

Technical University of Denmark



COLUMN2 - A Computer Program for Simulating Migration

Nielsen, Ole John; Bo, P.; Carlsen, Lars

Publication date:
1985

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

[Link back to DTU Orbit](#)

Citation (APA):
Nielsen, O. J., Bo, P., & Carlsen, L. (1985). COLUMN2 - A Computer Program for Simulating Migration. (Denmark. Forskningscenter Risoe. Risoe-R; No. 514).

DTU Library

Technical Information Center of Denmark

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

COLUMN2

- A Computer program for simulating migration

Ole John Nielsen, Peter Bo, and Lars Carlsen

RISØ-R-514

COLUMN2

- A computer program for simulating migration

Ole John Nielsen, Peter Bo, and Lars Carlsen
Chemistry Department

Abstract. COLUMN2 is a 1D FORTRAN77 computer program designed for studies of the effects of various physicochemical processes on migration. It solves the solute transport equation and can take into account dispersion, sorption, ion exchange, first and second order homogeneous chemical reactions. Spatial variations of input pulses and retention factors are possible. The method of solution is based on a finite difference discretion followed by the application of the method of characteristics and two separate grid systems. This report explains the mathematical and numerical methods used, describes the necessary input, contains a number of test examples, provides a listing of the program and explains how to acquire the program, adapt it to other computers and run it. This report serves as a manual for the program.

INIS-descriptors: COMPUTERIZED SIMULATION, COMPUTER CODES, MANUALS, RADIONUCLIDE MIGRATION, SOLUTES, ABSORPTION, HYDROLYSIS, COMPLEXES, RISK ASSESSMENT, CHEMICAL REACTION KINETICS

Oktober 1985

Risø National Laboratory, DK 4000 Roskilde, Denmark

ISBN 87-550-1148-9

ISSN 0106-2840

Grafisk Service Center, Risø 1985

CONTENTS

	Page
Preface	5
1. INTRODUCTION	7
1.1. General	7
1.2. Reactions which may influence the migration behaviour of solutes	8
2. MATHEMATICAL DESCRIPTIONS	10
2.1. Convection and dispersion	10
2.2. Equilibrium adsorption/absorption phenomena	11
2.3. First order reactions in solution and radio- active decay	16
2.4. Second order reactions in solution	17
3. NUMERICAL METHODS	18
3.1. Introduction	18
3.2. Convection and dispersion	18
3.3. Equilibrium sorption phenomena	22
3.4. Chemical reactions in solution and radioactive decay	23
3.5. Solid phase reactions	24
4. DESCRIPTION OF INPUT	24
5. PROGRAM LIMITATIONS	27
6. TEST EXAMPLES	27
6.1. Introduction	27
6.2. Test example 1	28
6.3. Test example 2	29
6.4. Test example 3A	31
6.5. Test example 3B	33
6.6. Test example 3C	34
6.7. Test example 3D	35

	Page
6.8. Test example 3E	36
6.9. Test case 4A and 4B	37
6.10. Test example 5	39
6.11. Test example 6	40
7. HOW TO RUN COLUMN2 ON THE RISØ B7800 COMPUTER	42
8. ADAPTION OF THE CODE FOR OTHER COMPUTERS	43
9. HOW TO ACQUIRE THE COLUMN2 PROGRAM	44
10. CONCLUSION	45
REFERENCES	46
APPENDIX I: Alphabetical list of major variables, parameters, identifiers and units	47
APPENDIX II: Listing of the COLUMN2 program	51
APPENDIX III: Listing of the COLUMNINPUT program	65
FIGURES	75

PREFACE

This report presents the latest developments in a computer program constructed at Risø National Laboratory in 1978 (Bo 1978) to deal with studies of the effects of various physicochemical processes on migration. The purpose is to present the code for the program COLUMN2 and the physical and chemical concepts and assumptions used in COLUMN2. This report also serves as a manual to the program and the general subject of chemical and transport modelling has not been treated in all detail.

The basic framework of the first COLUMN program (Bo 1978) has been maintained. However, the new COLUMN2 program has been made user-friendly in contrast to the first version. COLUMN2 is written in standard Fortran 77 and has easy-to-handle input routines and has been expanded to include second order kinetics.

1. INTRODUCTION

1.1. General

The purpose of this work was first to construct a numerical framework and then a computer program to aid in the study of the effects of various physicochemical processes on migration.

Simulation of the behaviour of geochemical barriers in connection with risk assessment is not the primary use of this program, since such simulations are most often very site-specific and require complex geometrical considerations in addition to the chemical. In other cases they are so simplified that they lack both specific geometrical and chemical features. In the latter case the aquifer in question is treated as the column and the chemistry as a general physicochemical retention. Such simplified simulations are very useful and often give results with the accuracy needed without requiring an excessive amount of time and work.

When complex physicochemical processes are expected a close examination of the effects on migration is needed before the description of the process in question can be included in a more complex geochemical barrier simulation.

For purposes like this, a one-dimensional description will be adequate and will at the same time be able to simulate laboratory column experiments that are part of fundamental experimental migration studies (Carlsen et al. 1981).

For radioactive substances the decay and the decay of radionuclide chains are important processes whose influence on the migration has been described earlier (Burkholder et al. 1974). These processes can also be studied with COLUMN2. However, it should be emphasized here that COLUMN2 by no means is specially coupled to treatment of radioactive substances, but includes general coupling with more complex physicochemical processes.

The PERCOL model (Routson and Serne 1972) does treat complex chemical processes, but only in an empirical manner as far as the microcomponents are concerned. This partly excludes the identification of the physicochemical processes involved in the retention of these substances. Furthermore, the model is site-specific in the sense that it includes only a limited number of variable chemical phenomena.

The original COLUMN program has been included and reviewed in a recent comprehensive directory of geochemical computer programs (Broyd et al. 1983).

1.2. Reactions that may influence the migration behaviour of solutes.

A wide range of geochemical processes involving possible pollutants, often present as microcomponents, in the terrestrial environment can be expected a priori to influence the migration behaviour of the latter. In principle, all processes can be regarded as more or less sophisticated combinations of uni- and bimolecular reactions. However, in practice the majority of bimolecular reactions will involve the macrocomponents in the ground water, the concentrations of these being up to several orders of magnitude higher than those of the microcomponents under investigation. Hence, these reactions can be treated as pseudo first-order reactions with respect to the micro components. In Table I the different elementary reactions that should be taken into account are summarized.

Table I. Characterization of elementary reactions, which may influence the migration. (FO: first-order, PFO: pseudo first-order, SO: second order)

Reaction	Type	Forward	Backward
Radioactive Decay Chains	$A \rightarrow B \rightarrow \dots \rightarrow C$	FO	-
Sorption/Desorption	$A \rightleftharpoons B$	FO	FO
Precipitation/Dissolution	$A \rightleftharpoons B$	FO	FO
Redox Reactions ^a	$A \rightleftharpoons B$	PFO	PFO
Degradations	$A \rightarrow \sum B_i$	FO	-
-	$A+X \rightarrow \sum B_i$	SO/PFO	-
Substitutions ^b	$A+X \rightarrow B$	SO/PFO	-
Hydrolysis	$A+CH \rightarrow B$	PFO	-
Complex Formation ^c	$A+L \rightleftharpoons AL$	SO/PFO	FO
Colloid Formation ^d	$nA \rightarrow A_n$	SO	-

^a The redox potential will generally be controlled by the macro composition of the ground water (e.g. by the Fe(II)/Fe(III) couple)

^b The substitution reactions may be of importance in cases of organic pollutants

^c In the case of complex formation, which can be expected to influence the migration behaviour of metal cations significantly, it is to be expected that in most cases it can be treated as a PFO reaction, owing to a large excess of the ligand, or a constant concentration of the latter (e.g. naturally occurring ligands in ground water).

^d Colloid formation can in principle be regarded as a consecutive series of SO reactions.

Numerous combinations of the above mentioned elementary reactions can be imagined. Especially parallel reactions, e.g. as a result of concurring chemical and microbiological transformations, and consecutive reactions, e.g. hydrolysis subsequently followed by a

precipitation/dissolution reaction of the hydrolyzed species, shall be mentioned.

2. MATHEMATICAL DESCRIPTIONS

2.1. Convection and dispersion

Common to all the solutes present in the liquid phase is the transport due to convection and dispersion which we describe by:

$$\frac{\partial c_i}{\partial t} = D_i \frac{\partial^2 c_i}{\partial x^2} - V \frac{\partial c_i}{\partial x} \quad (1)$$

where c_i is the concentration of component i .

D_i is the dispersion coefficient of component i .

V is the interstitial velocity of the liquid phase
($V = q/\epsilon$)

q is the volume flow per unit cross-sectional area.

ϵ is the porosity (volume fraction of liquid) of the porous medium.

t is the time.

x is the length coordinate.

Equation (1) assumes that neither the dispersion nor the interstitial velocity changes in the x -direction. Most often the interstitial velocity will be so high that all components have the same dispersion coefficient, but at very low velocities differences between the individual components do occur as a result of their individual diffusion coefficients in the liquid phase.

It sometimes happens that it is necessary to account for a component that does not move with the fluid flow, e.g. precipitates and absorption in the solid phase under non-equilibrium conditions. This can be done by setting the dispersion coefficient and the interstitial velocity equal to zero for such components.

2.2. Equilibrium adsorption/absorption phenomena

2.2.1. General description

When, in addition to dispersion and convection, components are transferred between the liquid (flowing) and the solid (stationary) phases, the transport equation (1) is modified by adding a term which describes the effect of these processes on the concentration in the liquid phase:

$$\frac{\partial c_i}{\partial t} = D_i \frac{\partial^2 c_i}{\partial x^2} - v \frac{\partial c_i}{\partial x} - \frac{\epsilon_s}{\epsilon} \frac{\partial \bar{c}_i}{\partial t} \quad (2)$$

where \bar{c}_i is the concentration of i in the solid phase, and ϵ_s is the volume fraction of the solid phase. If the process is an adsorption on a surface, ϵ_s and \bar{c}_i has the meaning of specific surface area and surface concentration, respectively.

The form of equation (2) is especially useful for fast reversible reactions in which equilibrium between the liquid and solid phases can be assumed at all times, because in this case

$$\frac{\partial \bar{c}_i}{\partial t}$$

does not depend explicitly on time, but can be expressed in terms of c_j and

$$\frac{\partial c_i}{\partial t}$$

which relates only to the liquid phase, i.e. when

$$\bar{c}_i = f_i(c_j, \dots) \quad (3)$$

then

$$\frac{\partial \bar{c}_i}{\partial t} = \sum_i \frac{\partial f_i(c_j, \dots)}{\partial c_1} \frac{\partial c_1}{\partial t} \quad (4)$$

Expressions such as (3) and (4) gives rise to the distinction between micro-chemistry (micro-components) and macro-chemistry (macro-components) because very often a few components completely determines the value of the functions $f_i(c_j, \dots)$ because of their relatively large concentrations. This is true also for components i present in small (micro) concentrations. Use of expressions like (3) and (4) requires, however, that the components i are present in amounts that makes their properties thermodynamically well defined, i.e. not subject to statistical fluctuations that are too large. This leaves us with a third category - submicro-chemistry - that cannot be treated by thermodynamics. Instances of the importance of this type of component may be found in the migration of some radionuclides. These problems will, however, not be treated here.

2.2.2. Ion Exchange

Ion exchange is one of the more important mechanisms by which positively charged ions are retained in soil. The exchange process is relatively rapid and reversible. Therefore the assumption of maintaining equilibrium at all times during the migration is most often fulfilled.

The equilibrium is determined by the equality of the electrochemical potentials in the ion exchange phase and in the liquid phase for each of the components present. Expressed in terms of concentrations this equilibrium can be written:

$$\bar{c}_i = c_i \frac{\gamma_i}{\bar{\gamma}_i} K_i Z^{z_i/\omega} \quad (5)$$

where

$$Z = \exp(\omega F(\psi - \bar{\psi}^-)/RT) \quad (6)$$

$$K_i = \exp((\mu_i^O - \bar{\mu}_i^O)/RT) \quad (7)$$

and γ_i is the activity coefficient of component i
 z_i is the valence factor of the component i
 ω is the change factor for the fixed ions in the exchange phase (-1 for a cation exchanger)
 μ_i^O is the standard chemical potential of component i
 ψ is the electrical potential of the phases
 F is Faradays number
 R is the gas constant
 T is the absolute temperature
a bar ($\bar{\quad}$) over a symbol refers to a property in the ion exchange phase.
 K : distribution coefficient
 Z : potential function

Z is determined by the condition of electroneutrality in the ion exchange phase:

$$\sum_j z_j \bar{c}_j + \omega X = 0 \quad (8)$$

where X is the concentration of the fixed ionic groups in the exchanger phase. Inserting (5) in (8) gives a polynomial expression in Z which can be solved when the constants and the concentrations in the liquid phase are known.

With the above expressions for the equilibrium we find:

$$\frac{\partial \bar{c}_i}{\partial t} = K_i \frac{\gamma_i}{\bar{\gamma}_i} Z^{z_i/\omega} \frac{\partial c_i}{\partial t} + \frac{z_i}{\omega} c_i K_i \frac{\gamma_i}{\bar{\gamma}_i} Z^{z_i/\omega} \frac{\partial \ln Z}{\partial t}$$

which with

$$K_{Di} = K_i \frac{\gamma_i}{\bar{\gamma}_i} Z^{z_i/\omega} \quad (9)$$

gives

$$\frac{\partial \bar{c}_i}{\partial t} = K_{Di} \frac{\partial c_i}{\partial t} + \frac{z_i}{\omega} c_i K_{Di} \frac{\partial \ln Z}{\partial t} \quad (10)$$

When inserted into the migration Eq. (2) with

$$A_i = 1 + K_{Di} \frac{\epsilon_s}{\epsilon} \quad (11)$$

where A_i is inverse retention factors, we get

$$\frac{\partial c_i}{\partial t} = \frac{D_i}{A_i} \frac{\partial^2 c_i}{\partial x^2} - \frac{v}{A_i} \frac{\partial c_i}{\partial x} - c_i \frac{A_i - 1}{A_i} \frac{\partial \ln Z}{\partial t} \quad (12)$$

in which

$$\frac{\partial \ln Z}{\partial t}$$

can be expressed in terms of

$$\frac{\partial c_j}{\partial t}$$

through

$$\frac{\partial \ln Z}{\partial t} = \sum_i \frac{1}{Z} \frac{\partial Z}{\partial c_j} \frac{\partial c_j}{\partial t}$$

or using the electroneutrality condition in the form

$$\sum_j z_j \frac{\partial \bar{c}_j}{\partial t} = 0 .$$

The microchemistry, i.e. the components that need not be considered when calculating Z , can often be described by the simplified equation:

$$\frac{\partial c_i}{\partial t} = \frac{D_i}{A_i} \frac{\partial^2 c_i}{\partial x^2} - \frac{v}{A_i} \frac{\partial c_i}{\partial x} \quad (13)$$

However, this requires either that the macrochemistry (and therefore Z) be independent of time or A_i be very close to 1.

Finally it should be mentioned that neither Z nor the individual K_i 's are thermodynamically well defined and, in the actual calculation, should be replaced by products and ratios of these parameter that are thermodynamically well defined.

2.2.3. Other types of sorption

When the sorption isotherm is known:

$$\bar{c}_i = f_i(c_j, \dots) \quad (3)$$

as a function of all components present, the time derivative can be found

$$\frac{\partial \bar{c}_i}{\partial t} = \frac{\partial f_i(c_j, \dots)}{\partial c_l} \frac{\partial c_l}{\partial t} \quad (4)$$

$$\text{Defining } K_{Di} = \frac{f_i(c_j, \dots)}{c_i} \text{ and } A_i = 1 + \frac{\epsilon_s}{\epsilon} K_{Di} \quad (14)$$

We get the transport equation:

$$\begin{aligned} \frac{\partial c_i}{\partial t} = & \frac{D_i}{A_i} \frac{\partial^2 c_i}{\partial x^2} - \frac{v}{A_i} \frac{\partial c_i}{\partial x} + \frac{A_i - 1}{A_i} \frac{\partial c_i}{\partial t} \\ & - \frac{\epsilon_s}{\epsilon} \sum_l \frac{\partial f_i(c_j, \dots) / \partial c_l}{A_i} \frac{\partial c_l}{\partial t} \end{aligned} \quad (15)$$

This again takes the form (13) if for all $l \neq i$ either

$$\frac{\partial f_i(c_j, \dots)}{\partial c_l} = 0, \text{ or } \frac{\partial c_l}{\partial t} = 0, \text{ and for } l = i$$

$$\frac{\partial f_i(c_j, \dots)}{\partial c_l} = \frac{f_i(c_j, \dots)}{c_i}$$

The first and last condition is likely to hold true for many microcomponents whereas the second will sometimes be a good approximation for macrocomponents.

2.3. First order reactions in solution and radioactive decay

Homogeneous chemical reactions and that part of heterogeneous reactions that concerns dissolved species and radioactive decay are introduced into the transport equation by adding terms of the form

$$\sum_l \sum_j S_{ij}^l c_j^{a(l,j)}$$

where l represents the chemical reaction in question and j is a component in this reaction which enters the reaction rate for component i with the coefficient S_{ij}^l and the exponent $a(l,j)$.

For first order reactions and radioactive decay the above expression simplifies to

$$\sum_l S_{il} c_l$$

since, in this case, the reaction can be identified with the component involved in the reaction. This expression gives a transport equation of the form:

$$\frac{\partial c_i}{\partial t} = \frac{D_i}{A_i} \frac{\partial^2 c_i}{\partial x^2} - \frac{v}{A_i} \frac{\partial c_i}{\partial x} + \sum_l \frac{S_{il}}{A_i} \tag{16}$$

in which also retention due to sorption, in its simplest form, has been taken into account.

Since

$$S_{il} = \left(\frac{\partial c_i}{\partial t} \right)^l \frac{1}{c_i} \tag{17}$$

where the superscript l means that part of the derivative that is due to the reactor l , S_{il} is, for a simple chemical reaction, a normal rate constant.

For radioactive decay, which is going on both in the liquid and solid phases

$$S_{i1} = \lambda_1 + \lambda_1 \frac{\epsilon_s}{\epsilon} K_{D1} = \lambda_1 A_1 \quad (18)$$

where λ_1 is the decay constant - taken positive for parent radionuclides ($l \neq i$) and negative for the component i itself. The two terms in the middle part of (18) arises because of decay in the liquid phase and of reestablishing of equilibrium between and solid phase following decay in the solid phase.

An other important application of these equations is to pseudo-first order reactions in which e.g. a complexing agent (L^{z_1}) reacts with positively charged ions (M^{z_m}) to form another charged species



with a retention (A_i) different from the simple ion. When the concentration of the complexing agent is high compared with the concentration of the ion M^{z_m} , the reactions (19) can be treated as first order reactions. Hydrolysis and precipitation of micro-components in a buffered system or one in which the concentration of the precipitating component is high compared with the concentration of the microcomponent can also, at least to a certain extent, be treated as first order reactions.

2.4. Second order reactions in solution

The second order reactions are not introduced directly into the transport equation but treated seperately. Terms of the form

$$SR_r \times R1_r \times R2_r \times dt$$

are subtracted and added, respectively, to the concentrations of the relevant components for each reaction, r , where

SR_r is the second order rate constant
 $R1_r$ is the concentration of the first reactant
 $R2_r$ is the concentration of the second reactant.

3. NUMERICAL METHODS

3.1. Introduction

When making a program for the calculation of migration phenomena, a choice must be made: should we plan a large program, comprising all possible phenomena, in which the problem to be studied in each specific case is selected in the input, leaving the program proper unmodified at all times ? Or should we plan a program comprising only the basic scheme common to all the individual problems to be solved, leaving the user of the program modify it according to his needs.

For the following two reasons we have chosen the second of the above-mentioned alternatives:

- 1) We cannot possibly foresee all the problems, hopefully to be solved with the program, and their numerical peculiarities
- 2) Solving specific detail problems with an all comprising program is often a waste of computer time, and will often leave the user in doubt as to whether the program function, all right for the specific problem thus necessitating a number of test calculations with the large program.

3.2. Convection and Dispersion

Setting up algorithms to solve Eq. (1) for one dimensional transport by convection and dispersion, we have to choose: (a) between finite difference and finite elements methods, and (b) between explicite and implicite algorithms. The criteria for the

choice should be both ease of handling modifications of the program, and minimizing the computer time needed for the problems. This, in our opinion, leads to choose finite difference explicit algorithms. They are the easiest to modify but not necessarily the fastest for calculation. However, the use of a modified method of characteristics to solve the hyperbolic part of the problem ($V\partial c_i/\partial x$) to some extent compensates for this.

Using methods of characteristics, Eqs. (1) or (13) is rearranged thus:

$$\frac{dc_i}{dt} = \frac{\partial c_i}{\partial t} + \frac{v}{A_i} \frac{\partial c_i}{\partial x} = \frac{D_i}{A_i} \frac{\partial^2 c_i}{\partial x^2} \quad (20)$$

where the substantial derivative on the left is the concentration change in a local liquid volume that follows the movement (V/A_i) of the component i in the liquid phase. In this way the stability problem with the otherwise hyperbolic equation is circumvented - but it necessitates the introduction of a set of grid points (space points at which concentrations etc. are calculated) that are moving with the velocity V/A_i relative to the stationary phase - i.e. the column. In many applications of the method of characteristics the actual numerical integration is, however, performed on an equidistant stationary set of grid points. This gives the following integration cycle:

- 1) Calculate the concentrations at the fixed grid points from the values at the moving points. This is done by averaging over flowing grid points that are within half a fixed grid point interval from a given fixed grid point.
- 2) Perform the numerical integration on the equidistant fixed grid points.
- 3) Calculate the new concentrations on the moving grid points by adding the increment found in 2) for each of the fixed grid points, to all the moving grid points within half a fixed grid point interval from the given fixed grid point.
- 4) Move the flowing grid points by adding to their corresponding x -values - V/A_i times the time increment.

On the average this calculating procedure gives very accurate results but has a tendency to give local (in space) errors when sharp concentration profiles on the moving grid point pass a fixed grid point.

The logical way to remove these difficulties is to dispense with the fixed grid points and perform the integration on the moving grid points. This makes the integration cycle consist of only two operations:

- 1) Perform the integration on the moving grid points.
- 2) Move the grid points by adding V/A_i times the time increment to their x-values, thus keeping track on the whereabouts of the different components.

This sounds as a greater simplification than it actually is because of two types of phenomena:

The first is that A_i may vary explicitly or implicitly with x-coordinate in the column making flowing grid points crowd at some places and spread at others. This necessitates an integration procedure that is able to work with grid points that are not equidistant.

The second is that when the different components are coupled e.g. through chemical reactions in the liquid phase, and they do not move at the same speed relative to the stationary phase, then one components concentration has to be calculated at other components grid points. This is done by interpolation.

The first order finite difference representation of first and second order spatial differential coefficients for nonequidistant grid points, as used here, are:

$$\left(\frac{\partial c}{\partial x}\right)_{x(L)} = c_L' = \frac{1}{2} \left(\frac{c_{L+1} - c_L}{h_{L+}} + \frac{c_L - c_{L-1}}{h_{L-}} \right) \quad (21)$$

$$\left(\frac{\partial^2 c}{\partial x^2}\right)_{x(L)} = c_L'' = \frac{2}{h_{L+} + h_{L-}} \left(\frac{c_{L+1} - c_L}{h_{L+}} - \frac{c_L - c_{L-1}}{h_{L-}} \right) \quad (22)$$

Both reduce to first order central differences approximations for $h_{L+} = h_{L-}$. The terminology can be inferred from Fig. 3.1. For simplicity of exposition the component identification subscript has been omitted.

Using a forward difference for the time derivative:

$$\left(\frac{dc}{dt}\right)_{x(L)} = \frac{c_{L,n+1} - c_{L,n}}{k} \quad (23)$$

where n enumerates the time and k is the time interval, and introducing the abbreviation:

$$H = \frac{2 h_{L-}}{h_{L+} + h_{L-}}$$

we get for the differential equation:

$$\frac{dc}{dt} = \frac{D}{A} \frac{\partial^2 c}{\partial x^2}$$

the explicit finite difference expression:

$$c_{L,n+1} = c_{L,n} + \frac{D-k}{A \times h_{L+} \times h_{L-}} (H(c_{L+1,n} - c_{L,n}) - (2-H)(c_{L,n} - c_{L-1,n})) \quad (24)$$

from which the concentration profile at time $(n+1)k$ can be calculated from the profile at time $n+k$.

At the boundary $x=0$, the input concentration to the column must be specified as a function of time i.e. $c_0 = g(t)$. This is the mathematical boundary condition at this point. Since integration will be made between grid points normally, and since we cannot be sure whether we have a grid point at $x=0$ or not, the inte-

gration for the first point in the column, i.e. with $X(I, J+1)$:



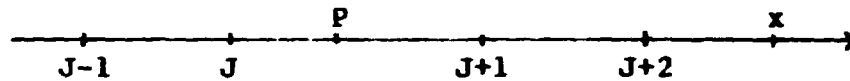
is made using (24) but substituting $x=0$ and c_0 for the point at $L-1$.

At the end of the column of length P , i.e. at $x=P$ the mathematical boundary condition is:

$$\left(\frac{\partial c}{\partial x}\right)_P = 0 .$$

Since we do not always have a grid point at $x=P$ we have to make an interpolation:

$$c'_P = c'_J + (P - x_J)(c'_{J+1} - c'_J)/(x_{J+1} - x_J) = C .$$



which results, using (21), in the following expression for $c_{J+2, n+1}$:

$$c_{J+2, n+1} = c_{J+1, n+1} - \frac{x_{J+2} - x_{J+1}}{P - x_J} (c_{J+1, n+1} - c_{J, n+1}) + \frac{P - x_{J+1}}{P - x_J} \frac{x_{J+2} - x_{J+1}}{x_J - x_{J-1}} (c_{J, n+1} - c_{J-1, n+1}) \quad (25)$$

This means that the normal integration using (24) must be carried out up to and including the grid point $J+1$.

3.3. Equilibrium sorption phenomena

For ion exchange and other equilibrium sorption phenomena, A_i is calculated in a subroutine prepared especially for the specific problem at hand.

As seen from Eqs. (12) and (15) it is sometimes necessary to solve a set of partial differential equations that are coupled through their partial time derivatives (not through the substantial time derivatives). The correct way to solve this problem would be to solve the resulting m equations in m unknown partial time derivatives ($\partial c_i / \partial t$). Such a procedure does not fit into the general scheme used here, where only the substantial derivative is calculated. It is furthermore likely to give stability problems because of the mixed parabolic ($\partial^2 c_i / \partial x^2$) and hyperbolic ($\partial c_i / \partial x$) nature of the set of equations that result from considering the partial time derivatives as the unknowns.

An approximation that is numerically easier to handle is to substitute the value of the partial time derivatives that can be calculated from the known concentration profiles at time $(n-1) \times k$ and $n \times k$ - for the ones that appear on the right side of (12) and (15) and should correspond to the time interval $n \times k$ to $(n+1) \times k$. This adds to the right side of (24), considered to describe the concentration of component i , a term of the form:

$$\sum_j Q_{ij} \times k \times (c_{j, x_i}(L), n - c_{j, x_x}(L), n-1) \quad (26)$$

where the subscript " $x_i(L)$ " means that the concentration of component j should be evaluated at this x -value which is a grid point for component i but most often not for component j when $j \neq i$. In the same way Q_{ij} , whose meaning depends on the specific problem to be solved, should be evaluated at the same place.

3.4. Chemical reactions in solution and radioactive decay

First order chemical reactions in solution and radioactive decay adds to the right side of (24) a term of the form:

$$\sum_i \frac{S_{i1} - k}{A_i} c_{1, x_i}(L), n$$

where the meaning of S_{i1} for the different processes has already been discussed.

3.5. Solid phase reactions

When non-equilibrium sorption processes, precipitation or other processes, in which the amount of a component in the solid phase is not determined solely by the concentrations that, at the time of interest, exists in the liquid phase, then it is necessary to keep track on the amount of this component. This is because, it is impossible to dissolve as precipitate that is no longer there! If the transfer of the component in question, back to solution, is not in some way proportional to a positive power of the concentration in the solid phase, then logical statements has to decide whether, at a given time and place, it is possible to transfer more of the component from the solid to the liquid phase. The meaning of "concentrations" and possibly ϵ_g has to be redefined according to the needs in each specific case. The fact that such components do not move with the liquid phase is taken care of by setting the interstitial liquid velocity (for these components only) and their dispersion coefficients equal to zero.

4. DESCRIPTIONS OF INPUT

COLUMN2 is designed to perform only one simulation in a single computer run. The program reads the input from a diskfile, file 1 with the name IFILE.

A special program, COLUMNINPUT, has been constructed to set up input files in a very easy way. COLUMNINPUT uses a finite automatic table generator system (FATGS) for manipulation of input. An input file can be constructed interactively starting from scratch. Or you can enter one input file already constructed, modify the input, and save it as another input file on disk. This is very useful when doing long series of parameter studies. COLUMNINPUT has been listed in appendix III.

All input are read in free field format. Required input begins with a title card followed by an output option card. The program can be run both interactively, which means that output plots are obtained on a graphics terminal, or in batch mode, which means that you will submit the run as a job and get the output plot on some external plotter, e.g. lineprinter or calcomp plotter or stored in a plotfile for later inspection. The present version of COLUMN2 uses a very strong computer grafics system developed at Risø Computer Installation (Rahbek and Hansen, 1983) for handling plotting. However, a version containing only a simple lineprinter option is available on request to the authors. In the input cards remember that input reals have to contain a decimal point. Examples of input will be given in Chapter 6.

1. TITLE, Title

maximum 36 characters

2a PLNAC, Name for the concentration plot, maximum 6 characters

2b PLNAE, Name for the effluent plot, maximum 6 characters.

3. Output options, (IOUT(I), I = 1,3)

IOUT(1) = 0 , print of input

IOUT(1) = 1 , no print of input

IOUT(2) = 0 , allows the amount of computing time
stated in job card

IOUT(2) = 1 , allows a maximum of 300 seconds
computing time

IOUT(2) = 2 , allows a maximum of 60 seconds compu-
ting time

IOUT(3) = 0 , negative concentrations are set equal
to zero

IOUT(3) = 1 , negative concentrations are carried on
in calculations

IOUT(4) = 0 , integration of each component is car-
ried out

IOUT(4) = 1 , no integration of peaks

IOUT(5) = 0 , hardcopy plot in batch job
IOUT(5) = 1 , no hardcopy plot in batch job

4. N, number of components
5. M, number of points uneven integer less than 801
6. P, length of column
7. II, number of timesteps between outputting a concentration profile
8. III, number of timesteps between outputting an effluent profile
9. V, velocity
10. TMAX, maximum time
11. DCINT, integration constant - should be fixed at 100000
12. DELTAT, time between integration steps
13. Input options (IOPT(I), I = 1,7)

IOUT(I) = 0 , different parameters for different components
= 1 , all parameters, e.g. D, equal for all component

I = 1 : D

I = 2 : retention factors

I = 3 : minimum value on y-axis for concentrations

I = 4 : maximum value on y-axis for concentrations

I = 5 : minimum value on y-axis for effluents

I = 6 : maximum value on y-axis for effluents

I = 7 : first order rate constant.

14. D, dispersion coefficients
15. AS, AE, retention factor at the start and end of the column
16. YCMIN, minimum value on y-axis for concentrations
17. YCMAX, maximum value on y-axis for concentrations
18. YEMIN, minimum value on y-axis for effluents
19. YEMAX, maximum value on y-axis for effluents
20. INSR, number of second order reactions
21. R1, R2, P1, P2, SR, number of reactant 1
number of reactant 2
number of product 1
number of product 2
second order rate constants
22. S(i,j), first order rate constants for components i to j.
S(j,i) is -S(i,j)

23. Concentration input options, (ICIN(I), I = 1,N)
ICIN(I) = 0 continuous flow through column
ICIN(I) = 1 time-limited input pulse
ICIN(I) = 2 point-limited input pulse
- 24a. IFORM(I), COTIME(I), CO(I)
IFORM(I) = 0 constant input
IFORM(I) = 1 variable input that must be specified
COTIME(I), time the pulse lasts
CO(I) , value of the input concentration
- 24b. IFORM(I), IENPOI(I), ISTPOI(I)
IENPOI(I), endpoint of pulse (highest number)
ISTPOI(I), startpoint of pulse (lowest number)
25. C(III,I,ISTPOI(I)), input concentration

5. PROGRAM LIMITATIONS

As already indicated COLUMN2 is an extremely general program. The limitations lie in the dimensioning of arrays. For computational reasons the number of components has been limited to 5 and a maximum number of 10 second order reactions. However, a redimensioning of all relevant arrays will allow for any number of components and reactions desired. Arrays should never be dimensioned larger than needed in order to save computation time. 5 components and 10 second order reactions may seem as small numbers. However, larger simulations are often divided into smaller subproblems for clarification purposes. The maximum number of grid points has been set to 801. This number can also be enlarged by a redimensioning of relevant arrays which causes longer computation times.

6. TEST EXAMPLES

6.1. Introduction

The following test examples have been designed not only to illustrate how to set up input files but also to show the output and check the program in different ways. For each test example the input file is listed and the resulting plots from COLUMN2 are listed. This may be useful for comparison purposes in making COLUMN2 operate on other computer installations. The results are discussed briefly.

6.2. Test example 1

This test example which is probably the simplest one can think of is made as a test of the numerical accuracy.

```
Input data: 100 TEST CASE NO. 1
            200 PLOT1C
            300 PLOT1E
            400 1,2,1,1
            500 1
            600 401
            700 100
            800 10
            900 1
           1000 1.0
           1100 100.0
           1200 100000
           1300 2
           1400 1,1,1,1,1,1,1
           1500 0.03
           1600 1.,1
           1700 0
           1800 1
           1900 0
```

2000 1
2100 0
2200 0
2300 2
2400 0,240,236
2500 1.535

There is one component present. 401 grid points are used giving initially 200 grid points inside the column whose length is 100 m. The interstitial velocity is 1 m/year and the dispersion-coefficient is $0.03 \text{ m}^2/\text{year}$ corresponding to the diffusion coefficient in water, that is, the lowest possible.

There are no reactions going on. The timestep is 2 years and concentration profiles are collected for every 10 time steps and effluent concentration for each time step. The initial concentration profile is square from grid points 246 to 240, with a concentration of 1.535 moles/m^3 . The retention factor is equal to 1 all over the column which corresponds to no absorption on the solid phase. In Fig. 6.1 the output of this test example is shown.

For this test example there exists an analytical solution:

$$c = 1/2 c_0 (\text{erf}((h-x)/2\sqrt{Dt}) + \text{erf}((h+x)/2\sqrt{Dt})) ,$$

where erf is the error function, h is the half width of the initial concentration profile. The analytical solution is plotted on top of the output in Fig. 6.1. No difference between the numerical and the analytical solution is seen.

6.3. Test example 2

In this test example a retention factor that varies linearly with distance in the column is introduced. It varies from 1 at the entrance to 3 at the exit of the column:

$$A(I,L) = 1. + 2 \cdot X(I,L)/P .$$

Input data:

```
100 THIS IS TEST CASE NO. 2
200 PLOT2C
300 PLOT2E
400 1,2,1,1
500 2
600 401
700 100
800 10
900 1
1000 1.0
1100 250.0
1200 100000
1300 2
1400 1,0,1,1,1,1,1
1500 0.03
1600 1.,3.,1.,3
1700 0
1800 1
1900 0
2000 1
2100 0
2200 0
2300 2,0
2400 0,200,196
2500 1.535
2600 1.
```

The input data are the same as in the previous example, except for the number of components, namely, two chemically independent ones. Component 1 has a square input concentration profile from grid point 196 to 200 just in front of the column while 2 has a continuous input with a concentration equal to 1 mole/m³. From part of the terminal written output:

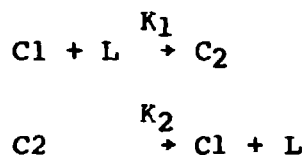
REDISTRIBUTION AT T=	0.8000E+02
REDISTRIBUTION AT T=	0.8000E+02
REARRANGEMENT OF X-AXES AT T=	0.9800E+02
REARRANGEMENT OF X-AXES AT T=	0.9800E+02
REDISTRIBUTION AT T=	0.1600E+03
REDISTRIBUTION AT T=	0.1600E+03
REARRANGEMENT OF X-AXES AT T=	0.1840E+03
REARRANGEMENT OF X-AXES AT T=	0.1840E+03
REDISTRIBUTION AT T=	0.2400E+03
REDISTRIBUTION AT T=	0.2400E+03

It is seen the two different operations on the grid points have been performed: (a) It has been necessary to redistribute the grid points every 80 years because the distance between some of the grid points gets too small. The faster-moving points at the beginning of the column approach the slower-moving ones at the end of the column. (b) It has also been necessary to rearrange the x-axis every 90 years because the first grid point gets too close to $x=0$ and many of grid points have left the column at the other end.

Figure 6.2 shows the plotted output. It is seen that the peaks lie closer and closer to each other due to the linearly increasing value of A . From the eluate plot it is seen that both the peak of component 1 and the half-maximum value of 2 both occur at $T=200$ years corresponding to a mean value of A equal to 2.

6.4. Test example 3A

In this and the five following test examples the algorithm for first and pseudo first order reactions is tested. Component 1 has retention $A(1) = 2$ due to e.g. ion exchange. It reacts with a ligand-ion present in great excess compared to component 1 at all places. The product is component 2 having no retention $A(2)=1$. The reaction is assumed reversible:



Each of these reactions can be characterized by half lives which are:

$$t_{1/2} = \ln 2 / (K_1 [L])$$

$$t_{1/2} = \ln 2 / K_2$$

respectively, where [L] is the concentration of the ligand-ion, L. A comparison of half-lives to the residence time for the two components in the column will tell whether equilibrium is maintained during the migration or not. The residence times in this case will be 200 and 100 years for component 1 and 2, respectively. Three different situations are considered, the equilibrium, intermediate, and non-equilibrium cases:

$K_1 [L] = K_2$ in years ⁻¹	$t_{1/2}$ in years	case
1	0.693	equilibrium
0.1	6.93	intermediate
0.01	63.3	non-equilibrium

Input data:

```

100 THIS IS TEST CASE NO. 3A
200 PLO3AC
300 PLO3AE
400 1,2,1,1
500 2
    
```

600 401
700 100
800 40
900 4
1000 1.0
1100 200
1200 100000
1300 0.5
1400 1,0,1,1,1,1,0
1500 0.03
1600 2.,2.,1.,1
1700 0
1800 1
1900 0
2000 1
2100 0
2200 -1.,1
2300 1.,-1
2400 2,2
2500 0,200,196
2600 1.535
2700 0,200,196
2800 1.535

The two components are put on the column in equal amounts and under equal conditions.

From the plotted output in Fig. 6.3 it is seen that the two components migrate with the same velocity equal to the mean of the velocities of the individual components. This behaviour is expected because of the high reaction rates, which maintain chemical equilibrium at all times.

6.5. Test example 3B

This test example is identical to the previous one, except that the reaction rates are changed to shift the equilibrium in favour of component 2.

Input data:

```
100 THIS IS TEST CASE NO. 3B
200 PLO3BC
300 PLO3BE
400 1,1,1,1
500 2
600 401
700 100
800 40
900 4
1000 1.0
1100 200
1200 100000
1300 0.5
1400 1,0,1,1,1,1,0
1500 0.03
1600 2.,2.,1.,1
1700 0
1800 1
1900 0
2000 1
2100 0
2200 -2.,1
2300 2.,-1
2400 2,2
2500 0,200,196
2600 1.535
2700 0,200,196
2800 1.535
```

From the plotted output in Fig. 6.4 it is seen that the relative amounts are shifted to the equilibrium ratio (1:2) and also the common migration velocity is increased in favour of component 2 compared with the previous example.

6.6. Test example 3C

This is the intermediate case.

Input data:

```
100 THIS IS TEST CASE NO. 3C
200 PLO3CC
300 PLO3CE
400 1,2,1,1
500 2
600 401
700 100
800 40
900 4
1000 1.0
1100 280
1200 100000
1300 0.5
1400 1,0,1,1,1,1,0
1500 0.03
1600 2.,2.,1.,1
1700 0
1800 1
1900 0
2000 1
2100 0
2200 -.1,.1
2300 .1,-.1
2400 2,2
2500 0,200,196
2600 1.535
2700 0,200,196
2800 1.535
2900 210
```

From the plotted output in Fig. 6.5 it is seen that the effect is to broaden the eluate and the concentration profiles. The ini-

tially produced skewness in the concentration profiles is damped during migration. The original skewness is due to the different retention of the two components.

6.7. Test example 3D

In this test example the half-lives of the reactions is comparable to the residence time in the column.

Input data:

```
100 THIS IS TEST CASE NO. 3D
200 PLO3DC
300 PLO3DE
400 1,2,1,1
500 2
600 401
700 100
800 40
900 4
1000 1.0
1000 260
1200 100000
1300 0.5
1400 1,0,1,1,1,1,0
1500 .03
1600 2.,2.,1.,1
1700 0
1800 1
1900 0
2000 1
2100 0
2200 -.01.,01
2300 .01,-.01
2400 2,2
2500 0,200,196
2600 1.535
2700 0,200,196
2800 1.535
```

From the plotted output in Fig. 6.6 it is seen that the compounds show peaks that move with the velocities of the individual components. Besides these peaks the components have a "tail" in front and behind the peak, respectively.

6.8. Test example 3E

The only change here compared to example 3D is that of the relative reaction rates.

Input data:

```
100 THIS IS TEST CASE NO. 3E
200 PLO3EC
300 PLO3EE
400 1,2,1,1
500 2
600 401
700 100
800 40
900 4
1000 1.0
1100 260
1200 100000
1300 0.5
1400 1,0,1,1,1,1,0
1500 0.03
1600 2.,2.,1.,i
1700 0
1800 1
1900 0
2000 0.3
2100 0
2200 -.02, .01
2300 .02, -.01
2400 2,2
2500 0,200,196
2600 1.535
2700 0,200,196
```

It is seen from the plotted output in Fig. 6.7 that this gives rise to a change in the relative amounts of the two components, as expected.

6.9. Test case 4A and 4B

In order to test the retention subroutines, test examples 4A and 4B were constructed.

Input data:

100 TEST CASE NO. 4A: A=1; V=1;	100 TEST CASE NO. 4B: A=10; V=10;
200 AV1C	200 AV1C
300 AV1E	300 AV1E
400 1,2,1,1	400 1,2,1,1
500 1	500 1
600 401	600 401
700 100	700 100
800 20	800 20
900 1	900 1
1000 1	1000 10.0
1100 200.0	1100 200
1200 100000	1200 100000
1300 1	1300 1
1400 1,1,1,1,1,1,1	1400 1,1,1,1,1,1,1
1500 0.03	1500 0.03
1600 1.,1	1600 10.,10
1700 0	1700 0
1800 1	1800 1
1900 0	1900 0
2000 1	2000 1
2100 0	2100 0
2200 0	2200 0
2300 2	2300 2
2400 0,205,201	2400 0,205,201
2500 1.0	2500 1.0

The only difference between these two inputs is that 4A has $V=1$ and $A=1$, and 4B has $V=10$ and $A=10$. In Fig. 6.8 the plotted output is shown with the two eluate plots superimposed. From the eluate plots it is seen that the peak leaves the column at the same time in both cases as they must. Also it is seen that the concentration profiles are narrower in case 4B, where only one tenth of the concentration is available for dispersion.

6.10. Test example 5

In test examples 5A, 5B, and 5C the algorithm for second order reactions alone are tested. Components 1 and 2 react to give 3 and 4 and vice versa. The input for the three cases are identical except for the concentration input. In 5A all 4 components have been put on the column in equal concentrations. In 5B only components 1 and 2 are added in equal concentrations and in 5C only components 3 and 4 are added to the column also in equal concentrations.

Input data:

```
100 TEST EXAMPLE 5A
200 F4C
300 F4E
400 0,0,1,0
500 4
600 401
700 800
800 100
900 4
1000 5
1100 200
1200 10000
1300 .2
1400 1,1,1,1,1,1,1
1500 .1
1600 1.,1
1700 0
```

1800 1
1900 0
2000 2
2100 2
2200 1,2,3,4,.02
2300 2,3,1,2,.92
2400 0
2500 1,1,1,1
2600 1,1,..5
2700 1,1,..5
2800 1,1,..5
2900 1,1,..5

The plotted output from these three test examples are all shown in Fig. 6.9. Since all components are already in equilibrium when introduced in the column in 5A it looks like they proceed through the column as if no reactions were taking place. In 5B only component 1 and 2 are added in twice the concentration as in 5A. Here component 3 and 4 build up through the column and the same equilibrium is obtained at the end of the column where the four peaks are identical to what is seen in 5A. In 5C components 3 and 4 are added analogously to 5B. Components 1 and 2 build up through the column and the same equilibrium peaks are observed at the end as expected.

6.11. Test example 6

In test examples 6A, 6B, and 6C combinations of first and second order kinetics is shown along with the effect of variation of the time step. Component 1 and 2 react to give component 3, which reacts back to give 1 and 2. In 6A the input concentration was given as component 3 only and in 6B input concentration was given as component 1 and 2 in equal concentrations.

Input data:

```
100 TEST EXAMPLE 6A
200 ABC6AC
300 ABC6AE
400 0,2,1,0
500 3
600 401
700 120
800 20
900 1
1000 1
1100 200
1200 100000
1300 1
1400 1,1,1,1,1,1,0
1500 0.03
1600 1.,1
1700 0
1800 1
1900 0
2000 1
2100 1
2200 2,1,3,0,.4
2300 0.,0.,.4
2400 0.,0.,.4
2500 0.,0.,-.4
2600 2,2,2
2700 0,205,201
2800 0,205,201
2800 0
2900 0,205,201
3000 0
3100 0,205,201
3200 1
```

The plotted output is shown in Fig. 6.10. It is seen that equilibrium is already obtained after 20 years and the same peaks are observed in 6A and 6B as observed. A closer inspection of the

peaks reveals that they are a little skew. This can be due only to the discrete nature of the second order calculation. Application of very small timesteps, however, will remove the skewness, but the process will be very time consuming. Example 6C shows what can happen if you choose a timestep that is too large. From 6A to 6C the only change was to increase the timestep from 1 to 2. The amount to be reacted in a second order reaction is calculated as:

$$\text{DELTAT} * \text{SR} * \text{R1} * \text{R2} ,$$

where DELTAT is the time step, SR the second order rate constant, R1 and R2 the concentrations of the two reactants. The factors in this expression may have values that causes the amount to be reacted to be greater than what is available for reaction causing "negative" concentrations. If this happens a warning is given at the terminal. An output option (IOUT(3)) exists that sets small negative concentrations that may occur in the beginning of a calculation equal to zero.

7. HOW TO RUN COLUMN2 ON THE RISØ B7800 COMPUTER

COLUMN2 can be run either interactively or as a batch job. In both cases the input file has to be made up and saved on the disk in advance. After log-on the program is run with the following run statement:

```
Run COLUMN2; file file1 = inputfilename.
```

File 1 has been defined as a disk file titled: IFILE. However, the run statement shown will allow any input file name, which is of great practical importance. The terminal has to be a graphics terminal in order to get the output plots shown on the screen.

The alpha-numeric output of the previous run statement will depend on how the output options are set. The title is always shown on the terminal as a signal indicating that reading of the input file has begun. Errors in the input file will give standard error messages on the terminal. When the calculation is finished the terminal will show "PLEASE", which means that the output plots are ready for inspection and/or further treatment, e.g. stored in an imagefile or plotted on paper at the computer installation or on a side plotter. When run interactively there is a maximum run time of 60 processor seconds set by the computer installation at Risø. The run time depends on a number of parameters: time step, number of components, number of reaction, length of column, and the number of years (TMAX) for which the calculation is performed. If the calculation is stopped because of lack of computing time the terminal will show "INTERRUPTED BECAUSE OF TIMELIMIT" and "PLEASE" on the next line.

When COLUMN2 is run as a batch job a job-file of the type shown below is needed:

```
100 ?BEGIN JOB TEST; CLASS=0 ; CHARGE=3520001; MAXPROCTIME=600;
110 MAXIOTIME=600; PRINTLIMIT=1000;
200 RUN OBJECT/COLUMN2
210 FILE FILE1=INPUT FILENAME
220 FILE FILE6 (PRINTER)
300 ?END JOB
```

"TEST" is just a name of the job for identification purposes. There exists different classes of jobs with different limits to processor time printout, etc. The show job is of class 0 with unlimited resources, which have to be specified. "Charge" is the account number from which the costs are paid. "MAXPROCTIME" is the specification of the maximum processor-time, here 600 seconds. In the second line is specified maximum io-time in seconds and maximum print output in lines. The third line has the run statement. Line 400 defines the input file name, line 500 defines the output file to be the printer, and line 600 ends the job file. The job is started with a statement: start jobfilename.

When COLUMN2 is run as a batch job the resulting plots are stored in an image file named IMSAVE.

8. ADAPTION OF THE CODE FOR OTHER COMPUTERS

COLUMN2 is written in standard FORTRAN77 and should as such be portable to any other computer having a FORTRAN77 compiler. The program uses Risø Interactive Graphics System, RIGS (Rahbek and Hansen 1983) and an integration routine, SPLINT, from Risø Computer Library, RCL. It should be no problem to substitute SPLINT which evaluates the integral of a tabulated function in non-equidistant prints using cubic spline quadrature with some other standard integration routine.

The missing availability of RIGS on another computer may cause more serious problems. It will not be attempted here to give a complete list of the problems and the modifications needed to make COLUMN2 run the same way as on our computer. Plotting of the concentration profiles should be no problem since only simple plot statements are used. However, frames, text and storing of the plots have to be altered completely.

9. HOW TO GET THE COLUMN 2 PROGRAM

A copy of the program with listing can be obtained free of charge by sending a blank 9 track tape to:

Ole John Nielsen
Chemistry Department
Risø National Laboratory
DK 4000 Roskilde
Denmark

Also your request should contain information as to what density (6250, 1600 or 800 bpi), record length, and blocksize you want. Assistance for modifications and implementation will be given on a commercial basis.

10. CONCLUSION

On the basis of the old COLUMN program a new COLUMN2 program has been constructed and tested. The various physical and chemical processes that may influence migration have been listed. Mathematical descriptions and numerical methods have been treated in great detail. A large number of test examples with both input and output have been given. As far as the authors know, COLUMN2 presents the first attempt to include second order kinetics in a computer program for simulation of migration. The discrete nature of the calculation of second order kinetics can give rise to numerical problems. Several solutions, e.g. variable time steps to this problem could have been suggested. However, the authors found it more important to keep the program so simple that users would have a better feeling for what is taking place. It was not intended to construct a fully automatized program. The program and this manual are available on request.

REFERENCES

- BO, P. (1978). COLUMN - Numerical solution of migration equations involving various physicochemical processes. (Chemistry Department, Risø National Laboratory) 59 pp.
- BROYD, T.W., DEAN, R.B., KOBBS, G.D., KNOWLES, N.C., PUTNEY, J.M., and WRINGLEY, J. (1983). A directory of computer programs for assessment of radioactive waste disposal in geological formations. Commission of the European Communities, EUR 8669 EN. 529 pp.
- BURKHOLDER, H.C., LESTER, D.H., and JANSEN, G. (1974). Migration of radionuclide chains through an adsorbing medium. BNWL-SA-5079.
- CARLSEN, L., BATSBERG, W., JENSEN, B.S., and BO, P. (1981). Permeability, porosity, dispersion-, diffusion-, and sorption characteristics of chalk samples from Erslev, Mors, Denmark. Risø-R-451. 47 pp.
- RAHBEK, S. and HANSEN, E. (1983). Risø interactive graphics system - RIGS. Risø-R-493. 149 pp.
- ROUTSON, R.C. and SERNE, R.J. (1972). One dimensional model of the movement of trace radioactive solute through soil columns - The Percol Model. BNWL-1718, VC 70.1972.

APPENDIX I

Alphabetic list of variables, parameters, identifiers, and units

A	Two-dimensional array of retention factors
AE	Retention factor for a component at the end of the column
AS	Retention factor for a component at the start of the column
C	Three-dimensional array for concentrations
CE	Two-dimensional array for effluent concentrations
CINT	Integrated concentration
CINT1	Maximum integrated sum of concentration
CIRCTI	Processor time between two concentration profiles
CLASS	Input identifier for IOU(2); 0, 1 or 2
CONE	One-dimensional array for concentration
CONTIME	Duration time for a concentration input pulse
CT	Interpolated concentration
CO	Input concentration
C1	Three dimensional array for concentrations used during redistribution of points
D	Dispersion coefficients
DCINT	Fraction to which the summed concentrations must be removed from the column before calculation is stopped. Permanently set equal to 100000
DELTAT	Timestep
DIFD	Input identifier for different dispersion coefficients
DIFMAXC	Input identifier for different YCMAX
DIFMAXE	Input identifier for different YEMAX
DIFMINC	Input identifier for different YCMIN

DIFMINE Input identifier for different YEMIN

DIFRATE Input identifier for different S

DIFRET Input identifier for different AS and AE

HC Input identifier for IOUT(5); YES/NO

ICIN Parameter (0,1, or 2) determining the input mode of the concentrations (continuous, pulse, etc.)

IENPOI Endpoint of the input concentration

IFORM Parameter (0 or 1) determining the form of the input concentration (square or user-defined)

II Number of time steps between concentration profiles

III Number of time steps between effluent collection

IMSAVE Name of the image file for saving output plots

INPUT Input identifier for getting input

INSR Number of second order reactions

INT Input identifier for IOUT(4); YES/NO

IOLDNI Array denoting the first point inside the column in the previous calculation cycle

IOPT Array giving input options for D, A, YCMIN, YCMAX, YEMIN, YEMAX, and S

IOUT Array giving output and calculation options

IP Input identifier for IOUT(1); YES/NO

IPLCOU Counter for the number of concentration profiles plotted

ISTPOI Start point for input concentration

KNOTS Number of points in integration of a concentration profile

LIST Input identifier listing the input on the terminal

M Number of points in the calculation (must be uneven)

N Number of components

NEG Boolean, default value, false. Can be set true using IOUT(3) = 0 causing negative concentrations to be set equal to zero

NEG Input identifier for IOUT(3); YES/NO

NEWFIL Boolean permanently set false to indicate that image file exists

NI First point inside the column

NNI Last point inside the column

OLDTI Summed processor time

P Length of column

PLNAC Variable containing the name by which the concentration output plot is identified

PLNAE Variable containing the name by which the effluent output plot is identified

P1 Number of first product component

P2 Number of second product component

PF Input identifier for input pulses, forms and concentrations

RN Counter for second order reactions

R1 Number of first reactant component

R2 Number of second reactant component

S Two-dimensional array giving first order rate constants

SAVE Input identifier for saving the input

SOR Input identifier for second order reactions

SR Second order rate constants

STOP Input identifier for stopping COLUMNINPUT

T Time

TIME Processor time

TITLE Input identifier for title

TMAX Maximum time (not processor time) before the calculation is stopped

V Velocity of the flow

VAL Value of the integrated concentration profile

WORK Working array for the subroutine SPLINT

X Two-dimensional array giving position of points

XX One-dimensional array giving position of points for the subroutine SPLINT

X1 Two-dimensional array giving position of points used in redistribution of points

YCMAX Maximum ordinate value in the concentration plots

YCMIN Minimum ordinate value in the concentration plots

YEMAX Maximum ordinate value in the effluent plot

YEMIN Minimum ordinate value in the effluent plot

Units

time: year

length: m

concentration: moles m⁻³

dispersion coefficient: m² year⁻¹

first order rate constants: year⁻¹

second order rate constants: m³ moles⁻¹ year⁻¹

velocity: m year⁻¹

APPENDIX II: Listing of the COLUMN2 program

```
100 $RESET FREE
200 C *****
300 C
400 C          PROGRAM COLUMN2
500 C
600 C          OLE JOHN NIELSEN, LARS CARLSEN, AND PETER BO
700 C
800 C          CHEMISTRY DEPARTMENT, RISOE NATIONAL LABORATORY
900 C
1000 C          DK-4000 ROSKILDE, DENMARK
1100 C
1200 C          PHONE: 2-371212 EXT. 5331
1300 C
1400 C *****
1500 FILE 1(KIND='DISK', TITLE='IFILE.', FILETYPE=7)
1500 FILE 2(KIND='PRINTER')
1700 FILE 3(KIND='DISK', TITLE='IMSAVE.', FILETYPE=7)
1800 $INCLUDE 'RISOE/BLOCKGLOBALS/F77'
1900 $INCLUDE 'RISOE/RIGS/F77'
2000 $INCLUDE '*RISOE/RISOE/F77'
2100     DIMENSION C(3,5,801),X(5,801),CE(5,801),NI(5),NHI(5),YEMAX(5)
2200     DIMENSION P2(10),D(5),CO(5),A(5,801),YCMIN(5),YCMAX(5),YEMIN(5)
2300     DIMENSION COTIME(5),X1(5,801),C1(2,5,801)
2400     DIMENSION S(5,5),JI(5),AS(5),AE(5),SR(10),R1(10),R2(10),P1(10)
2500     DIMENSION XX(801),COME(801),WORK(2500)
2600     DIMENSION IOPT(7),IOUT(5),ISTPOI(5),IENPOI(5)
2700     DIMENSION IOLDNI(5),ICIN(5),IFORM(5)
2800     INTEGER R1,R2,P1,P2,RN
2900     LOGICAL NEG,NEWFIL
3000     CHARACTER*36 TEXT2,TEXT3,TEXT12,TEXT13
3100     CHARACTER*72 TITLE
3200     CHARACTER*6 PLNAC,PLNAE,IMSAVE
3300 C *****
3400 C *** START TEXT *****
3500 C *****
3600     DATA TEXT2/'CONC. PROFILES DT=          YEARS '/'
3700     DATA TEXT3/'O          X M          '/'
3800     DATA TEXT12/'ELUATE CONC. L=          M          '/'
3900     DATA TEXT13/'O          T YEARS          '/'
4000 C *****
4100 C *** END TEXT *****
4200 C *****
4300 C ****INITIATION OF PLOTTING DEVICES ****
4400 C *****
4500     CALL RINIT
4600     CALL SIMFIL('IMSAVE',NEWFIL)
4700     NEWFIL=.FALSE.
4800     CALL SDEVI('TERM','TEX4014')
4900     CALL SDEVI('PLOT','C1012')
5000     CALL SDEVI('HARD','T4662')
5100     CALL SWINDO(0.,0.,30.,40.)
5200     CALL SMAP('TERM',1)
5300     CALL SMAP('PLOT',1)
5400     CALL SMAP('HARD',1)
5500     CALL SHEIGH('PLOT',20.)
5600     CALL SBAUD(9600)
5700     CALL SAUTO(.FALSE.)
5800 C *****
5900 C **END INITIATION OF PLOTTING DEVICES **
6000 C *****
```

```
6100 C *** START INPUT-DATA *****
6200 C *****
6300      READ (1,701) TITLE
6400      701 FORMAT (A72)
6500      702 FORMAT (A6)
6600      WRITE (6,801) TITLE
6700      READ(1,702) PLMAC
6800      READ(1,702) PLMAE
6900      801 FORMAT (1X, A72)
7000      READ (1,*) (IOUT(I), I=1,5)
7100      READ (1,*) N, M, P, II, III
7200      READ (1,*) V, TMAX, DCINT, DELTAT
7300      READ (1,*) (IOPT(I), I=1,7)
7400      IF (IOPT(1).EQ.0) GO TO 201
7500      READ (1,*) D(1)
7600      DO 301 I = 1, M
7700      301 D(I) = D(1)
7800      GO TO 202
7900      201 READ (1,*) (D(I), I= 1,M)
8000      202 CONTINUE
8100      IF (IOPT(2).EQ.0) GO TO 203
8200      READ (1,*) AS(1), AE(1)
8300      DO 307 I = 1, M
8400      AS(I)=AS(1)
8500      AE(I)=AE(1)
8600      307 CONTINUE
8700      GO TO 204
8800      203 READ(1,*) (AS(I),AE(I), I= 1,M)
8900      204 CONTINUE
9000      DO 308 I=1,M
9100      IF(AS(I).EQ.AE(I)) THEN
9200          DO 309 L=1,M
9300          A(I,L)=AS(I)
9400      309 CONTINUE
9500      ENDOF
9600      308 CONTINUE
9700      IF (IOPT(3).EQ.0) GO TO 205
9800      READ (1,*) YCHIN(1)
9900      DO 303 I = 1, M
10000     303 YCHIN(I) = YCHIN(1)
10100     GO TO 206
10200     205 READ (1,*) (YCHIN(I), I= 1,M)
10300     206 CONTINUE
10400     IF (IOPT(4).EQ.0) GO TO 207
10500     READ (1,*) YCMAX(1)
10600     DO 304 I = 1, M
10700     304 YCMAX(I) = YCMAX(1)
10800     GO TO 208
10900     207 READ (1,*) (YCMAX(I), I= 1,M)
11000     208 CONTINUE
11100     IF (IOPT(5).EQ.0) GO TO 209
11200     READ (1,*) YEMIN(1)
11300     DO 305 I = 1, M
11400     305 YEMIN(I) = YEMIN(1)
11500     GO TO 210
11600     209 READ (1,*) (YEMIN(I), I= 1,M)
11700     210 CONTINUE
11800     IF (IOPT(6).EQ.0) GO TO 211
11900     READ (1,*) YEMAX(1)
12000     DO 306 I = 1, M
```

```
12100 306 YEMAX(I) = YEMAX(1)
12200 GO TO 212
12300 211 READ (1,*) (YEMAX(I), I= 1,N)
12400 212 CONTINUE
12500 READ(1,*) INSR
12600 IF(INSR.EQ.0) GO TO 219
12700 DO 218 I=1,INSR
12800 READ(1,*) R1(I), R2(I), P1(I), P2(I), SR(I)
12900 218 CONTINUE
13000 219 CONTINUE
13100 IF (IOPT(7).EQ.0) GO TO 213
13200 READ(1,*) S(1,1)
13300 DO 310 I=1,N
13400 DO 311 L=1,M
13500 S(I,L)=S(1,1)
13600 311 CONTINUE
13700 310 CONTINUE
13800 GO TO 214
13900 213 DO 312 I=1,N
14000 312 READ(1,*) (S(I,L),L=1,M)
14100 214 CONTINUE
14200 C *****
14300 C *** END INPUT-DATA *****
14400 C *****
14500 C INITIAL VALUES OF COUNTERS ETC.
14600 C IJK REGULATES THE PLOT OF FRAMES AND TEXT
14700 C IJK=1
14800 C II2 AND II1 IDENTIFIES THE CONCENTRATION ARRAYS FOR T-DT AND
14900 C T RESPECTIVELY
15000 C II1=1
15100 C II2=2
15200 C II3 CHANGES THE VALUE (1 OR 2) OF II1 AND II2
15300 C II3=-1
15400 C II4 IS COUNTER FOR CONC. PROFILE OUTPUT,COUNTING UP TO II
15500 C II4=0
15600 C II5 IS COUNTER FOR COLLECTING EFFLUENT CONC,COUNTING UP TO III
15700 C II5=0
15800 C T IS THE TIME FROM START
15900 C T=0.
16000 C II6 IS TIME COUNTER FOR EFFLUENT CONC. CE(I,II6)
16100 C II6=0
16200 C INT1=0
16300 C IPLCOU=0
16400 C NEG=.TRUE.
16500 C *****
16600 C ***** START INITIAL CONCENTRATION PROFILE *****
16700 C *****
16800 READ(1,*) (ICIN(I), I=1,N)
16900 DO 313 I=1,N
17000 CO(I)=0.
17100 NI(I)=(M-1)/2
17200 NNI(I)=M
17300 DO 316 L=1,M
17400 X(I,L)=P+2.*P*L/(M-1)
17500 C(II1,I,L)=0.
17600 C(II2,I,L)=0.
17700 C(3,I,L)=0.
17800 316 CONTINUE
17900 IF (ICIN(I).EQ.0) GO TO 215
18000 IF(ICIN(I).EQ.2) GO TO 220
```

```
18100      READ(1,*) IFORM(I),COTIME(I),CO(I)
18200      IF (IFORM(I).EQ.0) GO TO 216
18300      GO TO 217
18400      216 CONTINUE
18500 C *****
18600 C *** IN 216 YOU CAN ENTER AN INPUT-SUBROUTINE *****
18700 C *****
18800      GO TO 217
18900      215 READ(1,*) CO(I)
19000      DO 318 L=1,(M-1)/2
19100      C(II1,I,L)=CO(I)
19200      C(II2,I,L)=CO(I)
19300      318 CONTINUE
19400      GO TO 217
19500      220 READ(1,*) IFORM(I),ISTPOI(I),IENPOI(I)
19600      IF(IFORM(I).EQ.0) GO TO 221
19700      READ(1,*) (C(II1,I,L), L=IENPOI(I),ISTPOI(I))
19800      DO 222 L=IENPOI(I),ISTPOI(I)
19900      222 C(II2,I,L)=C(II1,I,L)
20000      GO TO 217
20100      221 READ(1,*) C(II1,I,ISTPOI(I))
20200      IF (ISTPOI(I).GE.(M-1)/2)-1.AND.IE. POI(I).LE.(M-1)/2) CO(I)=C(II
20300      1,I,ISTPOI(I))
20400      DO 315 L=IENPOI(I),ISTPOI(I)
20500      C(II1,I,L)=C(II1,I,ISTPOI(I))
20600      C(II2,I,L)=C(II1,I,ISTPOI(I))
20700      315 CONTINUE
20800      217 CONTINUE
20900      313 CONTINUE
21000 C *****
21100 C ***** END INITIAL CONCENTRATION PROFILE *****
21200 C *****
21300 C *** PRINTING OF INPUT *****
21400 C *****
21500      IF(IOUT(1).EQ.1) GO TO 102
21600      WRITE(2,*) TITLE
21700      WRITE(2,*) PLNAC, ' ',PLNAE
21800      WRITE(2,*) ('IOUT OF ',I,' = ',IOUT(I),' ', I=1,5)
21900      WRITE(2,*) 'NUMBER OF COMPONENTS, N= ',N
22000      WRITE(2,*) 'NUMBER OF POINTS, M= ',M
22100      WRITE(2,*) 'LENGTH OF COLUMN, P= ',P
22200      WRITE(2,*) 'TIME STEPS BETWEEN CONCENTRATION PROFILES, II= ',II
22300      WRITE(2,*) 'TIME STEPS BETWEEN EFFLUENT COLLECTION, III= ',III
22400      WRITE(2,*) 'VELOCITY, V= ',V
22500      WRITE(2,*) 'MAXIMUM TIME, TMAX= ',TMAX
22600      WRITE(2,*) 'DCINT= ',DCINT
22700      WRITE(2,*) 'DELTAT= ',DELTAT
22800      WRITE(2,*) ('IOPT OF ',I,' = ',IOPT(I),' ', I=1,7)
22900      WRITE(2,*) ('D OF ',I,' = ',D(I),' ', I=1,N)
23000      WRITE(2,*) ('AS OF ',I,' = ',AS(I),' ', AE OF ',I,' = ',AE(I),' ',
23100      1=1,N)
23200      WRITE(2,*) ('YCMIN OF ',I,' = ',YCMIN(I),' YCMAX OF ',I,' = ',YC
23300      1AX(I),' ', I=1,N)
23400      WRITE(2,*) ('YEMIN OF ',I,' = ',YEMIN(I),' YEMAX OF ',I,' = ',YF
23500      1AX(I),' ', I=1,N)
23600      WRITE(2,*) 'NUMBER OF SECOND ORDER REACTIONS, INSR= ',INSR
23700      IF(INSR.EQ.0) GO TO 106
23800      DO 107 I=1,INSR,1
23900      WRITE(2,*) ('REACTION: ',I,' ',R1(I),'+',R2(I),' GIVES ',P1(I),'
24000      1',P2(I),' RATE CONSTANT= ',SR(I),' ', I=1,INSR)
```



```
24100 107 CONTINUE
24200 106 DO 108 I=1,N,1
24300 WRITE(2,*) 'FIRST ORDER RATE CONSTANTS: ',(L, ' TO ',I, '= ',S(I,L),
24400 1L=1,N)
24500 108 CONTINUE
24600 WRITE(2,*) ('ICIN OF ',I, ' = ',ICIN(I),', ',I=1,N)
24700 WRITE(2,*) ('FORM OF ',I, ' = ',IFORM(I),', ',I=1,N)
24800 DO 103 I=1,N
24900 IF(ICIN(I).EQ.0) GO TO 104
25000 IF(IFORM(I).EQ.2) GO TO 109
25100 WRITE(2,*) 'COMPONENT: ',I, ' COTIME= ',COTIME(I), ' CO= ',CO(I)
25200 103 CONTINUE
25300 GO TO 105
25400 104 WRITE(2,*) 'CONTINUOUS INPUT OF ',I, ' CONCENTRATION= ',CO(I)
25500 GO TO 103
25600 109 WRITE(2,*) 'COMPONENT: ',I, ' START= ',ISTPOI(I), ' END= ',IENPOI(
25700 1), ' "CONCENTRATION"= ',C(II1,I,ISTPOI(I))
25800 GO TO 103
25900 105 CONTINUE
26000 C *****
26100 C *** END PRINTING OF INPUT *****
26200 C *****
26300 102 CONTINUE
26400 GO TO 100
26500 101 CONTINUE
26600 502 FORMAT(3E15.4,2I10)
26700 CALL REDISP(X,C,X1,C1,II2,I,M,P)
26800 DO 503 L=1,M,1
26900 X(I,L)=X1(I,L)
27000 C(II2,I,L)=C1(II2,I,L)
27100 C(II1,I,L)=C1(II2,I,L)
27200 503 CONTINUE
27300 WRITE(6,500) T
27400 500 FORMAT(1H0,20X,'REDISTRIBUTION AT T=',E15.4/1X)
27500 II3=-II3
27600 II1=II1+II3
27700 II2=II2-II3
27800 NI(I)=(M-1)/2
27900 NNI(I)=M
28000 C FIND THE BOUNDARY OF THE COLUM
28100 100 DO 3 I=1,N,1
28200 IOLDNI(I)=NI(I)
28300 4 J=NI(I)
28400 IF(X(I,J).GE.0.) NI(I)=NI(I)-1
28500 IF(NI(I).EQ.0) GO TO 101
28600 IF(X(I,J).GE.0.) GO TO 4
28700 JI(I)=NI(I)+1
28800 5 J=NNI(I)
28900 IF(X(I,J).GE.P) NNI(I)=NNI(I)-1
29000 IF(X(I,J).GE.P) GO TO 5
29100 3 CONTINUE
29200 C CHANGE THE CONCENTRATION ARRAYS
29300 II3=-II3
29400 II1=II1+II3
29500 II2=II2-II3
29600 C CALC.RETENTION FACTORS
29700 CALL RETSUB(II2,X,C,A,N,M,P,AS,AE)
29800 C *****
29900 C *** START BOUNDARY CONDITION AT X=0 *****
30000 C *****
```

```

30100      DO 66 I=1,N,1
30200      IF (ICIN(I).EQ.0) GO TO 66
30300      IF(ICIN(I).EQ.2) GO TO 319
30400      IF(T.LE.COTIME(I)) THEN
30500      DO 67 L=NI(I)+1,IOLDNI(I)+1,1
30600      C(II1,I,L)=CO(I)
30700      C(II2,I,L)=CO(I)
30800 67   CONTINUE
30900      ELSE
31000      ENDIF
31100      IF(T.GT.COTIME(I)) CO(I)=0.
31200      GO TO 66
31300 319  IF(IENPOI(I).GT.(M-1)/2) GO TO 66
31400      IF(T.GT.((ISTPOI(I)-IENPOI(I)+1)/((V*M)/(2*P)))) CO(I)=0.
31500 66   CONTINUE
31600 C *****
31700 C *** END BOUNDARY CONDITION AT X=0 *****
31800 C *****
31900      DO 6 I=1,N,1
32000      JJ=1
32100      J=NI(I)
32200      IF(X(I,J+1).LT.P/(500.*M)) JJ=0
32300      IF(JJ.EQ.0) C(II1,I,J+1)=CO(I)
32400      IF(JJ.EQ.0) GO TO 7
32500      H=2.*X(I,J+1)/X(I,J+2)
32600      C(II1,I,J+1)=C(II2,I,J+1)+D(I)*DELTA*(H*(C(II2,I,J+2)-C(II2,I,J+
32700      1))-(2.-H)*(C(II2,I,J+1)-CO(I)))/(A(I,J+1)*X(I,J+1)*(X(I,J+2)-X(I,
32800      2+1)))
32900 C *****
33000 C ***** START REACTIONS AT BEGINNING OF COLUMN *****
33100 C *****
33200 C *** FIRST ORDER REACTIONS *****
33300 C *****
33400      DO 70 JK=1,N,1
33500      IF(S(I,JK).EQ.0) GO TO 70
33600      IF(X(JK,JI(JK)).LT.P/(500.*M)) JI(JK)=JI(JK)+1
33700      CT=CO(JK)+X(I,J+1)*(C(II2,JK,JI(JK))-CO(JK))/X(JK,JI(JK))
33800      C(II1,I,J+1)=C(II1,I,J+1)+DELTA*S(I,JK)*CT/A(I,J+1)
33900 70   CONTINUE
34000 7    CONTINUE
34100 6    CONTINUE
34200 C *****
34300 C *** SECOND ORDER REACTIONS *****
34400 C *****
34500      IF(INSR.EQ.0) GO TO 725
34600      DO 723 I=1,N,1
34700      DO 724 JK=1,N,1
34800      IF(X(JK,JI(JK)).LT.P/(500.*M)) JI(JK)=JI(JK)+1
34900 724  CONTINUE
35000      DO 71 RN=1,INSR,1
35100      IF(I.EQ.R1(RN)) GO TO 721
35200      GO TO 72
35300 721  CT=CO(R2(RN))+X(I,J+1)*(C(II2,R2(RN),JI(R2(RN)))-CO(R2(RN)))/X(R2
35400      1RN),JI(R2(RN)))
35500      C(3,I,J+1)=C(3,I,J+1)-DELTA*SR(RN)*CT*C(II2,I,J+1)/A(I,J+1)*.
35600      15
35700      C(3,R2(RN),J+1)=C(3,R2(RN),J+1)-DELTA*SR(RN)*CT*C(II2,I,J+1)/
35800      1A(R2(RN),J+1)*.5
35900      C(3,P1(RN),J+1)=C(3,P1(RN),J+1)+DELTA*SR(RN)*CT*C(II2,I,J+1)/
36000      1A(P1(RN),J+1)*.5

```

```
36100      IF(P2(RN).EQ.0) GO TO 71
36200      C(3,P2(RN),J+1)=C(3,P2(RN),J+1)+DELTAT*SR(RN)*CT*C(II2,I,J+1)/
36300      1A(P2(RN),J+1)*.5
36400      GO TO 71
36500 72   IF(I.EQ.R2(RN)) GO TO 722
36600      GO TO 71
36700 722  CT=CO(R1(RN))+X(I,J+1)*(C(II2,R1(RN),JI(R1(RN)))-CO(R1(RN)))/X(R1(
36800      1RN),JI(R1(RN)))
36900      C(3,I,J+1)=C(3,I,J+1)-DELTAT*SR(RN)*CT*C(II2,I,J+1)/A(I,J+1)*.
37000      15
37100      C(3,R1(RN),J+1)=C(3,R1(RN),J+1)-DELTAT*SR(RN)*CT*C(II2,I,J+1)/
37200      1A(R1(RN),J+1)*.5
37300      C(3,P1(RN),J+1)=C(3,P1(RN),J+1)+DELTAT*SR(RN)*CT*C(II2,I,J+1)/
37400      1A(P1(RN),J+1)*.5
37500      IF(P2(RN).EQ.0) GO TO 71
37600      C(3,P2(RN),J+1)=C(3,P2(RN),J+1)+DELTAT*SR(RN)*CT*C(II2,I,J+1)/
37700      1A(P2(RN),J+1)*.5
37800 71   CONTINUE
37900 723  CONTINUE
38000 725  CONTINUE
38100      DO 726 I=1,N,1
38200      C(II1,I,J+1)=C(II1,I,J+1)+C(3,I,J+1)
38300 726  CONTINUE
38400 C *****
38500 C ***** END REACTIONS *****
38600 C *****
38700 C      INTEGRATION 0<X<P
38800      DO 80 I=1,N,1
38900      DO 880 J=1,N,1
39000 880  JI(J)=NI(J)-1
39100      DO 90 L=NI(I)+2,NNI(I)+1,1
39200      IF((X(I,L)-X(I,L-1))*(X(I,L+1)-X(I,L)).LE.D(I)*DELTAT/(0.5*A(I,L)
39300      1) GO TO 101
39400      H=2.*(X(I,L)-X(I,L-1))/(X(I,L+1)-X(I,L-1))
39500      H1=D(I)*DELTAT/(A(I,L)*(X(I,L+1)-X(I,L))*(X(I,L)-X(I,L-1)))
39600      C(II1,I,L)=C(II2,I,L)+H1*(H*(C(II2,I,L+1)-C(II2,I,L))-(2.-H)*(C(I
39700      12,I,L)-C(II2,I,L-1)))
39800      90 CONTINUE
39900      80 CONTINUE
40000 C *****
40100 C ***** START REACTIONS INSIDE THE COLUMN *****
40200 C *****
40300 C ***** FIRST ORDER REACTIONS *****
40400 C *****
40500      DO 8 I=1,N,1
40600      DO 88 J=1,N,1
40700 88  JI(J)=NI(J)-1
40800      DO 9 L=NI(I)+2,NNI(I)+1,1
40900      DO 99 J=1,N,1
41000      IF(S(I,J).EQ.0) GO TO 99
41100 87  CONTINUE
41200      IF(X(J,JI(J)).LE.X(I,L)) GO TO 85
41300      CT=C(II2,J,JI(J)-1)+(X(I,L)-X(J,JI(J)-1))*(C(II2,J,JI(J))-C(II2,
41400      1JI(J)-1))/(X(J,JI(J))-X(J,JI(J)-1))
41500      C(II1,I,L)=C(II1,I,L)+DELTAT*S(I,J)*CT/A(I,L)
41600      GO TO 86
41700 85  JI(J)=JI(J)+1
41800      IF(JI(J).EQ.M+1) GO TO 101
41900      GO TO 87
42000 86  CONTINUE
```

```
42100 99    CONTINUE
42200 9     CONTINUE
42300 8     CONTINUE
42400 C *****
42500 C *** SECOND ORDER REACTIONS *****
42600 C *****
42700      IF(INSR.EQ.0) GO TO 873
42800      DO 862 I=1,N,1
42900      DO 863 L=NI(I)+2,NNI(I)+1,1
43000      DO 865 IF=1,N,1
43100 872    CONTINUE
43200      IF(X(IF,JI(IF)).LE.X(I,L)) GO TO 850
43300 866    CONTINUE
43400      IF(LESS.EQ.1) GO TO 867
43500      IF(X(IF,JI(IF)).GE.X(I,L)) GO TO 864
43600      GO TO 867
43700 850    JI(IF)=JI(IF)+1
43800      LESS=1
43900      GO TO 872
44000 864    JI(IF)=JI(IF)-1
44100      GO TO 866
44200 867    LESS=0
44300 865    CONTINUE
44400      DO 73 RN=1,INSR,1
44500      IF(I.EQ.R1(RN)) GO TO 870
44600      GO TO 74
44700 870    CONTINUE
44800      CT=C(II2,R2(RN),JI(R2(RN))-1)+(X(I,L)-X(R2(RN),JI(R2(RN))-1))*(C(
44900 1I2,R2(RN),JI(R2(RN)))-C(II2,R2(RN),JI(R2(RN))-1))/(X(R2(RN),JI(R2
45000 2RN))-X(R2(RN),JI(R2(RN))-1))
45100      C(3,I,L)=C(3,I,L)-DELTAT*SR(RN)*CT*C(II2,I,L)/A(I,L)*.5
45200      C(3,R2(RN),JI(R2(RN)))=C(3,R2(RN),JI(R2(RN)))-DELTAT*SR(RN)*CT
45300 1*C(II2,I,L)/A(R2(RN),JI(R2(RN)))*.5
45400      C(3,P1(RN),JI(P1(RN)))=C(3,P1(RN),JI(P1(RN)))+DELTAT*SR(RN)*CT
45500 1*C(II2,I,L)/A(P1(RN),JI(P1(RN)))*.5
45600      IF(P2(RN).EQ.0) GO TO 74
45700      C(3,P2(RN),JI(P2(RN)))=C(3,P2(RN),JI(P2(RN)))+DELTAT*SR(RN)*CT
45800 1*C(II2,I,L)/A(P2(RN),JI(P2(RN)))*.5
45900      GO TO 73
46000 74     IF(I.EQ.R2(RN)) GO TO 871
46100      GO TO 73
46200 871    CONTINUE
46300      CT=C(II2,R1(RN),JI(R1(RN))-1)+(X(I,L)-X(R1(RN),JI(R1(RN))-1))*(C(
46400 1I2,R1(RN),JI(R1(RN)))-C(II2,R1(RN),JI(R1(RN))-1))/(X(R1(RN),JI(R
46500 2RN))-X(R1(RN),JI(R1(RN))-1))
46600      C(3,I,L)=C(3,I,L)-DELTAT*SR(RN)*CT*C(II2,I,L)/A(I,L)*.5
46700      C(3,R1(RN),JI(R1(RN)))=C(3,R1(RN),JI(R1(RN)))-DELTAT*SR(RN)*CT
46800 1*C(II2,I,L)/A(R1(RN),JI(R1(RN)))*.5
46900      C(3,P1(RN),JI(P1(RN)))=C(3,P1(RN),JI(P1(RN)))+DELTAT*SR(RN)*CT
47000 1*C(II2,I,L)/A(P1(RN),JI(P1(RN)))*.5
47100      IF(P2(RN).EQ.0) GO TO 73
47200      C(3,P2(RN),JI(P2(RN)))=C(3,P2(RN),JI(P2(RN)))+DELTAT*SR(RN)*CT
47300 1*C(II2,I,L)/A(P2(RN),JI(P2(RN)))*.5
47400 73     CONTINUE
47500 863    CONTINUE
47600 862    CONTINUE
47700 873    CONTINUE
47800      DO 874 I=1,N,1
47900      DO 875 L=1,M,1
48000      C(II1,I,L)=C(II1,I,L)+C(3,I,L)
```

```
48100      C(3,I,L)=0.
48200 875 CONTINUE
48300 874 CONTINUE
48400 C *****
48500 C *** END REACTIONS *****
48600 C *****
48700 C *** START BOUNDARY CONDITION AT X=P *****
48800 C *****
48900      DO 10 I=1,N,1
49000      J=NMI(I)
49100      H=(X(I,J+2)-X(I,J+1))/(P-X(I,J))
49200      H1=H*(P-X(I,J+1))/(X(I,J)-X(I,J-1))
49300      C(II1,I,J+2)=C(II1,I,J+1)-H*(C(II1,I,J+1)-C(II1,I,J))+H1*(C(II1,I,
49400      1J)-C(II1,I,J-1))
49500 10 CONTINUE
49600 C *****
49700 C *** END BOUNDARY CONDITION AT X=P *****
49800 C *****
49900 C UPDATING TIME,COUNTERS AND POSITION
50000      T=T+DELTAT
50100      II4=II4+1
50200      II5=II5+1
50300      DO 11 I=1,N,1
50400      DO 12 L=1,M,1
50500      X(I,L)=X(I,L)+V*DELTAT/A(I,L)
50600 12 CONTINUE
50700 11 CONTINUE
50800 C COLLECTING DATA
50900      IF(II5.NE.III) GO TO 13
51000      II6=II6+1
51100      II5=0
51200      DO 14 I=1,N,1
51300 15 J=NMI(I)
51400      IF(X(I,J).GE.P) NNI(I)=NNI(I)-1
51500      IF(X(I,J).GE.P) GO TO 15
51600      J=NMI(I)
51700      IF(J.GT.801.OR.J.LT.0) WRITE(6,*) "J= ",J
51800      CE(I,II6)=C(II1,I,J)+(P-X(I,J))*(C(II1,I,J+1)-C(II1,I,J))/(X(I,J+
51900      1)-X(I,J))
52000 14 CONTINUE
52100 13 CONTINUE
52200      IF(II4.NE.II) GO TO 163
52300      CINT=0
52400 C INTEGRAL OF EACH COMPONENT IS CALCULATED IF DESIRED
52500      IF(IOUT(4).EQ.1) GO TO 24
52600      DO 16 I=1,N
52700      M1=1
52800      M2=M
52900      DO 17 L=1,M,1
53000      IF(X(I,L).LT.0) M1=L
53100      IF(X(I,L).GE.P) GO TO 19
53200 17 CONTINUE
53300 19 M2=L
53400      KNOTS=M2-M1-1
53500      DO 18 L=M1+1,M2-1,1
53600      XX(L-M1)=X(I,L)
53700      CONE(L-M1)=C(II1,I,L)
53800 18 CONTINUE
53900      CALL SPLINT(KNOTS,XX,CONE,WORK,VAL)
54000      WRITE(6,703) I,VAL
```

```
54100 703 FORMAT(' INTEGRAL OF COMPONENT ',I2,' = ',E10.3)
54200 CINT=CINT+VAL
54300 16 CONTINUE
54400 IF(CINT.GT.CINT1)CINT1=CINT
54500 IF(CINT.LT.CINT1/DCINT)TMAX=T
54600 WRITE(6,704) CINT,CINT1,CINT1/DCINT
54700 704 FORMAT(' CINT= ',E10.3,' CINT1= ',E10.3,' CINT1/DCINT= ',E10.3)
54800 WRITE(6,705) T
54900 705 FORMAT(' TIME= ',E10.2,' YEARS')
55000 24 CONTINUE
55100 CIRCTI=T-OLDTI
55200 OLDTI=T
55300 II4=0
55400 GO TO 399
55500 C REARRANGEMENT OF X-VALUES
55600 163 DO 160 I=1,N,1
55700 IF(X(I,1).GE.-2.*P/(M-1)-V*DELTAT/A(I,1)) GO TO 161
55800 160 CONTINUE
55900 GO TO 167
56000 161 DO 157 I=1,N,1
56100 162 J=NNI(I)
56200 IF(X(I,J).GE.P) NNI(I)=NNI(I)-1
56300 IF(X(I,J).GE.P) GO TO 162
56400 J=NNI(I)+2
56500 DO 166 JJ=1,M,1
56600 IF(JJ.GT.J) GO TO 164
56700 IF(X(I,J+1-JJ).LE.O.) GO TO 164
56800 X(I,M+1-JJ)=X(I,J+1-JJ)
56900 C(II1,I,M+1-JJ)=C(II1,I,J+1-JJ)
57000 C(II2,I,M+1-JJ)=C(II2,I,J+1-JJ)
57100 GO TO 165
57200 164 X(I,M+1-JJ)=X(I,M+1-JJ+1)-2.*P/(M-1)
57300 C(II1,I,M+1-JJ)=C(II1,I,M+1-JJ+1)
57400 C(II2,I,M+1-JJ)=C(II2,I,M+1-JJ+1)
57500 165 CONTINUE
57600 166 CONTINUE
57700 NNI(I)=M
57800 NI(I)=M
57900 WRITE(6,504) T
58000 504 FORMAT(1H0,20X,'REARRANGEMENT OF X-AXES AT T=',E15.4/1X)
58100 167 CONTINUE
58200 C CHECK FOR NEGATIVE CONCENTRATIONS
58300 DO 170 I=1,N,1
58400 DO 171 L=1,M,1
58500 IF(C(II1,I,L).LT.O.AND.(X(I,L).GE.O.AND.X(I,L).LE.P)) THEN
58600 IF(.NOT.NEG) GO TO 172
58700 WRITE(6,*) 'NEGATIVE CONCENTRATION OF COMPONENT ',I,' AT POINT ',
58800 1, ' AT TIME ',T,' AT ',X(I,L), ' M'
58900 WRITE(6,*) 'C WAS ',C(II1,I,L)
59000 NEG=.FALSE.
59100 172 IF(IOUT(3).EQ.0) C(II1,I,L)=0
59200 ENDIF
59300 171 CONTINUE
59400 170 CONTINUE
59500 C CHECK IF COMPUTING-TIME IS GETTING TOO CLOSE TO LIMIT
59600 IF(IOUT(2).EQ.2.AND.CIRCTI.GT.57.-TIME('PROCESSOR').AND.TIME('PRO
59700 1ESSOR').GE.54.) THEN
59800 IF(IOUT(2).EQ.2) WRITE(6,*) ' INTERRUPTED BECAUSE OF TIMELIMIT !'
59900 GO TO 406
60000 ENDIF
```

```
60100      IF(IOUT(2).EQ.1.AND.CIRCTI.GT.258.-TIME('PROCESSOR').AND.TIME('PR
60200      ICESSOR').GE.285.) THEN
60300      WRITE(6,*) ' INTERRUPTED BECAUSE OF TIMELIMIT !'
60400      GO TO 406
60500      ENDF
60600 C     NEW INTEGRATION STEP OR FINISH CALCULATION
60700      IF(T.GE.TMAX) GO TO 406
60800      GO TO 100
60900 C     $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
61000 C     $$$ START PLOT CONCENTRATION PROFILES $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
61100 C     $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
61200 399   CONTINUE
61300      CALL SWINDO(-2.,6.,24.,22.+10.*(REAL(N)-1.))
61400      T1=I*DELTAT
61500      IPLCOU=IPLCOU+1
61600      IF(IPLCOU.GT.1) GO TO 412
61700      DO 413 I=1,N,1
61800      CALL PFRAME(3.5,10.*REAL(I),23.5,10.+10.*REAL(I),2.,1.,0)
61900 413   CONTINUE
62000      CALL BRKIM
62100      CALL PTEXT(TITLE,0.,11.+10.*REAL(N),0.6)
62200      CALL BRKIM
62300      DO 414 I=N,1,-1
62400      CALL PREALE(YCMIN(I),8,2,-2.,0.2+10.*REAL(I),0.6)
62500      CALL PREALE(YCMAX(I),8,2,-2.,9.2+10.*REAL(I),0.6)
62600 414   CONTINUE
62700      CALL PTEXT(TEXT2,3.5,8.,0.6)
62800      CALL PTEXT(PLNAC,3.5,7.,.6)
62900      CALL PREALF(T1,7,1,14.9,8.0,0.6)
63000      CALL PTEXT(TEXT3,3.5,9.2,0.6)
63100      CALL PREALF(P,6,1,19.9,9.2,0.6)
63200      CALL BRKIM
63300      PX=23.5
63400      PY=10.
63500      CALL PLOT(PX,PY,1)
63600 412   CALL SORIGO(3.5,0.)
63700      J=0
63800      DO 401 I=1,N,1
63900      IF(YCMAX(I).LT.0.) GO TO 407
64000      J=J+1
64100      CALL SORIGO(0.,10.)
64200      IF(IPLCOU.GT.1) GO TO 440
64300      CALL PLOT(0.,0.,1)
64400      PX=0.
64500      PY=0.
64600 440   J=NNI(I)
64700      IF(X(I,J).GE.P) NNI(I)=NNI(I)-1
64800      IF(X(I,J).GE.P) GO TO 440
64900      PX=X(I,J)*20./P
65000      PY=YCMIN(I)+(C(II1,I,J)-YCMIN(I))*10./(YCMAX(I)-YCMIN(I))
65100      CALL PLOT(PX,PY,1)
65200      DO 405 JJ=1,M,1
65300      PX=X(I,J-JJ)*20./P
65400      IF(PX.LT.0.) GO TO 450
65500      PY=YCMIN(I)+(C(II1,I,J-JJ)-YCMIN(I))*10./(YCMAX(I)-YCMIN(I))
65600      CALL PLOT(PX,PY,0)
65700 405   CONTINUE
65800 450   CONTINUE
65900 407   CONTINUE
66000 401   CONTINUE
```



```
72100      PY=10.
72200      CALL PLOT(PX,PY,1)
72300      CALL SORIGO(3.5,0.)
72400      J=0
72500      DO 601 I=1,N,1
72600      IF(YEMAX(I).LT.0.) GO TO 607
72700      J=J+1
72800      CALL SORIGO(0.,10.)
72900      PX=20.
73000      PY=YEMIN(I)+(CE(I,II6)-YEMIN(I))*10./(YEMAX(I)-YEMIN(I))
73100      CALL PLOT(PX,PY,1)
73200      DO 605 JJ=1,II6-1,1
73300      PX=20.*(II6-JJ)/II6
73400      PY=YEMIN(I)+(CE(I,II6-JJ)-YEMIN(I))*10./(YEMAX(I)-YEMIN(I))
73500      CALL PLOT(PX,PY,0)
73600 605   CONTINUE
73700 607   CONTINUE
73800 601   CONTINUE
73900 C #####
74000 C #####
74100 C #####
74200      CALL SORIGO(-3.5,-10.*N)
74300      IF(IOUT(2).EQ.2) CALL IMHAND('PLEASE')
74400      IF(IOUT(2).EQ.2) CALL CLEARS
74500      IF(IOUT(2).EQ.1.OR.IOUT(2).EQ.0) CALL SAVEIM(PLNAE)
74600      STOP
74700      CALL RTERM
74800      END
74900 C #####
75000 C #####
75100 C #####
75200 C
75300      SUBROUTINE REDISP(X,C,X1,C1,II,I,M,P)
75400      DIMENSION X(5,801),X1(5,801),C(2,5,801),C1(2,5,801),R(4860)
75500      X1(I,1)=X(I,1)
75600      DO 1 L=2,M,1
75700      X1(I,L)=X1(I,L-1)+2*P/(M-1)
75800      IF(X1(I,L).GT.X(I,M)) C1(II,I,L)=C(II,I,M)
75900 1      CONTINUE
76000      E=0.000001
76100      DO 2 L=1,M,1
76200      R(2*L-1)=X(I,L)
76300      R(2*L)=C(II,I,L)
76400 2      CONTINUE
76500 3      R(4*M+1)=0.
76600      R(5*M)=0.
76700      D=4*(2.-SQRT(3.))
76800      DO 6 L=1,M-1,1
76900 4      R(3*M+L)=(R(2*L+2)-R(2*L))/(R(2*L+1)-R(2*L-1))
77000      R(2*M+L)=R(2*L+1)-R(2*L-1)
77100      IF(L.EQ.1) GO TO 5
77200      R(4*M+L)=(2*(R(3*M+L)-R(3*M+L-1)))/(R(2*M+L-1)+R(2*M+L))
77300      R(5*M+L)=3.*R(4*M+L)/2.
77400 5      CONTINUE
77500 6      CONTINUE
77600 7      EE=0.
77700      DO 10 L=2,M,1
77800      LL=4*M+L
77900      B=R(LL+1)
78000      BB=((R(LL+1)-R(LL-1)))/(1.+R(LL-2*M)/R(LL-2*M-1))-B)/2.
```

```
78100      B=D*(BB-R(LL)+R(LL+M))
78200      BB=SQRT(B*B)
78300      IF(BB.GT.EE) EE=BB
78400 10    R(LL)=R(LL)+B
78500      IF(E.LE.EE) GO TO 7
78600      DO 15 L=1,M-1,1
78700      LL=4*M+L
78800 15    R(LL-2*M)=(R(LL+1)-R(LL))/R(LL-2*M)
78900      DO 30 L=1,M,1
79000      IF(X1(I,L).GT.X(I,M)) GO TO 22
79100 23    CONTINUE
79200      Z=X1(I,L)
79300      DO 20 LL=1,M-1,1
79400      Q=Z-R(2*LL-1)
79500      Q1=Z-R(2*LL+1)
79600      IF(Q.GE.0..AND.Q1.LT.0.) GO TO 21
79700      IF(LL.EQ.M-1.AND.Q1.EQ.0.) GO TO 21
79800 20    CONTINUE
79900      GO TO 22
80000 21    D=R(4*M+LL)+Q*R(2*M+LL)
80100      D=(R(4*M+LL)+R(4*M+LL+1)+D)/6.
80200      D=R(2*LL)+Q*R(3*M+LL)+Q*Q1*D
80300      C1(II,I,L)=D
80400      IF(D.LE.0.) C1(II,I,L)=0.
80500 22    CONTINUE
80600 30    CONTINUE
80700      RETURN
80800      END
80900 C
81000      SUBROUTINE RETSUB(I,X,C,A,N,M,P,AS,AE)
81100      DIMENSION AS(5),AE(5),X(5,801),C(2,5,801),A(5,801)
81200      DO 1 J=1,N,1
81300      DO 2 L=1,M,1
81400      A(J,L)=AS(J)+(AE(J)-AS(J))*X(J,L)/P
81500      IF(X(J,L).LE.0.) A(J,L)=AS(J)
81600      IF(X(J,L).GT.P) A(J,L)=AE(J)
81700 2     CONTINUE
81800 1     CONTINUE
81900      RETURN
82000      END
82100 $RESET LIST
£
```

APPENDIX III: Listing of the COLUMNINPUT program

```
100
200 BEGIN
300 FILE SCREEN(KIND=REMOTE,MYUSE=OUT,MAXRECSIZE=90,UNITS=1),
400   KEYBOARD(KIND=REMOTE,MAXRECSIZE=90,UNITS=1,MYUSE=IN);
500 FILE INP(KIND=DISK,FILETYPE=7);
600 FILE OUT(KIND=DISK,MAXRECSIZE=14,BLOCKSIZE=420,AREASIZE=1080,
700   NEWFILE=TRUE);
800 BOOLEAN REMOTEINPUT,INPUTFINISHED,FILEFINISHED,REMOTEJOB,INPUTPRINT,
900   NEGCON,INTEGRATION,HARDPLOT;
1000 STRING PLNAC,PLNAE,TITLE,INPUTTITLE,OUTPUTTITLE;
1100 POINTER PT,PC;
1200 ALPHA ARRAY TITLEAR,CARD[0:70];
1300 INTEGER N,M,P,III,II,INSR,I,L,J;
1400 REAL TMAX,V,DCINT,DELTAT;
1500 INTEGER ARRAY IOPT[1:7],IOUT,ISTPOI,IENPOI,ICIN,IFORN[1:5],R1,R2,P1,
1600   P2[1:10];
1700 REAL ARRAY D,AE,AS,YCMIN,YCMAX,YEMIN,YEMAX[1:5],SR[1:10],S[1:5,1:5],
1800   C[1:5,1:801],CO,COTIME[1:5];
1900
2000 PROCEDURE PRINT(OUT);
2100 FILE OUT;
2200 BEGIN
2300   WRITE(OUT,<A*>,LENGTH(TITLE),TITLE);
2400   WRITE(OUT,<A6>,PLNAC);
2500   WRITE(OUT,<A6>,PLNAE);
2600   WRITE(OUT,<4(I1,""),I1>,FOR I:=1 STEP 1 UNTIL 5 DO IOUT[I]);
2700   WRITE(OUT,<I1>,N);
2800   WRITE(OUT,<I3>,M);
2900   WRITE(OUT,<I3>,P);
3000   WRITE(OUT,<I3>,II);
3100   WRITE(OUT,<I3>,III);
3200   WRITE(OUT,<F7.3>,V);
3300   WRITE(OUT,<P10.3>,TMAX);
3400   WRITE(OUT,<I6>,DCINT);
3500   WRITE(OUT,<P10.3>,DELTAT);
3600   WRITE(OUT,<6(I1,""),I1>,FOR I:=1 STEP 1 UNTIL 7 DO IOPT[I]);
3700   IF IOPT[1]=1 THEN WRITE(OUT,<P7.3>,D[1]) ELSE
3800     WRITE(OUT,<1(F7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO D[I]);
3900   IF IOPT[2]=1 THEN WRITE(OUT,<P10.3,"",P10.3>,AS[1],AE[1]) ELSE
4000     WRITE(OUT,<1(F7.3,"",P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N
4100       DO[AS[I],AE[I]]);
4200   IF IOPT[3]=1 THEN WRITE(OUT,<P7.3>,YCMIN[1]) ELSE
4300     WRITE(OUT,<1(F7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO YCMIN[I]);
4400   IF IOPT[4]=1 THEN WRITE(OUT,<P7.3>,YCMAX[1]) ELSE
4500     WRITE(OUT,<1(F7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO YCMAX[I]);
4600   IF IOPT[5]=1 THEN WRITE(OUT,<P7.3>,YEMIN[1]) ELSE
4700     WRITE(OUT,<1(F7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO YEMIN[I]);
4800   IF IOPT[6]=1 THEN WRITE(OUT,<P7.3>,YEMAX[1]) ELSE
4900     WRITE(OUT,<1(F7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO YEMAX[I]);
5000   WRITE(OUT,<I2>,INSR);
5100   IF INSR NEQ 0 THEN
5200     FOR I:=1 STEP 1 UNTIL INSR DO
5300       WRITE(OUT,<4(I1,""),P10.3>,R1[I],R2[I],P1[I],P2[I],SR[I]);
5400     IF IOPT[7]=1 THEN WRITE(OUT,<P11.5>,S[1,1]) ELSE
5500       FOR I:=1 STEP 1 UNTIL N DO
5600         WRITE(OUT,<1(F11.5,X1)>,FOR L:=1 STEP 1 UNTIL N DO S[I,L]);
5700     WRITE(OUT,<1(I1,X1)>,FOR I:=1 STEP 1 UNTIL N DO ICIN[I]);
5800     FOR I:=1 STEP 1 UNTIL N DO
5900       BEGIN
```

```

6000         IF ICIN[I]=0 THEN WRITE(OUT,<P11.5>,CO[I]);
6100         IF ICIN[I]=1 THEN
6200         WRITE(OUT,<I1,"",P10.3,"",P11.5>,IPORM[I],COTIME[I],CO[I]);
6300         IF ICIN[I]=2 THEN
6400         WRITE(OUT,<I1,"",I3,"",I3>,IPORM[I],IENPOI[I],ISTPOI[I]);
6500         IF ICIN[I]=2 AND IPORM[I]=0 THEN
6600         WRITE(OUT,<P11.5>,C[I,ISTPOI[I]]);
6700         IF ICIN[I]=2 AND IPORM[I]=1 THEN
6800         WRITE(OUT,<10(P10.5)>,FOR L:=IENPOI[I] STEP 1 UNTIL ISTPOI[I]
6900             DO C[I,L]);
7000         END;
7100     END PRINT;
7200
7300 PROCEDURE PRINTSCREEN;
7400 BEGIN
7500     WRITE(SCREEN,<A*,>,LENGTH(TITLE),TITLE);
7600     WRITE(SCREEN,<"PLOTNAMEC=",A6," PLOTNAMEE=",A6>,PLNAC,PLNAE);
7700     WRITE(SCREEN,<"IOUT: ",5(I1,"")>,FOR I:=1 STEP 1 UNTIL 5 DO
7800         IOUT[I]);
7900     WRITE(SCREEN,<"N=",I1," POINTS=",I3," LENGTH=",I3>,N,M,P);
8000     WRITE(SCREEN,<"TIMESTEPC= ",I3," TIMESTEPE= ",I3>,II,III);
8100     WRITE(SCREEN,<"V=",P7.3," TMAX=",P10.3," DCINT=",I7," DELTAT=",
8200         P10.3>,V,TMAX,DCINT,DELTAT);
8300     WRITE(SCREEN,<"IOPT: ",7(I1,"")>,FOR I:=1 STEP 1 UNTIL 7 DO
8400         IOPT[I]);
8500     IF IOPT[1]=1 THEN WRITE(SCREEN,<"D=",P7.3>,D[1]) ELSE
8600     WRITE(SCREEN,<"D= ",5(P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO D[I]);
8700     IF IOPT[2]=1 THEN
8800     WRITE(SCREEN,<"AS,AE= ",2P10.3,X1>,AS[1],AE[1]) ELSE
8900     WRITE(SCREEN,<"AS,AE= ",5(P7.3,X1,P7.3,X1)>,FOR I:=1 STEP 1 UNTIL
9000     N DO[AS[I],AE[I]]);
9100     IF IOPT[3]=1 THEN WRITE(SCREEN,<"YCMIN= ",P7.3>,YCMIN[1]) ELSE
9200     WRITE(SCREEN,<"YCMIN= ",5(P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO
9300     YCMIN[I]);
9400     IF IOPT[4]=1 THEN WRITE(SCREEN,<"YCMAX= ",P7.3>,YCMAX[1]) ELSE
9500     WRITE(SCREEN,<"YCMAX= ",5(P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO
9600     YCMAX[I]);
9700     IF IOPT[5]=1 THEN WRITE(SCREEN,<"YEMIN= ",P7.3>,YEMIN[1]) ELSE
9800     WRITE(SCREEN,<"YEMIN= ",5(P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO
9900     YEMIN[I]);
10000    IF IOPT[6]=1 THEN WRITE(SCREEN,<"YEMAX= ",P7.3>,YEMAX[1]) ELSE
10100    WRITE(SCREEN,<"YEMAX= ",5(P7.3,X1)>,FOR I:=1 STEP 1 UNTIL N DO
10200    YEMAX[I]);
10300    WRITE(SCREEN,<"NUMBER OF SO-REACTIONS= ",I2>,INSR);
10400    IF INSR NEQ 0 THEN
10500    FOR I:=1 STEP 1 UNTIL INSR DO
10600    WRITE(SCREEN,<"R",I1,4(X1,I1),P10.3>,I,R1[I],R2[I],P1[I],P2[I],
10700    SR[I]);
10800    IF IOPT[7]=1 THEN WRITE(SCREEN,<"S= ",P11.5>,S[1,1]) ELSE
10900    FOR I:=1 STEP 1 UNTIL N DO
11000    WRITE(SCREEN,<"S= ",5(P11.5,X1)>,FOR L:=1 STEP 1 UNTIL N DO
11100    S[I,L]);
11200    WRITE(SCREEN,<"ICIN= ",5(I1,X1)>,FOR I:=1 STEP 1 UNTIL N DO
11300    ICIN[I]);
11400    FOR I:=1 STEP 1 UNTIL N DO
11500    BEGIN
11600        IF ICIN[I]=0 THEN WRITE(SCREEN,<"C= ",P11.5>,CO[I]);
11700        IF ICIN[I]=1 THEN
11800        WRITE(SCREEN,<"IPORM= ",I1," COTIME= ",P10.3," C= ",P11.5>,

```

```
11900         IFORM[I],COTIME[I],CO[I]);
12000     IF ICIN[I]=2 THEN
12100         WRITE(SCREEN,<"IFORM= ",I1,"  ENDPOINT= ",I3,"  STARTPOINT= ",
12200             I3>,IFORM[I],IENPOI[I],ISTPOI[I]);
12300     IF ICIN[I]=2 AND IFORM[I]=0 THEN
12400         WRITE(SCREEN,<"C= ",P11.5>,C[I,ISTPOI[I]]);
12500     IF ICIN[I]=2 AND IFORM[I]=1 THEN
12600         WRITE(SCREEN,<"C= ",10(P10.5)>,FOR L:=IENPOI[I] STEP 1 UNTIL
12700             ISTPOI[I] DO C[I,L]);
12800     END;
12900     END PRINTSCREEN;
13000
13100 PROCEDURE READFILE;
13200     BEGIN
13300         READ(INP,<A70>,TITLE);
13400         READ(INP,<A6>,PLNAC);
13500         READ(INP,<A6>,PINAE);
13600         READ(INP,/,IOUT[1],IOUT[2],IOUT[3],IOUT[4],IOUT[5]);
13700         READ(INP,/,N);
13800         READ(INP,/,M);
13900         READ(INP,/,P);
14000         READ(INP,/,I1);
14100         READ(INP,/,I11);
14200         READ(INP,/,V);
14300         READ(INP,/,TMAX);
14400         READ(INP,/,DCINT);
14500         READ(INP,/,DELTAT);
14600         READ(INP,/,IOPT[1],IOPT[2],IOPT[3],IOPT[4],IOPT[5],IOPT[6],
14700             IOPT[7]);
14800         I:=1;
14900         IF IOPT[1]=1 THEN READ(INP,/,D[1]);
15000         IF IOPT[1]=0 THEN READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO D[I]);
15100         IF IOPT[2]=1 THEN READ(INP,/,AS[1],AE[1]);
15200         IF IOPT[2]=0 THEN
15300             READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO[AS[I],AE[I]]);
15400         IF IOPT[3]=1 THEN READ(INP,/,YCMIN[1]);
15500         IF IOPT[3]=0 THEN
15600             READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO YCMIN[I]);
15700         IF IOPT[4]=1 THEN READ(INP,/,YCMAX[1]);
15800         IF IOPT[4]=0 THEN
15900             READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO YCMAX[I]);
16000         IF IOPT[5]=1 THEN READ(INP,/,YEMIN[1]);
16100         IF IOPT[5]=0 THEN
16200             READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO YEMIN[I]);
16300         IF IOPT[6]=1 THEN READ(INP,/,YEMAX[1]);
16400         IF IOPT[6]=0 THEN
16500             READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO YEMAX[I]);
16600         READ(INP,/,INSR);
16700         IF INSR NEQ 0 THEN
16800             FOR I:=1 STEP 1 UNTIL INSR DO
16900                 READ(INP,/,R1[I],R2[I],P1[I],P2[I],SR[I]);
17000         IF IOPT[7]=1 THEN READ(INP,/,S[1,1]);
17100         IF IOPT[7]=0 THEN
17200             FOR I:=1 STEP 1 UNTIL N DO
17300                 READ(INP,/,FOR L:=1 STEP 1 UNTIL N DO S[I,L]);
17400         READ(INP,/,FOR I:=1 STEP 1 UNTIL N DO ICIN[I]);
17500         FOR I:=1 STEP 1 UNTIL N DO
17600             BEGIN
17700                 IF ICIN[I]=0 THEN READ(INP,/,CO[I]);
```

```
17800             IF ICIN[I]=1 THEN READ(INP,/,IFORM[I],COTIME[I],CO[I]);
17900             IF ICIN[I]=2 THEN READ(INP,/,IFORM[I],IENPOI[I],ISTPOI[I]);
18000             IF ICIN[I]=2 AND IFORM[I]=0 THEN READ(INP,/,C[I,ISTPOI[I]);
18100             IF ICIN[I]=2 AND IFORM[I]=1 THEN
18200             READ(INP,/,FOR L:=IENPOI[I] STEP 1 UNTIL ISTPOI[I] DO C[I,L]);
18300             END;
18400             END READFILE;
18500 $BEGINSEGMENT
18600
18700             BOOLEAN PROCEDURE PARSEINPUT(INARRAY,OUTFILE,ERRORLIMIT);
18800 %             #####
18900             VALUE ERRORLIMIT;
19000             INTEGER ERRORLIMIT;
19100             ARRAY INARRAY[*];
19200             FILE OUTFILE;
19300             BEGIN
19400             STRING NAMEVALUE,RESEVEDWORDVALUE,LITERALVALUE;
19500             REAL NUMBEVALUE;
19600             BOOLEAN ENDOFFILEVALUE,PARSEDONE,SYNTAXERROR,
19700             INTEGER STRINGLENGTH,CURRENTSTATE,COLUMNLEFT,INDEX,ARRAYSIZE,
19800             STARTCOLUMN,ENDCOLUMN;
19900             POINTER TOKENP,ENDP,STRINGP;
20000             TRUTHSET NUMBERS("0123456789.-+@"),DIGIT("0123456789"),
20100             LETTERS(ALPHA ANL NOT DIGIT);
20200             DEFINE RESERVEDWORDF=47:16#,OVERFLOWF=31:16#,TYPEVALUEF=15:16#;
20300             DEFINE NEXTSTATEF=47:12#,INPUTVALUEF=35:12#,STATEF=23:12#,
20400             CASENUMF=11:12#;
20500             DEFINE TOKENV=0#,SYNTAXV=1#,SEMANTICV=2#;
20600             DEFINE NAMEV=0#,NUMBERV=1#,LITV=2#,EOFV=3#;
20700             $INCLUDE "SYMBOL/TABLES"
20800 %#####
20900
21000             PROCEDURE SEMANTICS(WHICH);
21100             VALUE WHICH;
21200             INTEGER WHICH;
21300             FORWARD;
21400
21500             PROCEDURE ERROR(ERRORMESSAGE);
21600             VALUE ERRORMESSAGE;
21700             STRING ERRORMESSAGE;
21800             FORWARD;
21900             DEFINE HASHIT=INDEX:=(REAL(TOKENP,MIN(STRINGLENGTH,4))+
22000             STRINGLENGTH) MOD HASHMODV#;
22100
22200             INTEGER PROCEDURE LOOKUP;
22300 %             #####
22400             BEGIN
22500             BOOLEAN MATCHFOUND;
22600             HASHIT;
22700             DO
22800             BEGIN
22900             STRINGP:=POINTER(RESERVEDWORDS)+
23000             HASHTABLE[INDEX].[RESERVEDWORDF];
23100             IF STRINGLENGTH=INTEGER(STRINGP:STRINGP,2) THEN
23200             IF TOKENP=STRINGP FOR STRINGLENGTH THEN
23300             BEGIN
23400             MATCHFOUND:=TRUE;
23500             LOOKUP:=HASHTABLE[INDEX].[TYPEVALUEF];
23600             END;
```

```
23700         END UNTIL MATCHFOUND OR
23800         INDEX:=HASHTABLE[INDEX].[OVERFLOW]=4"FFFF";
23900     IF NOT MATCHFOUND THEN
24000     IF TOKENP IN LETTERS THEN LOOKUP:=NAMEV ELSE
24100         BEGIN
24200             LOOKUP:=-1;
24300             ERROR("*** UNRECOGNIZED SYNTAX ***");
24400         END;
24500     END LOOKUP;
24600
24700 INTEGER PROCEDURE GETTOKEN;
24800 BEGIN
24900     DEFINE EOFTOKEN=48"OO"#;
25000     SCAN TOKENP:ENDP FOR COLUMNLEFT: COLUMNLEFT WHILE=" ";
25100     STARTCOLUMN:=ARRAYSIZE-COLUMNLEFT;
25200     IF(IF COLUMNLEFT=0 THEN TRUE ELSE TOKENP=EOFTOKEN)THEN
25300         BEGIN
25400             GETTOKEN:=EOFV;
25500             ENDOFFILEVALUE:=TRUE;
25600         END
25700     ELSE
25800     IF(IF TOKENP IN DIGIT FOR 1 THEN TRUE ELSE
25900     TOKENP IN NUMBERS FOR 2) THEN
26000         BEGIN
26100             SCAN ENDP:TOKENP FOR COLUMNLEFT: COLUMNLEFT WHILE IN
26200             NUMBERS;
26300             ENDCOLUMN:=ARRAYSIZE-COLUMNLEFT;
26400             IF STRINGLENGTH:=ENDCOLUMN-STARTCOLUMN>13 THEN
26500                 BEGIN
26600                     GETTOKEN:=-1;
26700                     ERROR("*** ILLEGAL NUMBER ***");
26800                 END
26900             ELSE
27000                 BEGIN
27100                     GETTOKEN:=NUMBERV;
27200                     NUMBERVALUE:=DECIMAL(STRING(TOKENP,STRINGLENGTH));
27300                 END;
27400             END
27500         ELSE
27600         IF TOKENP IN LETTERS THEN
27700             BEGIN
27800                 SCAN ENDP:TOKENP FOR COLUMNLEFT: COLUMNLEFT WHILE IN ALPHA;
27900                 ENDCOLUMN:=ARRAYSIZE-COLUMNLEFT;
28000                 STRINGLENGTH:=ENDCOLUMN-STARTCOLUMN;
28100                 IF(GETTOKEN:=LOOKUP)=NAMEV THEN
28200                     NAMEVALUE:=STRING(TOKENP,STRINGLENGTH) ELSE
28300                     RESERVEDWORDVALUE:=STRING(TOKENP,STRINGLENGTH);
28400                 END
28500             ELSE
28600             IF TOKENP="" THEN
28700                 BEGIN
28800                     IF COLUMNLEFT>0 THEN
28900                         BEGIN
29000                             STARTCOLUMN:=*+1;
29100                             TOKENP:=TOKENP+1;
29200                             COLUMNLEFT:=*-1;
29300                         END;
29400                     SCAN ENDP:TOKENP FOR COLUMNLEFT: COLUMNLEFT UNTIL="";
29500                     ENDCOLUMN:=ARRAYSIZE-COLUMNLEFT;
```

```
29600         IF COLUMNLEFT=0 OR STRINGLENGTH:=ENDCOLUMN-STARTCOLUMN=0
29700         THEN
29800             BEGIN
29900                 GETTOKEN:=-1;
30000                 ERROR("*** MISSING END QUOTE ***");
30100             END
30200         ELSE
30300             BEGIN
30400                 GETTOKEN:=LIPV;
30500                 LITERALVALUE:=STRING(TOKENP,STRINGLENGTH);
30600                 IF COLUMNLEFT>0 THEN
30700                     BEGIN
30800                         ENDP:=ENDP+1;
30900                         COLUMNLEFT:=*-1;
31000                     END;
31100                 END;
31200             END
31300         ELSE
31400             BEGIN
31500                 STRINGLENGTH:=1;
31600                 IF COLUMNLEFT>0 THEN
31700                     BEGIN
31800                         COLUMNLEFT:=*-1;
31900                         ENDP:=TOKENP+1;
32000                     END;
32100                 GETTOKEN:=LOOKUP;
32200                 RESERVEDWORDVALUE:=STRING(TOKENP,STRINGLENGTH);
32300                 END SPECIAL CHARACTER;
32400             END GETTOKEN;
32500
32600     PROCEDURE PARSE;
32700     BEGIN
32800         INTEGER TOKEN,RULE;
32900         IF TOKEN:=GETTOKEN GEQ 0 THEN
33000             BEGIN
33100                 IF
33200 RULE:=ARRAYSEARCH(O&TOKEN[INPUTVALUEF]&CURRENTSTATE[STATEF],
33300 O&4"FFF"[INPUTVALUEF]&4"FFF"[STATEF],GRAMMAR)<0 THEN
33400                 IF TOKEN=EOPV THEN
33500                     BEGIN
33600                         STRINGLENGTH:=1;
33700                         ERROR("*** MORE INPUT REQUIRED ***");
33800                     END
33900                 ELSE ERROR("*** UNEF ECTED INPUT IN THIS CONTEXT ***") ELSE
34000                     BEGIN
34100                         CURRENTSTATE:=GRAMMAR[RULE].[NEXTSTATEF];
34200                         IF RULE:=GRAMMAR[RULE].[CASENUMF] NEQ 4"FFF" THEN
34300                             SEMANTICS(RULE);
34400                         END;
34500                     END;
34600                 END PARSE;
34700
34800     PROCEDURE SEMANTICS(WHICH);
34900     %     #####
35000     VALUE WHICH;
35100     INTEGER WHICH;
35200     BEGIN
35300         CASE WHICH OF
35400         BEGIN
```



```
35500 1: INPUTFINISHED:=PARSEDONE:=TRUE;
35600 2: REMOTEINPUT:=TRUE;
35700 3: REMOTEINPUT:=FALSE;
35800 INPUTTITLE:=LITERALVALUE;
35900 4: OUTPUTTITLE:=LITERALVALUE;
36000 10: TITLE:=LITERALVALUE;
36100 11: PLNAC:=NAMEVALUE;
36200 12: PLNAE:=NAMEVALUE;
36300 13: INPUTPRINT:=RESERVEDWORDVALUE="NO";
36400 14: IOUT[2]:=NUMBERVALUE;
36500 15: NEGCON:=RESERVEDWORDVALUE="NO";
36600 IF NEGCON THEN IOUT[3]:=1 ELSE IOUT[3]:=0;
36700 16: INTEGRATION:=RESERVEDWORDVALUE="NO";
36800 IF INTEGRATION THEN IOUT[4]:=0 ELSE IOUT[4]:=1;
36900 17: HARDPLOT:=RESERVEDWORDVALUE="NO";
37000 IF HARDPLOT THEN IOUT[5]:=0 ELSE IOUT[5]:=1;
37100 18: N:=NUMBERVALUE;
37200 19: M:=NUMBERVALUE;
37300 20: P:=NUMBERVALUE;
37400 21: II:=NUMBERVALUE;
37500 22: III:=NUMBERVALUE;
37600 23: V:=NUMBERVALUE;
37700 24: TMAX:=NUMBERVALUE;
37800 25: DELTAT:=NUMBERVALUE;
37900 26: IOPT[1]:=0;
38000 FOR I:=1 STEP 1 UNTIL N DO
38100 BEGIN
38200 WRITE(SCREEN,<"DISPERSION COEFFICIENT OF COMPONENT ",
38300 I1," ?">,I);
38400 READ(KEYBOARD,/,D[I]);
38500 END;
38600 27: IOPT[2]:=0;
38700 FOR I:=1 STEP 1 UNTIL N DO
38800 BEGIN
38900 WRITE(SCREEN,<"AS AND AE OF COMPONENT ",I1," ?">,I);
39000 READ(KEYBOARD,/,AS[I],AE[I]);
39100 END;
39200 28: IOPT[3]:=0;
39300 FOR I:=1 STEP 1 UNTIL N DO
39400 BEGIN
39500 WRITE(SCREEN,<"YCMIN OF COMPONENT ",I1," ?">,I);
39600 READ(KEYBOARD,/,YCMIN[I]);
39700 END;
39800 29: IOPT[4]:=0;
39900 FOR I:=1 STEP 1 UNTIL N DO
40000 BEGIN
40100 WRITE(SCREEN,<"YCMAX OF COMPONENT ",I1," ?">,I);
40200 READ(KEYBOARD,/,YCMAX[I]);
40300 END;
40400 30: IOPT[5]:=0;
40500 FOR I:=1 STEP 1 UNTIL N DO
40600 BEGIN
40700 WRITE(SCREEN,<"YEMIN OF COMPONENT ",I1," ?">,I);
40800 READ(KEYBOARD,/,YEMIN[I]);
40900 END;
41000 31: IOPT[6]:=0;
41100 FOR I:=1 STEP 1 UNTIL N DO
41200 BEGIN
41300 WRITE(SCREEN,<"YEMAX OF COMPONENT ",I1," ?">,I);
```

```

41400          READ(KEYBOARD,/,YEMAX[I]);
41500          END;
41600 32:      IOPT[7]:=0;
41700          FOR I:=1 STEP 1 UNTIL N DO
41800            BEGIN
41900              FOR L:=1 STEP 1 UNTIL N DO
42000                WRITE(SCREEN,<"S[" ,I1," ,",I1," ] ?">,I,L);
42100                READ(KEYBOARD,/,FOR L:=1 STEP 1 UNTIL N DO S[I,L]);
42200            END;
42300 33:      D[1]:=NUMBERVALUE;
42400 34:      AS[1]:=NUMBERVALUE;
42500 35:      AE[1]:=NUMBERVALUE;
42600 36:      YCMIN[1]:=NUMBERVALUE;
42700 37:      YCMAX[1]:=NUMBERVALUE;
42800 38:      YEMIN[1]:=NUMBERVALUE;
42900 39:      YEMAX[1]:=NUMBERVALUE;
43000 40:      INSR:=NUMBERVALUE;
43100          IF INSR NEQ 0 THEN
43200            FOR I:=1 STEP 1 UNTIL INSR DO
43300              BEGIN
43400                WRITE(SCREEN,<"NUMBER OF REACTANT ONE ?">);
43500                READ(KEYBOARD,/,R1[I]);
43600                WRITE(SCREEN,<"NUMBER OF REACTANT TWO ?">);
43700                READ(KEYBOARD,/,R2[I]);
43800                WRITE(SCREEN,<"NUMBER OF PRODUCT ONE ?">);
43900                READ(KEYBOARD,/,P1[I]);
44000                WRITE(SCREEN,<"NUMBER OF PRODUCT TWO ?">);
44100                READ(KEYBOARD,/,P2[I]);
44200                WRITE(SCREEN,<"RATE CONSTANT FOR REACTION ",I2," ?">,
44300                    I);
44400                READ(KEYBOARD,/,SR[I]);
44500              END;
44600 41:      S[1,1]:=NUMBERVALUE;
44700 42:
44800          WRITE(SCREEN,
44900            <"0:CONTINUOUS-INPUT ,1:TIME-INPUT ,2:POINT INPUT">);
45000          FOR I:=1 STEP 1 UNTIL N DO
45100            BEGIN
45200              WRITE(SCREEN,<"INPUT TYPE OF COMPONENT ",I1,
45300                " 0,1,OR 2 ?">,I);
45400              READ(KEYBOARD,/,ICIN[I]);
45500              IF ICIN[I]=0 THEN
45600                BEGIN
45700                  WRITE(SCREEN,<"CONCENTRATION ?">);
45800                  READ(KEYBOARD,/,CO[I]);
45900                END;
46000              IF ICIN[I]=1 THEN
46100                BEGIN
46200                  WRITE(SCREEN,<"TIME ? AND CONCENTRATION ?">);
46300                  READ(KEYBOARD,/,COTIME[I],CO[I]);
46400                  IFORM[I]:=1;
46500                END;
46600              IF ICIN[I]=2 THEN
46700                BEGIN
46800                  WRITE(SCREEN,
46900                    <"FORM (0 OR 1) ? ENDPOINT ? STARTPOINT ?">);
47000                  READ(KEYBOARD,/,IFORM[I],IENPOI[I],ISTPOI[I]);
47100                  IF IFORM[I]=0 THEN
47200                    WRITE(SCREEN,<"CONCENTRATION ?">);

```

```
47300          IF IPFORM[I]=0 THEN
47400          READ(KEYBOARD,/,C[I,ISTPOI[I]]);
47500          IF IPFORM[I]=1 THEN
47600          FOR L:=IENPOI[I] STEP 1 UNTIL ISTPOI[I] DO
47700              BEGIN
47800                  WRITE(SCREEN,<"CONCENTRATION AT ",I3," ?">,
47900                      L);
48000                  READ(KEYBOARD,/,C[I,L]);
48100              END;
48200          END;
48300          END;
48400 46:        PRINTSCREEN;
48500 47:        IF NOT INPUTFINISHED THEN
48600          IF NOT REMOTEINPUT THEN
48700              BEGIN
48800                  CLOSE(INP);
48900                  REPLACE PT:=POINTER(TITLEAR) BY INPUTTITLE,".";
49000                  INP(FILETYPE=7,TITLE=PT);
49100                  READFILE;
49200              END;
49300 48:        REPLACE PT:=POINTER(TITLEAR) BY OUTPUTTITLE,".";
49400          OUT(NEWFILE=FALSE,TITLE=PT);
49500          IF OUT.RESIDENT THEN
49600              BEGIN
49700                  WRITE(SCREEN,
49800                      <"RESIDENT FILE ! OVERWRITE ? YES=1; NO=0;">);
49900                  READ(KEYBOARD,/,J);
50000                  IF J=1 THEN CLOSE(OUT,PURGE);
50100              END
50200          ELSE J:=1;
50300          IF J=1 THEN
50400              BEGIN
50500                  OUT(NEWFILE=TRUE);
50600                  PRINT(OUT);
50700                  LOCK(OUT,CRUNCH);
50800              END;
50900          END CASE;
51000          IF WHICH GEQ 10 THEN PARSEDONE:=TRUE;
51100          END SEMANTICS;
51200
51300          PROCEDURE ERROR(ERRORMESSAGE);
51400          %      #####
51500          VALUE ERRORMESSAGE;
51600          STRING ERRORMESSAGE;
51700          BEGIN
51800              ARRAY UNDERLINE[0:12];
51900              POINTER P,PSTART;
52000              INTEGER UNITSPERWORD;
52100              UNITSPERWORD:=IF OUTFILE.UNITS=VALUE(CHARACTERS) THEN 6 ELSE
52200                  1;
52300              REPLACE P:=POINTER(UNDERLINE) BY "      " FOR 13 WORDS;
52400              IF OFFSET(TOKENP)+STRINGLENGTH>78 THEN
52500                  PSTART:=TOKENP-MIN(50,OFFSET(TOKENP)) ELSE
52600                  PSTART:=POINTER(INARRAY);
52700              WRITE(OUTFILE,(MIN(ARRAYSIZE-OFFSET(PSTART),78)+5)DIV 6*
52800                  UNITSPERWORD,PSTART);
52900              REPLACE P+(OFFSET(TOKENP)-OFFSET(PSTART)) BY "#" FOR
53000                  MIN(STRINGLENGTH,5);
53100              WRITE(OUTFILE,13*UNITSPERWORD,UNDERLINE);
```

```
53200          WRITE(OUTFILE,13*UNITS PERWORD,ERRORMESSAGE CAT REPEAT(" ",72-
53300              LENGTH(ERRORMESSAGE)));
53400          SYNTAXERROR:=TRUE;
53500          ENDOFFILEVALUE:=TRUE;
53600          IF(ERRORLIMIT:=*-1)=0 THEN PARSEDONE:=TRUE;
53700          END ERRORS;
53800 %
53900 %
54000 %
54100          ARRAYSIZE:=6*SIZE(INARRAY);
54200          COLUMNLEFT:=ARRAYSIZE;
54300          ENDP:=POINTER(INARRAY);
54400          DO PARSE UNTIL PARSEDONE;
54500          PARSEINPUT:=NOT SYNTAXERROR;
54600          END OF P A R S E I N P U T;
54700 $ENDSEGMENT
54800
54900          PROCEDURE READINPUT;
55000          BEGIN
55100              COMMENT***DEFAULTS***;
55200              FOR I:=1 STEP 1 UNTIL 7 DO IOPT[I]:=1;
55300              FOR I:=1 STEP 1 UNTIL 5 DO IOUT[I]:=IFORM[I]:=1;
55400              IOUT[2]:=2;
55500              DCINT:=100000;
55600              M:=401;
55700              N:=1;
55800              REMOTEJOB:=MYSELF.STATION<0;
55900              REMOTEINPUT:=REMOTEJOB;
56000              IF REMOTEINPUT THEN KEYBOARD.OPEN:=TRUE;
56100              INPUTFINISHED:=FALSE;
56200              PC:=POINTER(CARD);
56300              WHILE NOT INPUTFINISHED DO
56400                  BEGIN
56500                      FILEFINISHED:=FALSE;
56600                      IF REMOTEINPUT THEN
56700                          BEGIN
56800                              WRITE(SCREEN[STOP],<">">);
56900                              READ(KEYBOARD,72,PC);
57000                          END
57100                      ELSE
57200                          BEGIN
57300                              FILEFINISHED:=READ(INP,<A72>,PC);
57400                              IF FILEFINISHED THEN
57500                                  BEGIN
57600                                      IF REMOTEJOB THEN REMOTEINPUT:=TRUE ELSE
57700                                          INPUTFINISHED:=TRUE;
57800                                      END;
57900                                  END;
58000                              IF NOT INPUTFINISHED AND NOT FILEFINISHED THEN
58100                                  PARSEINPUT(CARD,SCREEN,1);
58200                              END WHILE;
58300                          END READINPUT;
58400                      READINPUT;
58500                  END.
#
```

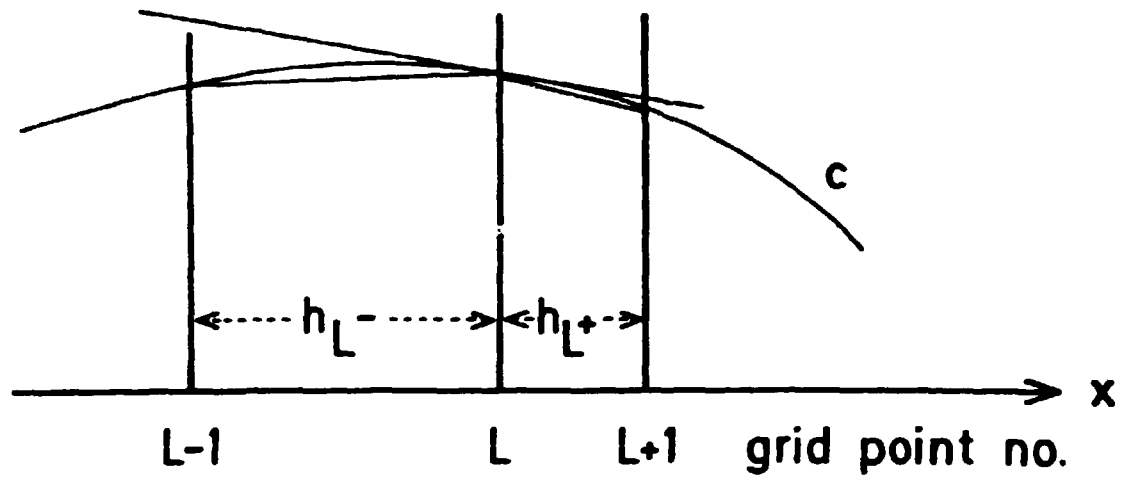


Fig. 3.1. Interpolation between non-equidistant grid points.

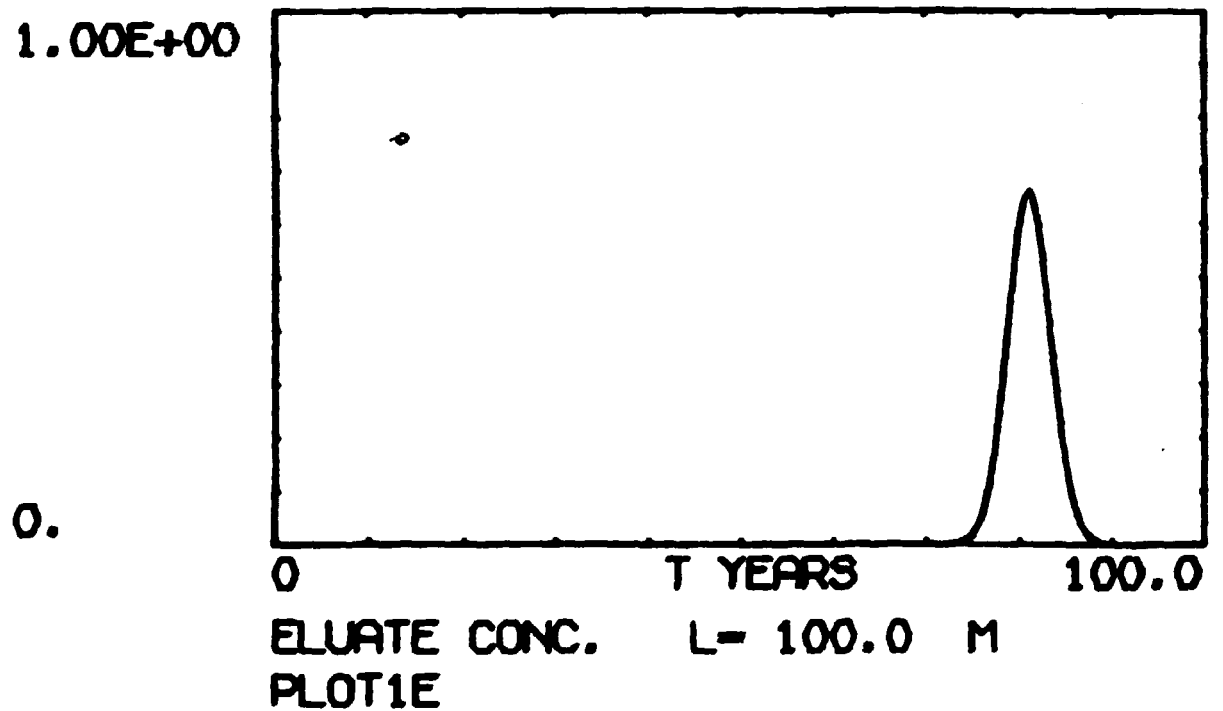
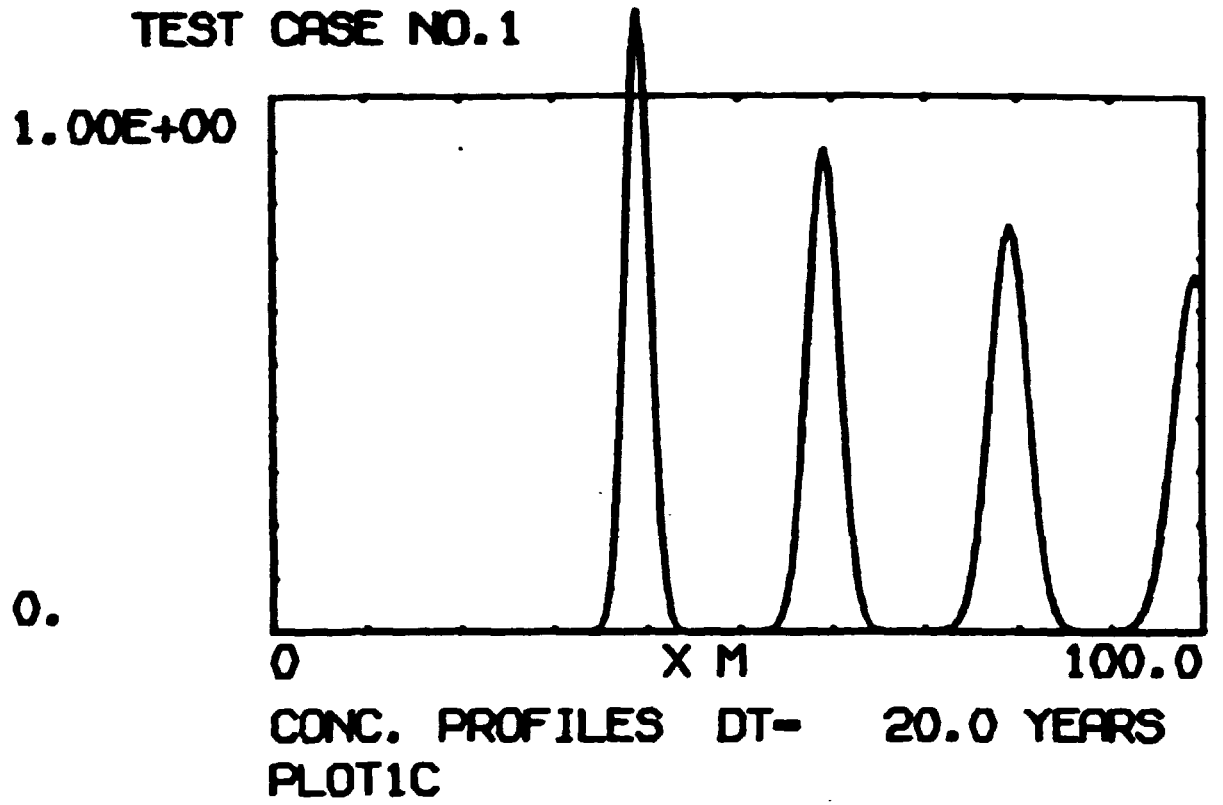


Fig. 6.1. Plotted output from test example 1

THIS IS TEST CASE NO.2

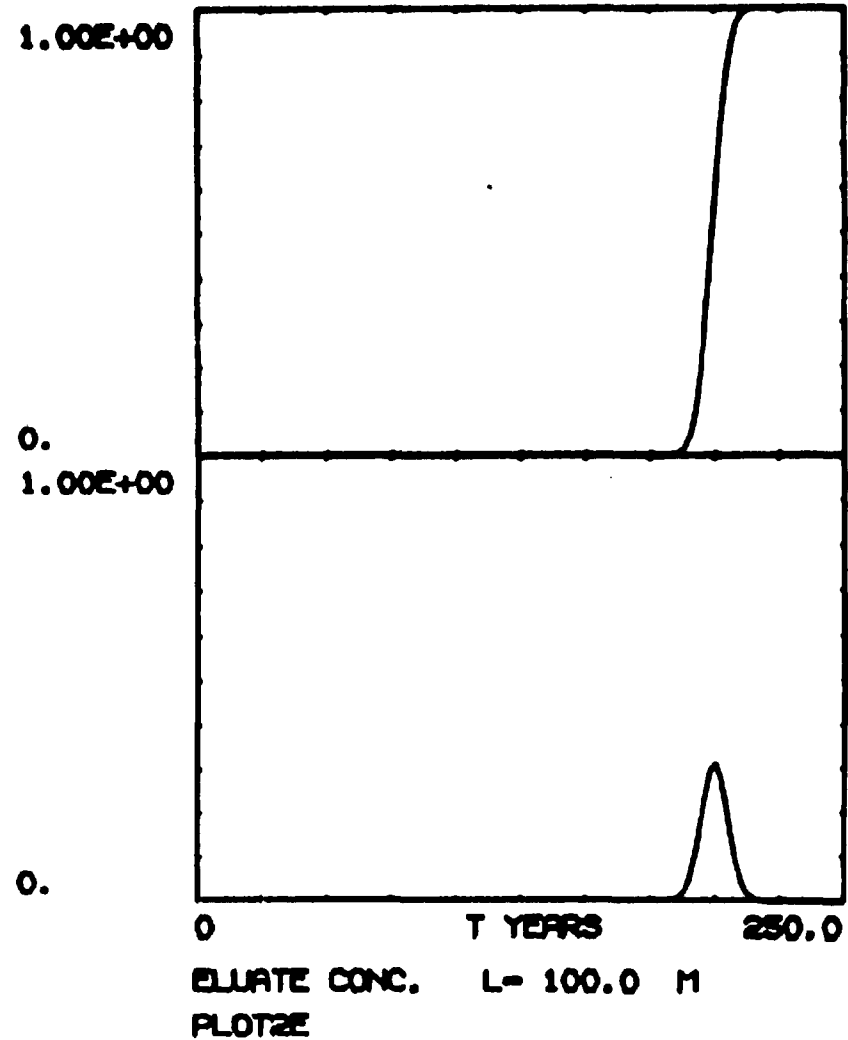
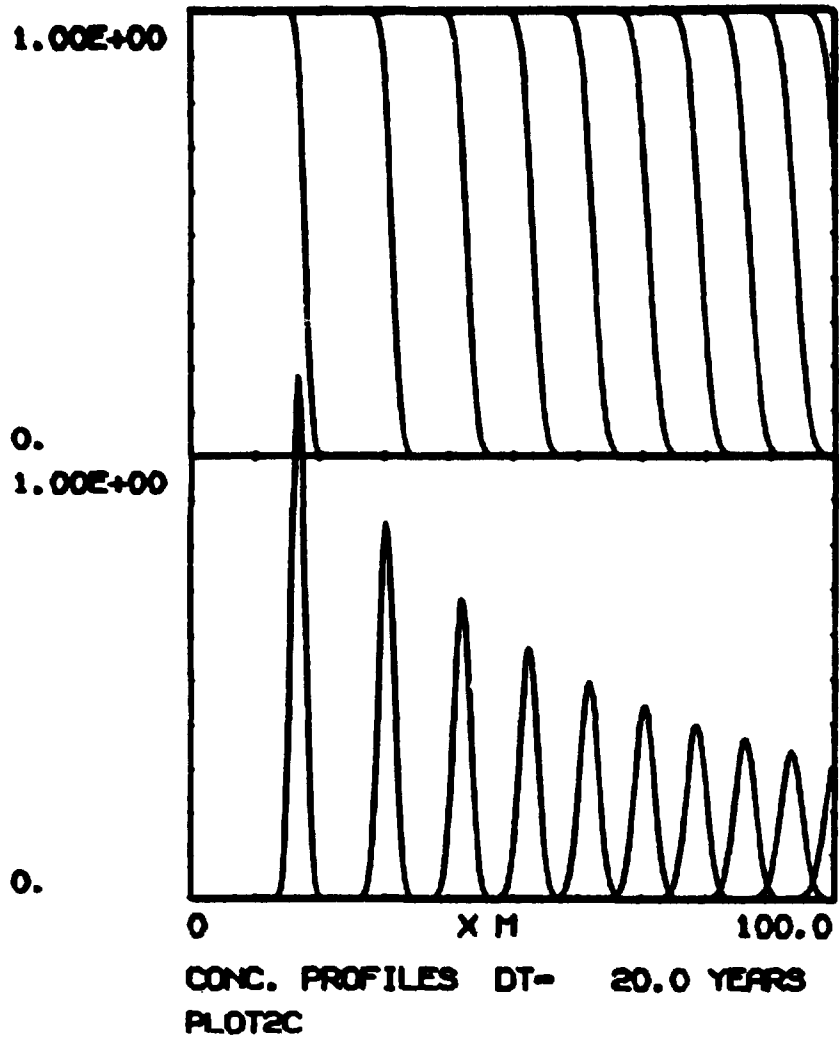


Fig. 6.2. Plotted output from test example 2

THIS IS TEST CASE NO. 3A

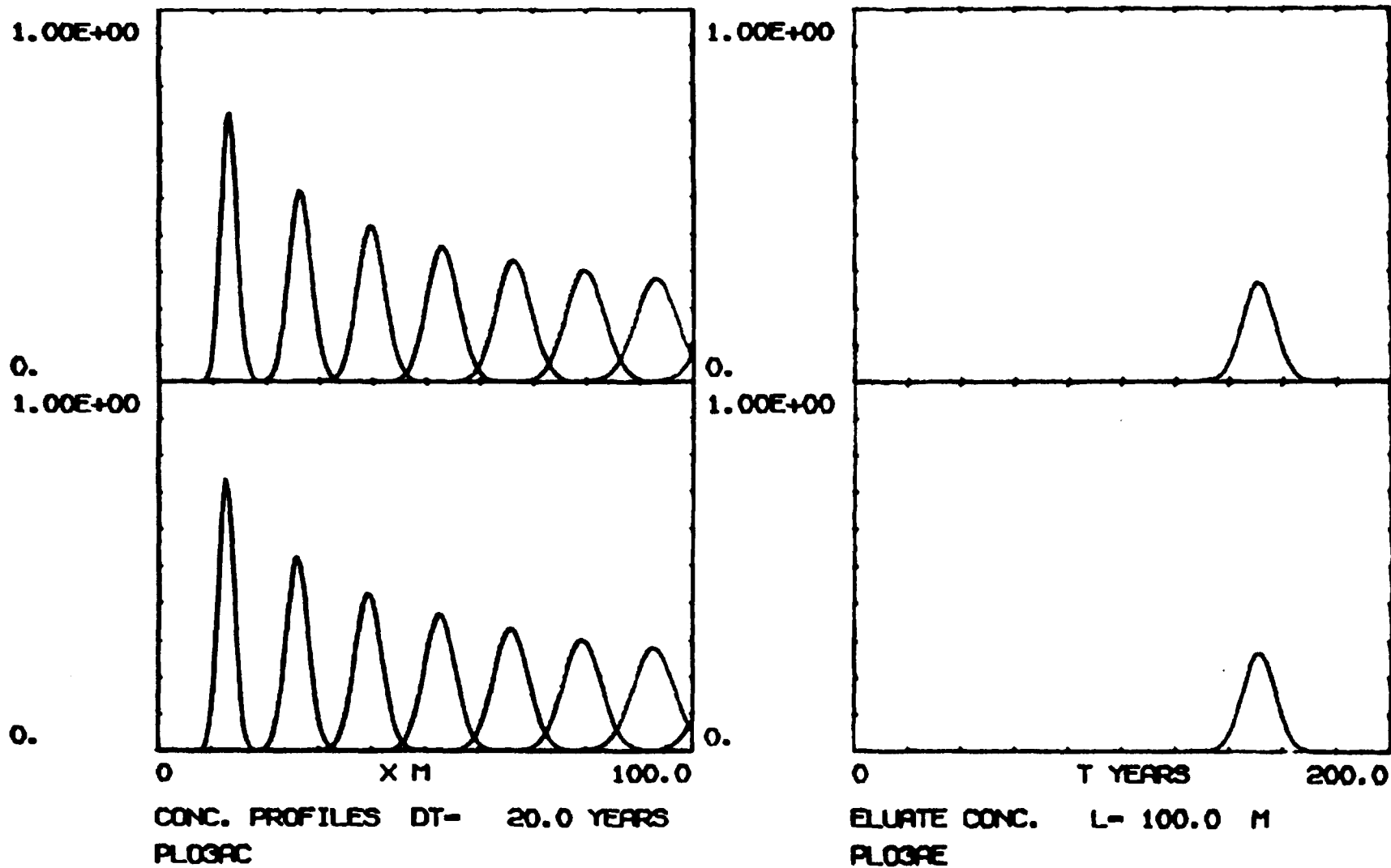


Fig. 6.3. Plotted output from test example 3A

THIS IS TEST CASE NO. 3B

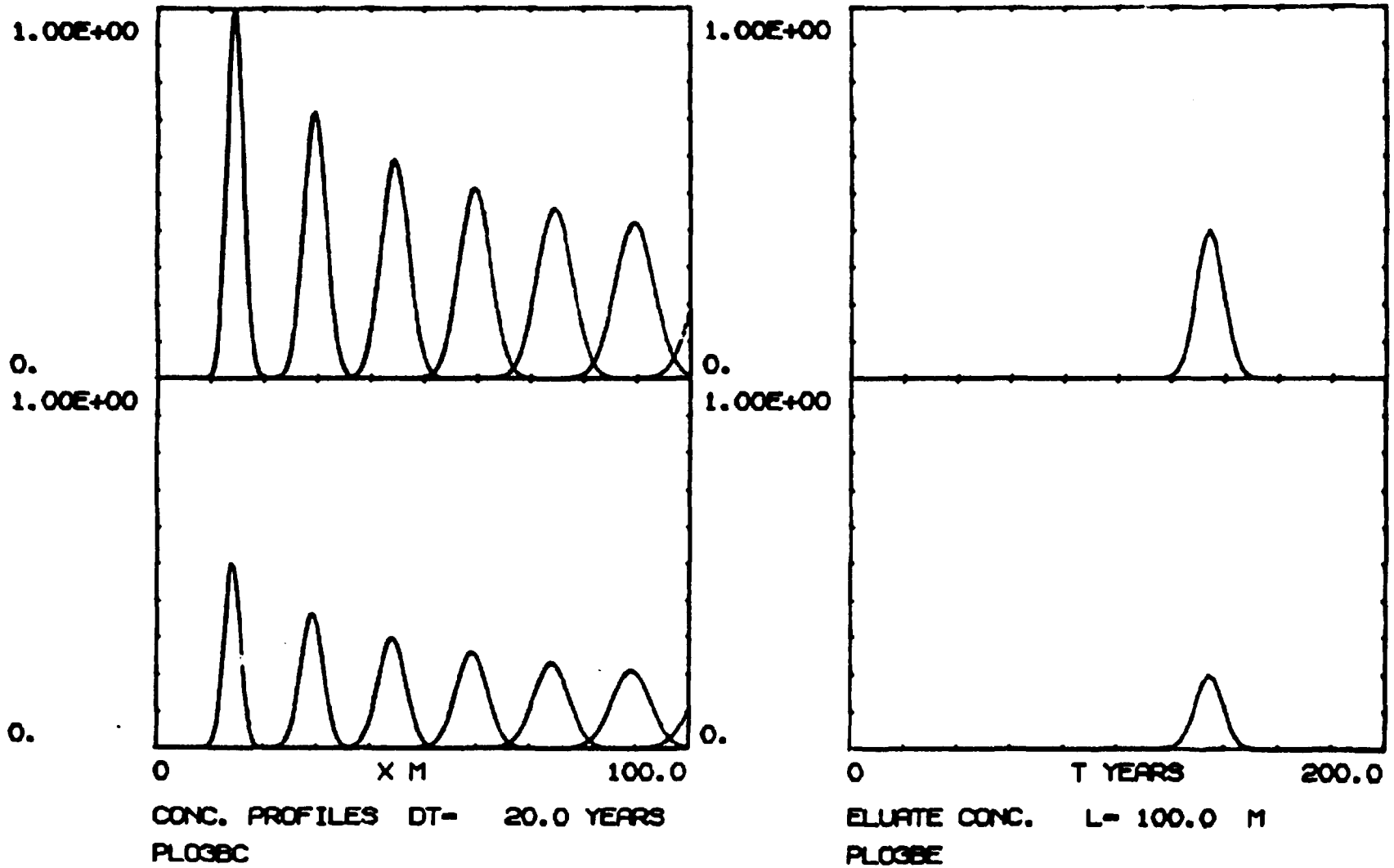


Fig. 6.4. Plotted output from test example 3B

THIS IS TEST CASE NO.3C

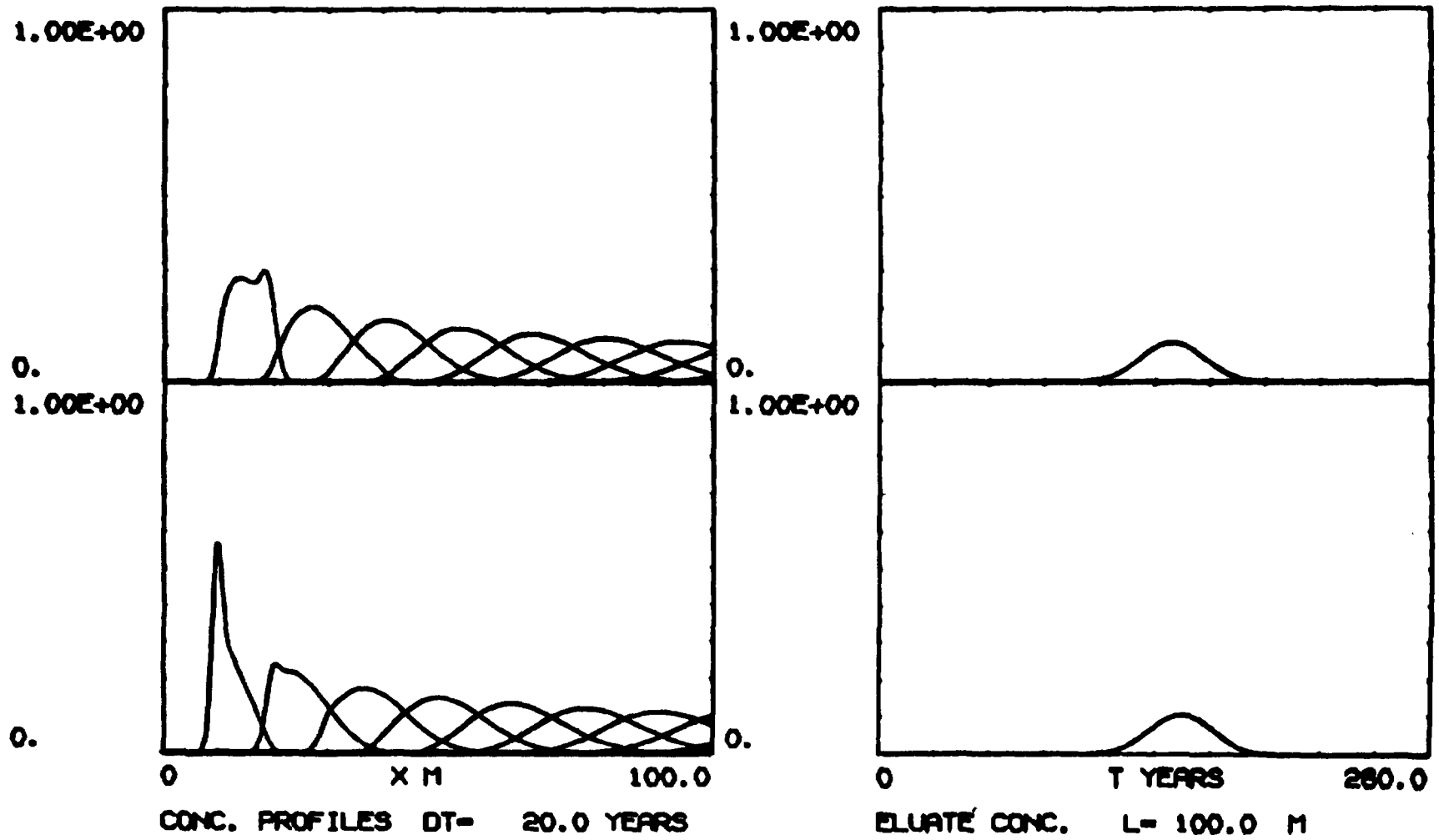


Fig. 6.5. Plotted output from test example 3C

THIS IS TEST CASE NO. 3D

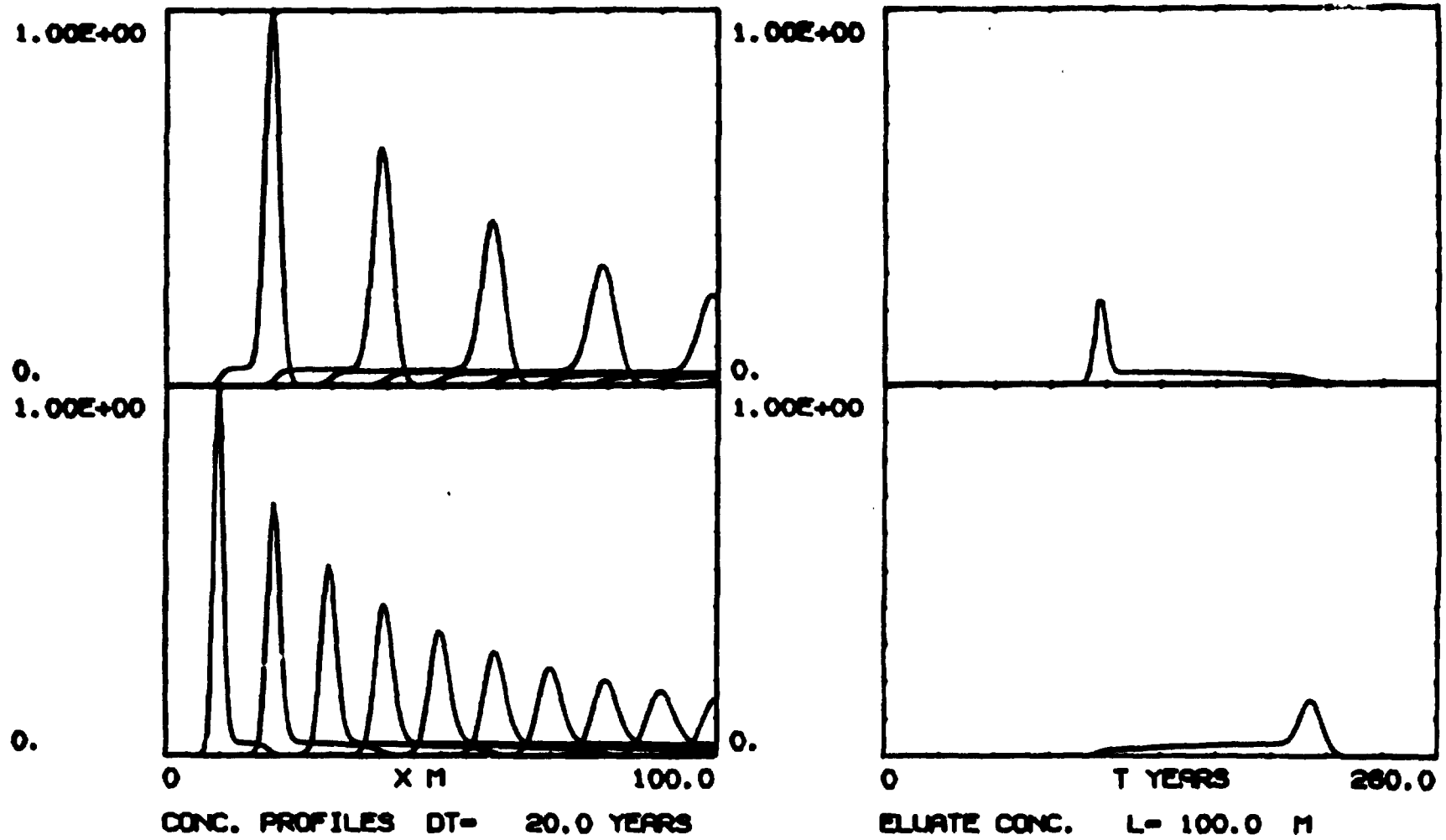


Fig. 6.6. Plotted output from test example 3D

THIS IS TEST CASE NO.3E

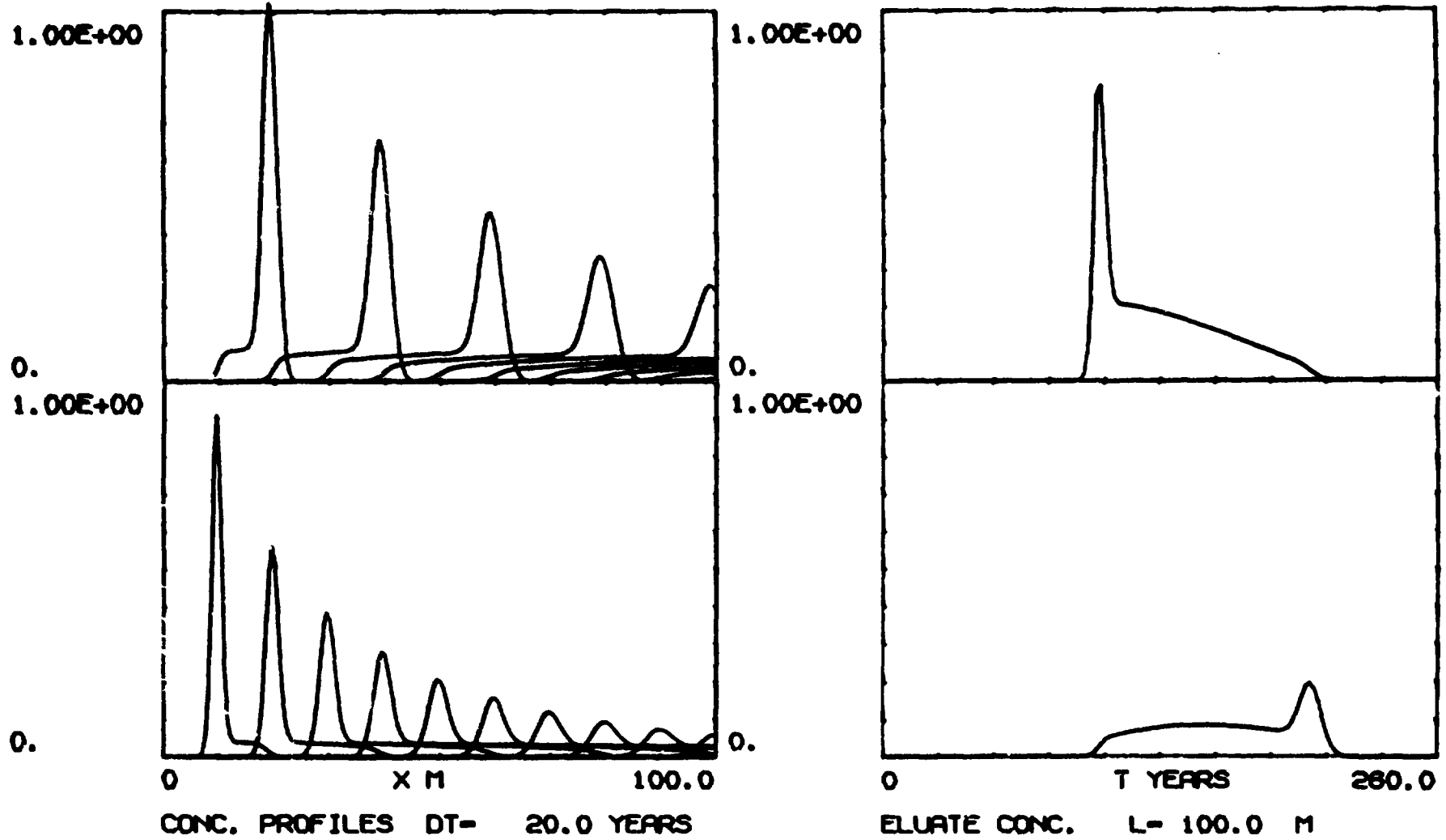
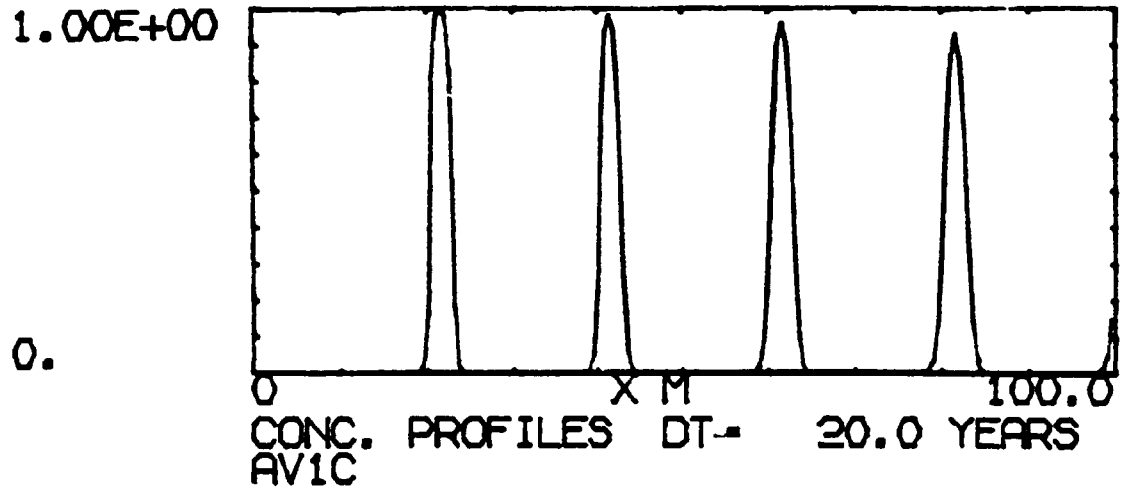


Fig. 6.7. Plotted output from test example 3E

TEST CASE NO. 4B: A=10; V=10;



TEST CASE NO. 4A: A=1; V=1;

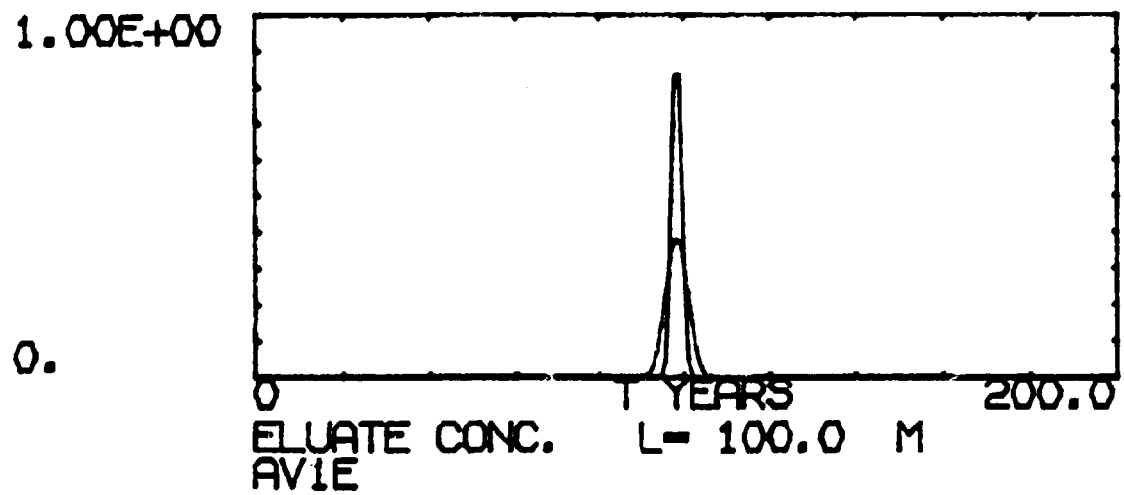
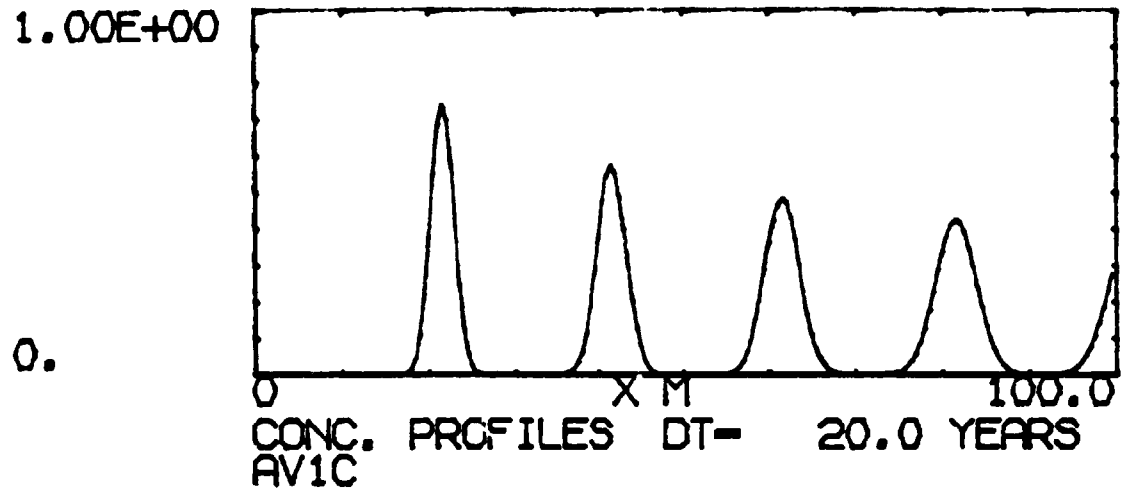


Fig. 6.8. Plotted output output from test example 4A and 4B

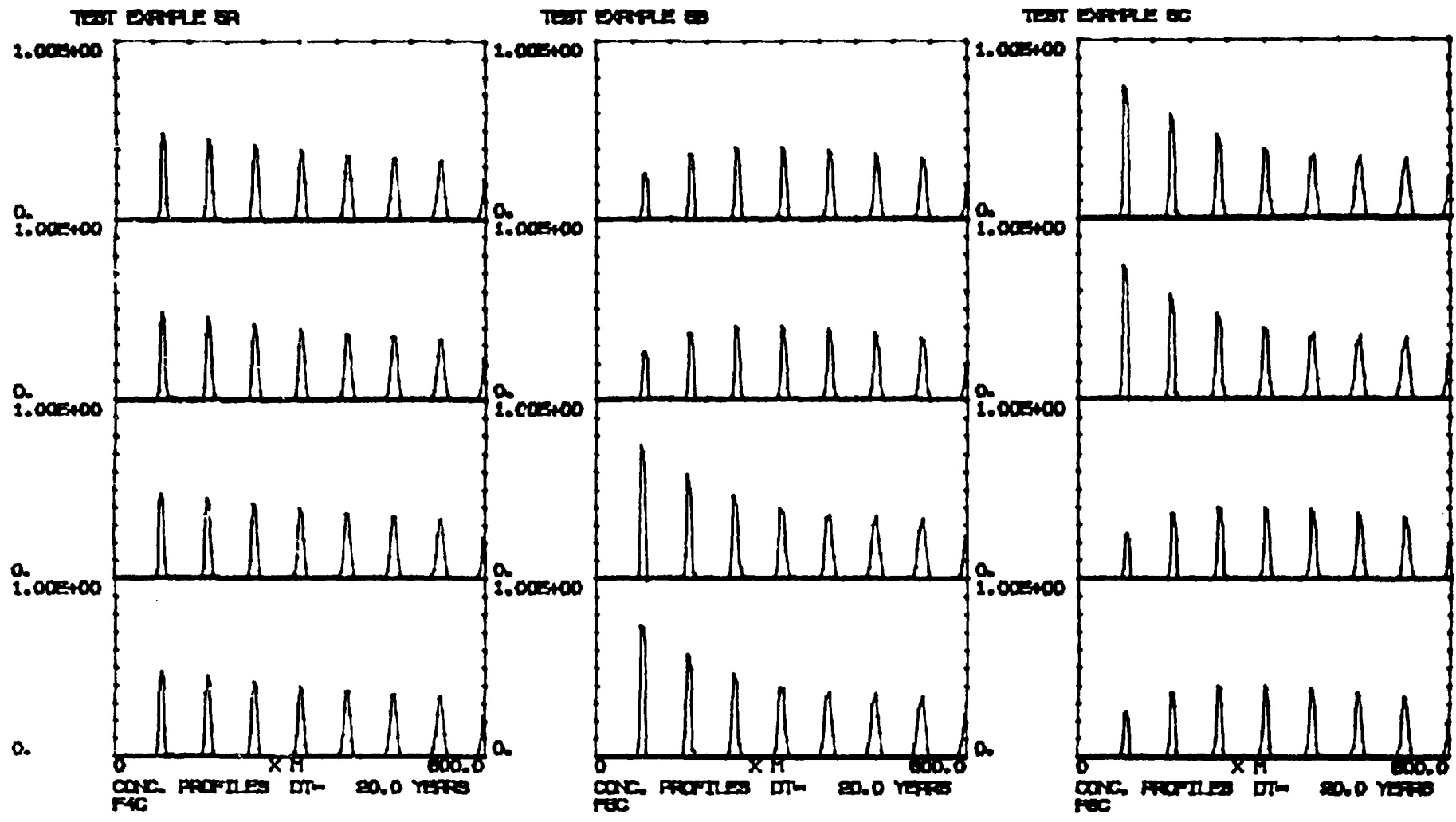


Fig. 6.9. Plotted output from test example 5A, 5B, and 5C

**Sales distributors:
G.E.C. Gad Strøget
Vimmelskiftet 32
DK-1161 Copenhagen K, Denmark**

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

**ISBN 87-550-1148-9
ISSN 0106-2840**