN -1 ELSE 1;
      PROD:=TRUE;
    END;
  END;
END ELSE
BEGIN
  S1:=HEAD(S,NOT SPEC1); S0:=S0 CAT S1;
  S :=DROP(S,LENGTH(S1));
  COMMENT THE FIRST CHARACTER IN STRING S IS
          A MEMBER OF THE TRUTHSET SPEC1;
  S2:=TAKE(S,MIN(1,LENGTH(S)));
  IF S2="[" THEN
  BEGIN
    COMMENT A BRACKET IS FOUND, INCREASE LEV BY 1;
```

```
            LEV:=LEV+1; SO:=SO CAT S2; S:=DROP(S,1);
          END ELSE
          IF S2="]" THEN
          BEGIN
            COMMENT A BRACKET IS FOUND, DECREASE LEV BY -1;
            LEV:=LEV-1; SO:=SO CAT S2; S:=DROP(S,1);
          END ELSE
          IF (S2="+" OR S2="-" OR S2=";") AND LEV=0 THEN
          BEGIN
            COMMENT CHECK WHETHER THE STRING SO IS THE NAME
                    OF A REACTANT;

            RFOUND:=FALSE;
            FOR RN:=1 STEP 1 WHILE RN<=NR AND NOT RFOUND DO
            IF SO=RNA[RN] THEN
            BEGIN
              COMMENT
                A REACTANT NAME IS FOUND, STORE RFAC,
                EN AND REACTANT NUMBER RN;
              PT:=PT+1; RFOUND:=TRUE; SO:=EMPTY;
              IF PT>MAXPT THEN
              BEGIN
                MAXPT:=MAXPT+(NPE+1-EN)*10;
                RESIZE(PFAC,MAXPT,RETAIN);
                RESIZE(PEN,MAXPT,RETAIN);
                RESIZE(PRN,MAXPT,RETAIN);
              END;
              PFAC[PT]:=RFAC;  PEN[PT]:=EN; PRN[PT]:=RN;
            END;
            IF NOT RFOUND THEN
            BEGIN
              INPUTERROR:=TRUE;
              WRITE(ERRFIL,MESS4,LENGTH(PE[EN]),PE[EN]);
            END;
            SO:=EMPTY; PROD:=FALSE;
            IF S2 = ";" THEN
            BEGIN
              SEM:=TRUE; S:=DROP(S,1);
            END;
          END ELSE
          BEGIN
            SO:=SO CAT S2; S:=DROP(S,1);
          END;
        END ELSESTATEMENT;
      END WHILESTATEMENT;
      IF LEV NEQ 0 THEN
      BEGIN
        INPUTERROR:=TRUE;
        WRITE(ERRFIL,MESS5,LENGTH(PE[EN]),PE[EN]);
      END;
      IF NOT SEM THEN
      BEGIN
        INPUTERROR:=TRUE;
        WRITE(ERRFIL,MESS3,LENGTH(PE[EN]),PE[EN]);
      END;
    END S NEQ EMPTY;
  END EN;
END TRANSPLOT;

PROCEDURE STOREG;
%************************************************
% THE PROCEDURE TAKES THE G-VALUES FROM THE      *
% STRING ARRAY GV, CONTROLS WHETHER THE REACTANT *
% EXIXTS IN THE REACTION SYSTEM AND FINALLY      *
```

```
% STORES THE VALUE IN REAL ARRAY RG.                    *
%                                                       *
% THE FOLLOWING VARIABLES ARE GLOBAL TO THE             *
% PROCEDURE:                                            *
% INTEGER       NR, NGV                                 *
% STRING ARRAY  RNA, GV                                 *
% REAL   ARRAY  RG                                      *
% BOOLEAN       INPUTERROR                              *
% FILE          ERRFIL                                  *
%*******************************************************
BEGIN

%*******************************************************
% DECLARATION OF LOCAL VARIABLES.                       *
%*******************************************************

INTEGER         % DECLARATION
  I, N;         % COUNTING VARIABLES.

BOOLEAN         % DECLARATION
  FOUND;        % WORKING VARIABLE

STRING          % DECLARATION
  S    ;        % WORKING VARIABLE.


%*******************************************************
% THE FOLLOWING FORMAT ARE ERROR MESSAGES WHICH         *
% ARE PRINTED IF INPUT ERRORS ARE DETECTED              *
%*******************************************************

FORMAT
MESS1("ERROR:WRONG NAME IN INPUTRECORD FOR G VALUE:"/A*);
  COMMENT RESIZE REAL ARRAY RG TO RG[1:NR];
  RESIZE(RG,NR,RETAIN);

  FOR I:=1 STEP 1 UNTIL NR DO RG[I]:=0;
  FOR I:=1 STEP 1 UNTIL NGV DO
  BEGIN
    FOUND:=FALSE; S:=HEAD(DROP(GV[I],2),NOT")");
    FOR N:=1 STEP 1 WHILE N<=NR AND NOT FOUND DO
    BEGIN
      IF S=RNA[N] THEN
      BEGIN
        FOUND:=TRUE;
        RG[N]:=DECIMAL(DROP(TAIL(GV[I],NOT")"),2));
      END;
    END;
    IF NOT FOUND THEN
    BEGIN
      WRITE(ERRFIL,MESS1,LENGTH(GV[I]),GV[I]);
      INPUTERROR:=TRUE;
    END;
  END I;
END STOREG;

PROCEDURE STOREK;
%*******************************************************
% THE PROCEDURE TAKES THE RATE CONSTANTS FROM           *
% THE STRING ARRAY RC, CONTROLS WHETHER THE             *
% REACTANT EXISTS IN THE REACTION SYSTEM AND            *
% FINALLY STORES THE VALUE IN REAL ARRAY RK.            *
%                                                       *
% THE FOLLOWING VARIABLES ARE GLOBAL TO THE             *
```

```
% PROCEDURE:                                          *
% INTEGER      NRE, NRC                               *
% STRING ARRAY RC                                     *
% REAL   ARRAY RK                                     *
% BOOLEAN      INPUTERROR                             *
% FILE         ERRFIL                                 *
%*****************************************************

BEGIN
%*****************************************************
% DECLARATION OF LOCAL VARIABLES                      *
%*****************************************************

INTEGER      % DECLARATION
  I,N,EN;    % COUNTING VARIABLES

BOOLEAN      % DECLARATION
  FOUND ;    % WORKING VARIABLE


%*****************************************************
% ERROR MESSAGE FOR INPUT ERRORS DETECTED             *
%*****************************************************

FORMAT
MESS2("ERROR: WRONG IDENTIFICATION FOR K VALUE:"/A*);
  FOR I:=1 STEP 1 UNTIL NRC DO
  BEGIN
    FOUND := FALSE;
    EN:=DECIMAL(HEAD(DROP(RC[I],1),NOT"="));
    FOR N:=1 STEP 1 WHILE N <= NRE AND NOT FOUND DO
    BEGIN
      IF EN = RK[1,N] THEN
      BEGIN
        FOUND:=TRUE;
        RK[2,N]:=DECIMAL(TAIL(TAIL(RC[I],NOT "="),"="));
      END
    END;
    IF NOT FOUND THEN
    BEGIN
      WRITE(ERRFIL,MESS2,LENGTH(RC[I]),RC[I]);
      INPUTERROR:=TRUE;
    END;
  END I;
END STOREK;

PROCEDURE STORECON;
%*****************************************************
% THE PROCEDURE TAKES THE INPUT VALUES FOR THE        *
% START CONCENTRATIONS OF REACTANTS FROM THE          *
% STRING ARRAY CON AND CONTROLS WHETHER THE REAC-     *
% TANT EXISTS IN THE REACTION SYSTEM. FINALLY IT      *
% STORES THE VALUE IN REAL ARRAY RCON.                *
%                                                     *
% INTEGER       NR, NCON                              *
% STRING ARRAY  RNA, CON                              *
% REAL   ARRAY  RCON                                  *
% BOOLEAN       INPUTERROR                            *
% FILE          ERRFIL                                *
%*****************************************************

BEGIN
%*****************************************************
% DECLARATION  OF LOCAL VARIABLES                     *
```

```
%************************************************

INTEGER             % DECLARATION
  I, N;             % COUNTING VARIABLES

STRING              % DECLARATION
  S;                % WORKING VARIABLE

BOOLEAN             % DECLARATION
  FOUND;            % WORKING VARIABLE

%************************************************
% ERROR MESSAGE FOR INPUT ERROR DETECTED        *
%************************************************

FORMAT
MESS1("ERROR:WRONG NAME IN INPUTRECORD FOR CON VALUE:"/A*");

  COMMENT RESIZE REAL ARRAY RCON TO RCON[1:NR];
  RESIZE(RCON,NR,RETAIN);
  RESIZE(RG,NR,RETAIN);

  FOR I:=1 STEP 1 UNTIL NR DO RCON[I]:=0;
  FOR I:=1 STEP 1 UNTIL NCON DO
  BEGIN
    FOUND:=FALSE; S:=HEAD(DROP(CON[I],4),NOT")");
    FOR N:=1 STEP 1 WHILE N<=NR AND NOT FOUND DO
    BEGIN
      IF S=RNA[N] THEN
      BEGIN
        FOUND:=TRUE;
        RCON[N]:=DECIMAL(DROP(TAIL(CON[I],NOT ")"),2));
      END;
    END;
    IF NOT FOUND THEN
    BEGIN
      WRITE(ERRFIL,MESS1,LENGTH(CON[I]),CON[I]);
      INPUTERROR:=TRUE;
    END;
  END I;
END STORECON;

PROCEDURE STOREXPD(XPD);
%************************************************
% THE PROCEDURE TAKES THE INPUT VALUES FOR EXPE- *
% RIMENTAL DATA FROM STRING ARRAY XPDATA AND     *
% STORES THEM IN REAL ARRAY XPD SUCH THAT THE    *
% EXPERIMENTAL POINT NO: I BELONGING TO PLOT     *
% EXPRESSION NO: N IS STORED IN                  *
% XPD[1,N,I] FOR THE X-COORDINATE, AND           *
% XPD[2,N,I] FOR THE Y-COORDINATE.               *
%                                                *
% FOLLOWING VARIABLES ARE GLOBAL:                *
% INTEGER       NPE                              *
% STRING ARRAY  XPDATA                           *
%************************************************

%************************************************
% SPECIFICATION OF PROCEDURE PARAMETER          *
%************************************************

REAL ARRAY
  XPD[*,*,*];
```

```
BEGIN
%************************************************
% DECLARATION  OF LOCAL VARIABLES                *
%************************************************

INTEGER       % DECLARATION
  N, I;       % COUNTING VARIABLES

STRING        % DECLARATION
  S, SN;      % WORKING VARIABLES

  FOR N:=1 STEP 1 UNTIL NPE DO
  BEGIN
    I:=0; SN:=XPDATA[N];
    WHILE LENGTH(SN)>0 DO
    BEGIN
      S:=HEAD(SN,NOT ";"); I:=I+1;
      XPD[1,N,I]:=DECIMAL(HEAD(S,NOT ","));
      XPD[2,N,I]:=DECIMAL(TAIL(TAIL(S,NOT ","),","));
      SN:=DROP(SN,LENGTH(S)+1);
    END;
  END N;
END STOREXPD;

PROCEDURE STARTVAL;
%************************************************
% OUTPUT OF G-VALUES AND START CONCENTRATIONS    *
% FOR THE REACTANTS.                            *
%                                               *
% FOLLOWING VARIABLES ARE GLOBAL:               *
% INTEGER       MAXRL, NR                       *
% STRING ARRAY  RNA                             *
% REAL   ARRAY  RG, RCON                        *
% FILE          OUT                             *
% BOOLEAN       NEWGV, NEWCON                   *
%************************************************

BEGIN
%************************************************
% DECLARATION OF LOCAL VARIABLES                 *
%************************************************

INTEGER       % DECLARATION
  J,K,N;      % COUNTING VARIABLES

  IF NEWGV AND NEWCON THEN
  BEGIN
    FOR N:=1 STEP 1 UNTIL NR DO
    BEGIN
      J:=LENGTH(RNA[N]); K:=MAXRL-J;
      WRITE(OUT,<"G(",A*,")",X*,"=",F6.3,X10,
                "CON(",A*,")",X*,"=",E10.3>,
            J,RNA[N],K,RG[N],J,RNA[N],K,RCON[N]);
    END N;
  END ELSE
  IF NEWGV THEN
  BEGIN
    WRITE(OUT,</"G - VALUES"/>);
     FOR N:=1 STEP 1 UNTIL NR DO
    BEGIN
    J:=LENGTH(RNA[N]); K:=MAXRL-J;
    WRITE(OUT,<"G(",A*,")",X*,"=",F6.3>,
              J,RNA[N],K,RG[N]);
    END;
```

```
  END ELSE
  IF NEWCON THEN
  BEGIN
    WRITE(OUT,</"START-CONCENTRATIONS"/>);
    FOR N:=1 STEP 1 UNTIL NR DO
    BEGIN
     J:=LENGTH(RNA[N]); K:=MAXRL-J;
     WRITE(OUT,<"CON(",A*,")",X*,"=",E10.3>,
               J,RNA[N],K,RCON[N]);
    END;
  END;
END STARTVAL;

PROCEDURE PRINTDIFFEQ;
%*************************************************
% THE PROCEDURE MAKES AN OUTPUT OF THE DIFFEREN- *
% TIAL EQUATION SYSTEM AS IT IS TRANSLATED FROM  *
% THE CHEMICAL REACTION SYSTEM. THE ZERO ORDER   *
% TERMS DEPENDING OF THE G-VALUES ARE ADDED IF   *
% THE INPUT VALUES OF G-VALUES ARE DIFFERENT     *
% FROM ZERO.                                     *
%                                                *
% FOLLOWING VARIABLES ARE GLOBAL:                *
% INTEGER         NRE,NR,NT,MAXRL,LINELE         *
% STRING  ARRAY   RNA                            *
% INTEGER ARRAY   ETERM, RG                      *
% REAL    ARRAY   RK                             *
% FILE            OUT                            *
%*************************************************

BEGIN
%*************************************************
% DECLARATION OF LOCAL VARIABLES                 *
%*************************************************

INTEGER          % DECLARATION
  N,I,J,         % COUNTING VARIABLES
  DK,            % NUMBER OF CHARACTERS RESERVED FOR
                 % THE NAME OF RATE CONSTANT.
  STOIC,         % USED FOR RESERVATION OF PLACE FOR
                 % STOICHIOMETRIC CONSTANT.
  TERMLENGTH,    % THE LENGTH OF A TERM ON THE RIGHT
                 % HAND SID OF A DIFFERENTIAL EQUA-
                 % TION.
  RL1,           % LENGTH OF THE FIRST REACTANT AND
  RL2,           % LENGTH OF THE SECOND REACTANT IN
                 % A TERM (WHICH IS A PRODUCT OF
                 % STOICHIOMETRIC CONSTANT AND THE
                 % TWO CONCENTRATIONS AND THE RATE
                 % CONSTANT).
  I1;            % WORKING VARIABLE

STRING           % DECLARATION
  S;             % WORKING VARIABLE.

  COMMENT
    THE NEXT 4 STATEMENTS ADJUST PARAMETER LINELE
    SUCH THAT IT CAN CONTAIN A LINE FOR PRINTING;

  STOIC:=0;
  FOR I:=1 STEP 1 UNTIL NT DO
  IF ABS(ETERM[(I-1)*5+2])>=STOIC THEN
      STOIC:=ABS(ETERM[(I-1)*5+2]);
```

```
COMMENT
  PLACE THE HEAD OF THE TABLE OF THE DIFFEREN-
  TIAL EQUATION SYSTEM AND PRINT;
N:=19+3*MAXRL+LENGTH(STRING(NRE,*))
            +LENGTH(STRING(STOIC,*));
IF N>LINELE THEN LINELE:=N;
N:= IF LINELE-28<0 THEN 0
                   ELSE (LINELE-28) DIV 2;
WRITE(OUT,</X*,"DIFFERENTIAL-EQUATION SYSTEM">,N);
WRITE(OUT,<X*,28("=")>,N);

FOR N:=1 STEP 1 UNTIL NR DO
BEGIN
  WRITE(OUT,</>);
  S:="D[" CAT RNA[N] CAT "]/DT"
          CAT REPEAT(" ",MAXRL-LENGTH(RNA[N]))
          CAT "=";
  J:=LINELE-7-MAXRL;
  FOR I:=1 STEP 1 UNTIL NT DO
  IF ETERM[(I-1)*5+1]=N THEN
  BEGIN
    I1:=(I-1)*5;
    DK:=LENGTH(STRING(ETERM[I1+3],*));
    STOIC:=IF ABS(ETERM[I1+2]) NEQ 1
           THEN LENGTH(STRING(ABS(ETERM[I1+2]),*))
           ELSE 0;
    TERMLENGTH:=4+DK+
                 (IF STOIC=0 THEN 0 ELSE STOIC+1);
    RL1:=LENGTH(RNA[ETERM[I1+4]]);
    IF RL1>0 THEN TERMLENGTH:=TERMLENGTH+RL1+3;
    RL2:=LENGTH(RNA[ETERM[I1+5]]);
    IF RL2>0 THEN TERMLENGTH:=TERMLENGTH+RL2+3;
    IF J-TERMLENGTH<0 THEN
    BEGIN
      WRITE(OUT,<A*>,LENGTH(S),S);
      J:=LINELE-7-MAXRL;
      S:=REPEAT(" ",MAXRL+7);
    END;
    IF ETERM[I1+2]<0 THEN S:=S CAT " - "
                     ELSE S:=S CAT " + ";
    IF ABS(ETERM[I1+2]) NEQ 1 THEN
    S:=S CAT STRING(ABS(ETERM[I1+2]),STOIC)
         CAT "*";
    S:=S CAT "K"
         CAT STRING(RK[1,ETERM[I1+3]],DK);
    IF RL1>0 THEN S:=S CAT "*["
                       CAT RNA[ETERM[I1+4]]
                       CAT "]";
    IF RL2>0 THEN S:=S CAT "*["
                       CAT RNA[ETERM[I1+5]]
                       CAT "]";
    J:=J-TERMLENGTH;
  END I;
  IF RG[N] NEQ 0 THEN
  BEGIN
    TERMLENGTH:=17+LENGTH(RNA[N]);
    IF J-TERMLENGTH<0 THEN
    BEGIN
      WRITE(OUT,<A*>,LENGTH(S),S);
      S:= REPEAT(" ",MAXRL+7);
    END;
    S:=S CAT " + G("
         CAT RNA[N]
         CAT ")*CONST*DOSE";
```

```
    END;
    WRITE(OUT,<A*>,LENGTH(S),S);
  END N;
  WRITE(OUT,</"THE VALUES OF THE RATE CONSTANTS"/>);
  DK:=LENGTH(STRING(NRE,*));
  J :=LINELE DIV (DK+15);
  I1:=NRE DIV J;
  IF I1*J < NRE THEN I1:=I1+1;
  FOR I:=1 STEP 1 UNTIL I1 DO
  WRITE(OUT,<*("K",I*," =",E10.3,X2)>,MIN(J,NRE-(I-1)*J),
  FOR N:=(I-1)*J+1 STEP 1 WHILE N<=I*J AND N<=NRE DO
    [DK,RK[1,N],RK[2,N]]);
  WRITE(OUT,</>);
END PRINTDIFFEQ;

PROCEDURE RESULTS(PRLINES,OUTARR);
%************************************************
% THE PROCEDURE PRINTS THE RESULTS OF THE SIMU- *
% LATION BY ARRANGING THE RESULTS IN A TABLE.   *
% THE TIME IS PRINTED IN THE FIRST COLUMN AND   *
% THE FOLLOWING COLUMNS CONTAIN THE CONCENTRA-  *
% TION OF THE REACTANTS CORRESPONDING TO THE    *
% TIME PRINTED IN THE FIRST COLUMN.             *
% THE FOLLOWING VARIABLES ARE GLOBAL:           *
% INTEGER       NR, MAXRL, LINELE, DEC          *
% STRING ARRAY  RNA                             *
% REAL   ARRAY  OUTARR                          *
% FILE          OUT                             *
%************************************************

%************************************************
% SPECIFICATION OF PROCEDURE PARAMETERS         *
%************************************************

INTEGER
  PRLINES;       % THE ACTUAL NUMBER OF PRIN-
                 % TED LINES IN THE RESULT
                 % TABLE OUTARR.

REAL ARRAY
  OUTARR[*,*];  % USED FOR RESULT TABLE.

BEGIN
%************************************************
% DECLARATION OF LOCAL VARIABLES                *
%************************************************

INTEGER          % DECLARATION
  C, I, N,       % COUNTING VARIABLES
  COL,I1,I2,
  L ,L1,L2,
  R1,R2,
  WIDTH;         % WORKING VARIABLES.

STRING           % DECLARATION
  S;             % VARIABLE USED FOR COLLECTING ALL
                 % CHARACTERS IN THE LINE TO BE
                 % PRINTED.


  WIDTH:=MAX(MAXRL+2,9+DEC,9);
  L2:=(WIDTH-7-DEC) DIV 2;
  R2:=WIDTH-7-DEC-L2;
  COL:=(LINELE-12) DIV WIDTH;
```

```
  FOR C:=1 STEP COL UNTIL NR DO
  BEGIN
    I1:=C; I2:=MIN(NR,C+COL-1);

    COMMENT PRINT HEAD OF TABLE;
    S:="TIME        ";
    FOR I:=I1 STEP 1 UNTIL I2 DO
    BEGIN
      L:= LENGTH(RNA[I]);
      L1:=(WIDTH-L) DIV 2;
      R1:=WIDTH-L-L1;
      S:=S CAT REPEAT(" ",L1)
           CAT RNA[I]
           CAT REPEAT(" ",R1);
    END;
    WRITE(OUT,<A*>,LENGTH(S),S);

    COMMENT PRINT TABLE;
    FOR N:=0 STEP 1 UNTIL PRLINES DO
    WRITE(OUT,<E12.5,*(X*,E*.*,X*)>,
              OUTARR[N,0],COL,
              FOR I:=I1 STEP 1 UNTIL I2 DO
                 [L2,7+DEC,DEC,OUTARR[N,I],R2]);
    WRITE(OUT,<//>);
  END C;
END RESULTS;

PROCEDURE CHEMPLOT(PRLINES,XPD,OUTARR);
%************************************************
% THE PROCEDURE PLOTS THE RESULTS.              *
% THE PLOTS ARE DONE BY USE OF THE PLOTTER PACK- *
% AGE 'RIGS' (=RISOE INTERACTIVE GRAPHIC SYSTEM).*
% BEFORE 'RIGS' IS CALLED FROM THE PROCEDURE,ALL *
% THE THE RESULTS ARE SCALED SUCH THAT THEY CAN  *
% BE PLOTTED WITHIN A FRAME. UP TO 3 CURVES CAN  *
% BE PLOTTED WHICH MEANS THAT UP TO 3 SCALES FOR *
% THE REACTANT CONCENTRATIONS ARE USED.          *
% IF THE PROGRAM IS RUNNED IN INTERACTIVE MODE   *
% FROM A GRAPHIC TERMINAL THE PROGRAM BREAKS     *
% WITH A MESSAGE ON THE TERMINAL AND THE CONTROL *
% IS GIVEN OVER TO 'RIGS' WAITING FOR INPUT ON   *
% THE TERMINAL FROM THE USER.                    *
%                                                *
% IF THE PROGRAM IS RUNNED IN BATCH MODE THE     *
% CONTROL IS TRANSFERED TO 'RIGS' AUTOMATICALLY  *
% AND AFTER END OF PLOTS IT IS RETURNED TO THE   *
% PROCEDURE.                                     *
%                                                *
% THE USE OF 'RIGS' IS DESCRIBED IN THE 'RIGS'   *
% MANUAL (REF) TO WHICH THE USER IS REFERRED FOR *
% FURTHER INFORMATION.                           *
%                                                *
% FOLLOWING VARIABLES ARE GLOBAL:                *
% INTEGER         PLONPA, NPE, PT, RESCLASS,     *
%                 RADPRS, MAXSET.                *
% REAL            TEND.                          *
% INTEGER ARRAY   PEN, PRN                       *
% REAL    ARRAY   PFAC, ADDCON                   *
% STRING          TEXT                           *
% STRING  ARRAY   PE                             *
% BOOLEAN         XVAL                           *
% FILE            OUT.                           *
%                                                *
% REFERENCE.                                     *
```

```
% STEEN RAHBEK & ERIK HANSEN, RISOE INTERACTIVE  *
% GRAPHICS SYSTEM 'RIGS' (1983) (RISOE-R-493).   *
%************************************************

%************************************************
% SPECIFICATION OF PROCEDURE PARAMETERS          *
%************************************************

INTEGER
  PRLINES ;       % ACTUAL NUMBER OF PRINTED
                  % LINES IN RESULT TABLE.

REAL ARRAY
 XPD[*,*,*],      % HOLDS THE EXPERIMENTAL VALUES
 OUTARR[*,*];     % HOLDS THE RESULT TABLE.

BEGIN
%************************************************
% DECLARATION OF LOCAL VARIABLES                 *
%************************************************

INTEGER           % DECLARATION
  N,P,PAGENO,I ,  % COUNTING VARIABLES
  PAGES,          % NO. OF PLOTTER SHEETS COMPUTED
                  % BY THE PROCEDURE.
  NCUR,           % NUMBER OF CURVES ON AN ACTUAL
                  % PLOTTER SHEET. 1 LEQ NCUR LEQ 3.
  WIDTH,          % LINEWIDTH USED FOR CURVES
 D,D1,D2,M,N1,N2; % WORKING VARIABLES.
REAL              % DECLARATION
  XMIN, XMAX,     % LEFT AND RIGHT END OF X-AXES
  YMIN, YMAX,     % LOWER AND UPPER END OF Y-AXES.
  XMARK, YMARK,   % USED BY PROCEDURE 'PLOTAXES' FOR
                  % MARKING POINTS ON THE AXES.
  WMIN, WMAX,     % LEFT AND RIGHT LIMITS OF THE
                  % FRAME ON X-AXES
  VMIN, VMAX,     % LOWER AND UPPER LIMITS OF THE
                  % FRAME ON Y-AXES.
  SW, SH,         % WIDTH AND HEIGHT OF SYMBOLS
                  % USED.
  MS,             % SIZE OF MARKS USED FOR EXPERI-
                  % MENTAL POINTS.
  MINF, MAXF,     % MINIMUM AND MAXIMUM VALUES OF
                  % RESULTS FOR A REACTANT. USED
                  % FOR ADAPTING THE CURVES TO
                  % THE FRAME.
  MACHEPS,        % SMALLEST NUMBER FOR WHICH
                  % 1+MACHEPS>1. USED FOR ADJUSTING
                  % SCALE NUMBERS ON X- AND Y-AXES.
  UP, LOW,
  TMAX, FMAX,
  FCT, TB,
  LOGUP,
  LOGTEND,
  LOGMAXF;        % WORKING VARIABLES.

REAL ARRAY                % DECLARATION
  FCTAR[0:NPE,0:PRLINES],% HOLDS THE INTEGRATION
                          % RESULTS ADAPTED FOR PLOT.
  X,Y[0:PRLINES],         % HOLDS THE X - AND Y
                          % COORDINATES FOR PLOT.
  XP[1:2,1:NPE,1:MAXSET];% HOLDS THE EXPERIMENTAL
                          % POINTS FOR PLOT.
```

```
INTEGER ARRAY                  % DECLARATION
  YY[1:3,1:2];                 % HOLDING THE MAX. VALUE OF Y
                               % PRINTED AT THE TOP OF Y-AXES.

STRING                         % DECLARATION
  S;                           % WORKING VARIABLE

STRING ARRAY                   % DECLARATION
  A[1:3];                      % HOLDS TEXTS FOR PRINTING.

IMAGE                          % DECLARATION, USED BY 'RIGS'.
  IM;

  SETFONT("EBCDIC");
  COMMENT COMPUTE NUMBER OF PAGES (SKETCHES);
  PAGES:=(NPE+PLONPA-1) DIV PLONPA;

  COMMENT INITIALIZE ARRAY FCTAR[*,*];
  FOR N:=1 STEP 1 UNTIL NPE DO
  FOR P:=0 STEP 1 UNTIL PRLINES DO FCTAR[N,P]:=ADDCON[N];

  FOR I:=1 STEP 1 UNTIL PT DO
  BEGIN
  COMMENT COMPUTE AND STORE PLOT VALUES IN ARRAY FCTAR;
    N:=PEN[I]; M:=PRN[I];
    FOR P:=0 STEP 1 UNTIL PRLINES DO
    FCTAR[N,P]:=FCTAR[N,P]+PFAC[I]*OUTARR[P,M];
  END I;
  COMMENT INITIALIZATION OF VARIABLES;
  XMAX:=20; XMIN:=YMIN:=0;
  YMAX:=IF TERMINAL AND DEVCLASS="HARDCOPY" THEN 10 ELSE 15;
  XMARK:=XMAX/10; YMARK:=YMAX/10;
  WMIN:=XMIN-6;WMAX:=XMAX+3;VMIN:=YMIN-2;VMAX:=YMAX+4;
  SETWINDOW(WMIN,VMIN,WMAX,VMAX);
  SW:=SH:=0.3; SETSYMBOLSIZE(SW,SH);
  MS:=0.2;     SETMARKSIZE(MS);

  MACHEPS:=4*8**(-13); LOGTEND:=LOG(TEND);
  UP:=10**ENTIER(LOGTEND);
  D:=IF TEND>5*UP*(1+MACHEPS) THEN 10 ELSE
      IF TEND>2*UP*(1+MACHEPS) THEN  5 ELSE
      IF TEND>1*UP*(1+MACHEPS) THEN  2 ELSE 1;
  TMAX:=D*UP;

  FOR PAGENO:=1 STEP 1 UNTIL PAGES DO
  BEGIN
    COMMENT MAKE THE SKETCHES 1, 2,.....PAGES.
            NCUR IS THE NUMBER (=1,2 OR 3) OF
            THE CURVES ON THE SKETCH;

    NCUR:=MIN(PLONPA,NPE-(PAGENO-1)*PLONPA);
    STARTCOLLECT; SETLINEDASH(0); SETLINEWIDTH(5);
    COMMENT PLOT A FRAME SURROUNDING THE PLOTS;
    PLOTLINE(WMIN,VMIN,WMAX,VMIN);
    PLOTLINE(WMAX,VMIN,WMAX,VMAX);
    PLOTLINE(WMAX,VMAX,WMIN,VMAX);
    PLOTLINE(WMIN,VMAX,WMIN,VMIN);

    COMMENT PLOTAXES AND PRINT X-SCALE;
    PLOTAXES(0,0,XMIN,XMAX,YMIN,YMAX,XMARK,YMARK);
    PLOTTEXT(DATE,12*SW,YMAX+10*SH,1);
    PLOTTEXT("    SCALE NO: ",-15*SW,YMAX+8*SH,1);

    COMMENT IDENTIFICATION TEXT FOR THE PLOTS;
```

```
IF TEXT NEQ EMPTY THEN
PLOTTEXT(TEXT,12*SW,YMAX+8*SH,1);

IF RESCLASS<=2 THEN
BEGIN
  COMMENT RESCLASS<=2, PRINT LINEAR X-SCALE;

  COMMENT
    TRANSFORM TIME VALUES TO THE INTERVAL 0-XMAX;
  FOR P:=0 STEP 1 UNTIL PRLINES DO
  FCTAR[0,P]:=OUTARR[P,0]*XMAX/TMAX;

  COMMENT PRINT THE COORDINATES TO THE TIME SCALE;
  FOR P:=0 STEP 1 UNTIL 10 DO
   PLOTWRITE((STRINGBUFF[*],<F4.1>,P/10*D),
              P/10*XMAX-2*SW,-2*SH,1);

  COMMENT PRINT THE SCALE FACTOR FOR THE TIME ;
  PLOTWRITE((STRINGBUFF[*],<"*10">),
             XMAX+2*SW,-2*SH,.8);
  PLOTWRITE((STRINGBUFF[*],<I3>,ENTIER(LOGTEND)),
             XMAX+4*SW,-1.4*SH,.8);
  PLOTWRITE((STRINGBUFF[*],<"TIME IN SECS">),
             XMAX-12*SW,-4*SH,1);
END
ELSE
BEGIN
  COMMENT RESCLASS > 2, PRINT LOG X - SCALE;

  COMMENT
    TRANSFORM LOG(TIME) TO THE INTERCAL 0-XMAX;
  TMAX:=IF LOGTEND>ENTIER(LOGTEND)
        THEN ENTIER(LOGTEND)+1
        ELSE ENTIER(LOGTEND);
  TB:=ENTIER(LOG(OUTARR[1,0]));
  FOR P:=1 STEP 1 UNTIL PRLINES DO
  FCTAR[0,P]:=(LOG(OUTARR[P,0])-TB)*XMAX/(TMAX-TB);

  COMMENT PRINT THE COORDINATES TO THE TIME SCALE;
  FOR P:=0 STEP 1 UNTIL 10 DO
   PLOTWRITE((STRINGBUFF[*],<F5.1>,P/10*(TMAX-TB)+TB),
              P/10*XMAX-2*SW,-2*SH,1);
  PLOTWRITE((STRINGBUFF[*],<"LOG(TIME IN SECS)">),
             XMAX-16*SW,-4*SH,1);
END;

N2:=0;
FOR N:=1 STEP 1 UNTIL NCUR DO
BEGIN
  M:=(PAGENO-1)*PLONPA+N; MAXF:=0;
  IF TAIL(TAIL(PE[M],NOT ":"),":") NEQ EMPTY THEN
  BEGIN
    FOR P:=0 STEP 1 UNTIL PRLINES DO
        MAXF:=MAX(MAXF,FCTAR[M,P]);
    LOGMAXF:=IF MAXF>@-46 THEN LOG(MAXF) ELSE -46;

    IF RESCLASS=1 OR RESCLASS=3 THEN
    BEGIN
      COMMENT COMPUTE LIN. Y-SCALES,STORE ATTENDING TEXTS
              IN A AND STORE PLOTVALUES IN FCTAR;
      UP:=10**ENTIER(LOGMAXF);
      IF MAXF>@-46 THEN
      BEGIN
        D1:= IF MAXF>5*UP THEN 10 ELSE
```

```
          IF MAXF>2*UP THEN  5 ELSE
          IF MAXF>1*UP THEN  2 ELSE 1;
      FMAX:=D1*UP;
      YY[N,1]:=D1;
      YY[N,2]:=D2:=ENTIER(LOGMAXF);
    END
    ELSE
    BEGIN
      D1:=YY[N,1]:=1; D2:=YY[N,2]:=-46;
    END;

    COMMENT STORE OUTPUT VALUES FOR PLOT;
    FOR P:=0 STEP 1 UNTIL PRLINES DO
    FCTAR[M,P]:=FCTAR[M,P]*YMAX/D1/UP;
    COMMENT STORE ATTENDING TEXT IN A[N];
    FOR I:=1 STEP 1 UNTIL N DO
    IF YY[I,1]=YY[N,1] AND
       YY[I,2]=YY[N,2] AND I<N THEN
    BEGIN COMMENT I<N;
      N1:=I; I:=N+1;
    END ELSE
    IF I = N THEN
    BEGIN
      N1:=N2:=N2+1;

      COMMENT PRINT THE Y CORRDINATES;
      D1:=YY[N,1]; D2:=YY[N,2];

      FOR P:=0 STEP 10/D1 UNTIL 10 DO
      BEGIN
        PLOTWRITE((STRINGBUFF[*],<I2>,P*D1/10),
                  (2-6*N2)*SW,P/10*YMAX-0.5*SH,1);
      END;
      PLOTWRITE((STRINGBUFF[*],<"10">),
                (2-6*N2)*SW,YMAX+2.5*SH,.8);
      PLOTWRITE((STRINGBUFF[*],<I3>,D2),
                (4-6*N2)*SW,YMAX+4.0*SH,.8);
      PLOTWRITE((STRINGBUFF[*],<I1>,N2),
                (2-6*N2)*SW,YMAX+5.5*SH,1);
    END;
    A[N]:="SCALE NO. " CAT STRING(N1,1) CAT ": "
          CAT HEAD(DROP(TAIL(PE[M],NOT":"),1),NOT";");
  END RESCLASS 1 AND 3
  ELSE
  BEGIN
    COMMENT RESCLASS 2 AND 4. COMPUTE LOG. Y-SCALE,STORE
            ATTENDING TEXTS IN A AND OUTPUT VALUES FOR
            PLOT IN FCTAR;
    MINF:=MAXF;
    UP:=IF LOGMAXF>ENTIER(LOGMAXF)
        THEN ENTIER(LOGMAXF)+1
        ELSE ENTIER(LOGMAXF);
    FOR P:=1 STEP 1 UNTIL PRLINES DO
        MINF:=MIN(MINF,FCTAR[M,P]);
    LOW:=IF MINF>@-46 THEN ENTIER(LOG(MINF)) ELSE -46;

    YY[N,1]:=LOW; YY[N,2]:=UP;
    FOR P:=0 STEP 1 UNTIL PRLINES DO
    BEGIN
     FCT:=FCTAR[M,P];
     FCT:=IF FCT>@-46 THEN LOG(FCT) ELSE -46;
     FCTAR[M,P]:=(FCT-LOW)*YMAX/(UP-LOW);
    END;
```

```
  COMMENT STORE ATTENDING TEXT IN ARRAY A[N,*];

  FOR I:=1 STEP 1 UNTIL N DO
  IF YY[I,1]=YY[N,1] AND
     YY[I,2]=YY[N,2] AND I<N THEN
  BEGIN COMMENT I<N;
   N1:=I; I:=N+1;
  END ELSE
  IF I = N THEN
  BEGIN
    N1:=N2:=N2+1;

    COMMENT PRINT Y COORDINATES IN LOGARITHMIC SCALES;

    FOR P:=0 STEP 1 UNTIL 10 DO
    BEGIN
     PLOTWRITE((STRINGBUFF[*],<F5.1>,
                P/10*(UP-LOW)+LOW),
                -6*SW*N2,P/10*YMAX-0.5*SH,1);
    END;
    PLOTWRITE((STRINGBUFF[*],<I3>,N2),
               -6*SW*N2,YMAX+5.5*SH,1);
  END;
  A[N]:="SCALE NO. " CAT STRING(N1,1) CAT ": "
        CAT HEAD(DROP(TAIL(PE[M],NOT":"),1),NOT";");
END RESCLASS 2 AND 4;

COMMENT PLOT EXPRESSION FOR THIS CURVE;

PLOTMARK(N,8*SW,YMAX+0.5*MS+2*(NCUR+1-N)*SH,1);
IF N=2 THEN WIDTH:=SETLINEWIDTH(6);
SETLINEDASH(N-1);
PLOTLINE( 9*SW,YMAX+0.5*MS+2*(NCUR+1-N)*SH,
         11*SW,YMAX+0.5*MS+2*(NCUR+1-N)*SH);
IF N=2 THEN SETLINEWIDTH(WIDTH);

PLOTWRITE((STRINGBUFF[*],<A*>,LENGTH(A[N]),A[N]),
          12*SW,YMAX+2*(NCUR+1-N)*SH,1);

COMMENT PLOT EXPERIMENTAL POINTS;

IF XVAL THEN
BEGIN
  FOR P:=1 STEP 1 UNTIL MAXSET DO
  IF XPD[1,M,P]>=0 THEN
  BEGIN
    IF RESCLASS<=2
    THEN XP[1,M,P]:=XPD[1,M,P]*XMAX/TEND
    ELSE
    BEGIN
      FCT:=XPD[1,M,P];
      FCT:=IF FCT>@-46 THEN LOG(FCT) ELSE -46;
      XP[1,M,P]:=(FCT-TB)*XMAX/(TMAX-TB);
    END;
    IF RESCLASS=1 OR RESCLASS=3
    THEN XP[2,M,P]:=XPD[2,M,P]*YMAX/D1/UP
    ELSE
    BEGIN
      FCT:=XPD[2,M,P];
      FCT:=IF FCT > @-46 THEN LOG(FCT) ELSE -46;
      XP[2,M,P]:=(FCT-LOW)*YMAX/(UP-LOW);
    END;
    PLOTMARK(M,XP[1,M,P],XP[2,M,P],1);
  END;
```

```
        END XVAL;

        COMMENT DRAW THE CURVES;

        SETLINEDASH(N-1);
        IF N=2 THEN WIDTH:=SETLINEWIDTH(6);

        IF RADPRS>0 THEN
        BEGIN COMMENT PLOT NCUR CURVES;
          FOR I:=0 STEP 1 UNTIL MIN(PRLINES,RADPRS) DO
          BEGIN
            X[I]:=FCTAR[0,I]; Y[I]:=FCTAR[M,I];
          END;
          PLOTSPLINE(X,Y,MIN(PRLINES,RADPRS),0.1,0,1);

          IF PRLINES>RADPRS THEN
          BEGIN
            FOR I:=0 STEP 1 UNTIL PRLINES-RADPRS DO
            BEGIN
              X[I]:=FCTAR[0,RADPRS+I];
              Y[I]:=FCTAR[M,RADPRS+I];
            END;
            PLOTSPLINE(X,Y,PRLINES-RADPRS,0.1,0,1);
          END;
        END ELSE
        BEGIN
          FOR I:=0 STEP 1 UNTIL PRLINES DO
          BEGIN
           X[I]:=FCTAR[0,I]; Y[I]:=FCTAR[M,I];
          END;
          PLOTSPLINE(X,Y,PRLINES,0.1,0,1);
        END;

        IF N=2 THEN SETLINEWIDTH(WIDTH);
      END PRINTEXPRESSION NEQ EMPTY;
    END N;

    IM:=COLLECT;

    IF TERMINAL THEN
    BEGIN
      IF DEVCLASS="HARDCOPY" THEN
      BEGIN
       SETMAPPING(DEVCLASS,0);
      END ELSE
      BEGIN
       SETMAPPING("TERMINAL",1);
      END;
      SETAUTOSHOW(FALSE);
      IMAGEHANDLER(IM,
                   "YOU ARE IN RIGS, USE RIGS - COMMANDS" CAT
                   ", AND/OR RETURN");
      WRITE(ERRFIL,<"YOU HAVE RETURNED FROM RIGS"
                    " - THE PROGRAM CONTINUES">);
    END ELSE
    BEGIN
     SETHEIGHT(DEVCLASS,14); SETMAPPING(DEVCLASS,2);
     SHOWIMAGE(IM,DEVCLASS); ENDSKETCH(DEVCLASS);
    END;
  END PAGENO;
END CHEMPLOT;
```

```
%*************************************************
%                                                 *
%         M A I N   P R O G R A M.                *
%                                                 *
%*************************************************

TERMINAL := MYSELF.STATION < 0;
RIGSINITIATION;
SETBAUDRATE(9600); SETORIGO(4,2); SETCENTREABS(0,0);

NEWSYS:=TRUE;

PROGRAMSTART:
%*************************************************
%                                                 *
% DATE AND TIME FOR IDENTIFICATION OF COMPUTATION *
%                                                 *
%*************************************************

D7:=TIME(7);
DATE:="DATE: ";
DATE:=DATE CAT STRING(D7.[47:12],4);
IF D7.[35:6]<10 THEN DATE:=DATE
                     CAT "/0"
                     CAT STRING(D7.[35:6],1)
                ELSE DATE:=DATE
                     CAT "/"
                     CAT STRING(D7.[35:6],2);
IF D7.[29:6]<10 THEN DATE:=DATE
                     CAT "/0"
                     CAT STRING(D7.[29:6],1)
                ELSE DATE:=DATE
                     CAT "/"
                     CAT STRING(D7.[29:6],2);
DATE:=DATE CAT "  TIME: ";
IF D7.[23:6]<10 THEN DATE:=DATE
                     CAT "0"
                     CAT STRING(D7.[23:6],1)
                ELSE DATE:=DATE
                     CAT STRING(D7.[23:6],2);
IF D7.[17:6]<10 THEN DATE:=DATE
                     CAT ":0"
                     CAT STRING(D7.[17:6],1)
                ELSE DATE:=DATE
                     CAT ":"
                     CAT STRING(D7.[17:6],2);
IF D7.[11:6]<10 THEN DATE:=DATE
                     CAT ":0"
                     CAT STRING(D7.[11:6],1)
                ELSE DATE:=DATE
                     CAT ":"
                     CAT STRING(D7.[11:6],2);

%*************************************
% CALL OF THE INPUT PROCEDURE CHEMINPUT *
%*************************************

CHEMINPUT;
IF INPUTERROR THEN
BEGIN
  DATAERR:=TRUE;
  GO TO PROGRAMSTART;
END;
IF DATAERR THEN GO TO FIN;
```

```
%************************************************
%                                                *
% IF THE BOOLEAN NEWRE HAS BEEN ASSIGNED THE     *
% VALUE 'TRUE' IN PROCEDURE CHEMINPUT THE THE    *
% THE PROCEDURE CHEMTRANS IS CALLED FOR TRANS-   *
% LATING THE REACTION EQUATIONS TO THE CORRES-   *
% PONDING DIFFERENTIAL EQUATION SYSTEM.          *
%                                                *
%************************************************

IF NEWRE THEN CHEMTRANS;

%************************************************
%                                                *
% IF THE BOOLEAN NEWPE HAS BEEN ASSIGNED THE     *
% VALUE 'TRUE' IN PROCEDURE CHEMINPUT THE THE    *
% PROCEDURE TRANSPLOT IS CALLED FOR TRANSLATING  *
% THE PLOT EXPRESSIONS.                          *
% IF ALSO THE BOOLEAN XVAL HAS BEEN ASSIGNED     *
% THE VALUE 'TRUE' THEN THE PROCEDURE STOREXPD   *
% IS CALLED FOR STORING THE EXPERIMENTAL DATA    *
% TO BE PLOTTED BY THE PROCEDURE CHEMPLOT.       *
%                                                *
%************************************************

IF NEWPE THEN TRANSPLOT;


%************************************************
%                                                *
% IF BOOLEAN NEWGV IS ASSIGNED THE VALUE 'TRUE'  *
% IN PROCEDURE CHEMINPUT, THE PROCEDURE STOREG   *
% IS CALLED FOR STORING THE G-VALUES IN ARRAY RG *
%                                                *
%************************************************

IF NEWGV THEN STOREG;

%************************************************
%                                                *
% IF BOOLEAN NEWRC IS ASSIGNED THE VALUE 'TRUE'  *
% IN PROCEDURE CHEMINPUT THEN PROCEDURE STOREK   *
% IS CALLED FOR STORING NEW RATE CONSTANTS IN    *
% ARRAY RK.                                      *
%                                                *
%************************************************

IF NEWRC THEN STOREK;

%************************************************
%                                                *
% IF ONE OF THE BOOLEANS NEWRE OR NEWCON ARE AS- *
% SIGNED THE VALUE 'TRUE' THE THE PROCEDURE      *
% STORECON IS CALLED FOR STORING NEW CONCENTRA-  *
% TION VALUES IN ARRAY RCON.                     *
%                                                *
%************************************************

IF NEWRE OR NEWCON THEN STORECON;

IF INPUTERROR THEN DATAERR:=TRUE;
IF DATAERR OR DATACON THEN GO TO PROGRAMSTART;

%************************************************
```

```
%                                                    *
% IF BOOLEAN NEWRE IS ASSIGNED THE VALUE 'TRUE'      *
% PROCEDURE CHEMINPUT AND BOOLEAN DIFFEQ IS SET      *
% TO 'TRUE' BY THE USER IN INPUT-FILE 'INP' THEN     *
% PROCEDURE PRINTDIFFEQ IS CALLED FOR PRINTING       *
% THE DIFFERENTIAL EQUATION SYSTEM.                  *
%                                                    *
%****************************************************

IF NEWRE AND DIFFEQ THEN PRINTDIFFEQ;

%****************************************************
%                                                    *
% IF BOOLEAN OUTINP IS TRUE THE PROCEDURE            *
% STARTVAL IS CALLED AND THE G-VALUES AND THE        *
% STARTVALUES OF CONCENTRATIONS ARE PRINTED          *
%                                                    *
%****************************************************

IF OUTINP THEN STARTVAL;

WRITE(OUT,<//"TOTAL DOSE",26("."),"=",E10.2," KILORAD">,
      DOSE);
WRITE(OUT,<"NUMBER OF RADIATIONS",16("."),"=",I3>,
      NRR);
WRITE(OUT,<"RADIATION-TIME",22("."),"=",E10.2," SEC.">,
      RADTIME);
WRITE(OUT,<"FIRST INTEGRATION STEP",14("."),"=",E10.2,
          " SEC.">,FSTSTP);
WRITE(OUT,<"MAXIMUM INTEGRATION STEP",12("."),"=",E10.2,
          " SEC.">,HMAXO);
WRITE(OUT,<"NUMBER OF RESULTS DURING RADIATION..=",I3>,
          RADPRS);
IF PRINTS<RADPRS THEN
BEGIN
  PRINTS:=RADPRS;
  WRITE(OUT,<
  "W A R N I N G: PRINTS IS SET EQUAL TO RADPRS"/
  "               BECAUSE THE INPUT VALUE OF   "/
  "               PRINTS IS SMALLER THAN RADPRS">);
END;
WRITE(OUT,<"TOTAL NUMBER OF RESULTS",13("."),"=",I3>,
          PRINTS);
WRITE(OUT,<"MAXIMUM INTEGRATION TIME",12("."),"=",E10.2,
          " SEC.">,TEND);
WRITE(OUT,<"RELATIVE ACCURACY IN EPISODE........=",E10.2>,
          EPS);

WRITE(OUT,</A*>,LENGTH(DATE),DATE);
IF TEXT NEQ EMPTY THEN WRITE(OUT,<A*>,LENGTH(TEXT),TEXT);

CASE MF OF
BEGIN
 10:11:12:13:
 WRITE(OUT,
       </"METHOD OF INTEGRATION: ADAMS METHOD: MF =",I2>,
       MF);
 20:21:22:23:
 WRITE(OUT,
       </"METHOD OF INTEGRATION: STIFF METHOD: MF =",I2>,
       MF);
END;
CASE RESCLASS OF
BEGIN
```

```
  1: WRITE(OUT,
           </"RESULTS: LIN. TIME-SCALE, LIN. Y-SCALE"/>);
  2: WRITE(OUT,
           </"RESULTS: LIN. TIME-SCALE, LIN. Y-SCALE"/>);
  3: WRITE(OUT,
           </"RESULTS: LOG. TIME-SCALE, LIN. Y-SCALE"/>);
  4: WRITE(OUT,
           </"RESULTS: LOG. TIME-SCALE, LOG. Y-SCALE"/>);
END;

BEGIN
%************************************************
%  INTEGRATION OF THE DIFFERENTIAL SYSTEM       *
%************************************************

%************************************************
% DECLARATION  OF LOCAL VARIABLES.              *
%************************************************
INTEGER          % DECLARATION
  I,J,K,N,P,     % COUNTING VARIABLES.
  FIRSTP,LASTP,% FIRST AND LAST PRINTED LINE.
  STARTTIME,     % USED FOR DETERMINATION OF
  ENDTIME,       % COMPUTING TIME.
  RADNO,         % THE RADIATION NO. NORMALLY 1.
  NT5,           % WORKING VARIABLE
  PRLINES,       % ACTUAL NUMBER OF OUTPUT RESULTS
  INDEX,         % INDICATE TYPE OF INTEGRATION.
                 % IN THIS PROGRAM WE USE INDEX=3.
  NSTEP,         % NUMBER OF INTEGRATION STEPS,
                 % RETURNED FROM EPISODE.
  NQUSED,        % LAST ORDER USED IN THE INTEGRATION,
                 % RETURNED FROM EPISODE.
  NFE,           % NUMBER OF FUNCTION EVALUATIONS
                 % IN INTEGRATION.
                 % RETURNED FROM EPISODE.
  NJE,           % NUMBER OF JACOBIAN EVALUATIONS
                 % IN INTEGRATION.
                 % RETURNED FROM EPISODE.
  IERROR;        % FLAG FOR ERROR CONTROL.
                 % RETURNED FROM EPISODE.
REAL             % DECLARATION
  RADEND,        % END OF RADIATION INTERVAL
  SUBTEND,       % END OF SUBINTERVAL FOR
                 % INTEGRATION. SUBTEND LEQ TEND.
  T,             % INDEPENDENT VARAIBLE, TIME.
  PRINTV,TOUT,  % WORKING VARIABLES.
  H,             % STEPSIZE IN CALL FOR INTEGRATION.
  CONST,         % CONST = 1.038E-6
  PRINTTIME,     % THE VALUE OF T FOR OUTPUT OF RESULT
  HUSED,         % LAST STEP USED IN INTEGRATION.
                 % RETURNED FROM EPISODE.
  LOGPTIME,      % USED FOR OUTPUT IN LOGARITHMIC
  LOGPINTV;      % TIME.
STRING           % DECLARATION
  S;             % USED FOR OUTPUT OF MESSAGES
                 % FROM EPISODE.
BOOLEAN          % DECLARATION
  RAD,           % TRUE FOR RADIATION, ELSE FALSE.
  INITOK;        % TRUE IF THE INITIATION OF COMMON
                 % VARIABLES IN 'FLIB IS OK, ELSE FALSE.
LABEL            % DECLARATION
  PRINTRESULT; %
REAL ARRAY        % DECLARATION
  Y[0:NR],        % HOLDS THE CONCENTRATION VALUES
```

```
  RGC[0:NR],      % HOLDS G-VALUES MULTIPLIED WITH
                  % CONST = 1.038E-6.
  ERK[0:NT]      ,% HOLDS STOICHIOMETRIC CONSTANTS
                  % MULTIPLIED WITH THE MATCHING RATE
                  % CONSTANTS.

  OUTARR[0:PRINTS*MAX(NRR,1),0:NR],
                  % HOLDS THE RESULT OF THE INTEGRATION
                  % WITH TIMES IN FIRST COLUMN AND THE
                  % THE CONCENTRATIONS IN THE REMAINING.
  XPD[1:2,1:MAXPE,1:MAXSET];
                  % CONTAINS THE DATA (X AND Y-COORDINA-
                  % TES) FOR THE EXPERIMENTAL POINTS TO
                  % BE PLOTTED.

COMMENT
  COMPUTE THE PRODUCTS OF STOICHIOMETRIC CONSTANTS
  AND RATE CONSTANTS AND STORE RESULTS IN ARRAY ERK.
  STORE ALSO RGC;

  FOR N:=1 STEP 1 UNTIL NT DO
   ERK[N]:=ETERM[5*(N-1)+2]*RK[2,ETERM[5*(N-1)+3]];
  CONST:=IF RADTIME>0 THEN 1.038@-6*DOSE/RADTIME ELSE 0;
  FOR N:=1 STEP 1 UNTIL NR DO RGC[N]:=RG[N]*CONST;
  IF MYSELF.STATION<0 THEN ERRFIL.KIND:=VALUE(REMOTE)
                      ELSE ERRFIL.KIND:=VALUE(PRINTER);
  MAXTIME:=MYSELF.MAXPROCTIME;
  COMMENT STORE STARTVALUES IN OUTPUT ARRAY OUTARR;
  PRINTTIME:=TOUT:=OUTARR[0,0]:=T:=0;
  IF RESCLASS>2 THEN LOGPTIME:=LOG(FSTSTP);
  NT5:=5*NT;
  FOR N:=1 STEP 1 UNTIL NR DO
  BEGIN
    OUTARR[0,N]:=Y[N]:=RCON[N];
  END N;

  %************************************************
  %                                                *
  % BEFORE THE CALL OF THE PROCEDURE DRIVE, WHICH  *
  % IS THE DRIVER FOR THE INTEGRATION SUBROUTINE   *
  % IN THE INTEGRATION PACKAGE 'EPISODE', THE CON- *
  % TENTS IN SOME OF THE ARRAYS DECLARED IN THE    *
  % ALGOL PROGRAM MUST BE CHANGED.                 *
  % THE CONNECTION BETWEEN ALGOL AND FORTRAN ARRAYS*
  % IS SO, THAT WHEN A FORTRAN SUBROUTINE REFERS   *
  % TO THE FIRST ELEMENT, SAY Y(1) WHERE Y IS DE-  *
  % CLARED AS AN ALGOL ARRAY, THEN THE ELEMENT     *
  % REFERRED IS Y[0] AND NOT Y[1] IN THE ALGOL     *
  % ARRAY. THEREFORE, THE CONTENT IN AN ALGOL AR-  *
  % RAY TO WHICH THERE IS REFERRED FROM A FORTRAN  *
  % SUBROUTINE, MUST BE MOVED TO LEFT SO THAT THE  *
  % CONTENT OF ELEMENT NO.:1 IS PLACED IN ELEMENT  *
  % NO.: 0, ELEMENT NO.: 2 IN ELEMENT NO.: 1 ETC.  *
  % IT IS BECAUSE OF THIS FEATURE THAT THE ARRAYS  *
  % IS DECLARED FROM 0 AND NOT FROM 1.             *
  %                                                *
  % THE ARRAY Y IS A PARAMETER IN DRIVE AND THE    *
  % ARRAYS RGC, ERK AND ETERM ARE TRANSFERRED TO   *
  % COMMON AREA IN 'FLIB'. THEY ARE USED BY THE    *
  % SUBROUTINES DIFFUN AND PEDERV.                 *
  % THE TRANSPORT OF CONTENT OF THE ARRAYS RGC,ERK *
  % AND ETERM IS DONE BY THE SUBROUTINE INITCM     *
  % (=INITIALIZATION OF COMMON, SEE BELOW).        *
  %                                                *
```

```
% 'EPISODE' AND ALL THE SUBROUTINES IN THE       *
% FORTRAN LIBRARY 'FLIB' ARE PRECOMPILED ON A    *
% DISK FILE WITH THE TITLE:                      *
%      'CHEMSIMUL/FORLIB'                        *
%                                                *
% SEE THE LISTING OF THE FORTRAN SUBROUTINES     *
% AFTER THIS ALGOL LISTING.                      *
%                                                *
%************************************************

FOR N:=1 STEP 1 UNTIL NR DO
BEGIN
 Y[N-1] := Y[N];  RGC[N-1] := RGC[N];
END N;
FOR N:=1 STEP 1 UNTIL NT DO ERK[N-1] := ERK[N];
FOR N:=1 STEP 1 UNTIL NT5 DO ETERM[N-1] := ETERM[N];

FIRSTP:=LASTP:=0; INDEX:=1;
STARTTIME:=TIME(2);
RADNO:=MIN(NRR,1);
FOR RADNO:=RADNO STEP 1 UNTIL NRR DO
BEGIN
  IF RADNO=0 THEN WRITE(OUT,</"NO RADIATION"/>)
             ELSE WRITE(OUT,</"RADIATION PERIODE NO:",I3/>,
                            RADNO);
  H:=FSTSTP;
  RADEND:=TOUT+RADTIME;
  SUBTEND:= TEND/MAX(1,NRR)+TOUT;
  WHILE TOUT<SUBTEND DO
  BEGIN
    IF TOUT<RADEND THEN
    BEGIN
     RAD:=TRUE; INDEX:=1; H:=FSTSTP;
     IF RESCLASS <= 2
       THEN PRINTV:=RADTIME/RADPRS
       ELSE LOGPINTV:=(LOG(RADTIME)-LOG(FSTSTP))/RADPRS;
     FIRSTP:=LASTP+1; LASTP:=FIRSTP+RADPRS-1;
    END ELSE
    BEGIN
      RAD:=FALSE; INDEX:=1; H:=FSTSTP;
      IF RESCLASS<=2
      THEN PRINTV:=(SUBTEND-RADEND)/MAX(1,PRINTS-RADPRS)
      ELSE LOGPINTV:=(LOG(SUBTEND)-LOG(MAX(FSTSTP,RADEND)))/
                     MAX(1,PRINTS-RADPRS);
      FIRSTP:=LASTP+1; LASTP:=FIRSTP+PRINTS-RADPRS-1;
    END;

    %************************************************
    %                                               *
    % STORE COMMON VARIABLES IN THE FORTRAN-LIBRARY  *
    %   'CHEMSIMUL/FORLIB'                          *
    %                                               *
    %************************************************

    INITCM(NR,NT,NT5,ETERM,ERK,RGC,RAD,INITOK);

    IF NOT INITOK THEN
    BEGIN
      WRITE(ERRFIL,<"NR=",I4,"  NT5=",I5," NT=",I5>,
                   NR,NT5,NT);
      WRITE(ERRFIL,
       <"THE COMMON STATEMENTS FOR ETERM, ERK AND RGC"/
        "IN SUBROUTINES: INITCM, DIFFUN AND PEDERV"/
        "MUST BE CHANGED AT LEAST TO:                "/
```

```
    "  RGC(",I4,")"/
    "  ETERM(",I5,")"/
    "  ERK(",I5,")">,NR+1,NT5+1,NT+1);
  WRITE(ERRFIL,
   <"NB: CHANGE ALSO THE IF-STATEMENT"/
    "    WHICH CONTROLS THE SIZE OF  "/
    "    ETERM,ERK AND RGC.          "/
    "AFTER CHANGE, RECOMPILE FORTRAN "/
    "LIBRARY AND RUN PROGRAM AGAIN   "/
    "SEE DESCRIPTION IN FORTRAN-PART ">);
  GO TO FIN;
END NOT INITOK;

FOR P:=FIRSTP STEP 1 UNTIL LASTP DO
BEGIN
  IF RESCLASS<=2 THEN PRINTTIME:=PRINTTIME+PRINTV ELSE
  BEGIN
    LOGPTIME:=LOGPTIME+LOGPINTV;
    PRINTTIME:=10**(LOGPTIME);
  END;

  IF P=LASTP THEN
  BEGIN
    PRINTTIME:=IF RAD THEN RADEND ELSE SUBTEND;

  END;
  TOUT:=PRINTTIME;

  IF TIME(2)/60 + 7 > MAXTIME THEN
  BEGIN
    WRITE(OUT,<40("*")/
    "PROCESSTIME SURPASSED - COMPUTATION INTERRUPTED"/
    40("*")>);
    PRLINES:=P;
    OUTARR[P,0]:=TOUT;
    FOR N:=1 STEP 1 UNTIL NR DO OUTARR[P,N]:=Y[N-1];
    GO TO PRINTRESULT;
  END;

  IERROR:=3;

  %*****************************************
  %                                        *
  % CALL OF THE INTEGRATION ROUTINE        *
  %                                        *
  %*****************************************
  DRIVE(NR,T,H,HMAX0,Y,TOUT,EPS,IERROR,MF,INDEX);

  IF INDEX < 0 THEN
  BEGIN
  %*****************************************
  % OUTPUT OF ERROR MESSAGES FROM THE      *
  % INTEGRATION ROUTINE. THE VALUE OF      *
  % PARAMETER INDEX DETERMINES THE TYPE    *
  % OF MESSAGE.                           *
  %*****************************************

    CASE -INDEX OF
    BEGIN
     1:WRITE(OUT,
       <"THE INTEGRATION WAS HALTED AFTER FAILING"/
        "TO PASS THE ERROR TEST EVEN AFTER REDUC-"/
        "ING H BY A FACTOR OF 1.E10 FROM ITS INI-"/
        "TIAL VALUE.">);
```

```
          2:WRITE(OUT,
            <"AFTER SOME INITIAL SUCCESS, THE INTEGRATION"/
             "WAS HALTED EITHER BY REPEATED ERROR TEST"/
             "FAILURES OR BY A TEST ON EPS. TOO MUCH AC-"/
             "CURACY HAS BEEN REQUESTED.">);
          3:WRITE(OUT,
            <"THE INTEGRATION WAS HALTED AFTER FAILING TO"/
             "ACHIEVE CORRECTOR CONVERGENCE EVEN AFTER"/
             "REDUCING H BY A FACTOR OF 1.E10 FROM ITS"/
             "INITIAL VALUE.">);
          4:WRITE(OUT,
            <"IMIDIATE HALT BECAUSE OF ILLEGAL VALUES OF"/
             "INPUT PARAMETERS. SEE PRINTED MESSAGE.">);
          5:WRITE(OUT,
            <"INDEX WAS -1 ON INPUT, BUT THE DESIRED"/
             "CHANGES OF PARAMETERS WERE NOT IMPLEMENTED"/
             "BECAUSE TOUT WAS NOT BEOND T. INTERPOLATION"/
             " TO T = TOUT WAS PERFORMED AS ON A NORMAL "/
             "RETURN. TO TRY AGAIN, SIMPLY CALL AGAIN"/
             "WITH INDEX = -1 AND A NEW TOUT.">);
        END;

        WRITE(OUT,
              <" AFTER OUTPUT OF RESULTS OBTAINED"
               " THE COMPUTATION OF THIS PROBLEM"/
               " IS ABORTED AND THE PROGRAM"
               " CONTINUES WITH THE NEXT PROBLEM"
               " IF SUCH EXISTS">);

        GO TO PRINTRESULT;
      END INDEX;

      COMMENT STORE TOUT AND ARRAY Y;
      OUTARR[P,0]:=TOUT; PRLINES:=P;
      FOR N:=1 STEP 1 UNTIL NR DO OUTARR[P,N]:=Y[N-1];
      IF P=LASTP THEN
      BEGIN
        INDEX:=1; T:=TOUT; H:=FSTSTP;
      END ;
    END P;
  END TOUT LT SUBTEND;
END RADNO;

%************************************************
%  RE-ESTABLISH THE ARRAYS:Y, RGC, ERK AND ETERM *
%  I.E. MOVE ALL ELEMENTS ONE ELEMENT TO RIGHT.  *
%************************************************

FOR N:=NR STEP -1 UNTIL 1 DO
BEGIN
 Y[N]:=Y[N-1];  RGC[N]:=RGC[N-1];
END N;
FOR N:=NT  STEP -1 UNTIL 1 DO ERK[N]  :=ERK[N-1];
FOR N:=NT5 STEP -1 UNTIL 1 DO ETERM[N]:=ETERM[N-1];

%************************************************
% PRINT THE RESULTS IF THE BOOLEAN TABLE IS TRUE *
%************************************************

PRINTRESULT:
IF TABLE THEN RESULTS(PRLINES,OUTARR);

%************************************************
% IF BOOLEAN PLOTTING IS TRUE AND IF THERE ARE AT*
```

```
% LEAST TWO OUTPUT LINES IN THE RESULT TABLE THEN*
% THE PROCEDURE CHEMPLOT  IS CALLED FOR PLOTS OF *
% THE RESULTS.                                    *
%************************************************

IF PLOTTING AND PRLINES GEQ 2 THEN
BEGIN
  IF XVAL THEN
  BEGIN
    %**********************************************
    % THE EXPERIMENTA DATA (IF SUCH EXIST) ARE    *
    % STORED IN THE REAL ARRAY XPD BY THE PROCE-  *
    % DURE, STOREXPD, AND ARE USED BY THE PLOT-   *
    % TER PROCEDURE, CHEMPLOT. ONLY POSITIVE VA-  *
    % LUES IN XPD ARE USED AND THEREFORE ARE ALL  *
    % ELEMENTS INITIATED TO '-1' BEFORE THE STO-  *
    % RAGE IS DONE.                               *
    %**********************************************
    FOR I:=1,2 DO
    FOR J:=1 STEP 1 UNTIL MAXPE  DO
    FOR K:=1 STEP 1 UNTIL MAXSET DO XPD[I,J,K]:= -1;

    STOREXPD(XPD);
  END XVAL;
  CHEMPLOT(PRLINES,XPD,OUTARR);
END PLOTTING;

%************************************************
% TRANSFER THE CONTENTS OF THE VARIABLES:       *
% HMIN, HMAX, HUSED, NQUSED, NSTEP, NFE AND NJE *
% TO THE ALGOL PROGRAM FOR OUTPUT.              *
%************************************************

INFORM(HMIN,HMAX,HUSED,NQUSED,NSTEP,NFE,NJE);

WRITE(OUT,
  <"NUMBER OF INTEGRATIONSTEPS          =",I10/
   "MAXIMUMSTEP                         =",E10.3/
   "MINIMUMSTEP                         =",E10.3/
   "LAST STEP USED                      =",E10.3/
   "THE LAST ORDER USED                 =",I2/
   "NUMBER OF FUNCTION EVALUATIONS      =",I5/
   "NUMBER OF DERIVATIVE EVALUATIONS    =",I5/>,
  NSTEP,HMAX,HMIN,HUSED,NQUSED,NFE,NJE);

ENDTIME:=TIME(2);
WRITE(OUT,
  <"COMPUTATION TIME",20(".")"=",F9.2," SEC">,
  (ENDTIME-STARTTIME)/60);
WRITE(OUT,<//"END OF THIS COMPUTATION"/>);
WRITE(OUT,<*("*")>,LINELE);
LOCKF7;
IF INTMSG.AVAILABLE = 1 THEN
BEGIN
  IF INTMES THEN
  WHILE NOT READ(INTMSG,LINELE,S) DO WRITE(OUT,LINELE,S);
  CLOSE(INTMSG,PURGE);
END;

END INTEGRATION;
GO TO PROGRAMSTART;
END;
FIN: IF DATACON THEN
BEGIN
```

```
    WRITE(ERRFIL,<38("*")>);
    WRITE(ERRFIL,<"*",36(" "),"*">);
    IF DATAERR THEN
    WRITE(ERRFIL,<"*",X6,"ERRORS IN DATA FILE: INP",X6,"*">)
    ELSE
    WRITE(ERRFIL,<"*",X9,"INPUT DATA CORRECT",X9,"*">);
    WRITE(ERRFIL,<"*",36(" "),"*">);
    WRITE(ERRFIL,<38("*")>);
  END;
  RIGSTERMINATION;
END.
```

```
C-------------------------------------------------------------
C     THE FORTRAN-66 PACKAGE CONTAINS:
C     1. THE 'EPISODE' PACKAGE
C     2. THE SUBROUTINE 'DIFFUN' TO BE CALLED FROM EPISODE
C        FOR COMPUTING THE RIGHT-HAND SIDE OF THE DIFFE-
C        RENTIAL EQUATION SYSTEM WHICH SIMULATES THE
C        CHEMICAL REACTION SYSTEM.
C     3. THE SUBROUTINE 'PEDERV' TO BE CALLED FROM EPISODE
C        FOR COMPUTATION  OF THE JACOBIAN MATRIX OF THE
C        RIGHT-HAND SIDE OF THE DIFFERENTIAL SYSTEM.
C     4. THE SUBROUTINE 'INITCM' (=INITIATION OF COMMONS)
C        WHICH TRANSFERS VARIABLES FROM THE ALGOL PROGRAM
C        TO COMMON AREAS IN THIS FORTRAN PACKAGE.
C     5. THE SUBROUTINE 'INFORM' WHICH GIVES INFORMATION
C        BACK TO THE ALGOL PROGRAM ABOUT THE CONTENTS OF
C        SOME VARIABLES IN THE FORTRAN PACKAGE.
C     6. THE SUBROUTINE  'LOCKF7' LOCKS THE DISK FILE
C        WITH THE EXTERNAL NAME, 'CHEM/FILE6' (SEE BELOW).
C        MESSAGES ABOUT THE EXECUTION OF THE INTEGRATION
C        IS ACCUMULATED IN THIS FILE AND WHEN LOCKED, IT
C        IS READY FOR OUTPUT VIA THE ALGOL PROGRAM.
C
C     THE FIRST SUBPROGRAM, BLOCK GLOBALS SUBPROGRAM, IS
C     USED FOR DECLARATION OF OUTPUT FILE, FILE6, AND FOR
C     DECLARATION OF THE EXPORT LIST.(SEE BELOW).
C
C     THIS FORTRAN PACKAGE IS COMPILED ON A DISK FILE
C     WITH THE EXTERNAL NAME: CHEMSIMUL/FORLIB.
C     BY THE PROGRAM:
C
C     BEGIN
C       JOB CHEMFORLIB; CLASS=2;
C       USER=USERNAME/PASSWORD; CHARGE=NNNNNNN;
C       JOBSUMMARY = SUPPRESSED;
C       COMPILE CHEMSIMUL/FORLIB FORTRAN LIBRARY;
C       JOBSUMMARY = SUPPRESSED;
C       COMPILER EBCDIC CARD
C     $ RESET LIST
C     $ INCLUDE "CHEMSIMUL/FORSUB"
C     ?END JOB
C
C     USERNAME/PASSWORD IS SELF-EXPLANATORY, NNNNNNN IS
C     THE USERS SEVEN-DIGIT CHARGE-CODE IN THE COMPUTER
C     SYSTEM.
C
C     CAUTION:
C     THE MAXIMUM NUMBER OF DIFFERENTIAL EQUATIONS WHICH
C     CAN BE SOLVED WITH THIS PACKAGE IS 100.FOR SOLVING
C     GREATER SYSTEM, SOME CHANGES MUST BE DONE IN THE
C     SUBROUTINE DRIVE. FOR THESE CHANGES, THE USER IS
C     RECOMMENDED TO READ THE DESCRIPTION FOR EPISODE
C     WHICH IS FOUND IN REF. [1] IN THE ALGOL PART.
C
C     ETERM, ERK AND RGC IN COMMON STATEMENTS FOR SUBROU-
C     TINES, INITCM, DIFFUN AND PEDERV MUST ALSO BE
C     CHANGED IF THEY ARE TOO SMALL. THE NECESSARY CHANGES
C     WILL BE GIVEN BY MESSAGES FROM THE ALGOL PROGRAM.
C     ALSO CHANGE THE ARGUMENT IN THE IF.... GO TO 440
C     STATEMENT (AFTER THE COMMON STATEMENTS) IN THE
C     SUBROUTINE DRIVE.
C
C-------------------------------------------------------------
$RESET LIST SET LINEINFO OWN
FILE  7(KIND=DISK,TITLE='INTEGRATIONMSG.',NEWFILE=TRUE)
```

```
      BLOCK GLOBALS
      EXPORT DRIVE,INFORM,INITCM,LOCKF7
      END

      SUBROUTINE INITCM (NR,NTI,NT5I,ETERMI,ERKI,RGCI,RADI,
     1INITOK)
      INTEGER NR, NTI, NT5I, ETERMI
      REAL ERKI, RGCI
      LOGICAL RADI,RAD,INITOK
      DIMENSION ETERMI(NT5I), ERKI(NTI), RGCI(NR)

      COMMON/CHEMS1/ RAD
      COMMON/CHEMS2/ NT,NT5
      COMMON/CHEMS3/ ETERM(2000)
      COMMON/CHEMS4/ ERK(400)
      COMMON/CHEMS5/ RGC(100)

C-------------------------------------------------------------
C     THE SUBROUTINE TRANSFERS THE CONTENTS OF THE PARAME-
C     TERS TO THE COMMON VARIABLES.
C     INTEGER
C       NT      : THE DIMENSION OF ERK
C       NT5     : THE DIMENSION OF ETERM
C     REAL
C       ETERM   : THE ARRAY HOLDING THE INFORMATION FOR
C                 USED BY SUBROUTINES DIFFUN AND PEDERV.
C       ERK     : ARRAY, HOLDS STOICHIOMETRIC CONSTANTS
C               : MULTIPLIED WITH RATE CONSTANTS.
C       RGC     : ARRAY, HOLDS G-VALUES MULTIPLIED WITH
C               : RATE CONSTANTS.
C     LOGICAL
C       INITOK : SET TO TRUE BY THE SUBROUTINE IF THE
C              : COMMON AREAS ARE GREAT ENOUGH, ELSE
C              : SET TO FALSE.
C-------------------------------------------------------------

C     CHECK THE COMMON STATEMENTS FOR ETERM,ERK AND RGC

      INITOK = .TRUE.
      IF (NR.LE.100.OR.
     1    NTI.LE.400.OR.
     2    NT5I.LE.2000) GO TO 5

      INITOK = .FALSE.
      RETURN
 5    RAD = RADI
      NT    = NTI
      NT5   = NT5I
      DO 10 I = 1,NT
10    ERK(I)= ERKI(I)
      DO 20 I = 1,NR
20    RGC(I)= RGCI(I)
      DO 30 I = 1, NT5
30    ETERM(I)=ETERMI(I)

      RETURN
C------------- END OF SUBROUTINE INITCM ------------------
      END

      SUBROUTINE INFORM(HMI,HMA,HUS,NQUS,NST,NF,NJ)
      INTEGER NQUS,NST,NF,NJ
      REAL HMI,HMA,HUS

      COMMON/EPCOM1/DUMMY1(2), HMIN, HMAX, DUMMY2(7)
```

```
      COMMON/EPCOM9/HUSED,NQUSED,NSTEP,NFE,NJE

C-------------------------------------------------------
C     THE SUBROUTINE TRANSFERS THE CONTENTS OF SOME
C     COMMON VARIABLES TO THE SUBROUTINE PARAMETERS.
C-------------------------------------------------------

      HMI   = HMIN
      HMA   = HMAX
      HUS  = HUSED
      NQUS = NQUSED
      NST  = NSTEP
      NF    = NFE
      NJ    = NJE
      RETURN
C------------- END OF SUBROUTINE INFORM ------------------
      END

      SUBROUTINE DIFFUN(NR, T, Y, YDOT)
      INTEGER NR,NT,ETERM,N1,N2,N4,N5,NT5,N
      REAL T,Y,YDOT,ERK,RGC,Y4,Y5
      DIMENSION Y(NR),YDOT(NR)
      LOGICAL RAD

      COMMON/CHEMS1/ RAD
      COMMON/CHEMS2/ NT,NT5
      COMMON/CHEMS3/ ETERM(2000)
      COMMON/CHEMS4/ ERK(400)
      COMMON/CHEMS5/ RGC(100)

C-------------------------------------------------------
C     THE SUBROUTINE COMPUTES THE DERIVATIVES, YDOT(N)
C     FOR N  = 1, 2, 3, ......,NR.
C
C     NR    IS THE NUMBER OF DIFFERENTIAL EQUATIONS, WHICH
C           IS EQUAL TO THE NUMBER OF REACTANTS.
C     T     IS THE TIME (NOT USED EXPLICITELY IN THIS
C           SUBROUTINE).
C     Y     IS THE ARRAY HOLDING THE VALUES OF THE REACTANT
C           CONCENTRATIONS.
C     YDOT IS THE ARRAY HOLDING THE COMPUTED DERIVATIVES
C           OF Y.
C
C     THE VARIABLES IN THE COMMON AREA HAS THE SAME NAME
C     AS THE CORRESPONDING VARIABLES IN THE ALGOL PROGRAM
C     AND FOR THE DESCRIPTION OF THESE THE USER IS REFER-
C     RED TO THE DESCRIPTION THERE.
C-------------------------------------------------------

C-------------------------------------------------------
C     INITIATE YDOT(N) WITH 0 (ZERO) OR WITH RGC(N)
C     IF ZERO'TH ORDER REACTION(S) EXISTS.
C-------------------------------------------------------
      DO 10 N=1,NR
       IF (.NOT.RAD) GO TO 5
       YDOT(N)=RGC(N)
       GO TO 10
    5  YDOT(N)=0.0
   10 CONTINUE

      N2= -5
      DO 20 N=1,NT
       N2=N2+5
```

```
C      ETERM(N2+1) HOLDS ADDRESS FOR REACTANT
C      CONCENTRATION WHICH DERIVATIVE IS COM-
C      PUTED.
       N1=ETERM(N2+1)

C      ETERM(N2+4) HOLDS ADDRESS FOR FIRST REACTANT
C      CONCENTRATION IN THE PRODUCT.
       N4=ETERM(N2+4)

C      ETERM(N2+5) HOLDS THE ADDRESS FOR SECOND REACTANT
C      CONCENTRATION IN THE PRODUCT.
       N5=ETERM(N2+5)
       IF (N4.NE.0) GO TO 30
       Y4=1.0
       GO TO 40
   30  Y4=Y(N4)
   40  IF (N5.NE.0) GO TO 50
       Y5=1.0
       GO TO 60
   50  Y5=Y(N5)
   60  YDOT(N1)=YDOT(N1) + ERK(N) * Y4 * Y5
   20 CONTINUE
      RETURN
C------------- END OF SUBROUTINE DIFFUN ------------------
      END

      SUBROUTINE PEDERV(NR, T, Y, PD, NO)
      INTEGER NR,NO,NT,NT5,ETERM,I,J,W1,W4,W5
      REAL T,Y,PD,ERK,W23
      DIMENSION Y(NR),PD(NO,NO)

      COMMON/CHEMS2/ NT,NT5
      COMMON/CHEMS3/ ETERM(2000)
      COMMON/CHEMS4/ ERK(400)
      COMMON/CHEMS5/ RGC(100)

C-------------------------------------------------------------
C     THE SUBROUTINE, PEDERV, COMPUTES THE JACOBIAN
C     MATRIX OF THE RIGHT-HAND SIDE OF THE DIFFEREN-
C     TIAL EQUATION SYSTEM, AND STORES THE MATRIX
C     ELEMENTS IN THE ARRAY PW.
C
C-------------------------------------------------------------
      DO 10 I=1,NO
      DO 20 J=1,NO
   20  PD(I,J)=0.0
   10 CONTINUE

      J= -5
      DO 80 I=1,NT
       J=J+5
       W23=ERK(I)
       W1=ETERM(J+1)
       W4=ETERM(J+4)
       W5=ETERM(J+5)
       IF (W4.EQ.0) GO TO 30

C      W4.NE.0

       IF (W5.NE.0) GO TO 40
       WY=1.0
       GO TO 50
   40  WY=Y(W5)
   50  PD(W1,W4) = PD(W1,W4) + W23 * WY
```

```
C       W4.EQ.0

   30   IF (W5.EQ.0) GO TO 80
        IF (W4.NE.0) GO TO 60
        WY=1.0
        GO TO 70
   60   WY=Y(W4)
   70   PD(W1,W5) = PD(W1,W5) + W23 * WY
   80 CONTINUE
      RETURN
C------------- END OF SUBROUTINE PEDERV ------------------
      END

      SUBROUTINE LOCKF7
      LOCK 7
      RETURN
      END

C-------------------------------------------------------------
C
C     NOW FOLLOWS THE FORTRAN PACKAGE, EPISODE,
C     IN SINGLE PRECISION.
C     THE FIRST SUBROUTINE, DRIVE, IS THE DRIVER
C     FOR THE EPISODE PACKAGE. THE FOLLOWING
C     CHANGES, COMPARED TO THE ORIGINAL VERSION,
C     IS DONE:
C
C     THE PARAMETER, HMAXO, GIVING THE MAXIMUM
C     INTEGRATION STEP TO BE USED IS INTRODUCED
C     AND HMAXO IS ADDED IN THE DECLARATION:
C
C     DOUBLE PRECISION EPS,HO,TOUT,TO,YO, HMAXO
C
C     IN THE BEGINNG OF THE SUBROUTINE DRIVE.
C
C     FOLLOWING 3 FORTRAN STATEMENT IS CHANGED:
C
C     HMAX = DABS(TO - TOUT)*TEN
C     IS CHANGED TO
C     HMAX = HMAXO
C
C 20  HMAX = DABS(TOUT - TOP)*TEN
C     IS CHANGED TO
C 20  HMAX = HMAXO
C
C 25  HMAX = DABS(TOUT - TOP)*TEN
C     IS CHANGED TO
C 25  HMAX = HMAXO
C
C     THE ORIGINAL PARAMETERS, DIFFUN AND PEDERV, ARE RE-
C     MOVED TOGETHER WITH THE EXTERNAL DECLARATIONS
C
C     EXTERNAL DIFFUN, PEDERV
C
C     IN ALL SUBROUTINES INCLUDING  DRIVE. THIS IS DONE
C     BECAUSE IT IS AT THE PRESENT TIME NOT ALLOWED TO
C     HAVE SUBROUTINE PARAMETERS IN LIBRARY.
C
C-------------------------------------------------------------
```

Sales distributors:
Jul. Gjellerup, Sølvgade 87,
DK-1307 Copenhagen K, Denmark

Available on exchange from:
Risø Library, Risø National Laboratory,
P.O.Box 49, DK-4000 Roskilde, Denmark

ISBN 87-550-0984-0
ISSN 0106-2840