Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

Title of Thesis: Process Modelling of The Nitration of Toluene

Nitin K. Shah, M. S. In Chemical Engineering, 1989

Thesis directed by: Dr. Edward C. Roche, Jr.

The thesis on the process modelling of the nitration process focuses specifically on the product separation of the nitrated aromatic material from the reactor and wash section of the process. Included in the text is an overview of the main process steps associated with the production of para and ortho-nitrotoluene. The critical issues concerning the reaction of nitric acid and toluene were summerized, and resulted in this subject being deferred for future work.

The separation of the isomers into pure components is an important part of the MNT process because of the close high boiling points of the isomeric products. The most desirable product in the process is the para isomer even though the ortho isomer is usable. An existing computer model was modified and used to study the batch distillation columns used to recover toluene and the marketable isomers of MNT. A typical washed organic intermediate feed to the separation section (batch columns) was analyzed using the continuous reflux batch distillation model. The results of the modelling program indicate that the separation campaign is a lengthy process, but that high purity products can be obtained.

Process Modelling of The Nitration Of Toluene

by
Nitin K. Shah

Thesis submitted to the Faculty of the Graduate School of
the New Jersey Institute of Technology in partial fulfillment of
the requirements for the degree of
Master of Science in Chemical Engineering.
1989

APPROVAL SHEET

Title of Thesis: Process Modelling Of The Nitration Of Toluene.

Name of Candidate: Nitin K. Shah
Master of Science in Chemical Engineering, 1989

Thesis and Abstract Approved: _____ 21 September 89

Dr. Edward C. Roche, Jr. Date
Professor
Department of Chemical Engineering

_____ 21 Sept 89

Dr. Ernest N. Bart Date
Assistant Professor
Department of Chemical Engineering

_____ 22 Sept 89

Irwin Hundert Date
Visiting Professor
Department of Chemical Engineering

VITA

Name: Shah Nitin K.

Permanent address:

India.

Degree and date to be conferred: Master of Science in Chemical Engineering, 1989.

Date of birth:

Place of birth:

Secondary Education: Shree Vidyanagar High School, Ahmedabad, India, 1979.

Collegiate institutions attended	Dates	Degree	Date of Degree
N. J. Institute of Technology	01/88-12/89	M.S.Chem. Engg.	Dec, 1989
M. S. University of Baroda	06/79-06/84	B.S.Chem.Engg.	Nov, 1984

Major: Chemical Engineering.

ACKNOWLEDGMENT

I wish to express my gratitude to Dr. E. C. Roche for his assistance and appropriate guidance during the course of my work on this thesis. I would also like to thanks Mr. R. Springer for the use of the computer code which was taken from his project work on batch distillation.

I would like to express my thanks to numerous members of the faculty at the Chemical Engineering Department of the New Jersey Institute of Technology for their assistance and guidance during my years at the institute.

I would also like to express my thanks to my friends for their direct or indirect support for my work.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The nitration of aromatic hydrocarbons is a unit process of considerable industrial importance. Some nitration processes are carried out in the vapor phase but the majority are conducted under liquid conditions, and involve reactions between an organic hydrocarbon phase and an aqueous phase consisting of a mixture of nitric acid, sulfuric acid and water.

The nitration of toluene has been the subject of extensive study for over 100 years, but research continues because of the great synthetic and industrial utility of the nitrated products. Control of the isometric composition of the nitration derivatives of toluene and its homologues has been an important but elusive goal. The 2,4,6-TNT as it is formed creates a need for additional processing steps if a stable and safe explosive is to be made and also generates a major environmental problem in disposal of the products to which the minor isomers are converted. The manufacture of dinitrotoluenes (DNT), which is ultimately converted to toluene di-isocyanate for polyurethanes, is far more significant than that of TNT. A large variety of uses exist for the para and ortho isomers of mononitrotoluene, but seldom has the demand for the individual compounds matched their production.

In past decade paranitrotoluene was most needed, and extensive research is devoted to maximize its production. In recent years new products based on orthonitrotoluene have effected the demand. Application of metanitrotoluene have suffered because, as a minor co-product, its availability always depends

upon changing market conditions. Gaining flexible control of the isomers produced in nitration of toluene continues to be a valid, if difficult, objective providing scientific challenge as well as opportunities to achieve important level of conservation of resources. With improvement in understanding the factors that govern the products of nitration, our prospects for their regulations in production should also improve.

CHAPTER II

DEFINITION OF THE PROCESS

Aromatic nitration has been studied in great detail for several decades, and as a result, it stands as one of the relatively few well understood phenomena in organic chemistry. The reaction is still known only imperfectly and many misconceptions of its nature are widely held. We are primarily interested in the mononitration of toluene.

Liquid phase nitration of toluene has been performed industrially on a large scale for many years. Mononitrotoluenes are produced by the nitration of toluene. It is well known that nitration of aromatic molecules produces a number of isometric products. These mononitro compounds are used as intermediates for manufacture of dyestuffs, pharmaceuticals, etc. The presence of the methyl group on the aromatic ring facilitates the nitration at higher temperature but at the same time it increases the ease of oxidation which sometime results in undesirable products. Thus, the nitration of toluene generally is carried out at a lower temperatures than the nitration of benzene to minimize oxidative side reactions. Toluene is less soluble in the acid phase, thus vigorous agitation of the reaction mixture is necessary to maximize the interfacial area of the two phases and mass transfer of the reactants.

Mononitrotoluenes can be produced by either a batch or continuous process. We will focus our attention on the batch process because better control of the process. A wide range of nitrating agents have been used in laboratory experiments and industrial nitrations. The main consideration for a nitrating

agent is that it be available in quantity, reasonably cheap and relatively stable. The work described below is done using a mixture of nitric acid and sulphuric acid, called mixed acid. This nitrating agent has had wide industrial use for many years. The major steps involved in the process are explained below:

(2.1) Mixed acid preparation:

Nitric acid of a required quantity is first taken into a stainless steel mixing vessel. Sulphuric acid is added slowly with the temperature at 35 to 40 °C. The temperature of the mixture should not go above 40 °C.

(2.2) Nitration of toluene to mononitrotoluene (MNT):

Toluene with the required quantity is charged first in a stainless steel reactor containing a very efficient stirring and cooling arrangement. The mixed acid is then slowly added to it ensure that the temperature of the mass does not exceed 35 °C. It is most important that temperature is controlled by stirring and proper cooling. The temperature of the reaction can also be maintained by adjusting the acid rate of addition. The acid is added below the surface of the toluene. After the addition of required acid, the temperature is raised slowly to 40 °C. The reactor contents are agitated for an additional hour to complete the reaction. After completion of the reaction, the reaction mixture is put into a separator where the mass is allow to settle down for about an hour. The acidic layer is drained from the bottom and is reconcentrated. The crude nitrated organic product from the top layer is transferred to a wash tank.

(2.3) Washing of crude MNT:

Crude mononitrotoluenes after charging to the wash tank is first washed

with water and then washed with dilute alkali. Finally the organic material s washed with water to free it from alkali traces. The wash system is provided with a good stirring arrangement. The product is analyzed using chromatographic equipment for unreacted starting materials (toluene), ortho, meta, and para mononitrotoluenes and for the dinitration product. The crude MNT is stored in a storage tank before processing it for recovery of the toluene and separation of the isomers of mononitrotoluenes.

(2.4) Separation of crude MNT into pure isomers:

The crude MNT is separated into the pure isomers by distillation in various steps. The separation process involved following steps:

(i) Removal of toluene and moisture by distillation:

The crude MNT from the storage tank is charged to the primary batch distillation column in order to remove the unreacted toluene and the traces of water. Water and toluene both will be collected in the overhead receiver. The toluene is recycled to the next nitration campaign. The removal of water and toluene is accomplished easily in the atmospheric batch distillation column.

(ii) Recovery of o-NT from isomer mixture:

Due to the high boiling points of the isomers of MNT, a vacuum batch distillation is carried out in a second column. The batch distillation column has with two overhead condensers and a reboiler. The crude MNT is taken in after vaporizing it in a boiler. The first fraction will be o-NT and is removed via the condenser at the top of the column. It is then crystalized to facilitate the removal of trace quantities of p-NT.

(iii) Recovery of pure p-NT from isomer mixture by crystalization:

The second cut in the column will be m-NT. Since the boiling points of these isomers are very close, a p-NT rich mixture of both isomers will be obtained. The pure p-NT is then recovered by crystallization. The remaining mixture, called mother liquor, still contain all of the three isomers and some of the dinitro products, is transferred to mother liquor tank for a second distillation using the vacuum column as an additional campaign step.

(iv) Mother liquor distillation for m-NT rich fraction:

The mother liquor is charged in the column and remaining part of o-NT is recovered. The p-NT rich stream is directed to the storage tank and transferred to crystallizer for the recovery of pure p-NT. The residual from the column will contain mostly m-NT along with p-NT and dinitro compounds. It is heated and again distilled for the recovery of pure m-NT and the unrecovered part of p-NT. The pure m-NT is then crystallized.

A detailed process block diagram is shown in Figure 1A and 1B.

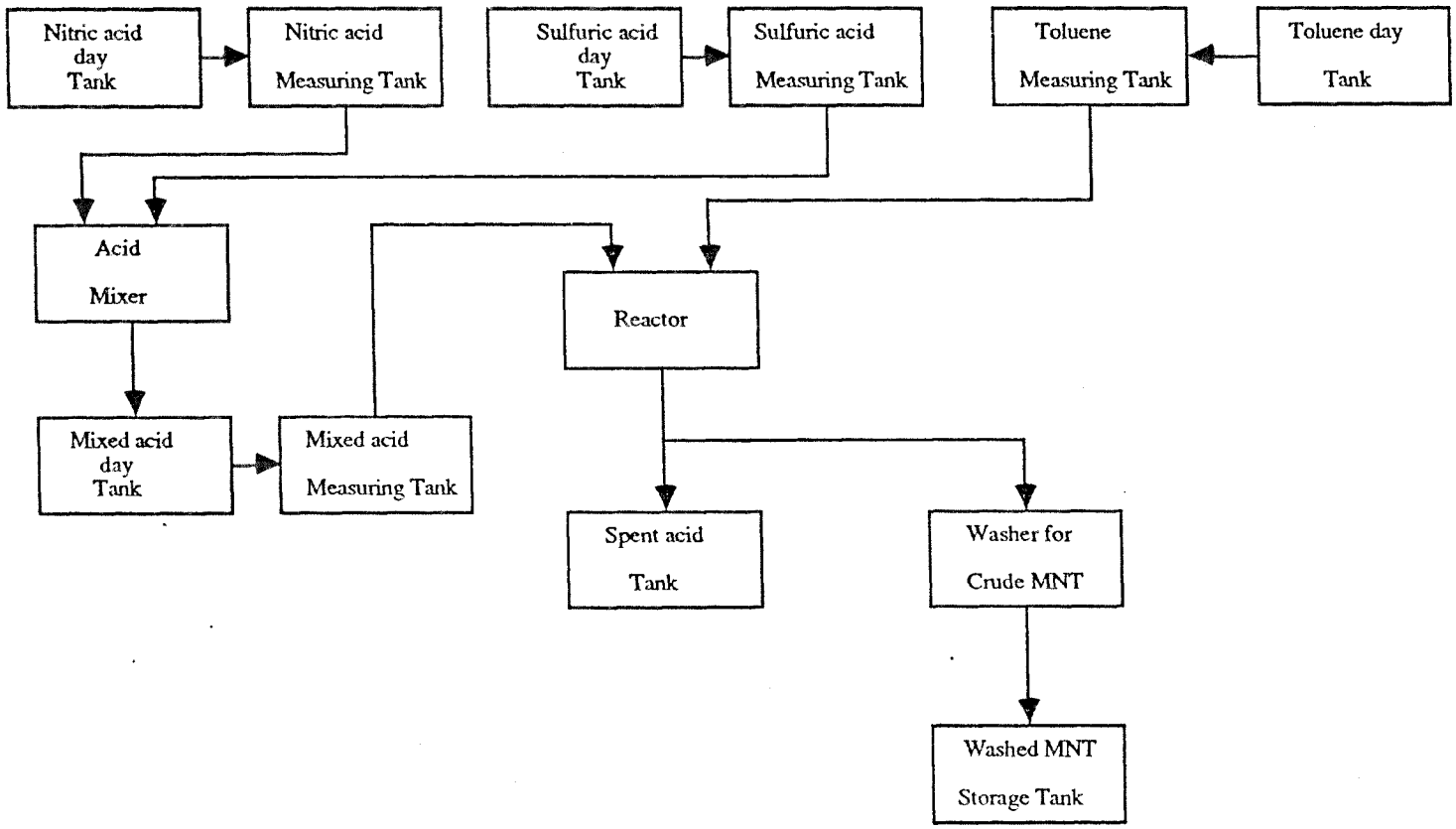

Figure 1A Process Block Diagram

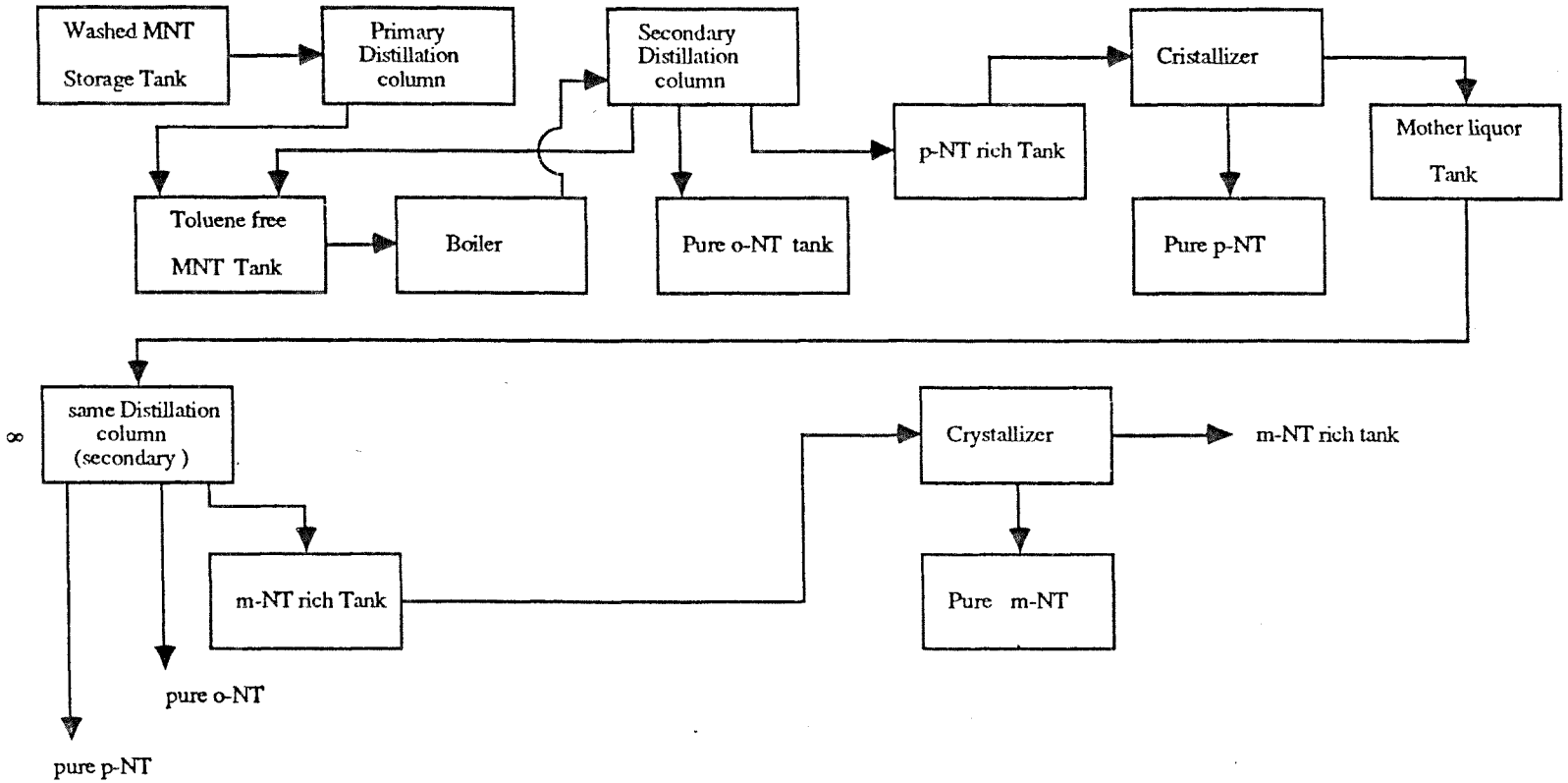

Figure 1B Process Block Diagram

CHAPTER III

TWO PRINCIPAL PROCESS STEPS

The commercial nitration of aromatic substances with mixed acid has been carried out in batch equipment for nearly a century. The process is divided into two major operations

(3.1) Reaction and,

(3.2) Separation of the isomers.

(3.1) Reaction:

For a commercial nitration of several aromatics such as benzene and toluene, the acid phase is a mixture of nitric acid, sulfuric acid and water, known as mixed acid. Mixed acid proportions are generally 52-56% wt. sulfuric acid (H_2SO_4), 28-32% wt. nitric acid (HNO_3) and 12-20% wt. water (H_2O). The design of a batch nitrator has been based mainly on empirical rules and heat transfer rates since it is an exothermic reaction. The nitration reactions certainly occur either in the acid phase or at least at the interface between liquid phases. The aromatic reactant must hence be transferred to the interface and possibly also into the acid phase or vice-versa. Thus the design and scale-up of an industrial nitrator is a problem of simultaneous mass transfer and chemical reaction, and it is essential to know which of these is the limiting step so that the proper parameters for scaleup can be choosen.

(3.2) Separation of the isomers:

Isomers are separated by batch distillation process in sequence. Batch distillation is much simpler in concept than continuous distillation. A mixture

is charged into a boiler (still pot) and is heated to vaporize it. The vapor is continuously condensed and collected in a series of fractions. Because the vapor has a higher concentration of the light component than the corresponding liquid, the condensed fraction has initially higher light concentrations than the original mixture, and the later fractions (or the part left in the reboiler) have lower concentrations. Because the separation from a single equilibrium stage is generally not sufficient, it is usual to incorporate a number of stages by inserting a column between the boiler and condenser, filled with mass transfer internals through which a part of the condensate flows as reflux.

The outstanding advantage of batch distillation is that one column can separate a mixture containing a number of components into relatively pure componential products. To achieve a similar result with a continuous distillation system requires n-1 columns for a system of n components. The second advantage of batch distillation is that it is more flexible than a continuous column, and more suitable as general purpose equipment for purification and solvent recovery within a plant where numerous separations are required.

Two batch columns are used in the total separation process. One, known as the primary column, is used for the removal of the unreacted toluene and trace amount of water from the crude mononitrotoluenes. The second column is operated for the separation of the isomers of MNT into the pure compounds and is called the secondary column. Due to the higher boiling points of the products, the distillation is undertaken under vacuum.

CHAPTER IV

REACTION MECHANISM ANALYSIS

Aromatic nitration is one of the oldest and most common industrial reactions. This reaction is still not well understood due to its nature. Only a small amount of useful kinetic data is available to the engineer, and this is restricted to a small range of process operating conditions. The extension and interpretation of the existing data is therefore of importance.

The stoichiometry of the reaction is misleadingly simple:

$$HNO_3 + ArH \longrightarrow ArNO_2 + H_2O$$

This chemical reaction is misleading because sulfuric acid does not appear in it, and for this reason sulfuric acid has often been assigned the role of promotor. It has long been known that aromatic nitration is an irreversible reaction. Past studies of the nitration of toluene have shown conclusively that nitration can be considered in many cases as an electrophilic attack of the nitronium ion, NO_2^+, on the toluene nucleus. A comprehensive survey of this theory has been presented by Gillespie and Millen[1], while Haun and Kobe[2] have shown how an understanding of the theory is helpful in the interpretation and correlation of experimental results.

Unfortunately, most of the evidence presented by these workers in support of the nitronium ion mechanism was obtained under homogeneous conditions that could not be compared directly with conditions generally employed in commercial nitration reactors. It is evident that the acids used for mononitration

lie outside the nitronium ion; therefore, it is doubtful that the nitronium ion mechanism applies.

Liquid phase toluene mononitration under normal industrial conditions is an example of mass transfer with simultaneous chemical reaction. The problem of determining the magnitude and the nature of the resistance to interphase transfer has been avoided by much of the research on nitration kinetics, by the simple expedient of working with a solvent where all the reactants are miscible. Early work on the rate of two liquid phase heterogeneous reaction kinetics has been reviewed by Albright and Hanson[3,4]. They pointed out that, contrary to the assumption commonly made, all mass transfer resistance has not necessarily been eliminated if the reaction rate becomes constant at high levels of agitation.

Hanson and co-workers[5,6] studied the heterogeneous mononitration of the toluene using a nitrating acid typical of that employed in industry i.e. 15 mole % HNO_3, 30 mole % H_2SO_4, and 55 mole % H_2O. The results were shown to be inconsistent with a simple kinetic model and it was concluded that diffusional resistances plays a more important part in determining the nitration in the two liquid phase region. This is in agreement with the extensive results for small scale reactors published during the last few years by Strachan and co-workers[7,8,9], who have demonstrated that the overall rate can be correlated by a simple kinetic model at low sulfuric acid concentrations but that diffusional resistances become important at higher sulfuric acid concentrations. At low concenration of H_2SO_4, the transition from kinetic to mass transfer control is obtained, and, at the higher concentration, the transition from slow to fast reaction is observed. The concentration of nitric acid becomes less important

as the overall acidity rises. At higher acidity, the concentration of sulfuric acid is more important than the nitric acid concentration. This is consistent with the effect of sulfuric acid on the kinetics of homogeneous nitration reaction. The sulfuric acid is seen as an ionizing medium for nitric acid to the nitronium, ion and so the availability of the latter increases with sulfuric acid concentrations.

As explained above, experimental data with nitrating acids of compositions typical of those employed industrially cannot be reconciled with theoretical models assuming a uniform rate of reaction through either or both phases. Such models will only apply to slow reactions. The other extreme is an instantaneous reactions, which would lead to reaction taking place at the interface. Development of a model to describe the overall rate demands a knowledge of the locale of the reaction. Conflicting opinions have been expressed in the literature as to whether any reaction takes place in the organic phase. Lewis and Suen[10], having been able to separate the two phases rapidly and follow the disappearance of HNO_3 from the organic layer, concluded that some reaction does takes place in the organic phase. Barduhn and Kobe [11], however, showed that while the HNO_3 disappears from the organic layer, there is no corresponding disappearance of toluene. Sohrabi[12] has studied reaction in the organic phase and demonstrated that the maximum rate of reaction is many orders of magnitudes less than that in the aqueous phase. Thus, while some reaction may takes place in the organic phase during heterogeneous nitration, its contributions to the overall rate will be extremely small ($< 0.001\%$) and can be justifiably neglected.

In the view of the above, the most realistic model to consider for toluene mononitration under these conditions appears to be that of a fast reaction tak-

ing place in a zone in the aqueous phase adjacent to the organic-acid interface. The toluene will diffuse through this zone while reacting. This is illustrated in Figure 2. The mathematical approach for this situation has been developed by Astarita[13] and Danckwerts[14].

The following steps are envisaged for the mononitration of toluene:

(1) Toluene diffuses through the organic phase toward the interface.

(2) From the interface the toluene diffuses into the bulk aqueous phase.

(3) While diffusing into the aqueous phase the toluene reacts to form mononitrotoluenes.

(4) The nitrotoluenes formed diffuses back through the aqueous phase to the interface.

(5) From the interface the nitrotoluenes diffuses into the bulk organic phase.

(6) Nitric acid diffuses from the bulk aqueous phase towards the interface.

(7) While diffusing, it reacts with the toluene to form mononitrotoluenes.

(8) The water formed diffuses back into the bulk aqueous phase.

(9) Some nitric acid diffuses from the interface into the bulk organic phase.

For the reasons discussed above, processes 2 and 3 are probably rate determining. To simplify derivation of a model, the following assumptions can be made:

(1) Reaction in the organic phase is negligible.

(2) The nitronium ion, NO_2^+, mechanism is partially applicable.

(3) The water concentration is essentially constant throughout the aqueous phase.

(4) The nitric acid and nitronium ion (NO_2^+) concentrations are constant

throughout the aqueous phase.

(5) There is a simple distribution relationship between the concentration of toluene in the organic phase (assumed uniform) and the interfacial concentration of toluene in the aqueous phase, i.e.

$$C_{T_0} = \Phi C_{T_A}^{i}$$

By material balance over a film of thickness dx in the reaction zone:

$$D\frac{\partial^2 C}{\partial x^2} = r(C)$$

The following boundary conditions are applicable:

At the interface:

$$C = C_{T_A}^{i} \text{ when} x = 0$$

In the bulk aqueous phase:

$$\frac{dC}{dx} = 0 \text{ when} C = C_{T_A}^{i} = 0$$

Solution of the differential equation gives the rate of reaction per unit volume of reacting mixture:

$$R = A\sqrt{2D\int_0^{C_{T_A}^{i}} r(C)dC}$$

The rate of reaction may be represented as:

$$r(C) = kC_{T_A}C_{N_A}$$

Substitution gives:

$$R = A\sqrt{DkC_{N_A}C_{T_A}}$$

$$R = \frac{A}{\Phi}\sqrt{DkC_{N_A}C_{T_0}}$$

In conclusion it is clear that whether the two phase nitration of toluene is kinetically or mass transfer control depends on the relative values of kC_{N_A} and the mass transfer coefficient. If the value of kC_{N_A} is very much smaller than the mass transfer coefficient, the rate will be kinetically controlled, whereas if it is very much greater than the mass transfer coefficient, the rate will be mass transfer controlled.

In other word most workers are agreed that in the two phase system the toluene nitration occurs almost exclusively in the acid phase. Batch reactor studies show that at low acid strengths the rate is kinetically controlled and given by:

$$R_a = kC_{N_A}C_{T_A}$$

where k is the rate constant for homogeneous nitration in the acid phase, and C_{N_A} and C_{T_A} are the acid phase concentrations of nitric acid and toluene respectively. Over a short range of acid strength a slow reaction diffusional regime in which the rate is given by:

$$R_a = a' k_m C_{T_A}$$

where a' is the interfacial area per unit volume of the acid phase and k_m is the overall mass transfer coefficient. Finally at high acid strengths a fast reaction regime is observed where the rate is given by:

$$R_a = a' \sqrt{D k_m C_{N_A} C_{T_A}}$$

where D is the diffusivity of toluene in the acid phase.

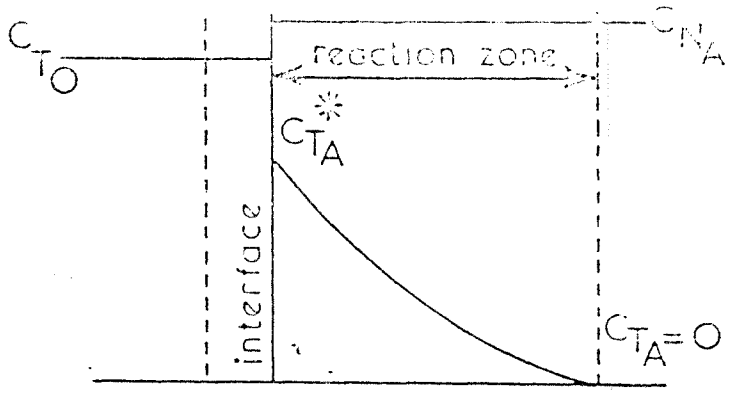

Figure: 2

Interfacial concentration profile assumed

in model

CHAPTER V

MONONITROTOLUENE ISOMER PRODUCTION

It is well known that nitration of aromatic molecules produces a number of isomeric products. According to literature data the mononitration of toluene can yield products in which the ratio of the orthonitrotoluene to paranitrotoluene varies between 1.41 and 1.62 depending on the nitrating agent and the reaction conditions used.

These mononitrations are carried out industrially on a considerable scale to produce compounds which are required as intermediates for the manufactures of dyestuffs, pharmaceuticals, etc. In most cases the para isomer is required while the ortho and meta products are waste products. Thus, with increasing cost of raw materials and the environmental problems in disposing of waste isomers, any increase in the percentage of the desired isomers represents a considerable process improvement.

A wide range of nitrating agents have been used in industrial scale nitration. The a nitrating agent should be available in sufficient quantity, reasonably cheap and relatively stable. A mixture of nitric acid and sulphuric acid, called mixed acid, is normally used as the nitrating agent for large scale industrial nitrations. The mixture has been used under varying conditions for industrial nitrations for many years. Information on the relationship between the composition of the nitrating mixture and the composition of the reaction product is, however, scarce and tend to be unsystematic.

There are three variables in the composition of the sulfuric/nitric acid nitrating mixture:

(i) Nitric acid content,

(ii) Water content,

(iii) Sulfuric acid concentration.

A few preliminary experiments done by different workers at different times indicate that the concentration of nitric acid did not effect the composition of the reaction product, as long as there was sufficient of nitric acid to mononitrate all of the toluene present. All further experiments were carried out with a slight excess of nitric acid, i.e. 1.04 moles nitric acid/mole of toluene. A series of experiments was then carried out to investigate the effect of water and sulfuric acid concentrations in the nitrating mixture on the composition of the reaction products. The results of these experiments are illustrated by Figures 3-7.

Figure 3 shows a plot of a reaction product composition vs. moles H_2SO_4 for a nitrating mixture which contains too little water for satisfactory mononitration. It is observed in the figure that an excessive amount of dinitration products are produced even at relatively low sulfuric acid concentrations, and consequently considerable quantities of toluene remain unreacted.

Figure 4 show a more satisfactory situation where there is more water (1.78 moles) in the nitrating mixture. The reaction products contain only traces of unreacted toluene and dinitration.

Figure 5 shows the effect of variation in water content in nitrating mixture containing 1.17 moles sulfuric acid -the optimum H_2SO_4 concentration in figure 3. In all of these experiments the amount of ortho- and paranitrotoluenes,

unconverted toluene and dinitration products varied with water or sulfuric acid concentration. Apart from extreme conditions, (Figure 3), the amount of metanitrotoluene remained constant at about 4%.

Figures 6 and 7 show the effect of variation in sulfuric acid and water content of the nitrating mixture on the ortho/para isomer ratio of mononitrotoluenes in the reaction products. The o/p isomer ratio decreases with increase in sulfuric acid content of the mixture and increases with the increase in water concentration. Examination of Figure 3 shows that some of the decrease in o/p ratio with increase in sulfuric acid concentration is due to the fact that the o-nitrotoluene dinitrates more readily than the para isomer. Thus the increase in the amount of para isomer, which can be produced by varying the composition of nitrating mixture, is limited if excessive amount of dinitration products are to be avoided.

The optimum conditions for mononitration are shown in Figure 3 - a nitrating mixture containing 1.17 moles sulfuric acid and 1.78 moles water giving a reaction product containing 36.5% p-nitrotoluene, 59.5% o-nitrotoluene and 4% m-nitrotoluene. o/p = 1.63.

1.04 moles HNO_3
0.63 moles H_2O

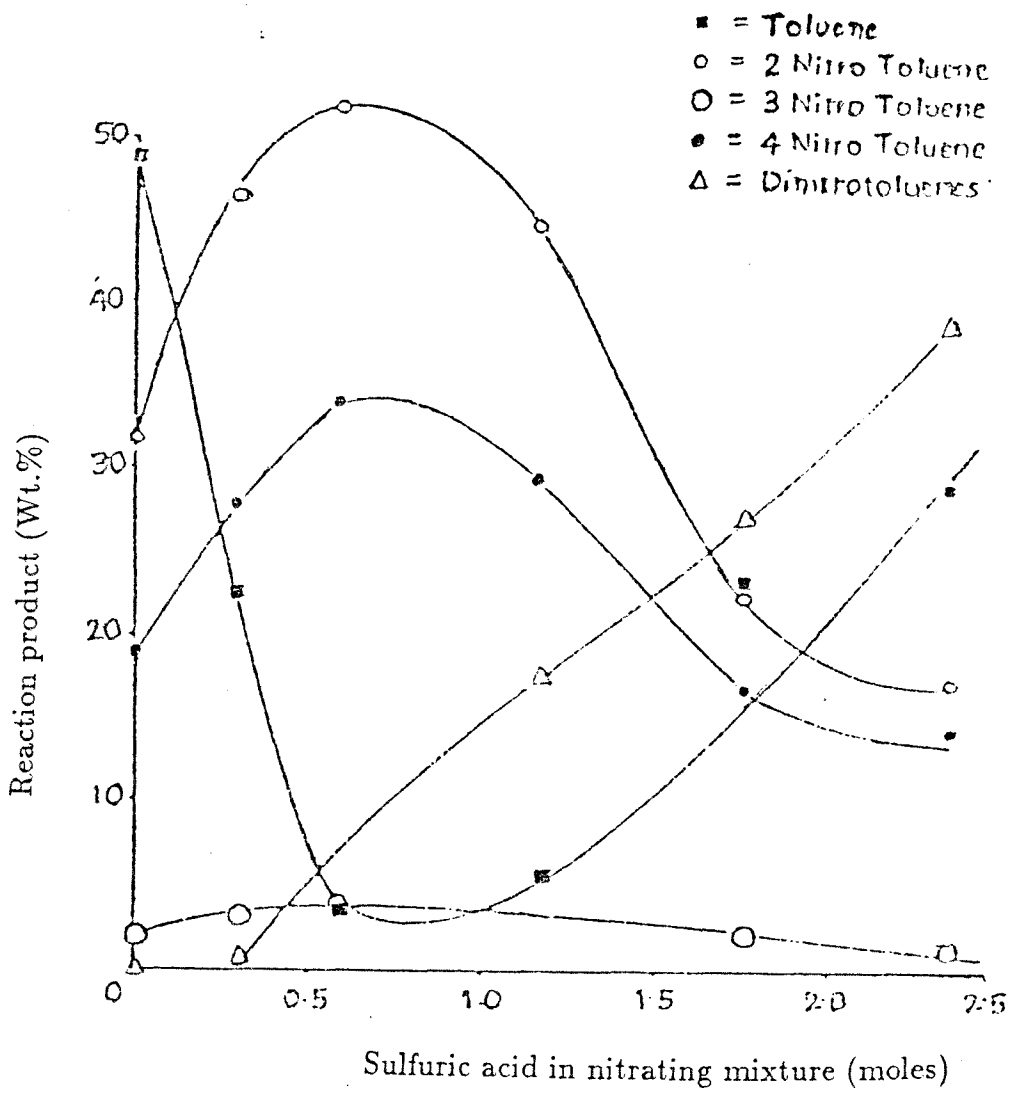

Figure: 3.

Reaction product composition vs. Sulfuric acid content

of nitrating mixture

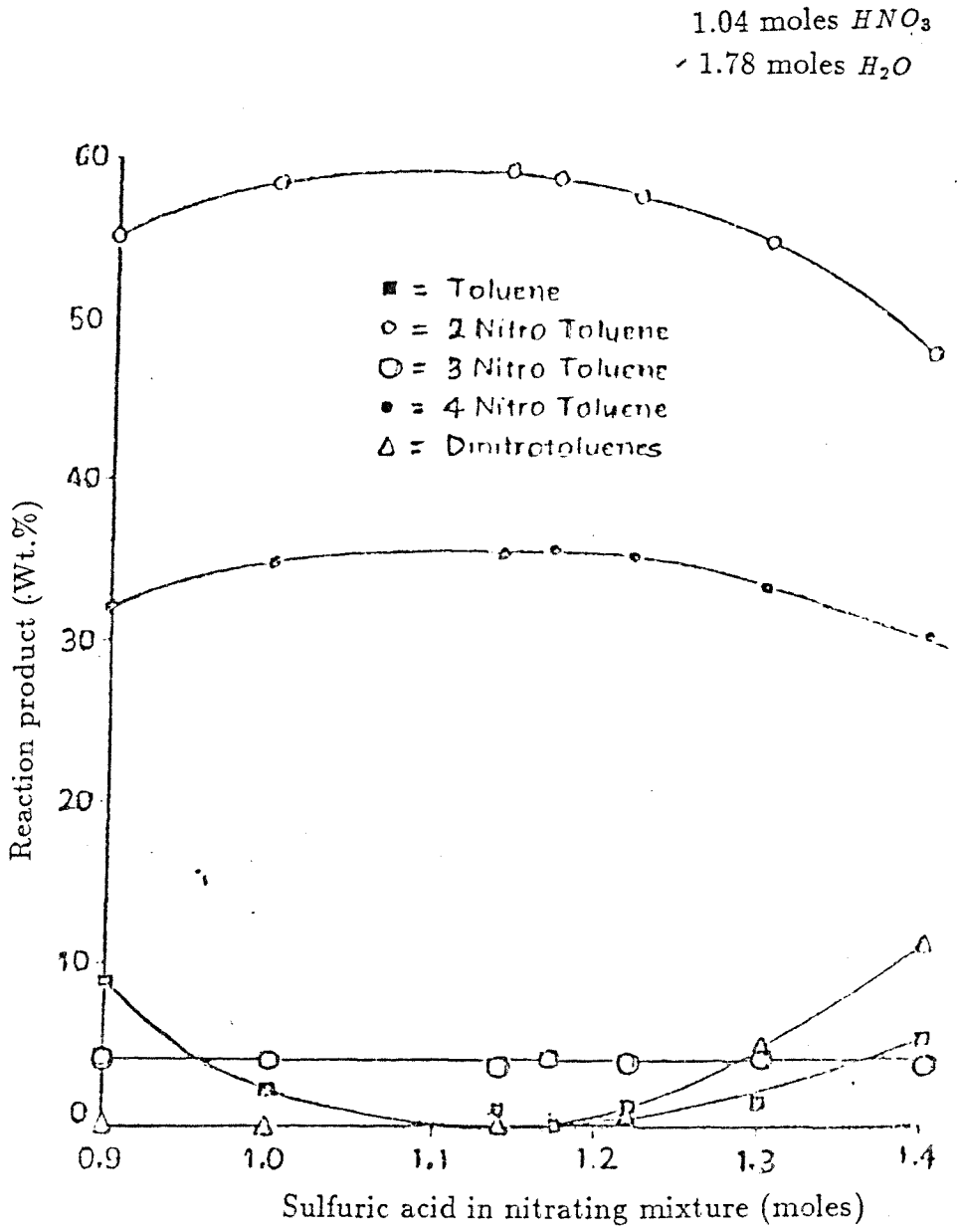

Figure: 4

Reaction product composition vs. Sulfuric acid content of nitrating mixture

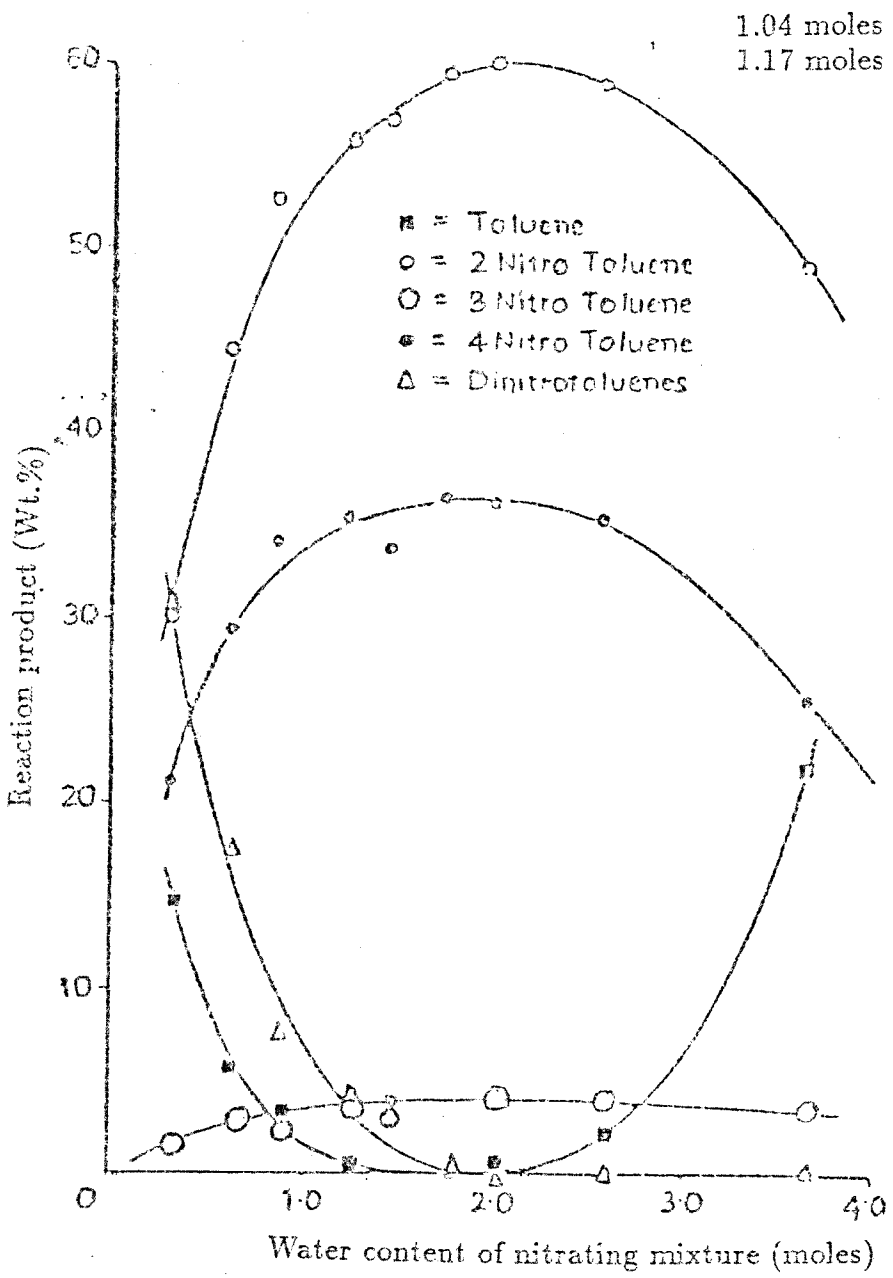

Figure: 5

Reaction product composition vs. Water content

of nitrating mixture

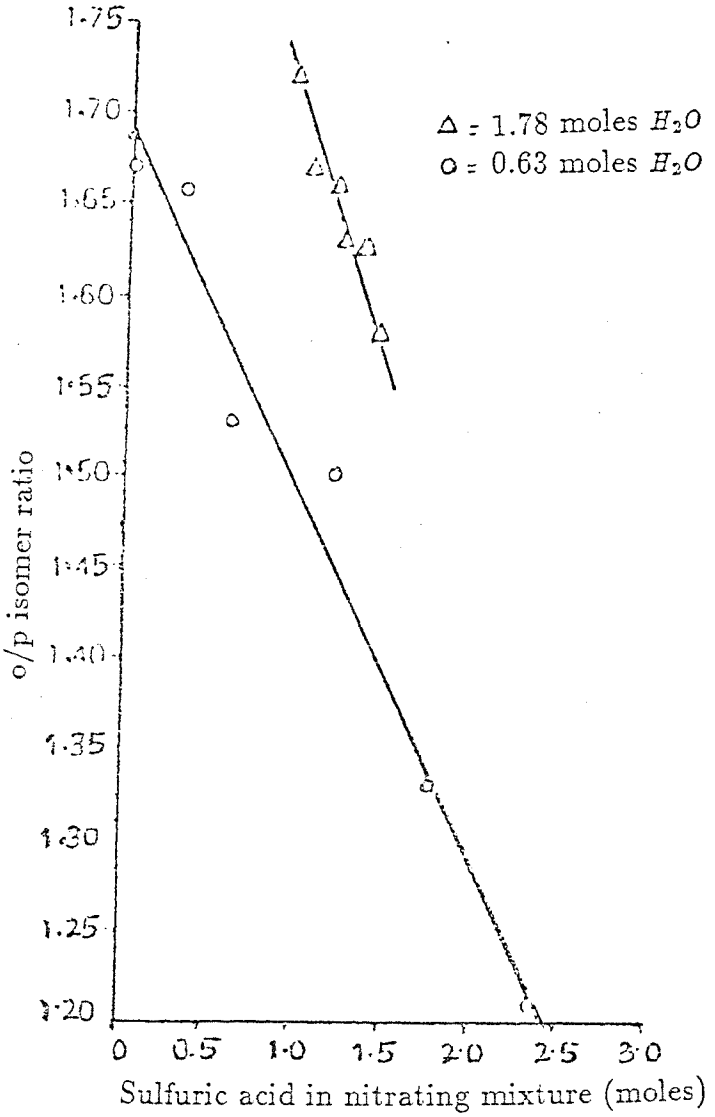

Figure: 6

o/p isomer ratio vs. sulfuric acid content

of nitrating mixture

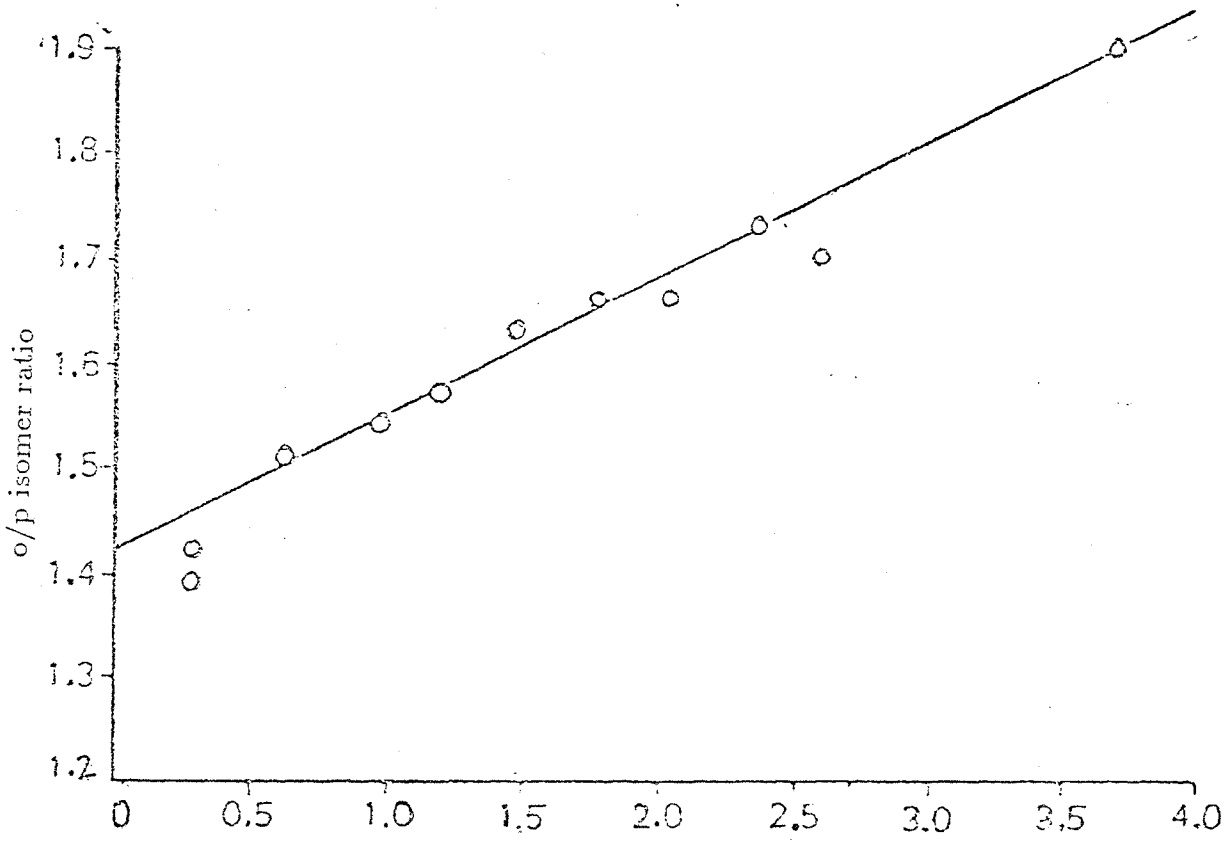

Figure: 7

o/p isomer ratio vs. Water content of nitrating mixture

(H2SO4 content 1.17 moles)

CHAPTER VI

FACTORS EFFECTING THE RATE OF REACTION

Effect of agitation speed

The results of the study of the reaction rate as a function of stirrer speed, with temperature as a parameter, are shown in Figure 8. Although this work greatly extends previously published data with respect to the range of agitation speed and temperature studied, the results follows the same pattern as those reported by McKinley and White[15], Biggs and White[16], and Barduhn and Kobe[11]. These authors claim that the lower part of the curve represents a mass transfer controlled region. After a transition period, a region is found where rate appears relatively insensitive to agitation speed. In this latter region it has been claimed previously that mass transfer resistances have been eliminated. The implication of this hypothesis is that the organic and acid solutions are mutually saturated and that, under steady state conditions, the rate of mass transfer is equal to the rate of reaction. If this is so, the reaction rate would be independent of agitation speed and would give a curve of zero slope. The present results extend well past the transitiom range and yet they still show a small but definite positive gradient. This naturally promps a deeper study of the inherent assumptions.

Effect of temperature

The other parameter to effect the rate is temperature. Previous measurement of the effect of temperature on the two phase nitration of toluene has

been variously interpreted. It is claimed that the rate of mononitration increases almost double with 10 °C increase in temperature up to some extent and beyond that limit where the dinitration products being are formed. There has been general agreement that where the rate is mass transfer controlled, a low temperature coefficient is to be expected. But, there has been dissgreement on the magnitude of the experimental temperature coefficient and on whether it is to be regarded as high or low. Such a variation would be consistent with the fact that any mass transfer resistance will tend to become more promient as the temperature is raised.

Since the temperature effect on mass transfer coefficient is very much less than that of chemical reactions, the validity must be questioned of using an overall temperature coefficient as a criterion of mass transfer control in such a system. Not only will temperature influence these resistances, it will also affect the distribution coefficients of the reactants between the phases, and hence the concentration of each phases, and also the concentration of the nitrating species. There are insufficient data available on the aqueous nitric acid-sulfuric acid system to allow calculations of nitronium ion concentration.

It can be readily seen that the effect of temperature on the nitration of toluene is very complex and cannot be used as evidence of kinetic control or mass transfer control of the reaction.

Effect of space time

Previous workers used space time as a test for a simple kinetic model, by operating it with a fixed feed rate, fixed agitation rate and at a constant temper-

ature. The results were analyzed on the assumption that the following kinetic rate equation applies:

$$R = k' C_{T_A} C_{T_B}$$

The constant k' includes the distribution coefficient for toluene between the two phases, the assumption being that reaction takes place homogeneously throughout the aqueous phase or atleast at the interface. The value of k' was found to vary with space time and this was taken as evidence for mass transfer control.

Previously, it was assumed that all the nitric acid would be present in the aqueous phase. As was shown earlier, a little but considerable quantities of nitric acid were found in the organic phase. This extraction of nitric acid into the organic phase effectively reduces the amount available for reaction and so decreases the rate of reaction. This was observed by McKinley and White[15] and by Barduhn and Kobe[11]. Thus it can only be concluded that the above equation is not the true rate equation, or that the measured rate includes a contribution from mass transfer.

Effect of phase inversion

A convenient criterion for the existance of a mass transfer resistance which has not been widely used in the past is the effect of phase inversion. This leads to change in both interfacial area per unit volume and mass transfer coefficients. The difference in the degree of conversion provides conclusive proof for the presence of a mass transfer resistance and a need for a rate model which takes

this in to account. It is also abundantly clear that the failure to eliminate the mass transfer resistance in the early work was not a function of scale and that the effect is likely to be of importance in industrial reactors, certainly with nitrating acid containing 30 mole % H_2SO_4.

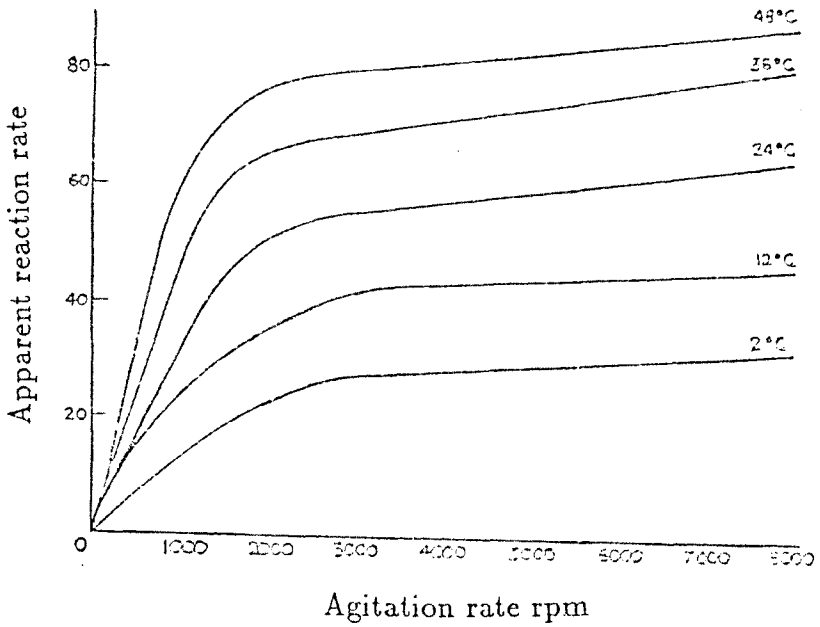

Figure: 8

Apparent reaction rate as a function of agitation

and temperature

CHAPTER VII

MONONITRATION PRODUCTS

o-Nitrotoluene is used in the synthesis of intermediates for azo dyes, sulfur dyes, rubber chemicals and agricultural chemicals. Typical intermediates are o-toluidine, o-nitrobenzaldehyde, 2-nitro-4-chlorotoluene, 2-nitro-6-chlo rotoluene, 2-amino-4-chlorotoluene, and 2-amini-6-chlorotoluene. p-Nitrotoluene is used principally in the production of intermediates for azo dyes and sulfur dyes. Typical intermediates are p-toluidine, p-nitrobenzaldehyde, and 4-nitro-2-chlorotoluene. m-Nitrotoluene has no known major uses, hence it is an undesirable byproduct, and should be avoided their selective operation of the reaction step in the mononitrotoluene synthesis.

Physical properties

o-Nitrotoluene is a clear yellow liquid. It is infinitely soluble in benzene, diethyl ether and ethanol. It is soluble in most organic solvent and only a slightly soluble in water. The electron-acceptor action of the nitro group in o-nitrotoluene confers increased reactivity on the methyl group, thus the methyl group is easily oxidized. The physical properties of o-nitrotoluene are listed in Table 1.

m-Nitrotoluene is a clear yellow liquid. It is infinitely soluble in ethanol, benzene and diethyl ether. It is soluble in most organic solvent and only sparingly soluble in water. m-Nitrotoluene does not have an active methyl group as do the ortho and para isomers. The physical properties of m-nitrotoluene are listed in Table 1.

p-Nitrotoluene crystallized in colorless rhombic crystals. It is only slightly soluble in water, moderately soluble in methanol and ethanol. It is readily soluble in acetone, benzene and diethyl ether. The physical properties of o-nitrotoluene are listed in Table 1.

Table: 1

The phisical properties of Mononitrotoluenes (MNT)

Properties	o-NT	m-NT	p-NT
B.P. °C	222	233	238
M.P. °C	9.5	16	52
Density gm/cc	1.163	1.157	1.104
Viscosity cP (15 °C)	0.0262	0.0178	0.0120
H_v KJ/Kg	344	364	366

Analytical and test methods

o-Nitrotoluene can be analzsed for purity and isomer content by IR spectroscopy with an accuracy of 1%. p-Nitrotoluene content can be estimated by the decomposition of the isometric toluene diazonium chlorides since the ortho and meta isomers decompose more rapidly than the para isomer. A colorimetric method for determining the contents of the various isomers is based upon the color which forms when the mononitrotoluenes are dissolved in sulfuric acid. From the absorption of the sulfuric acid solution at 436 and 305 nm, the ortho and para isomer content can be determined, and the meta isomer can be obtained by difference. However, this and other colorimetric methods are subject to possible interferences from other aromatic nitro compounds.

A titrametric method, which is based on the reduction of the nitro groups with titanium(III) sulfate or chloride, can be used to determine mononitrotoluenes. Chromatographic methods, eg. gas chromatography or higher pressure liquid chromatography, are well-suited for the determination of mononitrotoluenes as well as its individual isomers. Freezing points are used commonly as indicators of purity of the various isomers.

Health and safety consideration

The toxic effects of mononitrotoluenes as well as its individual isomers are similar to but less pronounced than those of nitrobenzene. The maximum allowable concentration for mononitrotoluenes is 5 ppm (30 mg/m^3). Mononitrotoluenes may be absorbed through the skin and respiratory tract. The mononitrotuluenes represent moderate fire hazards when expose to heat or flame. Impervious protective clothings should be worn in the area where risk of splash exists. Ordinary work cloths should be removed immediately, and the skin should be washed thoroughly with soap and warm water. In case of high vapor concentration, full facemasks with organic-vapor canisters or air supplied respirators should be used. Clean work clothing should be worn daily, and showering after each shift should me made mandatory.

CHAPTER VIII

PROGRAM DISCRIPTION

The objective of the computer program is to simulate a rigorous multicomponent batch distillation column. The computer model was originally developed by R. Springer for petroleum crude analysis. It has been being modified and made usable for nitrotoluene system. The source code is atteched herewith shown in Appendix A.

The input data for the program was calculated by various approximation methods since there is limited data available in the literature for the nitrotoluenes. A plant material balance is done according to the 1000 lbs of MNT production per batch. In the calculation of vapor enthalpy of dinitrotoluene, the enthalpy of vaporization was calculated by Carruth and Kobayashi equation. The vapor pressure and the liquid molar volume for all nitrotoluenes are calculated by the Lee-Kesler method.

As discussed earlier, we are using two batch distillation columns for the whole separation process. The first column is used to remove the unreacted toluene and the trace amount of water. The second column is used for the recovery of the isomers formed in the mononitration of toluene with the dinitration products left in the still pot. The computer model ras used to evaluate both separations. The input and output for both the cases are shown in the Appendix B and C respectively. Summarized in the following table is the results of the simulated batch separation.

Column 1

	Feed charged in lbmoles	Feed remaining in still, lbmole
Water	0.002	0.0
Toluene	1.316	0.8E-10
o-NT	9.486	0.3785
m-NT	0.643	0.1327
p-NT	5.865	1.930
DNT	0.002	0.002

Column 2

	Feed charged in lbmoles	Feed remaining in still, lbmole
o-NT	9.486	0.147E
m-NT	0.643	0.250E
p-NT	5.865	0.9362
DNT	0.002	0.002

CHAPTER IX

RECOMMENDATIONS

A subsidiary objective of the work described was to obtain an insight into scale-up criteria for a batch nitrator operating under industrial conditions. The rate controlling mechanism is still not completely known. Previous studies of the reaction mechanism were done by avoiding the mass transfer effect. Presently in the literature is evidence to prove that the mass transfer effect can not be completely neglected. Thus, it is very important to study the rate of diffusion taking place during the reaction. A knowledge of diffusional resistances associated with the toluene nitration will help in an understanding the of mechanism.

Other important criteria of the process is control of isomers distribution. One of the most desirable isomer is para MNT which has a wide use in the dyes industry. Some measure of control over the isomer distribution resulting from the mixed acid nitration of toluene is possible but only under severe constraints and far from the conditions normally employed for efficient mononitration. It is very important to have a nitrating agent which can maximize the desired product and can improve our control on the isomer distribution of MNT. The effects that are observable appear to be best explained by mass transfer influences. Although alternative methods of nitration do offer greater latitude in the determination of the isometric content, none can be accomplished with the convenience, speed, and low cost of the mixed acid, nitrating agent.

In summary, it is highly recommended to extend the study of this process to combine the reaction, crude MNT wash, and the separation sections of the

total process, and evaluate the process conditions of reaction to maximize the ortho-para isomers of MNT.

APPENDICES

(A) Source Program.

(B) Input and Output for the first column.

(C) Input and Output for the second column.

Appendix: A

Source Program

```
      BATCH DISTILLATION COMPUTER PROGRAM
C
C     AUTHOR - RONALD D. SPRINGER  (MAY,1985)
C     MASTERS THESIS AT NEW JERSEY INSTITUTE OF TECHNOLOGY
C
C     REVISED BY: NITIN K. SHAH
C                 NJIT, NEWARK, NJ
C                 21 SEPTEMBER, 1989
C
C     THIS VERSION IS IN DOUBLE PRECISION
C
C     INTERNAL SUBROUTINES:
C     MAIN   - CONTROLS ALL CALCULATIONS
C     CPU    - CALCULATES AND PRINTS CPU TIME (OPTIONAL)
C     INPUT  - READS INPUT
C     CDATA  - CURVE FITS THERMODYNAMIC DATA
C     REGRSS - PERFORMS LINEAR REGRESSION FOR THERMODYNAMIC COEFFICIENTS
C     CNSTNT - CALCULATES CONSTANTS
C     CONVRT - CALCULATES MOLE, WEIGHT, AND LIQUID VOLUME
C     STEADY - MAIN ROUTINE FOR STEADY-STATE TOTAL-REFLUX CALCULATIONS
C     INITAL - INITIALIZES STEADY-STATE TOTAL-REFLUX CALCULATION
C     BUBBLE - CONVERGES BUBBLE POINT TEMPERATURE
C     MATSOL - SOLVES STEADY-STATE TOTAL-REFLUX MATRIX
C     MATINV - INVERTS MATRIX USING JERSYLIB ROUTINE
C     SETUP  - SETS-UP STEADY-STATE TOTAL-REFLUX MATRIX
C     THERMO - CALCULATES THERMODYNAMIC DATA
C     CONVRG - CHECKS CONVERGENCE AND DETERMINES NEW VALUES FOR STDY-STE
C     HTBAL  - CALCULATES FLOWS FOR STEADY-STATE TOTAL-REFLUX
C     SPRINT - PRINTS STEADY-STATE TOTAL-REFLUX RESULTS
C     UNSTDY - MAIN ROUTINE FOR UNSTEADY-STATE BATCH DISTILLATION CALCS
C     MAXBON - CALCULATES NORMAL BOILING POINT USING MAXWELL-BONNELL
C     FUNCT  - CALCULATES DX/DT FOR LSODE
C     JACOB  - CALCULATES JACOBIAN FOR LSODE
C     UPRINT - PRINTS UNSTEADY-STATE BATCH DISTILLATION RESULTS
C     LSODE  - LIVERMORE SOLVER FOR ORDINARY DIFFERENTIAL EQUATIONS
C
C     EXTERNAL SUBROUTINES:
C     KSIMEQ - JERSYLIB ROUTINE FOR INVERTING A MATRIX
C     READ,WRITE - READS DATA FROM AND WRITES DATA TO FILES
C     WIPE   - SETS VARIABLES TO ZERO
C     SETCPU,GETCPU - USED FOR CPU TIME (OPTIONAL)
C     EXIT   - EXITS PROGRAM (OPTIONAL)
C
C     COMMON BLOCKS
C
C     FILE:
C     IIN    - READ FILE
C     IOUT   - WRITE FILE
C
C     PROP:
C     CNBP(50) - COMPONENT NORMAL BOILING POINT, DEGF
C     CMW(50) - COMPONENT MOLECULAR WEIGHT, LB/LBMOLE
C     CDEN(50) - COMPONENT LIQUID DENSITY AT 60 DEGF, LB/GAL
C     TF(8)  - TEMPERATURE DATA POINTS, DEGF
C     CVP(50,8) - COMPONENT VAPOR PRESSURE DATA, MMHG
C     CHV(50,8) - COMPONENT VAPOR ENTHALPY DATA, KBTU/LBMOLE
C     CHL(50,8) - COMPONENT LIQUID ENTHALPY DATA, KBTU/LBMOLE
C     CVM(50,8) - COMPONENT LIQUID MOLAR VOLUME DATA, GAL/LBMOLE
C     COEFVP(50,4) - COMPONENT VAPOR PRESSURE COEFFICIENTS
C                  - LN(VP) = A + B/T + C/T**2 + D/T**3, T=DEGR
```

```
C     COEFHV(50,4) - COMPONENT VAPOR ENTHALPY COEFFICIENTS
C                  - HV = A + B*T + C*T**2 + D*T**3, T=DEGR
C     COEFHL(50,4) - COMPONENT LIQUID ENTHALPY COEFFICIENTS
C                  - HL = A + B*T + C*T**2 + D*T**3, T=DEGR
C     COEFVM(50,4) - COMPONENT LIQUID MOLAR VOLUME COEFFICIENTS
C                  - VM = A + B*T + C*T**2 + D*T**3, T=DEGR
C
C     DATAIN:
C     FR      - FEED CHARGE READ
C     ZR(50)  - FEED COMPOSITION READ
C     EMR     - MURPHREE EFFICIENCY READ
C     PCOND   - CONDENSER PRESSURE, MMHG
C     DPCOND  - CONDENSER PRESSURE DROP, MMHG
C     DPTRAY  - STAGE PRESSURE DROP, MMHG
C     GCOND   - CONDENSER VOLUME HOLDUP, GALLON
C     GTRAY   - STAGE VOLUME HOLDUP, GALLON
C     QTRAY   - STAGE HEAT DUTY, KBTU/HR
C     TTIMES  - TOTAL TIME SAVE, HOURS
C     IFR     - FEED UNITS READ, 1=MOLE, 2=WEIGHT, 3=VOLUME
C
C     DATA:
C     F       - FEED CHARGE, LBMOLE
C     Z(50)   - FEED COMPOSITION, MOLE FRACTION
C     P(40)   - STAGE PRESSURE, MMHG
C     G(40)   - STAGE VOLUME HOLDUP, GALLON
C     Q(40)   - STAGE HEAT DUTY, KBTU/HR
C     EM(40,50) - MURPHREE EFFICIENCY
C     FC      - CONVRT VARIABLE, TOTAL
C     ZC(50)  - CONVRT VARIABLE, COMPOSITION
C     CF(6)   - MOLE, WEIGHT, VOLUME TOTALS AND PERCENTS
C     CZ(50,6) - MOLE, WEIGHT, VOLUME COMPOSITIONS AND PERCENTS
C     RR      - MOLAR REFLUX RATIO
C     DR      - LIQUID DISTILLATE RATE, LBMOLE/HR
C     FCUT    - FEED CUT CHARGE, LBMOLE
C     ZCUT(50) - FEED CUT COMPOSITION, MOLE FRACTION
C     TCUT(15) - TEMPERATURE CUT POINTS, DEGF
C     DMAX    - MAXIMUM PERCENT CHANGE (50.0)
C     TOL     - CONVERGENCE PERCENT TOLERANCE (0.001)
C     NAME(50,3) - COMPONENT NAME
C     MAXIT   - MAXIMUM ITERATIONS (15)
C     N       - NUMBER OF COMPONENTS, 2 TO 50
C     M       - NUMBER OF STAGES INCLUDING CONDENSER AND REBOILER,2 TO 40
C     M1      - M - 1
C     IC      - CONVRT VARIABLE, UNITS
C     INITL   - NUMBER OF INITIALIZATION IMPROVEMENTS
C     IPRNT   - PRINT LEVEL, 0=MINIMUM, 1=CUT COMPOSITIONS, 2=STAGE
C             - COMPOSITIONS, 3=INTERMEDIATE, 4=DEBUG
C
C     REGR:
C     X(8)    - REGRESSION INDEPENDENT VARIABLE
C     Y(8)    - REGRESSION DEPENDENT VARIABLE
C     C(4,8)  - REGRESSION MATRIX
C     D(4)    - REGRESSION RIGHT-HAND SIDE
C     COEF(4) - REGRESSION COEFFICIENTS
C
C     SCALC:
C     T(40)   - STAGE TEMPERATURE, DEGF
C     AL(40)  - STAGE LIQUID RATE, LBMOLE/HR
C     V(40)   - STAGE VAPOR RATE, LBMOLE/HR
C     U(40)   - STAGE MOLAR HOLDUP, LBMOLE
```

```
C     X(40,50) - STAGE LIQUID COMPOSITION, MOLE FRACTION
C     Y(40,50) - STAGE VAPOR COMPOSITION, MOLE FRACTION
C     DT(40) - STAGE DELTA T
C     DL(40) - HTBAL VARIABLE
C     DV(40) - HTBAL VARIABLE
C     DU(40) - STAGE DELTA U
C     DX(40,50) - STAGE DELTA X
C     SUMDX(40) - STAGE SUM DELTA X
C     AK(50) - COMPONENT EQUILIBRIUM CONSTANT, VP/P
C     HV(50) - COMPONENT VAPOR ENTHALPY, KBTU/LBMOLE
C     HL(50) - COMPONENT LIQUID ENTHALPY, KBTU/LBMOLE
C     VM(50) - COMPONENT LIQUID MOLAR VOLUME, GAL/LBMOLE
C     DKDT(50) - AK TEMPERATURE DERIVATIVE
C     DHVDT(50) - HV TEMPERATURE DERIVATIVE
C     DHLDT(50) - HL TEMPERATURE DERIVATIVE
C     DVMDT(50) - VM TEMPERATURE DERIVATIVE
C     SCALE  - CONVERGENCE SCALING FACTOR
C     CONV   - CONVERGENCE INDICATOR FACTOR
C     ICONV  - CONVERGENCE INDICATOR, 0=CONVERGED
C     ITER   - ITERATION NUMBER
C
C     MATRIX:
C     A(52,52) - A MATRIX
C     B(52,52) - B AND B INVERSE MATRIX
C     C(52,52) - C MATRIX
C     D(52)  - D MATRIX
C     DM(52) - D MATRIX FOR M STAGE
C     AHOLD(52,52) - A MATRIX SAVE
C     CHOLD(52,52) - C MATRIX SAVE
C     DHOLD(52) - D MATRIX SAVE
C     ILAST  - FILE MARKER
C     I1     - MATRIX CONSTANT
C     I2     - MATRIX CONSTANT
C
C     CLSODE:
C     XT(2000) - LSODE DEPENDENT VARIABLE, X
C     RWORK(26022) - LSODE REAL WORK ARRAY
C     IWORK(2020) - LSODE INTEGER WORK ARRAY
C
C     UCALC:
C     SAK(40) - STAGE SUM AK*X
C     SHV(40) - STAGE SUM HV*Y
C     SHL(40) - STAGE SUM HL*X
C     SVM(40) - STAGE SUM VM*X
C     SDKDT(40) - STAGE SUM DKDT*X
C     SDHLDT(40) - STAGE SUM DHLDT*X
C     SDVMDT(40) - STAGE SUM DVMDT*X
C     SAKY1(40) - STAGE SUM AK*Y(J+1)
C     SAKX1(40) - STAGE SUM AK*X(J-1)
C     SAKY(40) - STAGE SUM AK*Y
C     SHLY1(40) - STAGE SUM HL*Y(J+1)
C     SHLX1(40) - STAGE SUM HL*X(J-1)
C     SHLY(40) - STAGE SUM HL*Y
C     SVMY1(40) - STAGE SUM VM*Y(J+1)
C     SVMX1(40) - STAGE SUM VM*X(J-1)
C     SVMY(40) - STAGE SUM VM*Y
C     AKS(40,50) - AK SAVE
C     DTDT(40) - STAGE TEMPERATURE TIME DERIVATIVE
C     DUDT(40) - STAGE HOLDUP TIME DERIVATIVE
C     DHDT(40) - STAGE LIQUID ENTHALPY TIME DERIVATIVE
```

```
C
C     SUBROUTINE VARIABLES:
C     I       - COMPONENT INDICATOR
C     J       - STAGE INDICATOR
C     K       - LOOP VARIABLE
C     L       - ARGUMENT VARIABLE
C     ZZ(20)  - INPUT REPRINT
C     NOPER   - OPERATING STEP NUMBER
C     NT      - NUMBER CPU CALLS
C     NDUM    - CPU DUMMY VARIABLE
C     TIME    - TOTAL CPU TIME
C     DTIME   - DELTA CPU TIME
C     STIME   - START CPU TIME
C     II      - INPUT COMPONENT NUMBER
C     DEV     - CDATA PERCENT DEVIATION
C     SUM     - SUMMATION VARIABLE
C     JJ      - STAGE INDICATOR
C     UU      - INITAL HOLDUP VARIABLE
C     DTI     - DELTA BUBBLE POINT TEMPERATURE
C     TOLB    - BUBBLE POINT CONVERGENCE TOLERANCE
C     ERROR   - CONVERGENCE ERROR
C     SUMKX   - BUBBLE POINT SUM AK*X
C     SUMDKX  - BUBBLE POINT SUM DKDT*X
C     K1      - MATSOL LOOP VARIABLE
C     K2      - MATSOL LOOP VARIABLE
C     K3      - MATSOL LOOP VARIABLE
C     IFILE   - READ AND WRITE FILE NUMBER
C     BB      - KSIMEQ MATRIX
C     IN      - KSIMEQ MATRIX DIMENSION
C     IER     - KSIMEQ SINGULAR MATRIX INDICATOR
C     TR1     - THERMO TEMPERATURE, DEGR
C     TR2     - TR1**2
C     TR3     - TR1**3
C     TR4     - TR1**4
C     DELX    - DELTA X
C     XNEW    - NEW X
C     XSUM    - SUM X
C     XTOL    - X CONVERGENCE TOLERANCE
C     YSUM    - SUM Y
C     MF      - LSODE METHOD FLAG
C     LIW     - LSODE IWORK LENGTH
C     LRW     - LSODE RWORK LENGTH
C     NEQ     - LSODE NUMBER OF ORDINARY DIFFERENTIAL EQUATIONS
C     ATOL    - LSODE ABSOLUTE ERROR TOLERANCE
C     IOPT    - LSODE OPTIONAL INPUT FLAG
C     ITOL    - LSODE ERROR INDICATOR
C     RTOL    - LSODE RELATIVE ERROR TOLERANCE
C     TIME    - LSODE INDEPENDENT VARIABLE, TIME, HOURS
C     TOUT    - LSODE FIRST TIME
C     ITASK   - LSODE TASK INDEX
C     ISTATE  - LSODE STATE INDEX
C     ITC     - CUT NUMBER
C     TTIME   - TOTAL TIME, HOURS
C     TIMCUT  - CUT TIME, HOURS
C     SG      - SPECIFIC GRAVITY OF DISTILLATE
C     TF      - OVERHEAD TEMPERATURE, DEGF
C     PMMHG   - DISTILLATE PRESSURE, MMHG
C     TNBP    - OVERHEAD NORMAT BOILING POINT
C     TNBP12  - OVERHEAD NORMAL BOILING POINT (K=12)
C     A       - MAXBON VARIABLE
```

```
C      F       - MAXBON VARIABLE
C      TR      - MAXBON VARIABLE
C      WK      - MAXBON VARIABLE
C      PATM    - MAXBON VARIABLE
C      PLOG    - MAXBON VARIABLE
C      THIRD   - MAXBON VARIABLE
C      DXDT    - X COMPOSITION TIME DERIVATIVE
C      ML      - JACOBIAN HALF-BANDWIDTH
C      MU      - JACOBIAN HALF-BANDWIDTH
C      PD      - JACOBIAN MATRIX
C      NROWPD  - JACOBIAN MATRIX DIMENSION
C
C      MAIN ROUTINE CONTROLS ALL CALCULATIONS
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON /FILE/
      1 IIN,          IOUT
       REAL ZZ(20)
C
C      SET READ AND WRITE FILES
C
       IIN = 5
       IOUT = 6
C
C      PRINT INPUT
C
       REWIND IIN
     1 READ(IIN,2,END=4) ZZ
       WRITE(IOUT,3) ZZ
       GO TO 1
     2 FORMAT(20A4)
     3 FORMAT(1X,20A4)
     4 REWIND IIN
       CALL CPU
C
C      READ INPUT, CALCULATE THERMODYNAMIC DATA AND CONSTANTS
C
       NOPER = 0
    10 CALL INPUT ( NOPER, &99 )
       IF( NOPER .EQ. 0 ) CALL CDATA
       CALL CNSTNT(NOPER)
       CALL CPU
C
C      CALCULATE STEADY-STATE TOTAL REFLUX
C
       CALL STEADY(&99)
       CALL CPU
C
C      CALCULATE UNSTEADY-STATE BATCH DISTILLATION
C
       CALL UNSTDY(&99)
       CALL CPU
C
C      NEXT OPERATING STEP
C
       NOPER = NOPER + 1
       GO TO 10
C
C      TERMINATING
C
    99 WRITE(IOUT,900)
```

```
      CALL CONVRT(1)
      CALL CPU
      WRITE(IOUT,910)
      CALL EXIT
      STOP
C
  900 FORMAT(1H1/20X,'F E E D   C H A R G E   (REMAINING IN STILL)')
  910 FORMAT(/' BATCH TERMINATING')
      END
      SUBROUTINE CPU
C
C     CALCULATES AND PRINTS LAPSED CPU TIME IN SECONDS
C
      COMMON /FILE/  IIN,  IOUT
      DATA NT/-1/
C
C
      NT = NT + 1
      IF( NT .NE. 0 ) GO TO 10
C
C     FIRST CALL ... NT = 1
C
      STIME = SECNDS (0.0)
C
C     LAPSED CPU TIME ... NT > 1
C
   10 DTIME = SECNDS (STIME)
      WRITE (IOUT,900) DTIME, NT
      RETURN
C
  900 FORMAT(//' ( LAPSED CPU TIME = ',F8.2, ' SEC', 5X,
     1          '   ENTRY COUNT = ', I4,' )' )
C
      END
      SUBROUTINE INPUT(NOPER,*)
C     READS INPUT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /PROP/
     1 CNBP(50),     CMW(50),      CDEN(50),     TF(8),
     2 CVP(50,8),    CHV(50,8),    CHL(50,8),    CVM(50,8),
     3 COEFVP(50,4), COEFHV(50,4), COEFHL(50,4), COEFVM(50,4)
      COMMON /DATAIN/
     1 FR,         ZR(50),    EMR,        PCOND,    DPCOND,    DPTRAY,
     2 GCOND,      GTRAY,     QTRAY,      TTIMES,   IFR
      COMMON /DATA/
     1 F,          Z(50),     P(40),      G(40),    Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),      CZ(50,6), RR,        DR,
     3 FCUT,       ZCUT(50),  TCUT(15),   DMAX,     TOL,       NAME(50,3),
     4 MAXIT,      N,         M,          M1, IC,   INITL,     IPRNT
C
      IF( NOPER .NE. 0 ) GO TO 10
C
C     ZERO COMMON BLOCKS
C
      CALL WIPE ( CNBP(1), 2362 )
      CALL WIPE ( FR, 59 )
      IFR = 0
      CALL WIPE ( F, 2598 )
```

```
      CALL IWIPE ( NAME(1,1), 157)
C
C     READ INPUT
C
      READ(IIN,930) N
      READ(IIN,900) (II,(NAME(II,K),K=1,3),CNBP(II),CMW(II),CDEN(II),
     1            I=1,N)
      READ(IIN,910) (TF(K),K=1,8)
      READ(IIN,920) (II,(CVP(II,K),K=1,8),I=1,N)
      READ(IIN,920) (II,(CHV(II,K),K=1,8),I=1,N)
      READ(IIN,920) (II,(CHL(II,K),K=1,8),I=1,N)
      READ(IIN,920) (II,(CVM(II,K),K=1,8),I=1,N)
      READ(IIN,950) IFR, FR
      READ(IIN,950) (II,ZR(II),I=1,N)
   10 READ(IIN,930,END=99) M
      READ(IIN,940) GCOND, GTRAY, QTRAY, EMR
      READ(IIN,940) PCOND, DPCOND, DPTRAY, RR, DR
      DO 20 K=1,15
   20 TCUT(K)=0.0
      READ(IIN,960) (TCUT(K),K=1,5)
      READ(IIN,970) TOL, DMAX, MAXIT, INITL, IPRNT
      RETURN
C
   99 RETURN 1
C
  900 FORMAT(10X,I2,3A4,3G12.1)
  910 FORMAT(9X,8G9.1)
  920 FORMAT(7X,I2,8G9.1)
  930 FORMAT(12X,I6)
  940 FORMAT(12X,5G12.1)
  950 FORMAT(10X,I2,G12.1)
  960 FORMAT(12X,5G12.0)
  970 FORMAT(12X,2G12.1,3I6)
      END
      SUBROUTINE CDATA
C     CURVE FITS THERMODYNAMIC DATA
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /PROP/
     1 CNBP(50),     CMW(50),      CDEN(50),     TF(8),
     2 CVP(50,8),    CHV(50,8),    CHL(50,8),    CVM(50,8),
     3 COEFVP(50,4), COEFHV(50,4), COEFHL(50,4), COEFVM(50,4)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),    Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,        DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,     TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,   INITL,     IPRNT
      COMMON /REGR/
     1 X(8),       Y(8),      C(4,8),    D(4),     COEF(4)
C
C     PRINT COMPONENTS AND PROPERTIES
C
      WRITE(IOUT,900)
      WRITE(IOUT,905) (I,(NAME(I,K),K=1,3),CNBP(I),CMW(I),CDEN(I),I=1,N)
C
C     CURVE FIT VAPOR PRESSURE
C
      WRITE(IOUT,910)
      DO 5 K=1,8
```

```
      X(K) = 1.0D0 / ( TF(K) + 459.67D0 )
    5 CONTINUE
      CALL REGRSS(1)
      DO 40 I=1,N
      DO 10 K=1,8
      Y(K) = DLOG( CVP(I,K) )
   10 CONTINUE
      CALL REGRSS(2)
      DO 20 K=1,4
      COEFVP(I,K) = COEF(K)
   20 CONTINUE
      WRITE(IOUT,950) I, (NAME(I,K),K=1,3)
      DO 30 K=1,8
      Y(K) = DEXP( Y(K) )
      DEV = 100.0D0 * ( Y(K) - CVP(I,K) ) / CVP(I,K)
      WRITE(IOUT,960) TF(K), CVP(I,K), Y(K), DEV
   30 CONTINUE
   40 CONTINUE
C
C     CURVE FIT VAPOR ENTHALPY
C
      WRITE(IOUT,920)
      DO 45 K=1,8
      X(K) = TF(K) + 459.67D0
   45 CONTINUE
      CALL REGRSS(1)
      DO 80 I=1,N
      DO 50 K=1,8
      Y(K) = CHV(I,K)
   50 CONTINUE
      CALL REGRSS(2)
      DO 60 K=1,4
      COEFHV(I,K) = COEF(K)
   60 CONTINUE
      WRITE(IOUT,950) I, (NAME(I,K),K=1,3)
      DO 70 K=1,8
      DEV = 100.0D0 * ( Y(K) - CHV(I,K) ) / CHV(I,K)
      WRITE(IOUT,960) TF(K), CHV(I,K), Y(K), DEV
   70 CONTINUE
   80 CONTINUE
C
C     CURVE FIT LIQUID ENTHALPY
C
      WRITE(IOUT,930)
      DO 120 I=1,N
      DO 90 K=1,8
      Y(K) = CHL(I,K)
   90 CONTINUE
      CALL REGRSS(2)
      DO 100 K=1,4
      COEFHL(I,K) = COEF(K)
  100 CONTINUE
      WRITE(IOUT,950) I, (NAME(I,K),K=1,3)
      DO 110 K=1,8
      DEV = 100.0D0 * ( Y(K) - CHL(I,K) ) / CHL(I,K)
      WRITE(IOUT,960) TF(K), CHL(I,K), Y(K), DEV
  110 CONTINUE
  120 CONTINUE
C
C     CURVE FIT MOLAR VOLUME
```

```
C
      WRITE(IOUT,940)
      DO 160 I=1,N
      DO 130 K=1,8
      Y(K) = CVM(I,K)
  130 CONTINUE
      CALL REGRSS(2)
      DO 140 K=1,4
      COEFVM(I,K) = COEF(K)
  140 CONTINUE
      COEFVM(I,1)=1.0
      COEFVM(I,2)=0.0
      COEFVM(I,3)=0.0
      COEFVM(I,4)=0.0
      WRITE(IOUT,950) I, (NAME(I,K),K=1,3)
      DO 150 K=1,8
      DEV = 100.0D0 * ( Y(K) - CVM(I,K) ) / CVM(I,K)
      WRITE(IOUT,960) TF(K), CVM(I,K), Y(K), DEV
  150 CONTINUE
  160 CONTINUE
C
C     PRINT CURVE FIT COEFFICIENTS
C
      WRITE(IOUT,970)
      DO 170 I=1,N
      WRITE(IOUT,1010) I, (NAME(I,K),K=1,3), (COEFVP(I,K),K=1,4)
  170 CONTINUE
      WRITE(IOUT,980)
      DO 180 I=1,N
      WRITE(IOUT,1010) I, (NAME(I,K),K=1,3), (COEFHV(I,K),K=1,4)
  180 CONTINUE
      WRITE(IOUT,990)
      DO 190 I=1,N
      WRITE(IOUT,1010) I, (NAME(I,K),K=1,3), (COEFHL(I,K),K=1,4)
  190 CONTINUE
      WRITE(IOUT,1000)
      DO 200 I=1,N
      WRITE(IOUT,1010) I, (NAME(I,K),K=1,3), (COEFVM(I,K),K=1,4)
  200 CONTINUE
      RETURN
C
  900 FORMAT(1H1/20X,'C O M P O N E N T   P R O P E R T I E S'
     1 //'    NO  NAME           NBP          MOLE WT     DENSITY'/)
  905 FORMAT(I6,2X,3A4,3G12.4)
  910 FORMAT(1H1/' VAPOR PRESSURE DATA')
  920 FORMAT(1H1/' VAPOR ENTHALPY DATA')
  930 FORMAT(1H1/' LIQUID ENTHALPY DATA')
  940 FORMAT(1H1/' MOLAR VOLUME DATA')
  950 FORMAT(/I6,2X,3A4
     1       /6X,'   TEMP         EXP          CALC         %DEV')
  960 FORMAT(6X,4G12.4)
  970 FORMAT(1H1/' VAPOR PRESSURE COEFFICIENTS'/)
  980 FORMAT(/' VAPOR ENTHALPY COEFFICIENTS'/)
  990 FORMAT(/' LIQUID ENTHALPY COEFFICIENTS'/)
 1000 FORMAT(/' MOLAR VOLUME COEFFICIENTS'/)
 1010 FORMAT(I6,2X,3A4,4G13.5)
      END
      SUBROUTINE REGRSS (L)
C     PERFORMS LINEAR REGRESSION FOR THERMODYNAMIC COEFFICIENTS
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /REGR/
     1     X(8),      Y(8),      C(4,8),    D(4),     COEF(4)
C
C
      DIMENSION      BB(16), LHOLD(4), MHOLD(4)
C
      GO TO (10,40), L
C
C     SETUP AND INVERT MATRIX
C
C     GENERATE THE COEFFICIENT MATRIX
C
   10 CALL WIPE ( C(1,1), 32 )
      DO 20 I=1,8
      C(1,2) = C(1,2) + X(I)
      C(1,3) = C(1,3) + X(I)**2
      C(1,4) = C(1,4) + X(I)**3
      C(2,4) = C(2,4) + X(I)**4
      C(3,4) = C(3,4) + X(I)**5
      C(4,4) = C(4,4) + X(I)**6
   20 CONTINUE
      C(1,1) = 7.0D0
      C(2,1) = C(1,2)
      C(2,2) = C(1,3)
      C(2,3) = C(1,4)
      C(3,1) = C(1,3)
      C(3,2) = C(1,4)
      C(3,3) = C(2,4)
      C(4,1) = C(1,4)
      C(4,2) = C(2,4)
      C(4,3) = C(3,4)
C
      ISIZE = 4
      K = 0
      DO 31 J=1,ISIZE
      DO 30 I=1,ISIZE
      K=K+1
      BB(K) = C(I,J)
   30 CONTINUE
   31 CONTINUE
      CALL MINV (BB, ISIZE, DET, LHOLD, MHOLD )
      IF ( DET .EQ. 0.0D0 )  WRITE(IOUT,900)
      K = 0
      DO 39 J=1,ISIZE
      DO 38 I=1,ISIZE
      K=K+1
      C(I,J) = BB(K)
C
   38 CONTINUE
   39 CONTINUE
C
      RETURN
C
C     CALCULATE COEFFICIENTS, RETURN CALCULATED VALUES
C
   40 CALL WIPE ( D(1), 8 )
      DO 50 I=1,8
      D(1) = D(1) + Y(I)
```

```
      D(2) = D(2) + Y(I) * X(I)
      D(3) = D(3) + Y(I) * X(I)**2
      D(4) = D(4) + Y(I) * X(I)**3
   50 CONTINUE
      COEF(1)=0.0D0
      COEF(2)=0.0D0
      COEF(3)=0.0D0
      COEF(4)=0.0D0
      DO 60 I=1,4
      DO 60 J=1,4
      COEF(I) = COEF(I) + C(I,J) * D(J)
   60 CONTINUE
      DO 70 I=1,8
      Y(I) = COEF(1) + COEF(2)*X(I) + COEF(3)*X(I)**2 + COEF(4)*X(I)**3
   70 CONTINUE
      RETURN
C
  900 FORMAT(/' SINGULAR MATRIX')
C
      END
      SUBROUTINE CNSTNT(NOPER)
C     CALCULATES CONSTANTS
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATAIN/
     1 FR,          ZR(50),     EMR,        PCOND,     DPCOND,     DPTRAY,
     2 GCOND,       GTRAY,      QTRAY,      TTIMES,    IFR
      COMMON /DATA/
     1 F,           Z(50),      P(40),      G(40),     Q(40),      EM(40,50),
     2 FC,          ZC(50),     CF(6),      CZ(50,6), RR,         DR,
     3 FCUT,        ZCUT(50),   TCUT(15),   DMAX,      TOL,        NAME(50,3),
     4 MAXIT,       N,          M,          M1, IC,    INITL,      IPRNT
C
C     CALCULATE FIXED DISTILLATION VARIABLES
C
      M1 = M - 1
      P(1) = PCOND
      G(1) = GCOND
      Q(1) = 0.0D0
      DO 10 J = 2,M
      IF( J .EQ. 2 ) P(J) = P(J-1) + DPCOND
      IF( J .NE. 2 ) P(J) = P(J-1) + DPTRAY
      IF( J .NE. M ) G(J) = GTRAY
      IF( J .EQ. M ) G(J) = 0.0D0
      IF( J .NE. M ) Q(J) = QTRAY
      IF( J .EQ. M ) Q(J) = 0.0D0
   10 CONTINUE
      DO 20 J = 1,M
      DO 20 I = 1,N
      EM(J,I) = EMR
      IF( (J.EQ.1) .OR. (J.EQ.M) ) EM(J,I) = 1.0D0
   20 CONTINUE
C
C     CONVERT AND PRINT FEED CHARGE
C
      IF( NOPER .NE. 0 ) GO TO 40
      TTIMES = 0.0D0
      IC = IFR
      FC = FR
```

```
      DO 30 I=1,N
      ZC(I) = ZR(I)
   30 CONTINUE
   40 WRITE(IOUT,900)
      CALL CONVRT(1)
      F = FC
      FCUT = FC
      IF( NOPER .EQ. 0 ) FR = FC
      DO 50 I=1,N
      Z(I) = ZC(I)
      ZCUT(I) = ZC(I)
      IF( NOPER .EQ. 0 ) ZR(I) = ZC(I)
   50 CONTINUE
      RETURN
C
  900 FORMAT(1H1/20X,'F E E D   C H A R G E')
      END
      SUBROUTINE CONVRT(L)
C     CALCULATES MOLE, WEIGHT, AND LIQUID VOLUME
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /PROP/
     1 CNBP(50),     CMW(50),      CDEN(50),     TF(8),
     2 CVP(50,8),    CHV(50,8),    CHL(50,8),    CVM(50,8),
     3 COEFVP(50,4), COEFHV(50,4), COEFHL(50,4), COEFVM(50,4)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,        DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,     TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
C
C     NORMALIZE INPUT
C
      CALL WIPE ( CF(1), 306 )
      SUM = 0.0
      DO 10 I=1,N
      SUM = SUM + ZC(I)
   10 CONTINUE
      DO 20 I=1,N
      ZC(I) = FC * ZC(I) / SUM
   20 CONTINUE
C
C     CONVERT INPUT
C
      GO TO (30,50,70), IC
C
C     MOLE INPUT
C
   30 DO 40 I=1,N
      CZ(I,1) = ZC(I)
      CZ(I,3) = CZ(I,1) * CMW(I)
      CZ(I,5) = CZ(I,3) / CDEN(I)
   40 CONTINUE
      GO TO 90
C
C     WEIGHT INPUT
C
   50 DO 60 I=1,N
      CZ(I,3) = ZC(I)
```

```
      CZ(I,1) = CZ(I,3) / CMW(I)
      CZ(I,5) = CZ(I,3) / CDEN(I)
   60 CONTINUE
      GO TO 90
C
C     VOLUME INPUT
C
   70 DO 80 I=1,N
      CZ(I,5) = ZC(I)
      CZ(I,3) = CZ(I,5) * CDEN(I)
      CZ(I,1) = CZ(I,3) / CMW(I)
   80 CONTINUE
      GO TO 90
C
C     CALCULATE PERCENTS AND TOTALS
C
   90 DO 100 I=1,N
      CF(1) = CF(1) + CZ(I,1)
      CF(3) = CF(3) + CZ(I,3)
      CF(5) = CF(5) + CZ(I,5)
  100 CONTINUE
      DO 110 I=1,N
      CZ(I,2) = 100.0D0 * CZ(I,1) / CF(1)
      CZ(I,4) = 100.0D0 * CZ(I,3) / CF(3)
      CZ(I,6) = 100.0D0 * CZ(I,5) / CF(5)
      CF(2) = CF(2) + CZ(I,2)
      CF(4) = CF(4) + CZ(I,4)
      CF(6) = CF(6) + CZ(I,6)
  110 CONTINUE
C
C     RETURN MOLE
C
      FC = CF(1)
      DO 120 I=1,N
      ZC(I) = CZ(I,1) / CF(1)
  120 CONTINUE
C
C     PRINT
C
      IF( L .EQ. 3 ) RETURN
      WRITE(IOUT,900)
      IF( (L.EQ.1) .OR. (IPRNT.GE.1) )
     1 WRITE(IOUT,910) (I,(NAME(I,K),K=1,3),(CZ(I,K),K=1,6),I=1,N)
      WRITE(IOUT,920) (CF(K),K=1,6)
      RETURN
C
  900 FORMAT(/'    NO  NAME           LBMOLES     MOLE %      POUNDS
     1  WEIGHT %    GALLONS     VOLUME %'/)
  910 FORMAT(I6,2X,3A4,6G12.4)
  920 FORMAT(/8X,'TOTAL',7X,6G12.4)
      END
      SUBROUTINE STEADY(*)
C     MAIN ROUTINE FOR STEADY-STATE TOTAL-REFLUX CALCULATION
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATA/
     1 F,         Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,        ZC(50),    CF(6),     CZ(50,6), RR,        DR,
     3 FCUT,      ZCUT(50),  TCUT(15),  DMAX,     TOL,       NAME(50,3),
```

```
     4 MAXIT,     N,          M,          M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),     AL(40),     V(40),      U(40),     X(40,50),  Y(40,50),
     2 DT(40),    DL(40),     DV(40),     DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),    HV(50),     HL(50),     VM(50),    DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50),  SCALE,      CONV,      ICONV,     ITER
C
C      INITIALIZE
C
       CALL WIPE ( T(1), 6760 )
       CALL INITAL
       CALL WIPE( DT(1), 2600 )
       ITER = 0
C
       SCALE = 0.0
       CONV = 0.0
       ICONV = 0
C
       CALL SPRINT
C
C      CONVERGENCE LOOP
C
   10  ITER = ITER + 1
       CALL MATSOL(&99)
       CALL CONVRG
       IF( ICONV .EQ. 0 ) CALL HTBAL
       CALL SPRINT
       IF( (ICONV.NE.0) .AND. (ITER.LT.MAXIT) ) GO TO 10
       IF( ICONV .NE. 0 ) GO TO 90
       RETURN
C
C      NON-CONVERGENCE
C
   90  WRITE(IOUT,900) ITER
   99  RETURN 1
C
  900  FORMAT(/' STEADY-STATE TOTAL REFLUX CALCULATIONS DID NOT CONVERGE
     1IN',I3,' ITERATIONS')
       END
       SUBROUTINE INITAL
C      INITIALIZES STEADY-STATE TOTAL-REFLUX CALCULATION
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON /DATA/
      1 F,          Z(50),      P(40),      G(40),     Q(40),     EM(40,50),
      2 FC,         ZC(50),     CF(6),      CZ(50,6),  RR,        DR,
      3 FCUT,       ZCUT(50),   TCUT(15),   DMAX,      TOL,       NAME(50,3),
      4 MAXIT,      N,          M,          M1, IC,    INITL,     IPRNT
       COMMON /SCALC/
      1 T(40),      AL(40),     V(40),      U(40),     X(40,50),  Y(40,50),
      2 DT(40),     DL(40),     DV(40),     DU(40),    DX(40,50), SUMDX(40),
      3 AK(50),     HV(50),     HL(50),     VM(50),    DKDT(50),  DHVDT(50),
      4 DHLDT(50),  DVMDT(50),  SCALE,      CONV,      ICONV,     ITER
C
       INITL = INITL + 1
C
C      ZERO OUT THE X ARRAY
       DO 5 J=1,M
       DO 5 I=1,N
     5 X(J,I)=0.0D0
C
```

```
C     ESTIMATE U
C
C     ASSUME THAT THE COLUMN HAS A TEMPERATURE OF 60 f AND
C     EVALUATE THE LIQUID MOLAR VOLUME (GALLONS/LB-MOLE) TO
C     ESTIMATE THE MOLAR INVENTORY ON EACH STAGE.
C
      J = 1
      T(J) = 60.0D0
      CALL THERMO(J,4)
      DU(1) = 0.0D0
      DO 10 I=1,N
      DU(1) = DU(1) + Z(I) * VM(I)
C
C     F = FEED MOLES
C     G = VOLUMETRIC HOLDUP IN GALLONS
C     U = MOLAR HOLDUP, LB-MOLES
C
   10 CONTINUE
      U(M) = F
      DO 20 J=1,M1
      U(J) = G(J) / DU(1)
      U(M) = U(M) - U(J)
   20 CONTINUE
C
C     IMPROVE INITIALIZATION
C
      DO 110 K=1,INITL
C
C     INITIALIZE X
C
C     ALGORITHUM ASSUMES THAT THE FEED IS ORDERED BY INCREASING
C     BOILING POINT.  START BY PLACING THE LIGHTEST COMPONENT
C     ON THE TOP STAGE (#1 - CONDENSER) AND PROCEED DOWN THE
C     COLUMN.  WHEN COMPONENT IS EXHAUSTER USE THE NEXT HIGHER
C     BOILER.
C
      IF( K .NE. 1 ) CALL WIPE ( X(1,1), 2000 )
C
C     AK IS A DUMMY VARIABLE HERE (NOT K-VALUES) AND CONTAINS
C     THE MOLES OF EACH SPECIES IN THE CHARGE TO THE STILL.
C
      DO 30 I=1,N
      AK(I) = F * Z(I)
   30 CONTINUE
      I = 1
      DO 60 J=1,M
      UU = U(J)
   40 IF( AK(I) .GT. UU ) GO TO 50
      X(J,I) = AK(I)
      UU = UU - AK(I)
      I = I + 1
      IF( I .GT. N ) GO TO 60
      GO TO 40
   50 X(J,I) = UU
      AK(I) = AK(I) - UU
   60 CONTINUE
      DO 70 J=1,M
      DO 70 I=1,N
      X(J,I) = X(J,I) / U(J)
   70 CONTINUE
```

```
C
C     INITIALIZE T
C
      IF( K .EQ. 1 ) T(M) = 300.0D0
      CALL BUBBLE(M,2)
C
C     BUBBLE POINTS ARE ACCOMPLISHED FROM THE BOTTOM OF THE COLUMN
C     TO TH TOP (M = REBOLIER/STILL-POT, 1 = CONDENSER)
C
      DO 80 JJ=1,M1
      J = M1 - JJ + 1
      IF( K .EQ. 1 ) T(J) = T(J+1)
      CALL BUBBLE(J,2)
   80 CONTINUE
C
C     INITIALIZE U
C
C     REVALUATE THE LIQUID MOLAR VOLUME SINCE WE NOW HAVE
C     A COMPOSITION X'S AND TEMPERATURES T'S.
C
      DO 90 J=1,M
      CALL THERMO(J,4)
      DU(J) = 0.0D0
      DO 90 I=1,N
      DU(J) = DU(J) + X(J,I) * VM(I)
   90 CONTINUE
      U(M) = F
      DO 100 J=1,M1
      U(J) = G(J) / DU(J)
      U(M) = U(M) - U(J)
  100 CONTINUE
C
  110 CONTINUE
C
      RETURN
      END
C
      SUBROUTINE BUBBLE(J,L)
C     CONVERGES BUBBLE POINT TEMPERATURE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6),  RR,        DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,      TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),    V(40),     U(40),     X(40,50),  Y(40,50),
     2 DT(40),     DL(40),    DV(40),    DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),    HL(50),    VM(50),    DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,      ICONV,     ITER
C
C     CONVERGE T
C
      SUMX = 0.0
      DO 2 I=1,N
    2 SUMX= SUMX + X(J,I)
      DO 4 I=1,N
    4 X(J,I) = X(J,I)/SUMX
```

```
C
      TOLB = TOL * TOL
      ITER = 0
   10 ITER = ITER + 1
      CALL THERMO(J,1)
      SUMKX = 0.0D0
      SUMDKX = 0.0D0
      DO 20 I=1,N
      SUMKX = SUMKX + EM(J,I) * X(J,I) * AK(I)
      IF( J .NE. M ) SUMKX = SUMKX + ( 1.0D0 - EM(J,I) ) * Y(J+1,I)
      SUMDKX = SUMDKX + EM(J,I) * X(J,I) * DKDT(I)
   20 CONTINUE
      DTI = ( 1.0D0 - SUMKX ) / SUMDKX
      ICONV = 0
      SCALE = 1.0D0
      CONV = 1.0D0
      ERROR = DABS(DTI)
      IF( ERROR .LT. TOLB ) GO TO 30
      SCALE = DMIN1( SCALE, DMAX/ERROR )
      CONV = DMIN1( CONV, TOLB/ERROR )
      ICONV = 1
   30 T(J) = T(J) + SCALE * DTI
      IF( (ICONV.NE.0) .AND. (ITER.LT.MAXIT) ) GO TO 10
C
      IF( ICONV .NE. 0 ) WRITE(IOUT,900) J, ITER, SCALE, CONV, T(J), DTI
C
C     CALCULATE Y
C
      IF(  L .EQ. 2 ) RETURN
      CALL THERMO(J,1)
      SUM = 0.0D0
      DO 40 I=1,N
      Y(J,I) = EM(J,I) * X(J,I) * AK(I)
      IF( J .NE. M ) Y(J,I) = Y(J,I) + ( 1.0D0 - EM(J,I) ) * Y(J+1,I)
      SUM = SUM + Y(J,I)
   40 CONTINUE
      DO 50 I=1,N
      Y(J,I) = Y(J,I) / SUM
   50 CONTINUE
      RETURN
C
  900 FORMAT(' STAGE =',I3,'  ITER =',I3,'  SCALE =',G12.4,'CONV =',
     1       G12.4,'T =',G12.4,'DT =',G12.4)
      END
      SUBROUTINE MATSOL(*)
C
C     SOLVES STEADY-STATE TOTAL-REFLUX MATRIX
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,         DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,      TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),    V(40),     U(40),     X(40,50),  Y(40,50),
     2 DT(40),     DL(40),    DV(40),    DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),    HL(50),    VM(50),    DKDT(50),  DHVDT(50),
```

```
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,     ICONV,     ITER
      COMMON /MATRIX/
     1 A(52,52),  B(52,52),  C(52,52),  D(52),    DM(52),
     2 AHOLD(52,52),  CHOLD(52,52),  DHOLD(52),   ILAST,  I1,  I2
      DIMENSION CSAVE(52,52,40), DSAVE(52,40)
      DATA IFILE/'1   '/
C
C
C     CALCULATE MATRIX CONSTANTS
C     N = NUMBER OF COMONENTS
C     M = NUMBER OF STAGES
C
      I1 = N + 1
      I2 = N + 2
C
C     SETUP AND REDUCE MATRIX
C
      DO 70 J = 1,M
      CALL SETUP(J)
      IF( J .EQ. 1 ) GO TO 20
      IF( J .EQ. M ) GO TO 40
C
C     A-HOLD IS ELIMINATED VIA A SUBSTRACTION
C     DM = DM - AHOLD * DHOLD
C     A  = A  - AHOLD * CHOLD
C
      DO 10 K1 = 1,I2
      DO 10 K2 = 1,I2
      DM(K1) = DM(K1) - AHOLD(K1,K2) * DHOLD(K2)
      DO 10 K3 = 1,I2
      A(K1,K2) = A(K1,K2) - AHOLD(K1,K3) * CHOLD(K3,K2)
   10 CONTINUE
C
C     AHOLD = A
C     DHOLD = BINV * D
C     CHOLD = BINV * C
C     SAVE CHOLD, DHOLD
C
C     PUT A MATRIX INTO A-HOLD,
C     PREMULTIPLY C BY B-INVERSE AND SAVE IN C-HOLD,
C     PREMULTIPLY D BY D-INVERSE AND SAVE IN D-HOLD
C
   20 CALL WIPE( AHOLD(1,1), 5460 )
      DO 30 K1 = 1,I2
      DO 30 K2 = 1,I2
      AHOLD(K1,K2) = A(K1,K2)
      DHOLD(K1) = DHOLD(K1) + B(K1,K2) * D(K2)
      DO 30 K3 = 1,I2
      CHOLD(K1,K2) = CHOLD(K1,K2) + B(K1,K3) * C(K3,K2)
   30 CONTINUE
      DO 34 K1 = 1,I2
      DO 32 K2 = 1,I2
      CSAVE(K1,K2,J) = CHOLD(K1,K2)
   32 CONTINUE
      DSAVE(K1,J) = DHOLD(K1)
   34 CONTINUE
      GO TO 70
C
C     DM = DM - AHOLD * DHOLD
C     B  = B  - AHOLD * CHOLD
```

```
C
   40 DO 50 K1 = 1,I2
      DO 50 K2 = 1,I2
      DM(K1) = DM(K1) - AHOLD(K1,K2) * DHOLD(K2)
      DO 50 K3 = 1,I2
      B(K1,K2) = B(K1,K2) - AHOLD(K1,K3) * CHOLD(K3,K2)
   50 CONTINUE
C
C     INVERT B
C
      CALL MATINV (&99)
C
C     D = BINV * DM
C
      CALL WIPE ( D(1), 52 )
      DO 60 K1 = 1,I2
      DO 60 K2 = 1,I2
      D(K1) = D(K1) + B(K1,K2) * DM(K2)
   60 CONTINUE
   70 CONTINUE
C
C     SOLUTION
C     DHOLD = DHOLD - CHOLD * D
C
      DO 120 JJ = 1,M
      J = M - JJ + 1
      IF( J .EQ. M ) GO TO 100
C
      DO 74 K1 = 1,I2
      DO 72 K2 = 1,I2
      CHOLD(K1,K2) =  CSAVE(K1,K2,J)
   72 CONTINUE
      DHOLD(K1) = DSAVE(K1,J)
   74 CONTINUE
C
      DO 80 K1 = 1,I2
      DO 80 K2 = 1,I2
      DHOLD(K1) = DHOLD(K1) - CHOLD(K1,K2) * D(K2)
   80 CONTINUE
      DO 90 K1 = 1,I2
      D(K1) = DHOLD(K1)
   90 CONTINUE
C
  100 DO 110 I = 1,N
      DX(J,I) = D(I)
  110 CONTINUE
      DT(J) = D(I1)
      DU(J) = D(I2)
  120 CONTINUE
c
c     write (6,130) iter
c  130 format('1',4x,'iter = ', i5 /
c     1        5x, 'stage',5x,'temp dt',5x,'inventory')
c     write (6,140) (i,dt(i),du(i),i=1,m)
c  140 format(i8,2f14.2)
c
c     write (6,150)
c  150 format('1',4x,'stage',5x,'comps 1-5 dxs')
c     do 170 j=1,m
c     write (6,160) j,(dx(j,i),i=1,n)
```

```
c  160 format(i5,5(1x,d11.4))
c 170 continue
c
      RETURN
C
C     SINGULAR MATRIX
C
   99 WRITE(IOUT,900)
      RETURN 1
C
  900 FORMAT(/' SINGULAR MATRIX')
      END
      SUBROUTINE MATINV (*)
C
C     INVERTS MATRIX USING IBM-SSP ROUTINE 'MINV'
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /MATRIX/
     1 A(52,52),  B(52,52),  C(52,52),  D(52),    DM(52),
     2 AHOLD(52,52),  CHOLD(52,52),  DHOLD(52),   ILAST,  I1,  I2
      COMMON /CLSODE/
     1 XT(2000),   RWORK(26022),  IWORK(2020)
C
      DIMENSION BB(2704), LHOLD(52), MHOLD(52)
C
      DATA IN/52/
C
C     SETUP MATRIX FOR MINV
C
      CALL WIPE ( BB(1), 2704)
C
C     MOVE MATRIX INTO 'BB'
C
      K=0
      DO 11 I=1,I2
      DO 10 J=1,I2
      K=K+1
      BB(K) = B(I,J)
   10 CONTINUE
   11 CONTINUE
C
C     INVERT MATRIX USING MINV
C
      CALL MINV ( BB, I2, DET, LHOLD, MHOLD )
           IF (DET .EQ. 0.0D0) RETURN 1
C
C     MOVE INVERSE INTO 'B'
C
      K=0
      DO 21 I=1,I2
      DO 20 J=1,I2
      K=K+1
      B(I,J) = BB(K)
   20 CONTINUE
   21 CONTINUE
C
      END
      SUBROUTINE SETUP(J)
C     SETS-UP STEADY-STATE TOTAL-REFLUX MATRIX
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      COMMON /DATA/
     1 F,          Z(50),      P(40),      G(40),      Q(40),      EM(40,50),
     2 FC,         ZC(50),     CF(6),      CZ(50,6),   RR,         DR,
     3 FCUT,       ZCUT(50),   TCUT(15),   DMAX,       TOL,        NAME(50,3),
     4 MAXIT,      N,          M,          M1, IC,     INITL,      IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),     V(40),      U(40),      X(40,50),   Y(40,50),
     2 DT(40),     DL(40),     DV(40),     DU(40),     DX(40,50),  SUMDX(40),
     3 AK(50),     HV(50),     HL(50),     VM(50),     DKDT(50),   DHVDT(50),
     4 DHLDT(50),  DVMDT(50),  SCALE,      CONV,       ICONV,      ITER
      COMMON /MATRIX/
     1 A(52,52),   B(52,52),   C(52,52),   D(52),      DM(52),
     2 AHOLD(52,52),  CHOLD(52,52),  DHOLD(52),    ILAST,   I1,   I2
C
C       CALL WIPE ( A(1,1), 8164 )
      DO 16 K1=1,52
      DO 12 K2=1,52
      A(K2,K1)=0.0D0
      B(K2,K1)=0.0D0
      C(K2,K1)=0.0D0
   12 CONTINUE
      D(K1)=0.0D0
   16 CONTINUE
C
C     A MATRIX
C
      IF( J .EQ. M ) GO TO 20
      DO 10 I=1,N
      A(I,I) = U(J)
      A(I,I2) = X(J,I)
   10 CONTINUE
      A(I2,I2) = 1.0D0
C
C     B AND D MATRIX
C
C     THERMO(J,1) OBTAINS K-VALUES AND THE TEMP. DERIVATIVES FOR STAGE J
C     THERMO(J,4) OBTAINS LIQUID MOLAR VOL. AND THE TEMP. DERIVATIVES
C     FOR STAGE J
C
   20 CALL THERMO(J,1)
      IF( J .NE. M ) CALL THERMO(J,4)
      DO 40 I=1,N
      B(I1,I) = EM(J,I) * AK(I) + ( 1.0D0 - EM(J,I) )
      B(I1,I1) = B(I1,I1) + EM(J,I) * DKDT(I) * X(J,I)
      IF( J .EQ. M ) GO TO 30
      B(I,I) = 1.0D0
      B(I2,I) = U(J) * VM(I)
      B(I2,I1) = B(I2,I1) + U(J) * X(J,I) * DVMDT(I)
      B(I2,I2) = B(I2,I2) + X(J,I) * VM(I)
      D(I) = -X(J,I)
      D(I1) = D(I1) - ( EM(J,I) * AK(I) + ( 1.0D0 - EM(J,I) ) ) * X(J,I)
      D(I2) = D(I2) - U(J) * X(J,I) * VM(I)
      GO TO 40
   30 B(I,I) = U(J)
      B(I,I2) = X(J,I)
   40 CONTINUE
      IF( J .EQ. M ) B(I2,I2) = 1.0D0
      IF( J .EQ. M ) RETURN
      D(I1) = D(I1) + 1.0D0
      D(I2) = D(I2) + G(J)
```

```
C
C     INVERT B MATRIX
C
      DO 50 I=1,N
      B(I2,I) = (B(I1,I)*B(I2,I1)-B(I2,I)*B(I1,I1))/(B(I1,I1)*B(I2,I2))
      B(I1,I) = -B(I1,I) / B(I1,I1)
   50 CONTINUE
      B(I2,I1) = -B(I2,I1) / ( B(I1,I1) * B(I2,I2) )
      B(I1,I1) = 1.0D0 / B(I1,I1)
      B(I2,I2) = 1.0D0 / B(I2,I2)
C
C     C AND D MATRIX
C
C     THERMO(J+1,1) OBTAINS K-VALUES AND THE TEMP. DERIVATIVES FOR STAGE J+1
C
      CALL THERMO(J+1,1)
      DO 60 I=1,N
      C(I,I) = -( EM(J+1,I) * AK(I) + ( 1.0D0 - EM(J+1,I) ) )
      C(I,I1) = -EM(J+1,I) * DKDT(I) * X(J+1,I)
      D(I) = D(I) + ( EM(J+1,I) * AK(I) + (1.0D0-EM(J+1,I)) ) * X(J+1,I)
   60 CONTINUE
      IF( J .NE. 1 ) RETURN
C
C     DM MATRIX
C
c      CALL WIPE ( DM(1), 52 )
      DO 62 K1=1,52
   62 DM(K1)=0.0D0
C     THERMO(M,1) OBTAINS K-VALUES AND THE TEMP. DERIVATIVES FOR STAGE M
C
      CALL THERMO(M,1)
      DO 70 I=1,N
      DM(I) = F * Z(I)
      DM(I1) = DM(I1) - ( EM(M,I) * AK(I) + ( 1.0D0-EM(M,I) ) ) * X(M,I)
      DO 70 JJ = 1,M
      DM(I) = DM(I) - U(JJ) * X(JJ,I)
   70 CONTINUE
      DO 80 JJ = 1,M
      DM(I2) = DM(I2) - U(JJ)
   80 CONTINUE
      DM(I1) = DM(I1) + 1.0D0
      DM(I2) = DM(I2) + F
      RETURN
      END
C
      SUBROUTINE THERMO(J,L)
C     CALCULATES THERMODYNAMIC DATA
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /PROP/
     1 CNBP(50),     CMW(50),      CDEN(50),     TF(8),
     2 CVP(50,8),    CHV(50,8),    CHL(50,8),    CVM(50,8),
     3 COEFVP(50,4), COEFHV(50,4), COEFHL(50,4), COEFVM(50,4)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6),  RR,        DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,      TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),    V(40),     U(40),     X(40,50),  Y(40,50),
     2 DT(40),     DL(40),    DV(40),    DU(40),    DX(40,50), SUMDX(40),
```

```
     3 AK(50),    HV(50),    HL(50),    VM(50),   DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,     ICONV,     ITER
C
      TR1 = T(J) + 459.67D0
      TR2 = TR1 * TR1
      TR3 = TR2 * TR1
      TR4 = TR3 * TR1
      GO TO (1,2,3,4), L
C
C     K-VALUE
C
    1 DO 10 I=1,N
      AK(I) =COEFVP(I,1)+COEFVP(I,2)/TR1+COEFVP(I,3)/TR2+COEFVP(I,4)/TR3
      AK(I) = DEXP(AK(I)) / P(J)
      DKDT(I) = -COEFVP(I,2)/TR2-2.0D0*COEFVP(I,3)/TR3-3.0D0*COEFVP(I,4)
     1          /TR4
      DKDT(I) = DKDT(I) * AK(I)
   10 CONTINUE
      RETURN
C
C     VAPOR ENTHALPY
C
    2 DO 20 I=1,N
      HV(I) =COEFHV(I,1)+COEFHV(I,2)*TR1+COEFHV(I,3)*TR2+COEFHV(I,4)*TR3
      DHVDT(I) = COEFHV(I,2)+2.0D0*COEFHV(I,3)*TR1+3.0D0*COEFHV(I,4)*TR2
   20 CONTINUE
      RETURN
C
C     LIQUID ENTHALPY
C
    3 DO 30 I=1,N
      HL(I) =COEFHL(I,1)+COEFHL(I,2)*TR1+COEFHL(I,3)*TR2+COEFHL(I,4)*TR3
      DHLDT(I) = COEFHL(I,2)+2.0D0*COEFHL(I,3)*TR1+3.0D0*COEFHL(I,4)*TR2
   30 CONTINUE
      RETURN
C
C     MOLAR VOLUME
C
    4 DO 40 I=1,N
      VM(I) =COEFVM(I,1)+COEFVM(I,2)*TR1+COEFVM(I,3)*TR2+COEFVM(I,4)*TR3
      DVMDT(I) = COEFVM(I,2)+2.0D0*COEFVM(I,3)*TR1+3.0D0*COEFVM(I,4)*TR2
   40 CONTINUE
      RETURN
      END
C
      SUBROUTINE CONVRG
C     CHECKS CONVERGENCE AND DETERMINES NEW VALUES FOR STEADY-STATE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,         DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,      TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),    V(40),     U(40),     X(40,50),  Y(40,50),
     2 DT(40),     DL(40),    DV(40),    DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),    HL(50),    VM(50),    DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,      ICONV,     ITER
C
C     CALCULATE SCALE
```

```
C
      ICONV = 0
      SCALE = 1.0D0
      CONV = 1.0D0
      DO 20 J=1,M
      ERROR = DABS(DT(J))
      IF( ERROR .LT. TOL ) GO TO 10
      SCALE = DMIN1( SCALE, DMAX/ERROR )
      CONV = DMIN1( CONV, TOL/ERROR )
      ICONV = ICONV + 1
   10 ERROR = 100.0D0 * DABS( DU(J)/U(J) )
      IF( ERROR .LT. TOL ) GO TO 20
      SCALE = DMIN1( SCALE, DMAX/ERROR )
      CONV = DMIN1( CONV, TOL/ERROR )
      ICONV = ICONV + 1
   20 CONTINUE
C
C     CALCULATE NEW T AND U
C
      DO 30 J=1,M
      T(J) = T(J) + SCALE * DT(J)
      U(J) = U(J) + SCALE * DU(J)
   30 CONTINUE
C
C     CALCULATE SUMDX
C
      DO 40 J=1,M
      SUMDX(J) = 0.0D0
      DO 40 I=1,N
      SUMDX(J) = SUMDX(J) + DABS(DX(J,I))
   40 CONTINUE
C
C     CHECK X CONVERGENCE
C
      RTOL = 0.01D0 * TOL
      ATOL = RTOL * RTOL
      DO 50 J=1,M
      DO 50 I=1,N
      ERROR = DABS( DX(J,I) )
      XTOL = RTOL * DABS( X(J,I) ) + ATOL
      IF( ERROR .GT. XTOL ) ICONV = ICONV + 1
   50 CONTINUE
C
C     CALCULATE NEW X
C
      DO 100 J=1,M
      XSUM = 0.0D0
      DO 80 I=1,N
      DELX = SCALE * DX(J,I)
      XNEW = X(J,I) + DELX
      IF( XNEW .GE. 0.0D0 ) GO TO 70
      IF( X(J,I) .EQ. 0.0D0 ) GO TO 60
      IF( DELX/X(J,I) .LT. -30.0D0 ) GO TO 60
      XNEW = X(J,I) * DEXP( DELX / X(J,I) )
      GO TO 70
   60 XNEW = 0.0D0
   70 X(J,I) = XNEW
      XSUM = XSUM + X(J,I)
   80 CONTINUE
      DO 90 I=1,N
```

```
      X(J,I) = X(J,I) / XSUM
   90 CONTINUE
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE HTBAL
C     CALCULATES FLOWS FOR STEADY-STATE TOTAL-REFLUX
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DATA/
     1 F,          Z(50),      P(40),      G(40),     Q(40),      EM(40,50),
     2 FC,         ZC(50),     CF(6),      CZ(50,6), RR,         DR,
     3 FCUT,       ZCUT(50),   TCUT(15),   DMAX,      TOL,        NAME(50,3),
     4 MAXIT,      N,          M,          M1, IC,    INITL,      IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),     V(40),      U(40),     X(40,50),   Y(40,50),
     2 DT(40),     DL(40),     DV(40),     DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),     HL(50),     VM(50),    DKDT(50),   DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,      CONV,      ICONV,      ITER
C
C     CALCULATE Y
C
      DO 30 J=1,M
      CALL THERMO(J,1)
      YSUM = 0.0D0
      DO 10 I=1,N
      Y(J,I) = ( EM(J,I) * AK(I) + ( 1.0D0 - EM(J,I) ) ) * X(J,I)
      YSUM = YSUM + Y(J,I)
   10 CONTINUE
      DO 20 I=1,N
      Y(J,I) = Y(J,I) / YSUM
   20 CONTINUE
   30 CONTINUE
C
C     CALCULATE HV AND HL
C
      DO 40 J=1,M
      CALL THERMO(J,2)
      CALL THERMO(J,3)
      DV(J) = 0.0D0
      DL(J) = 0.0D0
      DO 40 I=1,N
      DV(J) = DV(J) + Y(J,I) * HV(I)
      DL(J) = DL(J) + X(J,I) * HL(I)
   40 CONTINUE
C
C     CALCULATE V, L, Q1, AND QM
C
      DO 80 J=1,M
      IF( J .EQ. 1 ) GO TO 50
      IF( J .EQ. M ) GO TO 70
      GO TO 60
C
C     CONDENSER
C
   50 V(J) = 0.0D0
      AL(J) = RR * DR + DR
      Q(J) = AL(J) * ( DL(J) - DV(J+1) )
      GO TO 80
C
```

```
C     STAGES
C
   60 V(J) = AL(J-1)
      AL(J) = ( AL(J-1) * ( DV(J)-DL(J-1) ) - Q(J) ) / ( DV(J+1)-DL(J) )
      GO TO 80
C
C     REBOILER
C
   70 V(J) = AL(J-1)
      AL(J) = 0.0D0
      Q(J) = AL(J-1) * ( DV(J) - DL(J-1) )
   80 CONTINUE
C
C     CALCULATE GM
C
      CALL THERMO(M,4)
      SUM = 0.0D0
      DO 90 I=1,N
      SUM = SUM + X(M,I) * VM(I)
   90 CONTINUE
      G(M) = U(M) * SUM
      RETURN
      END
C
      SUBROUTINE SPRINT
C     PRINTS STEADY-STATE TOTAL-REFLUX RESULTS
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATA/
     1 F,          Z(50),      P(40),      G(40),     Q(40),      EM(40,50),
     2 FC,         ZC(50),     CF(6),      CZ(50,6), RR,         DR,
     3 FCUT,       ZCUT(50),   TCUT(15),   DMAX,      TOL,        NAME(50,3),
     4 MAXIT,      N,          M,          M1, IC,    INITL,      IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),     V(40),      U(40),     X(40,50),   Y(40,50),
     2 DT(40),     DL(40),     DV(40),     DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),     HL(50),     VM(50),    DKDT(50),   DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,      CONV,      ICONV,      ITER
C
      IF( (ICONV.NE.0) .AND. (IPRNT.LE.2) ) RETURN
C
C     PRINT STEADY STATE PROFILE
C
      WRITE(IOUT,900) ITER
      WRITE(IOUT,910) (J,DT(J),T(J),V(J),AL(J),DU(J),U(J),G(J),Q(J),
     1                 P(J),SUMDX(J),J=1,M)
      WRITE(IOUT,920) SCALE, CONV, ICONV
C
      IF( (ICONV.NE.0) .OR. (IPRNT.LE.1) ) RETURN
C
C     PRINT X
C
      WRITE(IOUT,930)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 10 J=1,M
      WRITE(IOUT,1020) J, (X(J,I),I=1,N)
   10 CONTINUE
C
```

```
C     PRINT DX
C
      WRITE(IOUT,940)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 20 J=1,M
      WRITE(IOUT,1020) J, (DX(J,I),I=1,N)
   20 CONTINUE
C
C     PRINT Y
C
      WRITE(IOUT,950)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 30 J=1,M
      WRITE(IOUT,1020) J, (Y(J,I),I=1,N)
   30 CONTINUE
C
C     PRINT EM
C
      WRITE(IOUT,960)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 40 J=1,M
      WRITE(IOUT,1020) J, (EM(J,I),I=1,N)
   40 CONTINUE
      RETURN
C
  900 FORMAT(1H1/40X,'ITERATION NUMBER',I6
     1 //40X,'S T E A D Y - S T A T E   P R O F I L E'
     2 //' STAGE   DEL TEMP  TEMPERATURE  VAPOR RATE LIQUID RATE  DEL HO
     3LDUP  MOL HOLDUP  VOL HOLDUP   HEAT DUTY   PRESSURE    SUMDX'/)
  910 FORMAT(I6,10G12.4)
  920 FORMAT(/'   SCALE       CONV      ICONV'
     1       /2G12.4,I6)
  930 FORMAT(1H1/40X,'L I Q U I D   C O M P O S I T I O N')
  940 FORMAT(  //40X,'D E L   L I Q U I D   C O M P')
  950 FORMAT(1H1/40X,'V A P O R   C O M P O S I T I O N')
  960 FORMAT(  //40X,'M U R P H R E E   E F F I C I E N C Y')
 1000 FORMAT(/6X,10(I4,8X),4X
     1       /6X,10(I4,8X),4X
     2       /6X,10(I4,8X),4X
     3       /6X,10(I4,8X),4X
     4       /6X,10(I4,8X),4X    )
 1010 FORMAT(' STAGE',3X,30A4,1X
     1       /        9X,30A4,1X
     2       /        9X,30A4,1X
     3       /        9X,30A4,1X
     4       /        9X,30A4,1X )
 1020 FORMAT(I6,10G12.4,4X
     1      /6X,10G12.4,4X
     2      /6X,10G12.4,4X
     3      /6X,10G12.4,4X
     4      /6X,10G12.4,4X       )
      END
      SUBROUTINE UNSTDY(*)
C     MAIN ROUTINE FOR UNSTEADY-STATE BATCH DISTILLATION CALCULATIONS
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
```

```
      COMMON /DATAIN/
     1 FR,          ZR(50),     EMR,        PCOND,     DPCOND,     DPTRAY,
     2 GCOND,       GTRAY,      QTRAY,      TTIMES,    IFR
      COMMON /DATA/
     1 F,           Z(50),      P(40),      G(40),     Q(40),      EM(40,50),
     2 FC,          ZC(50),     CF(6),      CZ(50,6),  RR,         DR,
     3 FCUT,        ZCUT(50),   TCUT(15),   DMAX,      TOL,        NAME(50,3),
     4 MAXIT,       N,          M,          M1, IC,    INITL,      IPRNT
      COMMON /SCALC/
     1 T(40),       AL(40),     V(40),      U(40),     X(40,50),   Y(40,50),
     2 DT(40),      DL(40),     DV(40),     DU(40),    DX(40,50),  SUMDX(40),
     3 AK(50),      HV(50),     HL(50),     VM(50),    DKDT(50),   DHVDT(50),
     4 DHLDT(50),   DVMDT(50),  SCALE,      CONV,      ICONV,      ITER
      COMMON /CLSODE/
     1 XT(2000),    RWORK(26022),  IWORK(2020)
      EXTERNAL FUNCT, JACOB
      DATA LRW/26022/, LIW/2020/, IOPT/0/, ITASK/2/, ITOL/1/, MF/24/
C
      CALL WIPE ( XT(1), 28022 )
      CALL IWIPE ( IWORK(1), 2020 )
C
C     INITIALIZE VARIABLES FOR FIRST CALL TO LSODE
C
      ISTATE = 1
      IWORK(1) = 1
      IWORK(2) = 1
      NEQ = N * M
      RTOL = 0.01D0 * TOL
      ATOL = RTOL * RTOL
      TIME = 0.0D0
      TOUT = 1.0D0 / 60.0D0
      DO 10 J=1,M
      DO 10 I=1,N
      L = M * ( I - 1 ) + J
      XT(L) = X(J,I)
   10 CONTINUE
      ITC = 0
      TIMCUT = 0.0D0
C
C     CALL LSODE
C
   20 CALL LSODE( FUNCT, NEQ, XT, TIME, TOUT, ITOL, RTOL, ATOL, ITASK,
     1            ISTATE, IOPT, RWORK, LRW, IWORK, LIW, JACOB, MF)
      IF( ISTATE .LT. 0 ) GO TO 30
      IF( U(M)/F .LT. 0.05D0 ) GO TO 30
      IF( ITC .EQ. 0 ) GO TO 30
      IF( T(2) .GE. TCUT(ITC) ) GO TO 30
      GO TO 20
C
C     PRINT TIME AND OVERHEAD TEMPERATURE
C
   30 TTIME = TTIMES + TIME
      TIMCUT = TIME - TIMCUT
      IC = 1
      FC = DR
      DO 40 I=1,N
      ZC(I) = X(1,I)
   40 CONTINUE
      CALL CONVRT(3)
      SG = CF(3) / CF(5) / 8.345D0 / 0.999D0
```

```
      PMMHG = P(1)
      TF = T(2)
      CALL MAXBON( PMMHG, TF, SG, TNBP, TNBP12 )
      WRITE(IOUT,900) TTIME, TIME, TIMCUT, RWORK(11), T(2), TNBP, TNBP12
C
C     PRINT INSTANTANEOUS DISTILLATE
C
      IC = 1
      FC = DR
      DO 50 I=1,N
      ZC(I) = X(1,I)
   50 CONTINUE
      WRITE(IOUT,910)
      CALL CONVRT(2)
C
C     PRINT CUT DISTILLATE
C
      IC = 1
      FC = FCUT
      DO 60 J=1,M
      FC = FC - U(J)
   60 CONTINUE
      DO 70 I=1,N
      ZC(I) = FCUT * ZCUT(I)
      DO 70 J=1,M
      ZC(I) = ZC(I) - U(J) * X(J,I)
   70 CONTINUE
      WRITE(IOUT,920)
      CALL CONVRT(2)
C
C     PRINT CUMULATIVE DISTILLATE
C
      IC = 1
      FC = F
      DO 80 J=1,M
      FC = FC - U(J)
   80 CONTINUE
      DO 90 I=1,N
      ZC(I) = F * Z(I)
      DO 90 J=1,M
      ZC(I) = ZC(I) - U(J) * X(J,I)
   90 CONTINUE
      WRITE(IOUT,930)
      CALL CONVRT(2)
C
C     PRINT TOTAL DISTILLATE
C
      IC = 1
      FC = FR
      DO 100 J=1,M
      FC = FC - U(J)
  100 CONTINUE
      DO 110 I=1,N
      ZC(I) = FR * ZR(I)
      DO 110 J=1,M
      ZC(I) = ZC(I) - U(J) * X(J,I)
  110 CONTINUE
      WRITE(IOUT,940)
      CALL CONVRT(2)
C
```

```
C     ADDITIONAL PRINTS
C
      WRITE(IOUT,950) IWORK(11), IWORK(12), IWORK(13)
      IF( IPRNT .LE. 3 ) GO TO 120
      K1 = IWORK(17) - NEQ + 1
      K2 = IWORK(17)
      WRITE(IOUT,960) (RWORK(K),K=K1,K2)
  120 CALL UPRINT(TIME)
C
C     CHECK FOR ERRORS OR LIMITS
C
      IF( ISTATE .LT. 0 ) GO TO 210
      IF( U(M)/F .LT. 0.05D0 ) GO TO 170
      ITC = ITC + 1
      IF( ITC .GT. 15 ) GO TO 170
      IF( TCUT(ITC) .EQ. 0.0D0 ) GO TO 170
C
C     INITIALIZE FOR NEXT CUT
C
      TIMCUT = 0.0D0
      IF( ITC .EQ. 1 ) GO TO 20
      TIMCUT = TIME
      FCUT = 0.0D0
      DO 130 J=1,M
      FCUT = FCUT + U(J)
  130 CONTINUE
      DO 140 I=1,N
      ZCUT(I) = 0.0D0
      DO 140 J=1,M
      ZCUT(I) = ZCUT(I) + U(J) * X(J,I)
  140 CONTINUE
      SUM = 0.0D0
      DO 150 I=1,N
      SUM = SUM + ZCUT(I)
  150 CONTINUE
      DO 160 I=1,N
      ZCUT(I) = ZCUT(I) / SUM
  160 CONTINUE
      GO TO 20
C
C     CALCULATE FEED CHARGE FOR NEXT OPERATING STEP
C
  170 TTIMES = TTIME
      IC = 1
      FC = 0.0D0
      DO 180 J=1,M
      FC = FC + U(J)
  180 CONTINUE
      DO 200 I=1,N
      ZC(I) = 0.0D0
      DO 190 J=1,M
      ZC(I) = ZC(I) + U(J) * X(J,I)
  190 CONTINUE
      IF( ZC(I)/FC .LT. 1.0D-12 ) ZC(I) = 0.0D0
  200 CONTINUE
      RETURN
C
C     ERROR RETURN
C
  210 WRITE(IOUT,970) ISTATE
```

```
      RETURN 1
C
  900 FORMAT(1H1/10X,'TOTAL TIME, HOURS      ',G12.4
     1          /10X,'CUMULATIVE TIME, HOURS',G12.4
     2          /10X,'CUT TIME, HOURS        ',G12.4
     3          /10X,'TIME STEP, HOURS       ',G12.4
     4         //10X,'OVERHEAD TEMPERATURE   ',G12.4
     5          /10X,'AT 760 MM HG           ',G12.4
     6          /10X,'K = 12                 ',G12.4)
  910 FORMAT(//20X,'INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR')
  920 FORMAT(//20X,'CUT DISTILLATE DRAWOFF')
  930 FORMAT(//20X,'CUMULATIVE DISTILLATE DRAWOFF')
  940 FORMAT(//20X,'TOTAL DISTILLATE DRAWOFF')
  950 FORMAT(//10X,'NUMBER OF STEPS          ',I6
     1        /10X,'NUMBER OF FUNCTION CALLS',I6
     2        /10X,'NUMBER OF JACOBIAN CALLS',I6)
  960 FORMAT(10G12.4)
  970 FORMAT(/' LSODE ERROR, ISTATE =',I6)
      END
      SUBROUTINE MAXBON( PMMHG, TF, SG, TNBP, TNBP12 )
C     CALCULATES NORMAL BOILING POINT USING MAXWELL-BONNELL
      IMPLICIT REAL*8(A-H,O-Z)
C
      THIRD = 1.0D0 / 3.0D0
      PATM = PMMHG / 760.0D0
      TR = TF + 459.67D0
      PLOG = DLOG10(PATM)
      IF( PMMHG .LT. 2.0D0 )
     1 A = (4.741D0-PLOG) / (2876.663D0-43.0D0*PLOG) - 0.0002867D0
      IF( (PMMHG.GE.2.0D0) .AND. (PMMHG.LE.760.0D0) )
     1 A = (3.877D0-PLOG) / (2387.262D0-95.76D0*PLOG) - 0.0002867D0
      IF( PMMHG .GT. 760.0D0 )
     1 A = (4.326D0-PLOG) / (2666.376D0-36.0D0*PLOG) - 0.0002867D0
      TNBP = 748.1D0*A / ( 1.0D0/TR - 0.0002867D0 + 0.2145D0*A )
      TNBP12 = TNBP - 459.67D0
      WK = TNBP**THIRD / SG
      F = -1.0D0 + 0.005D0 * ( TNBP - 459.67D0 )
      IF( F .LT. 0.0D0 ) F = 0.0D0
      IF( F .GT. 1.0D0 ) F = 1.0D0
      TNBP = TNBP + 2.5D0 * ( WK - 12.0D0 ) * PLOG * F - 459.67D0
      RETURN
      END
      SUBROUTINE FUNCT( NEQ, TIME, XT, DXDT )
C     CALCULATES DX/DT FOR LSODE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,         DR,
     3 FCUT,       ZCUT(50),  TCUT(15),  DMAX,      TOL,       NAME(50,3),
     4 MAXIT,      N,         M,         M1, IC,    INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),      AL(40),    V(40),     U(40),     X(40,50),  Y(40,50),
     2 DT(40),     DL(40),    DV(40),    DU(40),    DX(40,50), SUMDX(40),
     3 AK(50),     HV(50),    HL(50),    VM(50),    DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,      ICONV,     ITER
      COMMON /UCALC/
     1 SAK(40),    SHV(40),   SHL(40),   SVM(40),  SDKDT(40), SDHLDT(40),
     2 SDVMDT(40),SAKY1(40), SAKX1(40), SAKY(40), SHLY1(40), SHLX1(40),
     3 SHLY(40),  SVMY1(40), SVMX1(40), SVMY(40), AKS(40,50),
     4 DTDT(40),  DUDT(40),  DHDT(40)
```

```
      DIMENSION XT(NEQ), DXDT(NEQ)
C
      CALL WIPE ( SAK(1), 2640 )
C
C     NORMALIZE X
C
      DO 30 J=1,M
      XSUM = 0.0D0
      DO 10 I=1,N
      L = M * ( I - 1 ) + J
      X(J,I) = XT(L)
      XSUM = XSUM + X(J,I)
   10 CONTINUE
      DO 20 I=1,N
      X(J,I) = X(J,I) / XSUM
   20 CONTINUE
   30 CONTINUE
C
C     CALCULATE T, Y, AND AK
C
      DO 40 JJ=1,M
      J = M - JJ + 1
      CALL BUBBLE(J,1)
      DO 40 I=1,N
      AKS(J,I) = AK(I)
      SAK(J) = SAK(J) + X(J,I) * AK(I)
      SDKDT(J) = SDKDT(J) + X(J,I) * DKDT(I)
      SAKY(J) = SAKY(J) + Y(J,I) * AK(I)
      IF( J .NE. M ) SAKY1(J) = SAKY1(J) + Y(J+1,I) * AK(I)
      IF( J .NE. 1 ) SAKX1(J) = SAKX1(J) + X(J-1,I) * AK(I)
   40 CONTINUE
C
C     CALCULATE HV, HL, AND VM
C
      DO 50 J=1,M
      CALL THERMO(J,2)
      CALL THERMO(J,3)
      CALL THERMO(J,4)
      DO 50 I=1,N
      SHV(J) = SHV(J) + Y(J,I) * HV(I)
      SHL(J) = SHL(J) + X(J,I) * HL(I)
      SVM(J) = SVM(J) + X(J,I) * VM(I)
      SDHLDT(J) = SDHLDT(J) + X(J,I) * DHLDT(I)
      SDVMDT(J) = SDVMDT(J) + X(J,I) * DVMDT(I)
      SHLY(J) = SHLY(J) + Y(J,I) * HL(I)
      SVMY(J) = SVMY(J) + Y(J,I) * VM(I)
      IF( J .NE. M ) SHLY1(J) = SHLY1(J) + Y(J+1,I) * HL(I)
      IF( J .NE. 1 ) SHLX1(J) = SHLX1(J) + X(J-1,I) * HL(I)
      IF( J .NE. M ) SVMY1(J) = SVMY1(J) + Y(J+1,I) * VM(I)
      IF( J .NE. 1 ) SVMX1(J) = SVMX1(J) + X(J-1,I) * VM(I)
   50 CONTINUE
C
C     CALCULATE U, V, L, Q1, QM, GM, AND DXDT
C
      U(M) = F
      DO 120 J=1,M
      IF( J .EQ. 1 ) GO TO 60
      IF( J .EQ. M ) GO TO 100
      GO TO 80
C
```

```
C      CONDENSER
C
   60 U(J) = G(J) / SVM(J)
      U(M) = U(M) - U(J)
      AL(J) = RR * DR
      V(J+1) = (AL(J)+DR)/(1.0D0-SDVMDT(J)/SDKDT(J)/SVM(J)
     1 *(SAKY1(J)-SAK(J))+(SVMY1(J)-SVM(J))/SVM(J))
      Q(J) = -SDHLDT(J)/SDKDT(J)*V(J+1)*(SAKY1(J)-SAK(J))
     1 +V(J+1)*(SHLY1(J)-SHV(J+1))
      DO 70 I=1,N
      DX(J,I) = V(J+1)*(Y(J+1,I)-X(J,I)) / U(J)
   70 CONTINUE
      GO TO 120
C
C      STAGES
C
   80 U(J) = G(J) / SVM(J)
      U(M) = U(M) - U(J)
      V(J+1) = (-SDHLDT(J)/SDKDT(J)*(AL(J-1)*(SAKX1(J)-SAK(J))
     1 -V(J)*(SAKY(J)-SAK(J)))+AL(J-1)*(SHLX1(J)-SHL(J-1))
     2 -V(J)*(SHLY(J)-SHV(J))-Q(J))/(SDHLDT(J)/SDKDT(J)
     3 *(SAKY1(J)-SAK(J))+SHV(J+1)-SHLY1(J))
      AL(J) = V(J+1)+AL(J-1)-V(J)-SDVMDT(J)/SDKDT(J)/SVM(J)
     1 *(V(J+1)*(SAKY1(J)-SAK(J))+AL(J-1)*(SAKX1(J)-SAK(J))
     2 -V(J)*(SAKY(J)-SAK(J)))+(V(J+1)*(SVMY1(J)-SVM(J))
     3 +AL(J-1)*(SVMX1(J)-SVM(J))-V(J)*(SVMY(J)-SVM(J)))/SVM(J)
      DO 90 I=1,N
      DX(J,I) = ( V(J+1)*(Y(J+1,I)-X(J,I)) + AL(J-1)*(X(J-1,I)-X(J,I))
     1 - V(J)*(Y(J,I)-X(J,I)) ) / U(J)
   90 CONTINUE
      GO TO 120
C
C      REBOILER
C
  100 U(J) = U(J) - DR * TIME
      G(J) = U(J) * SVM(J)
      Q(J) = -SDHLDT(J)/SDKDT(J)*(AL(J-1)*(SAKX1(J)-SAK(J))
     1 -V(J)*(SAKY(J)-SAK(J)))+AL(J-1)*(SHLX1(J)-SHL(J-1))
     2 -V(J)*(SHLY(J)-SHV(J))
      DO 110 I=1,N
      DX(J,I) = ( AL(J-1)*(X(J-1,I)-X(J,I)) - V(J)*(Y(J,I)-X(J,I)) )
     1 / U(J)
  110 CONTINUE
  120 CONTINUE
C
C      RETURN DXDT
C
      DO 130 J=1,M
      DO 130 I=1,N
      L = M * ( I - 1 ) + J
      DXDT(L) = DX(J,I)
  130 CONTINUE
      RETURN
      END
      SUBROUTINE JACOB( NEQ, TIME, XT, ML, MU, PD, NROWPD )
C     CALCULATES JACOBIAN FOR LSODE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DATA/
     1 F,          Z(50),     P(40),     G(40),     Q(40),     EM(40,50),
     2 FC,         ZC(50),    CF(6),     CZ(50,6), RR,         DR,
```

```
     3 FCUT,      ZCUT(50),  TCUT(15),  DMAX,     TOL,       NAME(50,3),
     4 MAXIT,     N,         M,         M1, IC,   INITL,     IPRNT
      COMMON /SCALC/
     1 T(40),     AL(40),    V(40),     U(40),    X(40,50),  Y(40,50),
     2 DT(40),    DL(40),    DV(40),    DU(40),   DX(40,50), SUMDX(40),
     3 AK(50),    HV(50),    HL(50),    VM(50),   DKDT(50),  DHVDT(50),
     4 DHLDT(50), DVMDT(50), SCALE,     CONV,     ICONV,     ITER
      COMMON /UCALC/
     1 SAK(40),   SHV(40),   SHL(40),   SVM(40),  SDKDT(40), SDHLDT(40),
     2 SDVMDT(40),SAKY1(40), SAKX1(40), SAKY(40), SHLY1(40), SHLX1(40),
     3 SHLY(40),  SVMY1(40), SVMX1(40), SVMY(40), AKS(40,50),
     4 DTDT(40),  DUDT(40),  DHDT(40)
      DIMENSION XT(NEQ), PD(NROWPD,NEQ)
C
C     CALCULATE JACOBIAN
C
      DO 70 J=1,M
      IF( J .EQ. 1 ) GO TO 10
      IF( J .EQ. M ) GO TO 50
      GO TO 30
C
C     CONDENSER
C
   10 DO 20 I=1,N
      L = M * ( I - 1 ) + J
      PD(2,L) = - V(J+1) / U(J)
      PD(1,L+1) = V(J+1) * EM(J+1,I) * AKS(J+1,I) / U(J)
   20 CONTINUE
      GO TO 70
C
C     STAGES
C
   30 DO 40 I=1,N
      L = M * ( I - 1 ) + J
      PD(3,L-1) = AL(J-1) / U(J)
      PD(2,L) = -(V(J+1)+AL(J-1)+V(J)*(EM(J,I)*AKS(J,I)-1.0D0)) / U(J)
      PD(1,L+1) = (V(J+1)*EM(J+1,I)*AKS(J+1,I)-V(J)*(1.0D0-EM(J,I))
     1            *EM(J+1,I)*AKS(J+1,I)) / U(J)
   40 CONTINUE
      GO TO 70
C
C     REBOILER
C
   50 DO 60 I=1,N
      L = M * ( I - 1 ) + J
      PD(3,L-1) = AL(J-1) / U(J)
      PD(2,L) = -(AL(J-1)+V(J)*(EM(J,I)*AKS(J,I)-1.0D0)) / U(J)
   60 CONTINUE
   70 CONTINUE
      RETURN
      END
      SUBROUTINE UPRINT(TIME)
C     PRINTS UNSTEADY-STATE BATCH DISTILLATION RESULTS
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /FILE/
     1 IIN,          IOUT
      COMMON /DATA/
     1 F,         Z(50),     P(40),     G(40),    Q(40),     EM(40,50),
     2 FC,        ZC(50),    CF(6),     CZ(50,6), RR,        DR,
     3 FCUT,      ZCUT(50),  TCUT(15),  DMAX,     TOL,       NAME(50,3),
```

```
     4 MAXIT,     N,          M,          M1, IC,    INITL,      IPRNT
      COMMON /SCALC/
     1 T(40),     AL(40),     V(40),      U(40),     X(40,50),   Y(40,50),
     2 DT(40),    DL(40),     DV(40),     DU(40),    DX(40,50),  SUMDX(40),
     3 AK(50),    HV(50),     HL(50),     VM(50),    DKDT(50),   DHVDT(50),
     4 DHLDT(50), DVMDT(50),  SCALE,      CONV,      ICONV,      ITER
      COMMON /UCALC/
     1 SAK(40),   SHV(40),    SHL(40),    SVM(40),   SDKDT(40),  SDHLDT(40),
     2 SDVMDT(40),SAKYl(40),  SAKXl(40),  SAKY(40),  SHLYl(40),  SHLXl(40),
     3 SHLY(40),  SVMYl(40),  SVMXl(40),  SVMY(40),  AKS(40,50),
     4 DTDT(40),  DUDT(40),   DHDT(40)
C
C     PRINT UNSTEADY STATE PROFILE
C
      WRITE(IOUT,900) TIME, DR
      WRITE(IOUT,910) (J,T(J),V(J),AL(J),U(J),G(J),Q(J),P(J),J=1,M)
C
      IF( IPRNT .LE. 1 ) RETURN
C
C     PRINT X
C
      WRITE(IOUT,920)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 10 J=1,M
      WRITE(IOUT,1020) J, (X(J,I),I=1,N)
   10 CONTINUE
C
C     PRINT DXDT
C
      WRITE(IOUT,930)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 20 J=1,M
      WRITE(IOUT,1020) J, (DX(J,I),I=1,N)
   20 CONTINUE
C
C     PRINT Y
C
      WRITE(IOUT,940)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
      DO 30 J=1,M
      WRITE(IOUT,1020) J, (Y(J,I),I=1,N)
   30 CONTINUE
C
      IF( IPRNT .LE. 2 ) RETURN
C
C     PRINT THERMODYNAMIC DATA
C
      CALL WIPE ( DTDT(1), 120 )
      IF( IPRNT .LE. 3 ) GO TO 40
      WRITE(IOUT,950)
      WRITE(IOUT,1000) (I,I=1,N)
      WRITE(IOUT,1010) ((NAME(I,K),K=1,3),I=1,N)
   40 DO 60 J=1,M
      CALL THERMO(J,1)
      CALL THERMO(J,2)
      CALL THERMO(J,3)
      CALL THERMO(J,4)
```

```
      IF( IPRNT .LE. 3 ) GO TO 50
      WRITE(IOUT,1020) J, (AK(I),I=1,N)
      WRITE(IOUT,1020) J, (DKDT(I),I=1,N)
      WRITE(IOUT,1020) J, (HV(I),I=1,N)
      WRITE(IOUT,1020) J, (DHVDT(I),I=1,N)
      WRITE(IOUT,1020) J, (HL(I),I=1,N)
      WRITE(IOUT,1020) J, (DHLDT(I),I=1,N)
      WRITE(IOUT,1020) J, (VM(I),I=1,N)
      WRITE(IOUT,1020) J, (DVMDT(I),I=1,N)
   50 DO 60 I=1,N
      DTDT(J) = DTDT(J) + DX(J,I) * AK(I)
      DUDT(J) = DUDT(J) + DX(J,I) * VM(I)
      DHDT(J) = DHDT(J) + DX(J,I) * HL(I)
   60 CONTINUE
C
C     PRINT DERIVATIVES
C
      DO 70 J=1,M
      DTDT(J) = -DTDT(J) / SDKDT(J)
      DUDT(J) = -U(J) / SVM(J) * ( SDVMDT(J) * DTDT(J) + DUDT(J) )
      DHDT(J) = SDHLDT(J) * DTDT(J) + DHDT(J)
   70 CONTINUE
      DUDT(M) = -DR
      DO 80 J=1,M1
      DUDT(M) = DUDT(M) - DUDT(J)
   80 CONTINUE
      WRITE(IOUT,960)
      WRITE(IOUT,970) (J,DTDT(J),DUDT(J),DHDT(J),SAK(J),SHV(J),SHL(J),
     1                 SVM(J),SDKDT(J),SDHLDT(J),SDVMDT(J),J=1,M)
      WRITE(IOUT,980)
      WRITE(IOUT,990) (J,SAKY1(J),SAKX1(J),SAKY(J),SHLY1(J),SHLX1(J),
     1                 SHLY(J),SVMY1(J),SVMX1(J),SVMY(J),J=1,M)
      RETURN
C
  900 FORMAT(1H1/40X,'TIME, HOURS',G12.4
     1         //40X,'DISTILLATE RATE',G12.4
     2 //40X,'U N S T E A D Y - S T A T E   P R O F I L E'
     3 //' STAGE TEMPERATURE  VAPOR RATE LIQUID RATE  MOL HOLDUP  VOL HO
     4LDUP   HEAT DUTY   PRESSURE'/)
  910 FORMAT(I6,7G12.4)
  920 FORMAT(1H1/40X,'L I Q U I D   C O M P O S I T I O N')
  930 FORMAT(  //40X,'D X / D T')
  940 FORMAT(1H1/40X,'V A P O R   C O M P O S I T I O N')
  950 FORMAT(1H1/40X,'T H E R M O D Y N A M I C   D A T A')
  960 FORMAT(1H1/40X,'D E R I V A T I V E S'
     1 //' STAGE   DTDT        DUDT        DHDT        SAK         SHV
     2       SHL         SVM         SDKDT       SDHLDT      SDVMDT'/)
  970 FORMAT(I6,10G12.4)
  980 FORMAT(
     1 //' STAGE   SAKY1       SAKX1       SAKY        SHLY1       SHLX1
     2       SHLY        SVMY1       SVMX1       SVMY'/)
  990 FORMAT(I6,9G12.4)
 1000 FORMAT(/6X,10(I4,8X),4X
     1       /6X,10(I4,8X),4X
     2       /6X,10(I4,8X),4X
     3       /6X,10(I4,8X),4X
     4       /6X,10(I4,8X),4X    )
 1010 FORMAT(' STAGE',3X,30A4,1X
     1       /        9X,30A4,1X
     2       /        9X,30A4,1X
```

```
     3       /        9X,30A4,1X
     4       /        9X,30A4,1X )
 1020 FORMAT(I6,10G12.4,4X
     1      /6X,10G12.4,4X
     2      /6X,10G12.4,4X
     3      /6X,10G12.4,4X
     4      /6X,10G12.4,4X       )
      END
      SUBROUTINE LSODE (F, NEQ, Y, T, TOUT, ITOL, RTOL, ATOL, ITASK,    LS00001
     1            ISTATE, IOPT, RWORK, LRW, IWORK, LIW, JAC, MF)        LS00002
      EXTERNAL F, JAC                                                   LS00003
      INTEGER NEQ, ITOL, ITASK, ISTATE, IOPT, LRW, IWORK, LIW, MF       LS00004
      DOUBLE PRECISION Y, T, TOUT, RTOL, ATOL, RWORK                    LS00005
      DIMENSION NEQ(1), Y(1), RTOL(1), ATOL(1), RWORK(LRW), IWORK(LIW)  LS00006
C-----------------------------------------------------------------------LS00007
C THIS IS THE SEPTEMBER 23, 1980 VERSION OF                             LS00008
C LSODE.. LIVERMORE SOLVER FOR ORDINARY DIFFERENTIAL EQUATIONS.         LS00009
C THIS VERSION IS IN DOUBLE PRECISION.                                  LS00010
C                                                                       LS00011
C LSODE SOLVES THE INITIAL VALUE PROBLEM FOR STIFF OR NONSTIFF          LS00012
C SYSTEMS OF FIRST ORDER ODE-S,                                         LS00013
C     DY/DT = F(T,Y) ,  OR, IN COMPONENT FORM,                          LS00014
C     DY(I)/DT = F(I) = F(I,T,Y(1),Y(2),...,Y(NEQ)) (I = 1,...,NEQ).    LS00015
C LSODE IS A PACKAGE BASED ON THE GEAR AND GEARB PACKAGES, AND ON THE   LS00016
C OCTOBER 23, 1978 VERSION OF THE TENTATIVE ODEPACK USER INTERFACE      LS00017
C STANDARD, WITH MINOR MODIFICATIONS.                                   LS00018
C AUTHOR AND CONTACT.. ALAN C. HINDMARSH,                               LS00019
C                      MATHEMATICS AND STATISTICS SECTION, L-300        LS00020
C                      LAWRENCE LIVERMORE LABORATORY                    LS00021
C                      LIVERMORE, CA 94550.                             LS00022
C-----------------------------------------------------------------------LS00023
C SUMMARY OF USAGE.                                                     LS00024
C                                                                       LS00025
C COMMUNICATION BETWEEN THE USER AND THE LSODE PACKAGE, FOR NORMAL      LS00026
C SITUATIONS, IS SUMMARIZED BELOW.  SEE THE FULL DESCRIPTION FOR        LS00027
C DETAILS, INCLUDING OPTIONAL COMMUNICATION, NONSTANDARD OPTIONS,       LS00028
C AND INSTRUCTIONS FOR SPECIAL SITUATIONS.  SEE ALSO THE EXAMPLE        LS00029
C PROBLEM (WITH PROGRAM AND OUTPUT) FOLLOWING THIS SUMMARY.             LS00030
C                                                                       LS00031
C A. FIRST PROVIDE A SUBROUTINE OF THE FORM..                           LS00032
C               SUBROUTINE F (NEQ, T, Y, YDOT)                          LS00033
C               DIMENSION Y(NEQ), YDOT(NEQ)                             LS00034
C WHICH SUPPLIES THE VECTOR FUNCTION F BY LOADING YDOT(I) WITH F(I).    LS00035
C                                                                       LS00036
C B. NEXT DETERMINE (OR GUESS) WHETHER OR NOT THE PROBLEM IS STIFF.     LS00037
C STIFFNESS OCCURS WHEN THE JACOBIAN MATRIX DF/DY HAS AN EIGENVALUE     LS00038
C WHOSE REAL PART IS NEGATIVE AND LARGE IN MAGNITUDE, COMPARED TO THE   LS00039
C RECIPROCAL OF THE T SPAN OF INTEREST.  IF THE PROBLEM IS NONSTIFF,    LS00040
C USE A METHOD FLAG MF = 10.  IF IT IS STIFF, THERE ARE FOUR STANDARD   LS00041
C CHOICES FOR MF, AND LSODE REQUIRES THE JACOBIAN MATRIX IN SOME FORM.  LS00042
C THIS MATRIX IS REGARDED EITHER AS FULL (MF = 21 OR 22),               LS00043
C OR BANDED (MF = 24 OR 25).  IN THE BANDED CASE, LSODE REQUIRES TWO    LS00044
C HALF-BANDWIDTH PARAMETERS ML AND MU.  THESE ARE, RESPECTIVELY, THE    LS00045
C WIDTHS OF THE LOWER AND UPPER PARTS OF THE BAND, EXCLUDING THE MAIN   LS00046
C DIAGONAL.  THUS THE BAND CONSISTS OF THE LOCATIONS (I,J) WITH         LS00047
C I-ML .LE. J .LE. I+MU, AND THE FULL BANDWIDTH IS ML+MU+1.             LS00048
C                                                                       LS00049
C C. IF THE PROBLEM IS STIFF, YOU ARE ENCOURAGED TO SUPPLY THE JACOBIAN LS00050
C DIRECTLY (MF = 21 OR 24), BUT IF THIS IS NOT FEASIBLE, LSODE WILL     LS00051
C COMPUTE IT INTERNALLY BY DIFFERENCE QUOTIENTS (MF = 22 OR 25).        LS00052
```

```
C IF YOU ARE SUPPLYING THE JACOBIAN, PROVIDE A SUBROUTINE OF THE FORM..
C                SUBROUTINE JAC (NEQ, T, Y, ML, MU, PD, NROWPD)
C                DIMENSION Y(NEQ), PD(NROWPD,NEQ)
C WHICH SUPPLIES DF/DY BY LOADING PD AS FOLLOWS..
C     FOR A FULL JACOBIAN (MF = 21), LOAD PD(I,J) WITH DF(I)/DY(J),
C THE PARTIAL DERIVATIVE OF F(I) WITH RESPECT TO Y(J).  (IGNORE THE
C ML AND MU ARGUMENTS IN THIS CASE.)
C     FOR A BANDED JACOBIAN (MF = 24), LOAD PD(I-J+MU+1,J) WITH
C DF(I)/DY(J), I.E. LOAD THE DIAGONAL LINES OF DF/DY INTO THE ROWS OF
C PD FROM THE TOP DOWN.
C     IN EITHER CASE, ONLY NONZERO ELEMENTS NEED BE LOADED.
C
C D. WRITE A MAIN PROGRAM WHICH CALLS SUBROUTINE LSODE ONCE FOR
C EACH POINT AT WHICH ANSWERS ARE DESIRED.  THIS SHOULD ALSO PROVIDE
C FOR POSSIBLE USE OF LOGICAL UNIT 6 FOR OUTPUT OF ERROR MESSAGES
C BY LSODE.  ON THE FIRST CALL TO LSODE, SUPPLY ARGUMENTS AS FOLLOWS..
C F      = NAME OF SUBROUTINE FOR RIGHT-HAND SIDE VECTOR F.
C          THIS NAME MUST BE DECLARED EXTERNAL IN CALLING PROGRAM.
C NEQ    = NUMBER OF FIRST ORDER ODE-S.
C Y      = ARRAY OF INITIAL VALUES, OF LENGTH NEQ.
C T      = THE INITIAL VALUE OF THE INDEPENDENT VARIABLE.
C TOUT   = FIRST POINT WHERE OUTPUT IS DESIRED (.NE. T).
C ITOL   = 1 OR 2 ACCORDING AS ATOL (BELOW) IS A SCALAR OR ARRAY.
C RTOL   = RELATIVE TOLERANCE PARAMETER (SCALAR).
C ATOL   = ABSOLUTE TOLERANCE PARAMETER (SCALAR OR ARRAY).
C          THE LOCAL ERROR IN Y(I) WILL BE CONTROLLED SO AS TO BE
C          ROUGHLY LESS THAN  RTOL*ABS(Y(I)) + ATOL  IF ITOL = 1,
C          OR  RTOL*ABS(Y(I)) + ATOL(I)  IF ITOL = 2.  USE RTOL = 0.0
C          FOR PURE ABSOLUTE ERROR CONTROL, ATOL (OR ATOL(I)) = 0.0
C          FOR PURE RELATIVE.  CAUTION.. ACTUAL (GLOBAL) ERRORS
C          MAY EXCEED THESE TOLERANCES, SO CHOOSE THEM CONSERVATIVELY.
C ITASK  = 1 FOR NORMAL COMPUTATION OF OUTPUT VALUES OF Y AT T = TOUT.
C ISTATE = INTEGER FLAG (INPUT AND OUTPUT).  SET ISTATE = 1.
C IOPT   = 0 TO INDICATE NO OPTIONAL INPUTS USED.
C RWORK  = REAL WORK ARRAY OF LENGTH AT LEAST..
C             20 + 16*NEQ                    FOR MF = 10,
C             22 +  9*NEQ + NEQ**2           FOR MF = 21 OR 22,
C             22 + 10*NEQ + (2*ML + MU)*NEQ  FOR MF = 24 OR 25.
C LRW    = DECLARED LENGTH OF RWORK (IN USER-S DIMENSION).
C IWORK  = INTEGER WORK ARRAY OF LENGTH AT LEAST..
C             20        FOR MF = 10,
C             20 + NEQ  FOR MF = 21, 22, 24, OR 25.
C          IF MF = 24 OR 25, INPUT IN IWORK(1),IWORK(2) THE LOWER
C          AND UPPER HALF-BANDWIDTHS ML,MU.
C LIW    = DECLARED LENGTH OF IWORK (IN USER-S DIMENSION).
C JAC    = NAME OF SUBROUTINE FOR JACOBIAN MATRIX (MF = 21 OR 24).
C          IF USED, THIS NAME MUST BE DECLARED EXTERNAL IN CALLING
C          PROGRAM.  IF NOT USED, PASS A DUMMY NAME.
C MF     = METHOD FLAG.  STANDARD VALUES ARE..
C          10 FOR NONSTIFF (ADAMS) METHOD, NO JACOBIAN USED.
C          21 FOR STIFF (BDF) METHOD, USER-SUPPLIED FULL JACOBIAN.
C          22 FOR STIFF METHOD, INTERNALLY GENERATED FULL JACOBIAN.
C          24 FOR STIFF METHOD, USER-SUPPLIED BANDED JACOBIAN.
C          25 FOR STIFF METHOD, INTERNALLY GENERATED BANDED JACOBIAN.
C
C E. THE OUTPUT FROM THE FIRST CALL (OR ANY CALL) IS..
C      Y = ARRAY OF COMPUTED VALUES OF Y(T) VECTOR.
C      T = CORRESPONDING VALUE OF INDEPENDENT VARIABLE (NORMALLY TOUT).
C ISTATE = 2  IF LSODE WAS SUCCESSFUL, NEGATIVE OTHERWISE.
C          -1 MEANS EXCESS WORK DONE ON THIS CALL (PERHAPS WRONG MF).
```

```
C          -2 MEANS EXCESS ACCURACY REQUESTED (TOLERANCES TOO SMALL).    LS0011
C          -3 MEANS ILLEGAL INPUT DETECTED (SEE PRINTED MESSAGE).        LS0011
C          -4 MEANS REPEATED ERROR TEST FAILURES (CHECK ALL INPUTS).     LS0011
C          -5 MEANS REPEATED CONVERGENCE FAILURES (PERHAPS BAD JACOBIAN LS0011
C             SUPPLIED OR WRONG CHOICE OF MF OR TOLERANCES).             LS0011
C          -6 MEANS ERROR WEIGHT BECAME ZERO DURING PROBLEM. (SOLUTION   LS0011
C             COMPONENT I VANISHED, AND ATOL OR ATOL(I) = 0.)            LS0011
C                                                                        LS0012
C F. TO CONTINUE THE INTEGRATION AFTER A SUCCESSFUL RETURN, SIMPLY       LS0012
C RESET TOUT AND CALL LSODE AGAIN.  NO OTHER PARAMETERS NEED BE RESET.   LS0012
C                                                                        LS0012
C-----------------------------------------------------------------------LS0012
C EXAMPLE PROBLEM.                                                       LS0012
C                                                                        LS0012
C THE FOLLOWING IS A SIMPLE EXAMPLE PROBLEM, WITH THE CODING             LS0012
C NEEDED FOR ITS SOLUTION BY LSODE.  THE PROBLEM IS FROM CHEMICAL        LS0012
C KINETICS, AND CONSISTS OF THE FOLLOWING THREE RATE EQUATIONS..         LS0012
C     DY1/DT = -.04*Y1 + 1.E4*Y2*Y3                                      LS0013
C     DY2/DT = .04*Y1 - 1.E4*Y2*Y3 - 3.E7*Y2**2                          LS0013
C     DY3/DT = 3.E7*Y2**2                                                LS0013
C ON THE INTERVAL FROM T = 0.0 TO T = 4.E10, WITH INITIAL CONDITIONS     LS0013
C Y1 = 1.0, Y2 = Y3 = 0.  THE PROBLEM IS STIFF.                          LS0013
C                                                                        LS0013
C THE FOLLOWING CODING SOLVES THIS PROBLEM WITH LSODE, USING MF = 21     LS0013
C AND PRINTING RESULTS AT T = .4, 4., ..., 4.E10.  IT USES               LS0013
C ITOL = 2 AND ATOL MUCH SMALLER FOR Y2 THAN Y1 OR Y3 BECAUSE            LS0013
C Y2 HAS MUCH SMALLER VALUES.                                            LS0013
C AT THE END OF THE RUN, STATISTICAL QUANTITIES OF INTEREST ARE          LS0014
C PRINTED (SEE OPTIONAL OUTPUTS IN THE FULL DESCRIPTION BELOW).          LS0014
C                                                                        LS0014
C     EXTERNAL FEX, JEX                                                  LS0014
C     DOUBLE PRECISION ATOL, RWORK, RTOL, T, TOUT, Y                     LS0014
C     DIMENSION Y(3), ATOL(3), RWORK(58), IWORK(23)                      LS0014
C     NEQ = 3                                                            LS0014
C     Y(1) = 1.D0                                                        LS0014
C     Y(2) = 0.D0                                                        LS0014
C     Y(3) = 0.D0                                                        LS0014
C     T = 0.D0                                                           LS0015
C     TOUT = .4D0                                                        LS0015
C     ITOL = 2                                                           LS0015
C     RTOL = 1.D-4                                                       LS0015
C     ATOL(1) = 1.D-6                                                    LS0015
C     ATOL(2) = 1.D-10                                                   LS0015
C     ATOL(3) = 1.D-6                                                    LS0015
C     ITASK = 1                                                          LS0015
C     ISTATE = 1                                                         LS0015
C     IOPT = 0                                                           LS0015
C     LRW = 58                                                           LS0016
C     LIW = 23                                                           LS0016
C     MF = 21                                                            LS0016
C     DO 40 IOUT = 1,12                                                  LS0016
C       CALL LSODE(FEX,NEQ,Y,T,TOUT,ITOL,RTOL,ATOL,ITASK,ISTATE,         LS0016
C    1     IOPT,RWORK,LRW,IWORK,LIW,JEX,MF)                              LS0016
C       WRITE(3,20)T,Y(1),Y(2),Y(3)                                      LS0016
C 20    FORMAT(7H AT T =,E12.4,6H   Y =,3E15.7)                          LS0016
C       IF (ISTATE .LT. 0) GO TO 80                                      LS0016
C 40    TOUT = TOUT*10.D0                                                LS0016
C     WRITE(3,60)IWORK(11),IWORK(12),IWORK(13)                           LS0017
C 60  FORMAT(/12H NO. STEPS =,I4,11H  NO. F-S =,I4,11H  NO. J-S =,I4)    LS0017
C     STOP                                                               LS0017
```

```
C 80   WRITE(3,90)ISTATE                                                    LS0017
C 90   FORMAT(///22H ERROR HALT.. ISTATE =,I3)                              LS0017
C      STOP                                                                 LS0017
C      END                                                                  LS0017
C                                                                           LS0017
C      SUBROUTINE FEX (NEQ, T, Y, YDOT)                                     LS0017
C      DOUBLE PRECISION T, Y, YDOT                                          LS0017
C      DIMENSION Y(3), YDOT(3)                                              LS0018
C      YDOT(1) = -.04D0*Y(1) + 1.D4*Y(2)*Y(3)                               LS0018
C      YDOT(3) = 3.D7*Y(2)*Y(2)                                             LS0018
C      YDOT(2) = -YDOT(1) - YDOT(3)                                         LS0018
C      RETURN                                                               LS0018
C      END                                                                  LS0018
C                                                                           LS0018
C      SUBROUTINE JEX (NEQ, T, Y, ML, MU, PD, NRPD)                         LS0018
C      DOUBLE PRECISION PD, T, Y                                            LS0018
C      DIMENSION Y(3), PD(NRPD,3)                                           LS0018
C      PD(1,1) = -.04D0                                                     LS0019
C      PD(1,2) = 1.D4*Y(3)                                                  LS0019
C      PD(1,3) = 1.D4*Y(2)                                                  LS0019
C      PD(2,1) = .04D0                                                      LS0019
C      PD(2,3) = -PD(1,3)                                                   LS0019
C      PD(3,2) = 6.D7*Y(2)                                                  LS0019
C      PD(2,2) = -PD(1,2) - PD(3,2)                                         LS0019
C      RETURN                                                               LS0019
C      END                                                                  LS0019
C                                                                           LS0019
C THE OUTPUT OF THIS PROGRAM (ON A CDC-7600 IN SINGLE PRECISION)            LS0020
C IS AS FOLLOWS..                                                           LS0020
C                                                                           LS0020
C   AT T =  4.0000E-01   Y =  9.851726E-01  3.386406E-05  1.479357E-02      LS0020
C   AT T =  4.0000E+00   Y =  9.055142E-01  2.240418E-05  9.446344E-02      LS0020
C   AT T =  4.0000E+01   Y =  7.158050E-01  9.184616E-06  2.841858E-01      LS0020
C   AT T =  4.0000E+02   Y =  4.504846E-01  3.222434E-06  5.495122E-01      LS0020
C   AT T =  4.0000E+03   Y =  1.831701E-01  8.940379E-07  8.168290E-01      LS0020
C   AT T =  4.0000E+04   Y =  3.897016E-02  1.621193E-07  9.610297E-01      LS0020
C   AT T =  4.0000E+05   Y =  4.935213E-03  1.983756E-08  9.950648E-01      LS0020
C   AT T =  4.0000E+06   Y =  5.159269E-04  2.064759E-09  9.994841E-01      LS0021
C   AT T =  4.0000E+07   Y =  5.306413E-05  2.122677E-10  9.999469E-01      LS0021
C   AT T =  4.0000E+08   Y =  5.494529E-06  2.197824E-11  9.999945E-01      LS0021
C   AT T =  4.0000E+09   Y =  5.129458E-07  2.051784E-12  9.999995E-01      LS0021
C   AT T =  4.0000E+10   Y = -7.170592E-08 -2.868236E-13  1.000000E+00      LS0021
C                                                                           LS0021
C   NO. STEPS = 330  NO. F-S = 405  NO. J-S =  69                           LS0021
C-----------------------------------------------------------------------    LS0021
C FULL DESCRIPTION OF USER INTERFACE TO LSODE.                              LS0021
C                                                                           LS0021
C THE USER INTERFACE TO LSODE CONSISTS OF THE FOLLOWING PARTS.              LS0022
C                                                                           LS0022
C I.    THE CALL SEQUENCE TO SUBROUTINE LSODE, WHICH IS A DRIVER            LS0022
C       ROUTINE FOR THE SOLVER.  THIS INCLUDES DESCRIPTIONS OF BOTH         LS0022
C       THE CALL SEQUENCE ARGUMENTS AND OF USER-SUPPLIED ROUTINES.          LS0022
C       FOLLOWING THESE DESCRIPTIONS IS A DESCRIPTION OF                    LS0022
C       OPTIONAL INPUTS AVAILABLE THROUGH THE CALL SEQUENCE, AND THEN       LS0022
C       A DESCRIPTION OF OPTIONAL OUTPUTS (IN THE WORK ARRAYS).             LS0022
C                                                                           LS0022
C II.   DESCRIPTIONS OF OTHER ROUTINES IN THE LSODE PACKAGE THAT MAY BE     LS0022
C       (OPTIONALLY) CALLED BY THE USER.  THESE PROVIDE THE ABILITY TO      LS0023
C       ALTER ERROR MESSAGE HANDLING, SAVE AND RESTORE THE INTERNAL         LS0023
C       COMMON, AND OBTAIN SPECIFIED DERIVATIVES OF THE SOLUTION Y(T).      LS0023
```

```
C                                                                       LS00233
C III. DESCRIPTIONS OF COMMON BLOCKS TO BE DECLARED IN OVERLAY          LS00234
C      OR SIMILAR ENVIRONMENTS, OR TO BE SAVED WHEN DOING AN INTERRUPT  LS00235
C      OF THE PROBLEM AND CONTINUED SOLUTION LATER.                     LS00236
C                                                                       LS00237
C IV.  DESCRIPTION OF TWO SUBROUTINES IN THE LSODE PACKAGE, EITHER OF   LS00238
C      WHICH THE USER MAY REPLACE WITH HIS OWN VERSION, IF DESIRED.     LS00239
C      THESE RELATE TO THE MEASUREMENT OF ERRORS.                       LS00240
C                                                                       LS00241
C-----------------------------------------------------------------------LS00242
C PART I.  CALL SEQUENCE.                                               LS00243
C                                                                       LS00244
C THE CALL SEQUENCE PARAMETERS USED FOR INPUT ONLY ARE                  LS00245
C     F, NEQ, TOUT, ITOL, RTOL, ATOL, ITASK, IOPT, LRW, LIW, JAC, MF,   LS00246
C AND THOSE USED FOR BOTH INPUT AND OUTPUT ARE                          LS00247
C     Y, T, ISTATE.                                                     LS00248
C THE WORK ARRAYS RWORK AND IWORK ARE ALSO USED FOR CONDITIONAL AND      LS00249
C OPTIONAL INPUTS AND OPTIONAL OUTPUTS.  (THE TERM OUTPUT HERE REFERS    LS00250
C TO THE RETURN FROM SUBROUTINE LSODE TO THE USER-S CALLING PROGRAM.)    LS00251
C                                                                       LS00252
C THE LEGALITY OF INPUT PARAMETERS WILL BE THOROUGHLY CHECKED ON THE     LS00253
C INITIAL CALL FOR THE PROBLEM, BUT NOT CHECKED THEREAFTER UNLESS A      LS00254
C CHANGE IN INPUT PARAMETERS IS FLAGGED BY ISTATE = 3 ON INPUT.          LS00255
C                                                                       LS00256
C THE DESCRIPTIONS OF THE CALL ARGUMENTS ARE AS FOLLOWS.                LS00257
C                                                                       LS00258
C F      = THE NAME OF THE USER-SUPPLIED SUBROUTINE DEFINING THE        LS00259
C          ODE SYSTEM.  THE SYSTEM MUST BE PUT IN THE FIRST-ORDER       LS00260
C          FORM DY/DT = F(T,Y), WHERE F IS A VECTOR-VALUED FUNCTION     LS00261
C          OF THE SCALAR T AND THE VECTOR Y.  SUBROUTINE F IS TO        LS00262
C          COMPUTE THE FUNCTION F.  IT IS TO HAVE THE FORM              LS00263
C               SUBROUTINE F (NEQ, T, Y, YDOT)                          LS00264
C               DIMENSION Y(1), YDOT(1)                                 LS00265
C          WHERE NEQ, T, AND Y ARE INPUT, AND THE ARRAY YDOT = F(T,Y)   LS00266
C          IS OUTPUT.  Y AND YDOT ARE ARRAYS OF LENGTH NEQ.             LS00267
C          (IN THE DIMENSION STATEMENT ABOVE, 1 IS A DUMMY              LS00268
C          DIMENSION.. IT CAN BE REPLACED BY ANY VALUE.)                LS00269
C          SUBROUTINE F SHOULD NOT ALTER Y(1),...,Y(NEQ).               LS00270
C          F MUST BE DECLARED EXTERNAL IN THE CALLING PROGRAM.          LS00271
C                                                                       LS00272
C          SUBROUTINE F MAY ACCESS USER-DEFINED QUANTITIES IN           LS00273
C          NEQ(2),... AND Y(NEQ(1)+1),... IF NEQ IS AN ARRAY            LS00274
C          (DIMENSIONED IN F) AND Y HAS LENGTH EXCEEDING NEQ(1).        LS00275
C          SEE THE DESCRIPTIONS OF NEQ AND Y BELOW.                     LS00276
C                                                                       LS00277
C NEQ    = THE SIZE OF THE ODE SYSTEM (NUMBER OF FIRST ORDER            LS00278
C          ORDINARY DIFFERENTIAL EQUATIONS).  USED ONLY FOR INPUT.      LS00279
C          NEQ MAY BE DECREASED, BUT NOT INCREASED, DURING THE PROBLEM. LS00280
C          IF NEQ IS DECREASED (WITH ISTATE = 3 ON INPUT), THE          LS00281
C          REMAINING COMPONENTS OF Y SHOULD BE LEFT UNDISTURBED, IF     LS00282
C          THESE ARE TO BE ACCESSED IN F AND/OR JAC.                    LS00283
C                                                                       LS00284
C          NORMALLY, NEQ IS A SCALAR, AND IT IS GENERALLY REFERRED TO   LS00285
C          AS A SCALAR IN THIS USER INTERFACE DESCRIPTION.  HOWEVER,    LS00286
C          NEQ MAY BE AN ARRAY, WITH NEQ(1) SET TO THE SYSTEM SIZE.     LS00287
C          (THE LSODE PACKAGE ACCESSES ONLY NEQ(1).)  IN EITHER CASE,   LS00288
C          THIS PARAMETER IS PASSED AS THE NEQ ARGUMENT IN ALL CALLS    LS00289
C          TO F AND JAC.  HENCE, IF IT IS AN ARRAY, LOCATIONS           LS00290
C          NEQ(2),... MAY BE USED TO STORE OTHER INTEGER DATA AND PASS  LS00291
C          IT TO F AND/OR JAC.  SUBROUTINES F AND/OR JAC MUST INCLUDE   LS00292
```

```
C          NEQ IN A DIMENSION STATEMENT IN THAT CASE.                    LS0029
C                                                                        LS0029
C Y      = A REAL ARRAY FOR THE VECTOR OF DEPENDENT VARIABLES, OF        LS0029
C          LENGTH NEQ OR MORE.  USED FOR BOTH INPUT AND OUTPUT ON THE    LS0029
C          FIRST CALL (ISTATE = 1), AND ONLY FOR OUTPUT ON OTHER CALLS.  LS0029
C          ON THE FIRST CALL, Y MUST CONTAIN THE VECTOR OF INITIAL       LS0029
C          VALUES.  ON OUTPUT, Y CONTAINS THE COMPUTED SOLUTION VECTOR,  LS0029
C          EVALUATED AT T.  IF DESIRED, THE Y ARRAY MAY BE USED          LS0030
C          FOR OTHER PURPOSES BETWEEN CALLS TO THE SOLVER.               LS0030
C                                                                        LS0030
C          THIS ARRAY IS PASSED AS THE Y ARGUMENT IN ALL CALLS TO        LS0030
C          F AND JAC.  HENCE ITS LENGTH MAY EXCEED NEQ, AND LOCATIONS    LS0030
C          Y(NEQ+1),... MAY BE USED TO STORE OTHER REAL DATA AND         LS0030
C          PASS IT TO F AND/OR JAC.  (THE LSODE PACKAGE ACCESSES ONLY    LS0030
C          Y(1),...,Y(NEQ).)                                             LS0030
C                                                                        LS0030
C T      = THE INDEPENDENT VARIABLE.  ON INPUT, T IS USED ONLY ON THE    LS0030
C          FIRST CALL, AS THE INITIAL POINT OF THE INTEGRATION.          LS0031
C          ON OUTPUT, AFTER EACH CALL, T IS THE VALUE AT WHICH A         LS0031
C          COMPUTED SOLUTION Y IS EVALUATED (USUALLY THE SAME AS TOUT).  LS0031
C          ON AN ERROR RETURN, T IS THE FARTHEST POINT REACHED.          LS0031
C                                                                        LS0031
C TOUT   = THE NEXT VALUE OF T AT WHICH A COMPUTED SOLUTION IS DESIRED.  LS0031
C          USED ONLY FOR INPUT.                                          LS0031
C                                                                        LS0031
C          WHEN STARTING THE PROBLEM (ISTATE = 1), TOUT MAY BE EQUAL     LS0031
C          TO T FOR ONE CALL, THEN SHOULD .NE. T FOR THE NEXT CALL.      LS0031
C          FOR THE INITIAL T, AN INPUT VALUE OF TOUT .NE. T IS USED      LS0032
C          IN ORDER TO DETERMINE THE DIRECTION OF THE INTEGRATION        LS0032
C          (I.E. THE ALGEBRAIC SIGN OF THE STEP SIZES) AND THE ROUGH     LS0032
C          SCALE OF THE PROBLEM.  INTEGRATION IN EITHER DIRECTION        LS0032
C          (FORWARD OR BACKWARD IN T) IS PERMITTED.                      LS0032
C                                                                        LS0032
C          IF ITASK = 2 OR 5 (ONE-STEP MODES), TOUT IS IGNORED AFTER     LS0032
C          THE FIRST CALL (I.E. THE FIRST CALL WITH TOUT .NE. T).        LS0032
C          OTHERWISE, TOUT IS REQUIRED ON EVERY CALL.                    LS0032
C                                                                        LS0032
C          IF ITASK = 1, 3, OR 4, THE VALUES OF TOUT NEED NOT BE         LS0033
C          MONOTONE, BUT A VALUE OF TOUT WHICH BACKS UP IS LIMITED       LS0033
C          TO THE CURRENT INTERNAL T INTERVAL, WHOSE ENDPOINTS ARE       LS0033
C          TCUR - HU AND TCUR (SEE OPTIONAL OUTPUTS, BELOW, FOR          LS0033
C          TCUR AND HU).                                                 LS0033
C                                                                        LS0033
C ITOL   = AN INDICATOR FOR THE TYPE OF ERROR CONTROL.  SEE              LS0033
C          DESCRIPTION BELOW UNDER ATOL.  USED ONLY FOR INPUT.           LS0033
C                                                                        LS0033
C RTOL   = A RELATIVE ERROR TOLERANCE PARAMETER, EITHER A SCALAR OR      LS0033
C          AN ARRAY OF LENGTH NEQ.  SEE DESCRIPTION BELOW UNDER ATOL.    LS0034
C          INPUT ONLY.                                                   LS0034
C                                                                        LS0034
C ATOL   = AN ABSOLUTE ERROR TOLERANCE PARAMETER, EITHER A SCALAR OR     LS0034
C          AN ARRAY OF LENGTH NEQ.  INPUT ONLY.                          LS0034
C                                                                        LS0034
C             THE INPUT PARAMETERS ITOL, RTOL, AND ATOL DETERMINE        LS0034
C          THE ERROR CONTROL PERFORMED BY THE SOLVER.  THE SOLVER WILL   LS0034
C          CONTROL THE VECTOR E = (E(I)) OF ESTIMATED LOCAL ERRORS       LS0034
C          IN Y, ACCORDING TO AN INEQUALITY OF THE FORM                  LS0034
C                      RMS-NORM OF ( E(I)/EWT(I) )   .LE.   1,           LS0035
C          WHERE       EWT(I) = RTOL(I)*ABS(Y(I)) + ATOL(I),             LS0035
C          AND THE RMS-NORM (ROOT-MEAN-SQUARE NORM) HERE IS              LS0035
```

```
C          RMS-NORM(V) = SQRT(SUM V(I)**2 / NEQ).  HERE EWT = (EWT(I))  LS00353
C          IS A VECTOR OF WEIGHTS WHICH MUST ALWAYS BE POSITIVE, AND    LS00354
C          THE VALUES OF RTOL AND ATOL SHOULD ALL BE NON-NEGATIVE.      LS00355
C          THE FOLLOWING TABLE GIVES THE TYPES (SCALAR/ARRAY) OF        LS00356
C          RTOL AND ATOL, AND THE CORRESPONDING FORM OF EWT(I).         LS00357
C                                                                       LS00358
C             ITOL    RTOL       ATOL          EWT(I)                  LS00359
C              1      SCALAR     SCALAR     RTOL*ABS(Y(I)) + ATOL       LS00360
C              2      SCALAR     ARRAY      RTOL*ABS(Y(I)) + ATOL(I)    LS00361
C              3      ARRAY      SCALAR     RTOL(I)*ABS(Y(I)) + ATOL    LS00362
C              4      ARRAY      ARRAY      RTOL(I)*ABS(Y(I)) + ATOL(I) LS00363
C                                                                       LS00364
C          WHEN EITHER OF THESE PARAMETERS IS A SCALAR, IT NEED NOT     LS00365
C          BE DIMENSIONED IN THE USER-S CALLING PROGRAM.                LS00366
C                                                                       LS00367
C          IF NONE OF THE ABOVE CHOICES (WITH ITOL, RTOL, AND ATOL      LS00368
C          FIXED THROUGHOUT THE PROBLEM) IS SUITABLE, MORE GENERAL      LS00369
C          ERROR CONTROLS CAN BE OBTAINED BY SUBSTITUTING               LS00370
C          USER-SUPPLIED ROUTINES FOR THE SETTING OF EWT AND/OR FOR     LS00371
C          THE NORM CALCUATION.  SEE PART IV BELOW.                     LS00372
C                                                                       LS00373
C          IF GLOBAL ERRORS ARE TO BE ESTIMATED BY MAKING A REPEATED    LS00374
C          RUN ON THE SAME PROBLEM WITH SMALLER TOLERANCES, THEN ALL    LS00375
C          COMPONENTS OF RTOL AND ATOL (I.E. OF EWT) SHOULD BE SCALED   LS00376
C          DOWN UNIFORMLY.                                              LS00377
C                                                                       LS00378
C ITASK  = AN INDEX SPECIFYING THE TASK TO BE PERFORMED.                LS00379
C          INPUT ONLY.  ITASK HAS THE FOLLOWING VALUES AND MEANINGS.    LS00380
C          1  MEANS NORMAL COMPUTATION OF OUTPUT VALUES OF Y(T) AT       LS00381
C             T = TOUT (BY OVERSHOOTING AND INTERPOLATING).             LS00382
C          2  MEANS TAKE ONE STEP ONLY AND RETURN.                      LS00383
C          3  MEANS STOP AT THE FIRST INTERNAL MESH POINT AT OR         LS00384
C             BEYOND T = TOUT AND RETURN.                               LS00385
C          4  MEANS NORMAL COMPUTATION OF OUTPUT VALUES OF Y(T) AT      LS00386
C             T = TOUT BUT WITHOUT OVERSHOOTING T = TCRIT.              LS00387
C             TCRIT MUST BE INPUT AS RWORK(1).  TCRIT MAY BE EQUAL TO   LS00388
C             OR BEYOND TOUT, BUT NOT BEHIND IT IN THE DIRECTION OF     LS00389
C             INTEGRATION.  THIS OPTION IS USEFUL IF THE PROBLEM        LS00390
C             HAS A SINGULARITY AT OR BEYOND T = TCRIT.                 LS00391
C          5  MEANS TAKE ONE STEP, WITHOUT PASSING TCRIT, AND RETURN.   LS00392
C             TCRIT MUST BE INPUT AS RWORK(1).                          LS00393
C                                                                       LS00394
C          NOTE..  IF ITASK = 4 OR 5 AND THE SOLVER REACHES TCRIT       LS00395
C          (WITHIN ROUNDOFF), IT WILL RETURN T = TCRIT (EXACTLY)        LS00396
C          INDICATE THIS (UNLESS ITASK = 4 AND TOUT COMES BEFORE TCRIT, LS00397
C          IN WHICH CASE ANSWERS AT T = TOUT ARE RETURNED FIRST).       LS00398
C                                                                       LS00399
C ISTATE = AN INDEX USED FOR INPUT AND OUTPUT TO SPECIFY THE            LS00400
C          THE STATE OF THE CALCULATION.                                LS00401
C                                                                       LS00402
C          ON INPUT, THE VALUES OF ISTATE ARE AS FOLLOWS.               LS00403
C          1  MEANS THIS IS THE FIRST CALL FOR THE PROBLEM              LS00404
C             (INITIALIZATIONS WILL BE DONE).  SEE NOTE BELOW.          LS00405
C          2  MEANS THIS IS NOT THE FIRST CALL, AND THE CALCULATION     LS00406
C             IS TO CONTINUE NORMALLY, WITH NO CHANGE IN ANY INPUT      LS00407
C             PARAMETERS EXCEPT POSSIBLY TOUT AND ITASK.                LS00408
C             (IF ITOL, RTOL, AND/OR ATOL ARE CHANGED BETWEEN CALLS     LS00409
C             WITH ISTATE = 2, THE NEW VALUES WILL BE USED BUT NOT      LS00410
C             TESTED FOR LEGALITY.)                                     LS00411
C          3  MEANS THIS IS NOT THE FIRST CALL, AND THE                 LS00412
```

```
C            CALCULATION IS TO CONTINUE NORMALLY, BUT WITH                  LS0041
C            A CHANGE IN INPUT PARAMETERS OTHER THAN                        LS0041
C            TOUT AND ITASK.  CHANGES ARE ALLOWED IN                        LS0041
C            NEQ, ITOL, RTOL, ATOL, IOPT, MF, ML, MU, AND ANY OF THE        LS0041
C            OPTIONAL INPUTS EXCEPT H0.                                     LS0041
C            (SEE IWORK DESCRIPTION FOR ML AND MU.)                         LS0041
C          NOTE..  A PRELIMINARY CALL WITH TOUT = T IS NOT COUNTED          LS0041
C          AS A FIRST CALL HERE, AS NO INITIALIZATION OR CHECKING OF        LS0042
C          INPUT IS DONE.  (SUCH A CALL IS SOMETIMES USEFUL FOR THE         LS0042
C          PURPOSE OF OUTPUTTING THE INITIAL CONDITIONS.)                   LS0042
C          THUS THE FIRST CALL FOR WHICH TOUT .NE. T REQUIRES               LS0042
C          ISTATE = 1 ON INPUT.                                             LS0042
C                                                                           LS0042
C          ON OUTPUT, ISTATE HAS THE FOLLOWING VALUES AND MEANINGS.         LS0042
C            1  MEANS NOTHING WAS DONE, AS TOUT WAS EQUAL TO T WITH         LS0042
C               ISTATE = 1 ON INPUT.  (HOWEVER, AN INTERNAL COUNTER WAS     LS0042
C               SET TO DETECT AND PREVENT REPEATED CALLS OF THIS TYPE.)     LS0042
C            2  MEANS THE INTEGRATION WAS PERFORMED SUCCESSFULLY.           LS0043
C           -1  MEANS AN EXCESSIVE AMOUNT OF WORK (MORE THAN MXSTEP         LS0043
C               STEPS) WAS DONE ON THIS CALL, BEFORE COMPLETING THE         LS0043
C               REQUESTED TASK, BUT THE INTEGRATION WAS OTHERWISE           LS0043
C               SUCCESSFUL AS FAR AS T.  (MXSTEP IS AN OPTIONAL INPUT       LS0043
C               AND IS NORMALLY 500.)  TO CONTINUE, THE USER MAY            LS0043
C               SIMPLY RESET ISTATE TO A VALUE .GT. 1 AND CALL AGAIN        LS0043
C               (THE EXCESS WORK STEP COUNTER WILL BE RESET TO 0).          LS0043
C               IN ADDITION, THE USER MAY INCREASE MXSTEP TO AVOID          LS0043
C               THIS ERROR RETURN (SEE BELOW ON OPTIONAL INPUTS).           LS0043
C           -2  MEANS TOO MUCH ACCURACY WAS REQUESTED FOR THE PRECISION     LS0044
C               OF THE MACHINE BEING USED.  THIS WAS DETECTED BEFORE        LS0044
C               COMPLETING THE REQUESTED TASK, BUT THE INTEGRATION          LS0044
C               WAS SUCCESSFUL AS FAR AS T.  TO CONTINUE, THE TOLERANCE     LS0044
C               PARAMETERS MUST BE RESET, AND ISTATE MUST BE SET            LS0044
C               TO 3.  THE OPTIONAL OUTPUT TOLSF MAY BE USED FOR THIS       LS0044
C               PURPOSE.  (NOTE.. IF THIS CONDITION IS DETECTED BEFORE      LS0044
C               TAKING ANY STEPS, THEN AN ILLEGAL INPUT RETURN              LS0044
C               (ISTATE = -3) OCCURS INSTEAD.)                              LS0044
C           -3  MEANS ILLEGAL INPUT WAS DETECTED, BEFORE TAKING ANY         LS0044
C               INTEGRATION STEPS.  SEE WRITTEN MESSAGE FOR DETAILS.        LS0045
C               NOTE..  IF THE SOLVER DETECTS AN INFINITE LOOP OF CALLS     LS0045
C               TO THE SOLVER WITH ILLEGAL INPUT, IT WILL CAUSE             LS0045
C               THE RUN TO STOP.                                            LS0045
C           -4  MEANS THERE WERE REPEATED ERROR TEST FAILURES ON            LS0045
C               ONE ATTEMPTED STEP, BEFORE COMPLETING THE REQUESTED         LS0045
C               TASK, BUT THE INTEGRATION WAS SUCCESSFUL AS FAR AS T.       LS0045
C               THE PROBLEM MAY HAVE A SINGULARITY, OR THE INPUT            LS0045
C               MAY BE INAPPROPRIATE.                                       LS0045
C           -5  MEANS THERE WERE REPEATED CONVERGENCE TEST FAILURES ON      LS0045
C               ONE ATTEMPTED STEP, BEFORE COMPLETING THE REQUESTED         LS0046
C               TASK, BUT THE INTEGRATION WAS SUCCESSFUL AS FAR AS T.       LS0046
C               THIS MAY BE CAUSED BY AN INACCURATE JACOBIAN MATRIX,        LS0046
C               IF ONE IS BEING USED.                                       LS0046
C           -6  MEANS EWT(I) BECAME ZERO FOR SOME I DURING THE              LS0046
C               INTEGRATION.  PURE RELATIVE ERROR CONTROL (ATOL(I)=0.0)     LS0046
C               WAS REQUESTED ON A VARIABLE WHICH HAS NOW VANISHED.         LS0046
C               THE INTEGRATION WAS SUCCESSFUL AS FAR AS T.                 LS0046
C                                                                           LS0046
C          NOTE..  SINCE THE NORMAL OUTPUT VALUE OF ISTATE IS 2,            LS0046
C          IT DOES NOT NEED TO BE RESET FOR NORMAL CONTINUATION.            LS0047
C          ALSO, SINCE A NEGATIVE INPUT VALUE OF ISTATE WILL BE             LS0047
C          REGARDED AS ILLEGAL, A NEGATIVE OUTPUT VALUE REQUIRES THE        LS0047
```

```
C          USER TO CHANGE IT, AND POSSIBLY OTHER INPUTS, BEFORE          LS0047
C          CALLING THE SOLVER AGAIN.                                     LS0047
C                                                                        LS0047
C IOPT    = AN INTEGER FLAG TO SPECIFY WHETHER OR NOT ANY OPTIONAL       LS0047
C           INPUTS ARE BEING USED ON THIS CALL.  INPUT ONLY.             LS0047
C           THE OPTIONAL INPUTS ARE LISTED SEPARATELY BELOW.             LS0047
C           IOPT = 0 MEANS NO OPTIONAL INPUTS ARE BEING USED.            LS0047
C                    DEFAULT VALUES WILL BE USED IN ALL CASES.           LS0048
C           IOPT = 1 MEANS ONE OR MORE OPTIONAL INPUTS ARE BEING USED.   LS0048
C                                                                        LS0048
C RWORK   = A REAL WORKING ARRAY (DOUBLE PRECISION).                     LS0048
C           THE LENGTH OF RWORK MUST BE AT LEAST                         LS0048
C              20 + NYH*(MAXORD + 1) + 3*NEQ + LWM    WHERE              LS0048
C           NYH    = THE INITIAL VALUE OF NEQ,                           LS0048
C           MAXORD = 12 (IF METH = 1) OR 5 (IF METH = 2) (UNLESS A       LS0048
C                    SMALLER VALUE IS GIVEN AS AN OPTIONAL INPUT),       LS0048
C           LWM    = 0              IF MITER = 0,                        LS0048
C           LWM    = NEQ**2 + 2     IF MITER IS 1 OR 2,                  LS0049
C           LWM    = NEQ + 2        IF MITER = 3, AND                    LS0049
C           LWM    = (2*ML+MU+1)*NEQ + 2 IF MITER IS 4 OR 5.            LS0049
C           (SEE THE MF DESCRIPTION FOR METH AND MITER.)                 LS0049
C           THUS IF MAXORD HAS ITS DEFAULT VALUE AND NEQ IS CONSTANT,    LS0049
C           THIS LENGTH IS..                                             LS0049
C              20 + 16*NEQ                    FOR MF = 10,              LS0049
C              22 + 16*NEQ + NEQ**2           FOR MF = 11 OR 12,        LS0049
C              22 + 17*NEQ                    FOR MF = 13,              LS0049
C              22 + 17*NEQ + (2*ML+MU)*NEQ    FOR MF = 14 OR 15,        LS0049
C              20 +  9*NEQ                    FOR MF = 20,              LS0049
C              22 +  9*NEQ + NEQ**2           FOR MF = 21 OR 22,        LS0050
C              22 + 10*NEQ                    FOR MF = 23,              LS0050
C              22 + 10*NEQ + (2*ML+MU)*NEQ    FOR MF = 24 OR 25.        LS0050
C           THE FIRST 20 WORDS OF RWORK ARE RESERVED FOR CONDITIONAL     LS0050
C           AND OPTIONAL INPUTS AND OPTIONAL OUTPUTS.                    LS0050
C                                                                        LS0050
C           THE FOLLOWING WORD IN RWORK IS A CONDITIONAL INPUT..         LS0050
C             RWORK(1) = TCRIT = CRITICAL VALUE OF T WHICH THE SOLVER    LS0050
C                        IS NOT TO OVERSHOOT.  REQUIRED IF ITASK IS      LS0050
C                        4 OR 5, AND IGNORED OTHERWISE.  (SEE ITASK.)    LS0050
C                                                                        LS0051
C LRW     = THE LENGTH OF THE ARRAY RWORK, AS DECLARED BY THE USER.      LS0051
C           (THIS WILL BE CHECKED BY THE SOLVER.)                        LS0051
C                                                                        LS0051
C IWORK   = AN INTEGER WORK ARRAY.  THE LENGTH OF IWORK MUST BE AT LEAST LS0051
C              20        IF MITER = 0 OR 3 (MF = 10, 13, 20, 23), OR     LS0051
C              20 + NEQ  OTHERWISE (MF = 11, 12, 14, 15, 21, 22, 24, 25).LS0051
C           THE FIRST FEW WORDS OF IWORK ARE USED FOR CONDITIONAL AND    LS0051
C           OPTIONAL INPUTS AND OPTIONAL OUTPUTS.                        LS0051
C                                                                        LS0051
C           THE FOLLOWING 2 WORDS IN IWORK ARE CONDITIONAL INPUTS..      LS0052
C             IWORK(1) = ML     THESE ARE THE LOWER AND UPPER            LS0052
C             IWORK(2) = MU     HALF-BANDWIDTHS, RESPECTIVELY, OF THE    LS0052
C                        BANDED JACOBIAN, EXCLUDING THE MAIN DIAGONAL.   LS0052
C                        THE BAND IS DEFINED BY THE MATRIX LOCATIONS     LS0052
C                        (I,J) WITH I-ML .LE. J .LE. I+MU.  ML AND MU    LS0052
C                        MUST SATISFY  0 .LE.  ML,MU  .LE. NEQ-1.        LS0052
C                        THESE ARE REQUIRED IF MITER IS 4 OR 5, AND      LS0052
C                        IGNORED OTHERWISE.  ML AND MU MAY IN FACT BE    LS0052
C                        THE BAND PARAMETERS FOR A MATRIX TO WHICH       LS0052
C                        DF/DY IS ONLY APPROXIMATELY EQUAL.              LS0053
C                                                                        LS0053
```

```
C LIW     = THE LENGTH OF THE ARRAY IWORK, AS DECLARED BY THE USER.      LS0053
C             (THIS WILL BE CHECKED BY THE SOLVER.)                      LS0053
C                                                                        LS0053
C NOTE..  THE WORK ARRAYS MUST NOT BE ALTERED BETWEEN CALLS TO LSODE     LS0053
C FOR THE SAME PROBLEM, EXCEPT POSSIBLY FOR THE CONDITIONAL AND          LS0053
C OPTIONAL INPUTS, AND EXCEPT FOR THE LAST 3*NEQ WORDS OF RWORK.         LS0053
C THE LATTER SPACE IS USED FOR INTERNAL SCRATCH SPACE, AND SO IS         LS0053
C AVAILABLE FOR USE BY THE USER OUTSIDE LSODE BETWEEN CALLS, IF          LS0053
C DESIRED (BUT NOT FOR USE BY F OR JAC).                                 LS0054
C                                                                        LS0054
C JAC     = THE NAME OF THE USER-SUPPLIED ROUTINE (MITER = 1 OR 4) TO    LS0054
C             COMPUTE THE JACOBIAN MATRIX, DF/DY, AS A FUNCTION OF       LS0054
C             THE SCALAR T AND THE VECTOR Y.  IT IS TO HAVE THE FORM     LS0054
C                  SUBROUTINE JAC (NEQ, T, Y, ML, MU, PD, NROWPD)        LS0054
C                  DIMENSION Y(1), PD(NROWPD,1)                          LS0054
C             WHERE NEQ, T, Y, ML, MU, AND NROWPD ARE INPUT AND THE ARRAY LS0054
C             PD IS TO BE LOADED WITH PARTIAL DERIVATIVES (ELEMENTS OF   LS0054
C             THE JACOBIAN MATRIX) ON OUTPUT.  PD MUST BE GIVEN A FIRST  LS0054
C             DIMENSION OF NROWPD.  T AND Y HAVE THE SAME MEANING AS IN  LS0055
C             SUBROUTINE F.  (IN THE DIMENSION STATEMENT ABOVE, 1 IS A   LS0055
C             DUMMY DIMENSION.. IT CAN BE REPLACED BY ANY VALUE.)        LS0055
C                  IN THE FULL MATRIX CASE (MITER = 1), ML AND MU ARE    LS0055
C             IGNORED, AND THE JACOBIAN IS TO BE LOADED INTO PD IN       LS0055
C             COLUMNWISE MANNER, WITH DF(I)/DY(J) LOADED INTO PD(I,J).   LS0055
C                  IN THE BAND MATRIX CASE (MITER = 4), THE ELEMENTS     LS0055
C             WITHIN THE BAND ARE TO BE LOADED INTO PD IN COLUMNWISE     LS0055
C             MANNER, WITH DIAGONAL LINES OF DF/DY LOADED INTO THE ROWS  LS0055
C             OF PD.  THUS DF(I)/DY(J) IS TO BE LOADED INTO PD(I-J+MU+1,J).LS0055
C             ML AND MU ARE THE HALF-BANDWIDTH PARAMETERS (SEE IWORK).   LS0056
C             THE LOCATIONS IN PD IN THE TWO TRIANGULAR AREAS WHICH      LS0056
C             CORRESPOND TO NONEXISTENT MATRIX ELEMENTS CAN BE IGNORED   LS0056
C             OR LOADED ARBITRARILY, AS THEY ARE OVERWRITTEN BY LSODE.   LS0056
C                  JAC NEED NOT PROVIDE DF/DY EXACTLY.  A CRUDE          LS0056
C             APPROXIMATION (POSSIBLY WITH A SMALLER BANDWIDTH) WILL DO. LS0056
C                  IN EITHER CASE, PD IS PRESET TO ZERO BY THE SOLVER,   LS0056
C             SO THAT ONLY THE NONZERO ELEMENTS NEED BE LOADED BY JAC.   LS0056
C             EACH CALL TO JAC IS PRECEDED BY A CALL TO F WITH THE SAME  LS0056
C             ARGUMENTS NEQ, T, AND Y.  THUS TO GAIN SOME EFFICIENCY,    LS0056
C             INTERMEDIATE QUANTITIES SHARED BY BOTH CALCULATIONS MAY BE LS0057
C             SAVED IN A USER COMMON BLOCK BY F AND NOT RECOMPUTED BY JAC, LS0057
C             IF DESIRED.  ALSO, JAC MAY ALTER THE Y ARRAY, IF DESIRED.  LS0057
C             JAC MUST BE DECLARED EXTERNAL IN THE CALLING PROGRAM.      LS0057
C                  SUBROUTINE JAC MAY ACCESS USER-DEFINED QUANTITIES IN  LS0057
C             NEQ(2),... AND Y(NEQ(1)+1),... IF NEQ IS AN ARRAY          LS0057
C             (DIMENSIONED IN JAC) AND Y HAS LENGTH EXCEEDING NEQ(1).    LS0057
C             SEE THE DESCRIPTIONS OF NEQ AND Y ABOVE.                   LS0057
C                                                                        LS0057
C MF      = THE METHOD FLAG.  USED ONLY FOR INPUT.  THE LEGAL VALUES OF  LS0057
C             MF ARE 10, 11, 12, 13, 14, 15, 20, 21, 22, 23, 24, AND 25. LS0058
C             MF HAS DECIMAL DIGITS METH AND MITER.. MF = 10*METH + MITER. LS0058
C             METH INDICATES THE BASIC LINEAR MULTISTEP METHOD..         LS0058
C               METH = 1 MEANS THE IMPLICIT ADAMS METHOD.                LS0058
C               METH = 2 MEANS THE METHOD BASED ON BACKWARD              LS0058
C                        DIFFERENTIATION FORMULAS (BDF-S).               LS0058
C             MITER INDICATES THE CORRECTOR ITERATION METHOD..           LS0058
C               MITER = 0 MEANS FUNCTIONAL ITERATION (NO JACOBIAN MATRIX LS0058
C                         IS INVOLVED).                                  LS0058
C               MITER = 1 MEANS CHORD ITERATION WITH A USER-SUPPLIED     LS0058
C                         FULL (NEQ BY NEQ) JACOBIAN.                    LS0059
C               MITER = 2 MEANS CHORD ITERATION WITH AN INTERNALLY       LS0059
```

```
C                       GENERATED (DIFFERENCE QUOTIENT) FULL JACOBIAN    LS0059
C                       (USING NEQ EXTRA CALLS TO F PER DF/DY VALUE).    LS0059
C             MITER = 3 MEANS CHORD ITERATION WITH AN INTERNALLY         LS0059
C                       GENERATED DIAGONAL JACOBIAN APPROXIMATION.       LS0059
C                       (USING 1 EXTRA CALL TO F PER DF/DY EVALUATION).  LS0059
C             MITER = 4 MEANS CHORD ITERATION WITH A USER-SUPPLIED       LS0059
C                       BANDED JACOBIAN.                                 LS0059
C             MITER = 5 MEANS CHORD ITERATION WITH AN INTERNALLY         LS0059
C                       GENERATED BANDED JACOBIAN (USING ML+MU+1 EXTRA   LS0060
C                       CALLS TO F PER DF/DY EVALUATION).                LS0060
C           IF MITER = 1 OR 4, THE USER MUST SUPPLY A SUBROUTINE JAC     LS0060
C           (THE NAME IS ARBITRARY) AS DESCRIBED ABOVE UNDER JAC.        LS0060
C           FOR OTHER VALUES OF MITER, A DUMMY ARGUMENT CAN BE USED.     LS0060
C-----------------------------------------------------------------------LS0060
C OPTIONAL INPUTS.                                                       LS0060
C                                                                        LS0060
C THE FOLLOWING IS A LIST OF THE OPTIONAL INPUTS PROVIDED FOR IN THE     LS0060
C CALL SEQUENCE.  (SEE ALSO PART II.)  FOR EACH SUCH INPUT VARIABLE,     LS0060
C THIS TABLE LISTS ITS NAME AS USED IN THIS DOCUMENTATION, ITS           LS0061
C LOCATION IN THE CALL SEQUENCE, ITS MEANING, AND THE DEFAULT VALUE.     LS0061
C THE USE OF ANY OF THESE INPUTS REQUIRES IOPT = 1, AND IN THAT          LS0061
C CASE ALL OF THESE INPUTS ARE EXAMINED.  A VALUE OF ZERO FOR ANY        LS0061
C OF THESE OPTIONAL INPUTS WILL CAUSE THE DEFAULT VALUE TO BE USED.      LS0061
C THUS TO USE A SUBSET OF THE OPTIONAL INPUTS, SIMPLY PRELOAD            LS0061
C LOCATIONS 5 TO 10 IN RWORK AND IWORK TO 0.0 AND 0 RESPECTIVELY, AND    LS0061
C THEN SET THOSE OF INTEREST TO NONZERO VALUES.                          LS0061
C                                                                        LS0061
C NAME    LOCATION      MEANING AND DEFAULT VALUE                        LS0061
C                                                                        LS0062
C H0      RWORK(5)  THE STEP SIZE TO BE ATTEMPTED ON THE FIRST STEP.     LS0062
C                   THE DEFAULT VALUE IS DETERMINED BY THE SOLVER.       LS0062
C                                                                        LS0062
C HMAX    RWORK(6)  THE MAXIMUM ABSOLUTE STEP SIZE ALLOWED.              LS0062
C                   THE DEFAULT VALUE IS INFINITE.                       LS0062
C                                                                        LS0062
C HMIN    RWORK(7)  THE MINIMUM ABSOLUTE STEP SIZE ALLOWED.              LS0062
C                   THE DEFAULT VALUE IS 0.  (THIS LOWER BOUND IS NOT    LS0062
C                   ENFORCED ON THE FINAL STEP BEFORE REACHING TCRIT     LS0062
C                   WHEN ITASK = 4 OR 5.)                                LS0063
C                                                                        LS0063
C MAXORD  IWORK(5)  THE MAXIMUM ORDER TO BE ALLOWED.  THE DEFAULT        LS0063
C                   VALUE IS 12 IF METH = 1, AND 5 IF METH = 2.          LS0063
C                   IF MAXORD EXCEEDS THE DEFAULT VALUE, IT WILL         LS0063
C                   BE REDUCED TO THE DEFAULT VALUE.                     LS0063
C                   IF MAXORD IS CHANGED DURING THE PROBLEM, IT MAY      LS0063
C                   CAUSE THE CURRENT ORDER TO BE REDUCED.               LS0063
C                                                                        LS0063
C MXSTEP  IWORK(6)  MAXIMUM NUMBER OF (INTERNALLY DEFINED) STEPS         LS0063
C                   ALLOWED DURING ONE CALL TO THE SOLVER.               LS0064
C                   THE DEFAULT VALUE IS 500.                            LS0064
C                                                                        LS0064
C MXHNIL  IWORK(7)  MAXIMUM NUMBER OF MESSAGES PRINTED (PER PROBLEM)     LS0064
C                   WARNING THAT T + H = T ON A STEP (H = STEP SIZE).    LS0064
C                   THIS MUST BE POSITIVE TO RESULT IN A NON-DEFAULT     LS0064
C                   VALUE.  THE DEFAULT VALUE IS 10.                     LS0064
C-----------------------------------------------------------------------LS0064
C OPTIONAL OUTPUTS.                                                      LS0064
C                                                                        LS0064
C AS OPTIONAL ADDITIONAL OUTPUT FROM LSODE, THE VARIABLES LISTED         LS0065
C BELOW ARE QUANTITIES RELATED TO THE PERFORMANCE OF LSODE               LS0065
```

```
C WHICH ARE AVAILABLE TO THE USER.  THESE ARE COMMUNICATED BY WAY OF
C THE WORK ARRAYS, BUT ALSO HAVE INTERNAL MNEMONIC NAMES AS SHOWN.
C EXCEPT WHERE STATED OTHERWISE, ALL OF THESE OUTPUTS ARE DEFINED
C ON ANY SUCCESSFUL RETURN FROM LSODE, AND ON ANY RETURN WITH
C ISTATE = -1, -2, -4, -5, OR -6.  ON AN ILLEGAL INPUT RETURN
C (ISTATE = -3), THEY WILL BE UNCHANGED FROM THEIR EXISTING VALUES
C (IF ANY), EXCEPT POSSIBLY FOR TOLSF, LENRW, AND LENIW.
C ON ANY ERROR RETURN, OUTPUTS RELEVANT TO THE ERROR WILL BE DEFINED,
C AS NOTED BELOW.
C
C NAME    LOCATION      MEANING
C
C HU      RWORK(11) THE STEP SIZE IN T LAST USED (SUCCESSFULLY).
C
C HCUR    RWORK(12) THE STEP SIZE TO BE ATTEMPTED ON THE NEXT STEP.
C
C TCUR    RWORK(13) THE CURRENT VALUE OF THE INDEPENDENT VARIABLE
C                   WHICH THE SOLVER HAS ACTUALLY REACHED, I.E. THE
C                   CURRENT INTERNAL MESH POINT IN T.  ON OUTPUT, TCUR
C                   WILL ALWAYS BE AT LEAST AS FAR AS THE ARGUMENT
C                   T, BUT MAY BE FARTHER (IF INTERPOLATION WAS DONE).
C
C TOLSF   RWORK(14) A TOLERANCE SCALE FACTOR, GREATER THAN 1.0,
C                   COMPUTED WHEN A REQUEST FOR TOO MUCH ACCURACY WAS
C                   DETECTED (ISTATE = -3 IF DETECTED AT THE START OF
C                   THE PROBLEM, ISTATE = -2 OTHERWISE).  IF ITOL IS
C                   LEFT UNALTERED BUT RTOL AND ATOL ARE UNIFORMLY
C                   SCALED UP BY A FACTOR OF TOLSF FOR THE NEXT CALL,
C                   THEN THE SOLVER IS DEEMED LIKELY TO SUCCEED.
C                   (THE USER MAY ALSO IGNORE TOLSF AND ALTER THE
C                   TOLERANCE PARAMETERS IN ANY OTHER WAY APPROPRIATE.)
C
C NST     IWORK(11) THE NUMBER OF STEPS TAKEN FOR THE PROBLEM SO FAR.
C
C NFE     IWORK(12) THE NUMBER OF F EVALUATIONS FOR THE PROBLEM SO FAR.
C
C NJE     IWORK(13) THE NUMBER OF JACOBIAN EVALUATIONS (AND OF MATRIX
C                   LU DECOMPOSITIONS) FOR THE PROBLEM SO FAR.
C
C NQU     IWORK(14) THE METHOD ORDER LAST USED (SUCCESSFULLY).
C
C NQCUR   IWORK(15) THE ORDER TO BE ATTEMPTED ON THE NEXT STEP.
C
C IMXER   IWORK(16) THE INDEX OF THE COMPONENT OF LARGEST MAGNITUDE IN
C                   THE WEIGHTED LOCAL ERROR VECTOR ( E(I)/EWT(I) ),
C                   ON AN ERROR RETURN WITH ISTATE = -4 OR -5.
C
C LENRW   IWORK(17) THE LENGTH OF RWORK ACTUALLY REQUIRED.
C                   THIS IS DEFINED ON NORMAL RETURNS AND ON AN ILLEGAL
C                   INPUT RETURN FOR INSUFFICIENT STORAGE.
C
C LENIW   IWORK(18) THE LENGTH OF IWORK ACTUALLY REQUIRED.
C                   THIS IS DEFINED ON NORMAL RETURNS AND ON AN ILLEGAL
C                   INPUT RETURN FOR INSUFFICIENT STORAGE.
C
C THE FOLLOWING TWO ARRAYS ARE SEGMENTS OF THE RWORK ARRAY WHICH
C MAY ALSO BE OF INTEREST TO THE USER AS OPTIONAL OUTPUTS.
C FOR EACH ARRAY, THE TABLE BELOW GIVES ITS INTERNAL NAME,
C ITS BASE ADDRESS IN RWORK, AND ITS DESCRIPTION.
C
```

```
C NAME    BASE ADDRESS      DESCRIPTION                                          LS00712
C                                                                                LS00713
C YH      21                THE NORDSIECK HISTORY ARRAY, OF SIZE NYH BY        LS00714
C                           (NQCUR + 1), WHERE NYH IS THE INITIAL VALUE        LS00715
C                           OF NEQ.  FOR J = 0,1,...,NQCUR, COLUMN J+1         LS00716
C                           OF YH CONTAINS HCUR**J/FACTORIAL(J) TIMES          LS00717
C                           THE J-TH DERIVATIVE OF THE INTERPOLATING           LS00718
C                           POLYNOMIAL CURRENTLY REPRESENTING THE SOLUTION,    LS00719
C                           EVALUATED AT T = TCUR.                             LS00720
C                                                                              LS00721
C ACOR     LENRW-NEQ+1      ARRAY OF SIZE NEQ USED FOR THE ACCUMULATED         LS00722
C                           CORRECTIONS ON EACH STEP, SCALED ON OUTPUT          LS00723
C                           TO REPRESENT THE ESTIMATED LOCAL ERROR IN Y        LS00724
C                           ON THE LAST STEP.  THIS IS THE VECTOR E IN         LS00725
C                           THE DESCRIPTION OF THE ERROR CONTROL.  IT IS       LS00726
C                           DEFINED ONLY ON A SUCCESSFUL RETURN FROM LSODE.    LS00727
C                                                                              LS00728
C-----------------------------------------------------------------------       LS00729
C PART II.  OTHER ROUTINES CALLABLE.                                           LS00730
C                                                                              LS00731
C THE FOLLOWING ARE OPTIONAL CALLS WHICH THE USER MAY MAKE TO                  LS00732
C GAIN ADDITIONAL CAPABILITIES IN CONJUNCTION WITH LSODE.                      LS00733
C (THE ROUTINES XSETUN AND XSETF ARE DESIGNED TO CONFORM TO THE                LS00734
C SLATEC ERROR HANDLING PACKAGE.)                                              LS00735
C                                                                              LS00736
C     FORM OF CALL                  FUNCTION                                   LS00737
C   CALL XSETUN(LUN)            SET THE LOGICAL UNIT NUMBER, LUN, FOR          LS00738
C                               OUTPUT OF MESSAGES FROM LSODE, IF              LS00739
C                               THE DEFAULT IS NOT DESIRED.                    LS00740
C                               THE DEFAULT VALUE OF LUN IS 6.                 LS00741
C                                                                              LS00742
C   CALL XSETF(MFLAG)           SET A FLAG TO CONTROL THE PRINTING OF          LS00743
C                               MESSAGES BY LSODE.                             LS00744
C                               MFLAG = 0 MEANS DO NOT PRINT. (DANGER..        LS00745
C                               THIS RISKS LOSING VALUABLE INFORMATION.)       LS00746
C                               MFLAG = 1 MEANS PRINT (THE DEFAULT).           LS00747
C                                                                              LS00748
C                               EITHER OF THE ABOVE CALLS MAY BE MADE AT       LS00749
C                               ANY TIME AND WILL TAKE EFFECT IMMEDIATELY.     LS00750
C                                                                              LS00751
C   CALL SVCOM (RSAV, ISAV)     STORE IN RSAV AND ISAV THE CONTENTS            LS00752
C                               OF THE INTERNAL COMMON BLOCKS USED BY          LS00753
C                               LSODE (SEE PART III BELOW).                    LS00754
C                               RSAV MUST BE A REAL ARRAY OF LENGTH 218        LS00755
C                               OR MORE, AND ISAVE MUST BE AN INTEGER          LS00756
C                               ARRAY OF LENGTH 35 OR MORE.                    LS00757
C                                                                              LS00758
C   CALL RSCOM (RSAV, ISAV)     RESTORE, FROM RSAV AND ISAV, THE CONTENTS      LS00759
C                               OF THE INTERNAL COMMON BLOCKS USED BY          LS00760
C                               LSODE.  PRESUMES A PRIOR CALL TO SVCOM         LS00761
C                               WITH THE SAME ARGUMENTS.                       LS00762
C                                                                              LS00763
C                               SVCOM AND RSCOM ARE USEFUL IF                  LS00764
C                               INTERRUPTING A RUN AND RESTARTING              LS00765
C                               LATER, OR ALTERNATING BETWEEN TWO OR           LS00766
C                               MORE PROBLEMS SOLVED WITH LSODE.               LS00767
C                                                                              LS00768
C   CALL INTDY(,,,,,)           PROVIDE DERIVATIVES OF Y, OF VARIOUS           LS00769
C        (SEE BELOW)            ORDERS, AT A SPECIFIED POINT T, IF             LS00770
C                               DESIRED.  IT MAY BE CALLED ONLY AFTER          LS00771
```

```
C                                A SUCCESSFUL RETURN FROM LSODE.          LS00772
C                                                                         LS00773
C THE DETAILED INSTRUCTIONS FOR USING INTDY ARE AS FOLLOWS.               LS00774
C THE FORM OF THE CALL IS..                                               LS00775
C                                                                         LS00776
C   CALL INTDY (T, K, RWORK(21), NYH, DKY, IFLAG)                         LS00777
C                                                                         LS00778
C THE INPUT PARAMETERS ARE..                                              LS00779
C                                                                         LS00780
C T         = VALUE OF INDEPENDENT VARIABLE WHERE ANSWERS ARE DESIRED     LS00781
C             (NORMALLY THE SAME AS THE T LAST RETURNED BY LSODE).        LS00782
C             FOR VALID RESULTS, T MUST LIE BETWEEN TCUR - HU AND TCUR.   LS00783
C             (SEE OPTIONAL OUTPUTS FOR TCUR AND HU.)                     LS00784
C K         = INTEGER ORDER OF THE DERIVATIVE DESIRED.  K MUST SATISFY    LS00785
C             0 .LE. K .LE. NQCUR, WHERE NQCUR IS THE CURRENT ORDER       LS00786
C             (SEE OPTIONAL OUTPUTS).  THE CAPABILITY CORRESPONDING       LS00787
C             TO K = 0, I.E. COMPUTING Y(T), IS ALREADY PROVIDED          LS00788
C             BY LSODE DIRECTLY.  SINCE NQCUR .GE. 1, THE FIRST           LS00789
C             DERIVATIVE DY/DT IS ALWAYS AVAILABLE WITH INTDY.            LS00790
C RWORK(21) = THE BASE ADDRESS OF THE HISTORY ARRAY YH.                   LS00791
C NYH       = COLUMN LENGTH OF YH, EQUAL TO THE INITIAL VALUE OF NEQ.     LS00792
C                                                                         LS00793
C THE OUTPUT PARAMETERS ARE..                                             LS00794
C                                                                         LS00795
C DKY       = A REAL ARRAY OF LENGTH NEQ CONTAINING THE COMPUTED VALUE    LS00796
C             OF THE K-TH DERIVATIVE OF Y(T).                             LS00797
C IFLAG     = INTEGER FLAG, RETURNED AS 0 IF K AND T WERE LEGAL,          LS00798
C             -1 IF K WAS ILLEGAL, AND -2 IF T WAS ILLEGAL.               LS00799
C             ON AN ERROR RETURN, A MESSAGE IS ALSO WRITTEN.              LS00800
C-----------------------------------------------------------------------LS00801
C PART III.  COMMON BLOCKS.                                               LS00802
C                                                                         LS00803
C IF LSODE IS TO BE USED IN AN OVERLAY SITUATION, THE USER                LS00804
C MUST DECLARE, IN THE PRIMARY OVERLAY, THE VARIABLES IN..                LS00805
C   (1) THE CALL SEQUENCE TO LSODE,                                       LS00806
C   (2) THE TWO INTERNAL COMMON BLOCKS                                    LS00807
C         /LS0001/  OF LENGTH  251  (218 DOUBLE PRECISION WORDS           LS00808
C                        FOLLOWED BY 33 INTEGER WORDS),                   LS00809
C         /EH0001/  OF LENGTH  2 (INTEGER WORDS).                         LS00810
C                                                                         LS00811
C IF LSODE IS USED ON A SYSTEM IN WHICH THE CONTENTS OF INTERNAL          LS00812
C COMMON BLOCKS ARE NOT PRESERVED BETWEEN CALLS, THE USER SHOULD          LS00813
C DECLARE THE ABOVE TWO COMMON BLOCKS IN HIS MAIN PROGRAM TO INSURE       LS00814
C THAT THEIR CONTENTS ARE PRESERVED.                                      LS00815
C                                                                         LS00816
C IF THE SOLUTION OF A GIVEN PROBLEM BY LSODE IS TO BE INTERRUPTED        LS00817
C AND THEN LATER CONTINUED, SUCH AS WHEN RESTARTING AN INTERRUPTED RUN    LS00818
C OR ALTERNATING BETWEEN TWO OR MORE PROBLEMS, THE USER SHOULD SAVE,      LS00819
C FOLLOWING THE RETURN FROM THE LAST LSODE CALL PRIOR TO THE              LS00820
C INTERRUPTION, THE CONTENTS OF THE CALL SEQUENCE VARIABLES AND THE       LS00821
C INTERNAL COMMON BLOCKS, AND LATER RESTORE THESE VALUES BEFORE THE       LS00822
C NEXT LSODE CALL FOR THAT PROBLEM.  TO SAVE AND RESTORE THE COMMON       LS00823
C BLOCKS, USE SUBROUTINES SVCOM AND RSCOM (SEE PART II ABOVE).            LS00824
C                                                                         LS00825
C-----------------------------------------------------------------------LS00826
C PART IV.  OPTIONALLY REPLACEABLE SOLVER ROUTINES.                       LS00827
C                                                                         LS00828
C BELOW ARE DESCRIPTIONS OF TWO ROUTINES IN THE LSODE PACKAGE WHICH       LS00829
C RELATE TO THE MEASUREMENT OF ERRORS.  EITHER ROUTINE CAN BE             LS00830
C REPLACED BY A USER-SUPPLIED VERSION, IF DESIRED.  HOWEVER, SINCE SUCH   LS00831
```

```
C A REPLACEMENT MAY HAVE A MAJOR IMPACT ON PERFORMANCE, IT SHOULD BE    LS0083
C DONE ONLY WHEN ABSOLUTELY NECESSARY, AND ONLY WITH GREAT CAUTION.     LS0083
C (NOTE.. THE MEANS BY WHICH THE PACKAGE VERSION OF A ROUTINE IS        LS0083
C SUPERSEDED BY THE USER-S VERSION MAY BE SYSTEM-DEPENDENT.)            LS0083
C                                                                       LS0083
C (A) EWSET.                                                            LS0083
C THE FOLLOWING SUBROUTINE IS CALLED JUST BEFORE EACH INTERNAL          LS0083
C INTEGRATION STEP, AND SETS THE ARRAY OF ERROR WEIGHTS, EWT, AS        LS0083
C DESCRIBED UNDER ITOL/RTOL/ATOL ABOVE..                                LS0084
C     SUBROUTINE EWSET (NEQ, ITOL, RTOL, ATOL, YCUR, EWT)               LS0084
C WHERE NEQ, ITOL, RTOL, AND ATOL ARE AS IN THE LSODE CALL SEQUENCE,    LS0084
C YCUR CONTAINS THE CURRENT DEPENDENT VARIABLE VECTOR, AND              LS0084
C EWT IS THE ARRAY OF WEIGHTS SET BY EWSET.                             LS0084
C                                                                       LS0084
C IF THE USER SUPPLIES THIS SUBROUTINE, IT MUST RETURN IN EWT(I)        LS0084
C (I = 1,...,NEQ) A POSITIVE QUANTITY SUITABLE FOR COMPARING ERRORS     LS0084
C IN Y(I) TO.  THE EWT ARRAY RETURNED BY EWSET IS PASSED TO THE         LS0085
C VNORM ROUTINE (SEE BELOW), AND ALSO USED BY LSODE IN THE COMPUTATION  LS0085
C OF THE OPTIONAL OUTPUT IMXER, THE DIAGONAL JACOBIAN APPROXIMATION,    LS0085
C AND THE INCREMENTS FOR DIFFERENCE QUOTIENT JACOBIANS.                 LS0085
C                                                                       LS0085
C IN THE USER-SUPPLIED VERSION OF EWSET, IT MAY BE DESIRABLE TO USE     LS0085
C THE CURRENT VALUES OF DERIVATIVES OF Y.  DERIVATIVES UP TO ORDER NQ   LS0085
C ARE AVAILABLE FROM THE HISTORY ARRAY YH, DESCRIBED ABOVE UNDER        LS0085
C OPTIONAL OUTPUTS.  IN EWSET, YH IS IDENTICAL TO THE YCUR ARRAY,       LS0085
C EXTENDED TO NQ + 1 COLUMNS WITH A COLUMN LENGTH OF NYH AND SCALE      LS0085
C FACTORS OF H**J/FACTORIAL(J).  ON THE FIRST CALL FOR THE PROBLEM,     LS0086
C GIVEN BY NST = 0, NQ IS 1 AND H IS TEMPORARILY SET TO 1.0.            LS0086
C THE QUANTITIES NQ, NYH, H, AND NST CAN BE OBTAINED BY INCLUDING       LS0086
C IN EWSET THE STATEMENTS..                                             LS0086
C     DOUBLE PRECISION H, RLS                                           LS0086
C     COMMON /LS0001/ RLS(218),ILS(33)                                  LS0086
C     NQ = ILS(29)                                                      LS0086
C     NYH = ILS(14)                                                     LS0086
C     NST = ILS(30)                                                     LS0086
C     H = RLS(213)                                                      LS0086
C THUS, FOR EXAMPLE, THE CURRENT VALUE OF DY/DT CAN BE OBTAINED AS      LS0087
C YCUR(NYH+I)/H  (I=1,...,NEQ)  (AND THE DIVISION BY H IS               LS0087
C UNNECESSARY WHEN NST = 0).                                            LS0087
C                                                                       LS0087
C (B) VNORM.                                                            LS0087
C THE FOLLOWING IS A REAL FUNCTION ROUTINE WHICH COMPUTES THE WEIGHTED  LS0087
C ROOT-MEAN-SQUARE NORM OF A VECTOR V..                                 LS0087
C     D = VNORM (N, V, W)                                               LS0087
C WHERE..                                                               LS0087
C   N = THE LENGTH OF THE VECTOR,                                       LS0087
C   V = REAL ARRAY OF LENGTH N CONTAINING THE VECTOR,                   LS0088
C   W = REAL ARRAY OF LENGTH N CONTAINING WEIGHTS,                      LS0088
C   D = SQRT( (1/N) * SUM(V(I)/W(I))**2 ).                              LS0088
C VNORM IS CALLED WITH N = NEQ AND A WEIGHT VECTOR OF W = EWT,          LS0088
C AS SET BY SUBROUTINE EWSET.                                           LS0088
C                                                                       LS0088
C IF THE USER SUPPLIES THIS FUNCTION, IT SHOULD RETURN A NON-NEGATIVE   LS0088
C VALUE OF VNORM SUITABLE FOR USE IN THE ERROR CONTROL IN LSODE.        LS0088
C THE WEIGHT ARRAY W MAY BE USED AS NEEDED, BUT SHOULD NOT BE ALTERED.  LS0088
C FOR EXAMPLE, A USER-SUPPLIED VNORM ROUTINE MIGHT..                    LS0088
C   -SUBSTITUTE A MAX-NORM OF (V(I)/W(I)) FOR THE RMS-NORM, OR          LS0089
C   -IGNORE SOME COMPONENTS OF V IN THE NORM, WITH THE EFFECT OF        LS0089
C    SUPPRESSING THE ERROR CONTROL ON THOSE COMPONENTS OF Y.            LS0089
C-----------------------------------------------------------------------LS0089
```

```
C-----------------------------------------------------------------------LS0089
C OTHER ROUTINES IN THE LSODE PACKAGE.                                  LS0089
C                                                                       LS0089
C IN ADDITION TO SUBROUTINE LSODE, THE LSODE PACKAGE INCLUDES THE       LS0089
C FOLLOWING SUBROUTINES AND FUNCTION ROUTINES..                         LS0089
C  INTDY    COMPUTES AN INTERPOLATED VALUE OF THE Y VECTOR AT T = TOUT. LS0089
C  STODE    IS THE CORE INTEGRATOR, WHICH DOES ONE STEP OF THE          LS0090
C           INTEGRATION AND THE ASSOCIATED ERROR CONTROL.               LS0090
C  CFODE    SETS ALL METHOD COEFFICIENTS AND TEST CONSTANTS.            LS0090
C  PREPJ    COMPUTES AND PREPROCESSES THE JACOBIAN MATRIX J = DF/DY     LS0090
C           AND THE NEWTON ITERATION MATRIX P = I - H*L0*J.             LS0090
C  SOLSY    MANAGES SOLUTION OF LINEAR SYSTEM IN CHORD ITERATION.       LS0090
C  EWSET    SETS THE ERROR WEIGHT VECTOR EWT BEFORE EACH STEP.          LS0090
C  VNORM    COMPUTES THE WEIGHTED R.M.S. NORM OF A VECTOR.              LS0090
C  SVCOM AND RSCOM   ARE USER-CALLABLE ROUTINES TO SAVE AND RESTORE,    LS0090
C           RESPECTIVELY, THE CONTENTS OF THE INTERNAL COMMON BLOCKS.   LS0090
C  DGEFA AND DGESL   ARE ROUTINES FROM LINPACK FOR SOLVING FULL         LS0091
C           SYSTEMS OF LINEAR ALGEBRAIC EQUATIONS.                      LS0091
C  DGBFA AND DGBSL   ARE ROUTINES FROM LINPACK FOR SOLVING BANDED       LS0091
C           LINEAR SYSTEMS.                                             LS0091
C  DAXPY, DSCAL, IDAMAX, AND DDOT   ARE BASIC LINEAR ALGEBRA MODULES    LS0091
C           (BLAS) USED BY THE ABOVE LINPACK ROUTINES.                  LS0091
C  D1MACH   COMPUTES THE UNIT ROUNDOFF IN A MACHINE-INDEPENDENT MANNER. LS0091
C  XERRWV, XSETUN, AND XSETF   HANDLE THE PRINTING OF ALL ERROR         LS0091
C           MESSAGES AND WARNINGS.  XERRWV IS MACHINE-DEPENDENT.        LS0091
C NOTE..  VNORM, IDAMAX, DDOT, AND D1MACH ARE FUNCTION ROUTINES.        LS0091
C ALL THE OTHERS ARE SUBROUTINES.                                       LS0092
C                                                                       LS0092
C THE INTRINSIC AND EXTERNAL ROUTINES USED BY LSODE ARE..               LS0092
C DABS, DMAX1, DMIN1, DFLOAT, MAX0, MIN0, MOD, DSIGN, DSQRT, AND WRITE. LS0092
C                                                                       LS0092
C A BLOCK DATA SUBPROGRAM IS ALSO INCLUDED WITH THE PACKAGE,            LS0092
C FOR LOADING SOME OF THE VARIABLES IN INTERNAL COMMON.                 LS0092
C                                                                       LS0092
C-----------------------------------------------------------------------LS0092
      EXTERNAL PREPJ, SOLSY                                             LS0092
      INTEGER I, I1, I2, IER, IFLAG, ILLIN, IMXER, INIT, IOWNS,         LS0093
     1   JSTART, KFLAG, KGO, L, LACOR, LENIW, LENRW, LENWM, LEWT, LF0,  LS0093
     2   LIWM, LSAVF, LWM, LYH, MAXORD, METH, MITER, ML, MORD, MU,      LS0093
     3   MXHNIL, MXHNL0, MXSTEP, MXSTP0, N, NFE, NHNIL, NJE, NQ, NQU,   LS0093
     4   NSLAST, NST, NTREP, NYH                                        LS0093
      DOUBLE PRECISION TRET, ROWNS, EL0, H, HMIN, HMXI, HU, TN, UROUND, LS0093
     1   ATOLI, AYI, BIG, EWTI, H0, HMAX, HMX, RH, RTOLI, TCRIT, TDIST, LS0093
     2   TNEXT, TOL, TOLSF, TP, SIZE, SUM, W0, D1MACH, VNORM            LS0093
      DIMENSION MORD(2)                                                 LS0093
      LOGICAL IHIT                                                      LS0093
C-----------------------------------------------------------------------LS0094
C THE FOLLOWING INTERNAL COMMON BLOCK CONTAINS                          LS0094
C (A) VARIABLES WHICH ARE LOCAL TO ANY SUBROUTINE BUT WHOSE VALUES MUST LS0094
C     BE PRESERVED BETWEEN CALLS TO THE ROUTINE (OWN VARIABLES), AND    LS0094
C (B) VARIABLES WHICH ARE COMMUNICATED BETWEEN SUBROUTINES.             LS0094
C THE STRUCTURE OF THE BLOCK IS AS FOLLOWS..  ALL REAL VAIABLES ARE     LS0094
C LISTED FIRST, FOLLOWED BY ALL INTEGERS.  WITHIN EACH TYPE, THE        LS0094
C VARIABLES ARE GROUPED WITH THOSE LOCAL TO SUBROUTINE LSODE FIRST,     LS0094
C THEN THOSE LOCAL TO SUBROUTINE STODE, AND FINALLY THOSE USED          LS0094
C FOR COMMUNICATION.  THE BLOCK IS DECLARED IN SUBROUTINES              LS0094
C LSODE, INTDY, STODE, PREPJ, AND SOLSY.  GROUPS OF VARIABLES ARE       LS0095
C REPLACED BY DUMMY ARRAYS IN THE COMMON DECLARATIONS IN ROUTINES       LS0095
C WHERE THOSE VARIABLES ARE NOT USED.                                   LS0095
C-----------------------------------------------------------------------LS0095
```

```
      COMMON /LS0001/ TRET, ROWNS(210),                                 LS0095
     1   EL0, H, HMIN, HMXI, HU, TN, UROUND,                            LS0095
     2   ILLIN, INIT, LYH, LEWT, LACOR, LSAVF, LWM, LIWM,               LS0095
     3   MXSTEP, MXHNIL, NHNIL, NTREP, NSLAST, NYH, IOWNS(6),           LS0095
     4   IER, JSTART, KFLAG, L, METH, MITER, MAXORD, N, NQ, NST, NFE,   LS0095
     5   NJE, NQU                                                       LS0095
C                                                                       LS0096
      DATA  MORD(1),MORD(2)/12,5/, MXSTP0/500/, MXHNL0/10/              LS0096
C-----------------------------------------------------------------------LS0096
C BLOCK A.                                                              LS0096
C THIS CODE BLOCK IS EXECUTED ON EVERY CALL.                            LS0096
C IT TESTS ISTATE AND ITASK FOR LEGALITY AND BRANCHES APPROPIATELY.     LS0096
C IF ISTATE .GT. 1 BUT THE FLAG INIT SHOWS THAT INITIALIZATION HAS      LS0096
C NOT YET BEEN DONE, AN ERROR RETURN OCCURS.                            LS0096
C IF ISTATE = 1 AND TOUT = T, JUMP TO BLOCK G AND RETURN IMMEDIATELY.   LS0096
C-----------------------------------------------------------------------LS0096
      IF (ISTATE .LT. 1 .OR. ISTATE .GT. 3) GO TO 601                   LS0097
      IF (ITASK .LT. 1 .OR. ITASK .GT. 5) GO TO 602                     LS0097
      IF (ISTATE .EQ. 1) GO TO 10                                       LS0097
      IF (INIT .EQ. 0) GO TO 603                                        LS0097
      IF (ISTATE .EQ. 2) GO TO 200                                      LS0097
      GO TO 20                                                          LS0097
 10   INIT = 0                                                          LS0097
      IF (TOUT .EQ. T) GO TO 430                                        LS0097
 20   NTREP = 0                                                         LS0097
C-----------------------------------------------------------------------LS0097
C BLOCK B.                                                              LS0098
C THE NEXT CODE BLOCK IS EXECUTED FOR THE INITIAL CALL (ISTATE = 1),    LS0098
C OR FOR A CONTINUATION CALL WITH PARAMETER CHANGES (ISTATE = 3).       LS0098
C IT CONTAINS CHECKING OF ALL INPUTS AND VARIOUS INITIALIZATIONS.       LS0098
C                                                                       LS0098
C FIRST CHECK LEGALITY OF THE NON-OPTIONAL INPUTS NEQ, ITOL, IOPT,      LS0098
C MF, ML, AND MU.                                                       LS0098
C-----------------------------------------------------------------------LS0098
      IF (NEQ(1) .LE. 0) GO TO 604                                      LS0098
      IF (ISTATE .EQ. 1) GO TO 25                                       LS0098
      IF (NEQ(1) .GT. N) GO TO 605                                      LS0099
 25   N = NEQ(1)                                                        LS0099
      IF (ITOL .LT. 1 .OR. ITOL .GT. 4) GO TO 606                       LS0099
      IF (IOPT .LT. 0 .OR. IOPT .GT. 1) GO TO 607                       LS0099
      METH = MF/10                                                      LS0099
      MITER = MF - 10*METH                                              LS0099
      IF (METH .LT. 1 .OR. METH .GT. 2) GO TO 608                       LS0099
      IF (MITER .LT. 0 .OR. MITER .GT. 5) GO TO 608                     LS0099
      IF (MITER .LE. 3) GO TO 30                                        LS0099
      ML = IWORK(1)                                                     LS0099
      MU = IWORK(2)                                                     LS0100
      IF (ML .LT. 0 .OR. ML .GE. N) GO TO 609                           LS0100
      IF (MU .LT. 0 .OR. MU .GE. N) GO TO 610                           LS0100
 30   CONTINUE                                                          LS0100
C NEXT PROCESS AND CHECK THE OPTIONAL INPUTS. ------------------------LS0100
      IF (IOPT .EQ. 1) GO TO 40                                         LS0100
      MAXORD = MORD(METH)                                               LS0100
      MXSTEP = MXSTP0                                                   LS0100
      MXHNIL = MXHNL0                                                   LS0100
      IF (ISTATE .EQ. 1) H0 = 0.0D0                                     LS0100
      HMXI = 0.0D0                                                      LS0101
      HMIN = 0.0D0                                                      LS0101
      GO TO 60                                                          LS0101
 40   MAXORD = IWORK(5)                                                 LS0101
```

```
      IF (MAXORD .LT. 0) GO TO 611                                      LS0101
      IF (MAXORD .EQ. 0) MAXORD = 100                                   LS0101
      MAXORD = MIN0(MAXORD,MORD(METH))                                  LS0101
      MXSTEP = IWORK(6)                                                 LS0101
      IF (MXSTEP .LT. 0) GO TO 612                                      LS0101
      IF (MXSTEP .EQ. 0) MXSTEP = MXSTP0                                LS0101
      MXHNIL = IWORK(7)                                                 LS0102
      IF (MXHNIL .LT. 0) GO TO 613                                      LS0102
      IF (MXHNIL .EQ. 0) MXHNIL = MXHNL0                                LS0102
      IF (ISTATE .NE. 1) GO TO 50                                       LS0102
      H0 = RWORK(5)                                                     LS0102
      IF ((TOUT - T)*H0 .LT. 0.0D0) GO TO 614                           LS0102
 50   HMAX = RWORK(6)                                                   LS0102
      IF (HMAX .LT. 0.0D0) GO TO 615                                    LS0102
      HMXI = 0.0D0                                                      LS0102
      IF (HMAX .GT. 0.0D0) HMXI = 1.0D0/HMAX                            LS0102
      HMIN = RWORK(7)                                                   LS0103
      IF (HMIN .LT. 0.0D0) GO TO 616                                    LS0103
C-----------------------------------------------------------------------LS0103
C SET WORK ARRAY POINTERS AND CHECK LENGTHS LRW AND LIW.                 LS0103
C POINTERS TO SEGMENTS OF RWORK AND IWORK ARE NAMED BY PREFIXING L TO    LS0103
C THE NAME OF THE SEGMENT.  E.G., THE SEGMENT YH STARTS AT RWORK(LYH).   LS0103
C SEGMENTS OF RWORK (IN ORDER) ARE DENOTED  YH, WM, EWT, SAVF, ACOR.     LS0103
C-----------------------------------------------------------------------LS0103
 60   LYH = 21                                                          LS0103
      IF (ISTATE .EQ. 1) NYH = N                                        LS0103
      LWM = LYH + (MAXORD + 1)*NYH                                      LS0104
      IF (MITER .EQ. 0) LENWM = 0                                       LS0104
      IF (MITER .EQ. 1 .OR. MITER .EQ. 2) LENWM = N*N + 2               LS0104
      IF (MITER .EQ. 3) LENWM = N + 2                                   LS0104
      IF (MITER .GE. 4) LENWM = (2*ML + MU + 1)*N + 2                   LS0104
      LEWT = LWM + LENWM                                                LS0104
      LSAVF = LEWT + N                                                  LS0104
      LACOR = LSAVF + N                                                 LS0104
      LENRW = LACOR + N - 1                                             LS0104
      IWORK(17) = LENRW                                                 LS0104
      LIWM = 1                                                          LS0105
      LENIW = 20 + N                                                    LS0105
      IF (MITER .EQ. 0 .OR. MITER .EQ. 3) LENIW = 20                    LS0105
      IWORK(18) = LENIW                                                 LS0105
      IF (LENRW .GT. LRW) GO TO 617                                     LS0105
      IF (LENIW .GT. LIW) GO TO 618                                     LS0105
C CHECK RTOL AND ATOL FOR LEGALITY. -----------------------------------LS0105
      RTOLI = RTOL(1)                                                   LS0105
      ATOLI = ATOL(1)                                                   LS0105
      DO 70 I = 1,N                                                     LS0105
        IF (ITOL .GE. 3) RTOLI = RTOL(I)                                LS0106
        IF (ITOL .EQ. 2 .OR. ITOL .EQ. 4) ATOLI = ATOL(I)               LS0106
        IF (RTOLI .LT. 0.0D0) GO TO 619                                 LS0106
        IF (ATOLI .LT. 0.0D0) GO TO 620                                 LS0106
 70     CONTINUE                                                        LS0106
      IF (ISTATE .EQ. 1) GO TO 100                                      LS0106
C IF ISTATE = 3, SET FLAG TO SIGNAL PARAMETER CHANGES TO STODE. --------LS0106
      JSTART = -1                                                       LS0106
      IF (NQ .LE. MAXORD) GO TO 90                                      LS0106
C MAXORD WAS REDUCED BELOW NQ.  COPY YH(*,MAXORD+2) INTO SAVF. ---------LS0106
      DO 80 I = 1,N                                                     LS0107
 80     RWORK(I+LSAVF-1) = RWORK(I+LWM-1)                               LS0107
C RELOAD WM(1) = RWORK(LWM), SINCE LWM MAY HAVE CHANGED. --------------LS01071A
 90   RWORK(LWM) = DSQRT(UROUND)                                        LS01071B
```

```
      IF (N .EQ. NYH) GO TO 200                                          LS0107
C NEQ WAS REDUCED.  ZERO PART OF YH TO AVOID UNDEFINED REFERENCES. -----LS0107
      I1 = LYH + L*NYH                                                   LS0107
      I2 = LYH + (MAXORD + 1)*NYH - 1                                    LS0107
      IF (I1 .GT. I2) GO TO 200                                          LS0107
      DO 95 I = I1,I2                                                    LS0107
 95     RWORK(I) = 0.0D0                                                 LS0107
      GO TO 200                                                          LS0107
C-----------------------------------------------------------------------LS0108
C BLOCK C.                                                              LS0108
C THE NEXT BLOCK IS FOR THE INITIAL CALL ONLY (ISTATE = 1).             LS0108
C IT CONTAINS ALL REMAINING INITIALIZATIONS, THE INITIAL CALL TO F,     LS0108
C AND THE CALCULATION OF THE INITIAL STEP SIZE.                         LS0108
C-----------------------------------------------------------------------LS0108
 100  UROUND = D1MACH(4)                                                LS0108
      TN = T                                                            LS0108
      IF (ITASK .NE. 4 .AND. ITASK .NE. 5) GO TO 110                    LS0108
      TCRIT = RWORK(1)                                                  LS0108
      IF ((TCRIT - TOUT)*(TOUT - T) .LT. 0.0D0) GO TO 625               LS0109
      IF (H0 .NE. 0.0D0 .AND. (T + H0 - TCRIT)*H0 .GT. 0.0D0)           LS0109
     1   H0 = TCRIT - T                                                 LS0109
 110  JSTART = 0                                                        LS0109
      RWORK(LWM) = DSQRT(UROUND)                                        LS0109
      NHNIL = 0                                                         LS0109
      NST = 0                                                           LS0109
      NJE = 0                                                           LS0109
      NSLAST = 0                                                        LS0109
      HU = 0.0D0                                                        LS0109
      NQU = 0                                                           LS0109
C INITIAL CALL TO F.  (LF0 POINTS TO YH(*,2). -------------------------LS0109
      LF0 = LYH + NYH                                                   LS0110
      CALL F (NEQ, T, Y, RWORK(LF0))                                    LS0110
      NFE = 1                                                           LS0110
C LOAD THE INITIAL VALUE VECTOR IN YH. --------------------------------LS0110
      DO 115 I = 1,N                                                    LS0110
 115    RWORK(I+LYH-1) = Y(I)                                           LS0110
C LOAD THE ERROR WEIGHT VECTOR EWT.  (H IS TEMPORARILY SET TO 1.0.) ----LS0110
      NQ = 1                                                            LS0110
      H = 1.0D0                                                         LS0110
      CALL EWSET (N, ITOL, RTOL, ATOL, RWORK(LYH), RWORK(LEWT))         LS0110
      DO 120 I = 1,N                                                    LS0111
        IF (RWORK(I+LEWT-1) .LE. 0.0D0) GO TO 621                       LS0111
 120  CONTINUE                                                          LS0111
C-----------------------------------------------------------------------LS0111
C THE CODING BELOW COMPUTES THE STEP SIZE, H0, TO BE ATTEMPTED ON THE   LS0111
C FIRST STEP, UNLESS THE USER HAS SUPPLIED A VALUE FOR THIS.            LS0111
C FIRST CHECK THAT TOUT - T DIFFERS SIGNIFICANTLY FROM ZERO.            LS0111
C A SCALAR TOLERANCE QUANTITY TOL IS COMPUTED, AS MAX(RTOL(I))          LS0111
C IF THIS IS POSITIVE, OR MAX(ATOL(I)/ABS(Y(I))) OTHERWISE, ADJUSTED    LS0111
C SO AS TO BE BETWEEN 100*UROUND AND 1.0E-3.                            LS0111
C THEN THE COMPUTED VALUE H0 IS GIVEN BY..                              LS0112
C                                     NEQ                               LS0112
C   H0**2 = TOL / ( W0**-2 + (1/NEQ) * SUM ( F(I)/YWT(I) )**2  )        LS0112
C                                      1                                LS0112
C WHERE   W0     = MAX ( ABS(T), ABS(TOUT) ),                           LS0112
C         F(I)   = I-TH COMPONENT OF INITIAL VALUE OF F,                LS0112
C         YWT(I) = EWT(I)/TOL  (A WEIGHT FOR Y(I)).                     LS0112
C THE SIGN OF H0 IS INFERRED FROM THE INITIAL VALUES OF TOUT AND T.     LS0112
C-----------------------------------------------------------------------LS0112
      IF (H0 .NE. 0.0D0) GO TO 180                                      LS0112
```

```
      TDIST = DABS(TOUT - T)                                            LS0113
      W0 = DMAX1(DABS(T),DABS(TOUT))                                    LS0113
      IF (TDIST .LT. 2.0D0*UROUND*W0) GO TO 622                         LS0113
      TOL = RTOL(1)                                                     LS0113
      IF (ITOL .LE. 2) GO TO 140                                        LS0113
      DO 130 I = 1,N                                                    LS0113
 130    TOL = DMAX1(TOL,RTOL(I))                                        LS0113
 140  IF (TOL .GT. 0.0D0) GO TO 160                                     LS0113
      ATOLI = ATOL(1)                                                   LS0113
      DO 150 I = 1,N                                                    LS0113
        IF (ITOL .EQ. 2 .OR. ITOL .EQ. 4) ATOLI = ATOL(I)               LS0114
        AYI = DABS(Y(I))                                                LS0114
        IF (AYI .NE. 0.0D0) TOL = DMAX1(TOL,ATOLI/AYI)                  LS0114
 150    CONTINUE                                                        LS0114
 160  TOL = DMAX1(TOL,100.0D0*UROUND)                                   LS0114
      TOL = DMIN1(TOL,0.001D0)                                          LS0114
      SUM = VNORM (N, RWORK(LF0), RWORK(LEWT))                          LS0114
      SUM = 1.0D0/(TOL*W0*W0) + TOL*SUM**2                              LS0114
      H0 = 1.0D0/DSQRT(SUM)                                             LS0114
      H0 = DMIN1(H0,TDIST)                                              LS0114
      H0 = DSIGN(H0,TOUT-T)                                             LS0115
C ADJUST H0 IF NECESSARY TO MEET HMAX BOUND. ----------------------------LS0115
 180  RH = DABS(H0)*HMXI                                                LS0115
      IF (RH .GT. 1.0D0) H0 = H0/RH                                     LS0115
C LOAD H WITH H0 AND SCALE YH(*,2) BY H0. -------------------------------LS0115
      H = H0                                                            LS0115
      DO 190 I = 1,N                                                    LS0115
 190    RWORK(I+LF0-1) = H0*RWORK(I+LF0-1)                              LS0115
      GO TO 270                                                         LS0115
C-------------------------------------------------------------------------LS0115
C BLOCK D.                                                              LS0116
C THE NEXT CODE BLOCK IS FOR CONTINUATION CALLS ONLY (ISTATE = 2 OR 3)  LS0116
C AND IS TO CHECK STOP CONDITIONS BEFORE TAKING A STEP.                 LS0116
C-------------------------------------------------------------------------LS0116
 200  NSLAST = NST                                                      LS0116
      GO TO (210, 250, 220, 230, 240), ITASK                            LS0116
 210  IF ((TN - TOUT)*H .LT. 0.0D0) GO TO 250                           LS0116
      CALL INTDY (TOUT, 0, RWORK(LYH), NYH, Y, IFLAG)                   LS0116
      IF (IFLAG .NE. 0) GO TO 627                                       LS0116
      T = TOUT                                                          LS0116
      GO TO 420                                                         LS0117
 220  TP = TN - HU*(1.0D0 + 100.0D0*UROUND)                             LS0117
      IF ((TP - TOUT)*H .GT. 0.0D0) GO TO 623                           LS0117
      IF ((TN - TOUT)*H .LT. 0.0D0) GO TO 250                           LS0117
      GO TO 400                                                         LS0117
 230  TCRIT = RWORK(1)                                                  LS0117
      IF ((TN - TCRIT)*H .GT. 0.0D0) GO TO 624                          LS0117
      IF ((TCRIT - TOUT)*H .LT. 0.0D0) GO TO 625                        LS0117
      IF ((TN - TOUT)*H .LT. 0.0D0) GO TO 245                           LS0117
      CALL INTDY (TOUT, 0, RWORK(LYH), NYH, Y, IFLAG)                   LS0117
      IF (IFLAG .NE. 0) GO TO 627                                       LS0118
      T = TOUT                                                          LS0118
      GO TO 420                                                         LS0118
 240  TCRIT = RWORK(1)                                                  LS0118
      IF ((TN - TCRIT)*H .GT. 0.0D0) GO TO 624                          LS0118
 245  HMX = DABS(TN) + DABS(H)                                          LS0118
      IHIT = DABS(TN - TCRIT) .LE. 100.0D0*UROUND*HMX                   LS0118
      IF (IHIT) GO TO 400                                               LS0118
      TNEXT = TN + H*(1.0D0 + 4.0D0*UROUND)                             LS0118
      IF ((TNEXT - TCRIT)*H .LE. 0.0D0) GO TO 250                       LS0118
```

```
      H = (TCRIT - TN)*(1.0D0 - 4.0D0*UROUND)                          LS0119(
      IF (ISTATE .EQ. 2) JSTART = -2                                   LS0119)
C-----------------------------------------------------------------------LS0119:
C BLOCK E.                                                             LS0119;
C THE NEXT BLOCK IS NORMALLY EXECUTED FOR ALL CALLS AND CONTAINS       LS0119<
C THE CALL TO THE ONE-STEP CORE INTEGRATOR STODE.                      LS0119=
C                                                                      LS0119>
C THIS IS A LOOPING POINT FOR THE INTEGRATION STEPS.                   LS01197
C                                                                      LS01198
C FIRST CHECK FOR TOO MANY STEPS BEING TAKEN, UPDATE EWT (IF NOT AT    LS01199
C START OF PROBLEM), CHECK FOR TOO MUCH ACCURACY BEING REQUESTED, AND  LS01200
C CHECK FOR H BELOW THE ROUNDOFF LEVEL IN T.                           LS01201
C-----------------------------------------------------------------------LS01202
 250  CONTINUE                                                         LS01203
      IF ((NST-NSLAST) .GE. MXSTEP) GO TO 500                          LS01204
      CALL EWSET (N, ITOL, RTOL, ATOL, RWORK(LYH), RWORK(LEWT))        LS01205
      DO 260 I = 1,N                                                   LS01206
        IF (RWORK(I+LEWT-1) .LE. 0.0D0) GO TO 510                      LS01207
 260    CONTINUE                                                       LS01208
 270  TOLSF = UROUND*VNORM (N, RWORK(LYH), RWORK(LEWT))                LS01209
      IF (TOLSF .LE. 1.0D0) GO TO 280                                  LS01210
      TOLSF = TOLSF*2.0D0                                              LS01211
      IF (NST .EQ. 0) GO TO 626                                        LS01212
      GO TO 520                                                        LS01213
 280  IF ((TN + H) .NE. TN) GO TO 290                                  LS01214
      NHNIL = NHNIL + 1                                                LS01215(
      IF (NHNIL .GT. MXHNIL) GO TO 290                                 LS01216(
      CALL XERRWV(50HLSODE--  WARNING..INTERNAL T (=R1) AND H (=R2) ARE,LS01217(
     1   50, 101, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                         LS01218(
      CALL XERRWV(                                                     LS01219(
     1  60H      SUCH THAT IN THE MACHINE, T + H = T ON THE NEXT STEP  ,LS01220(
     1   60, 101, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                         LS01221(
      CALL XERRWV(50H      (H = STEP SIZE). SOLVER WILL CONTINUE ANYWAY,LS01222(
     1   50, 101, 1, 0, 0, 0, 2, TN, H)                                LS01223(
      IF (NHNIL .LT. MXHNIL) GO TO 290                                 LS01224(
      CALL XERRWV(50HLSODE--  ABOVE WARNING HAS BEEN ISSUED I1 TIMES.  ,LS01225(
     1   50, 102, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                         LS01226(
      CALL XERRWV(50H      IT WILL NOT BE ISSUED AGAIN FOR THIS PROBLEM,LS01227(
     1   50, 102, 1, 1, MXHNIL, 0, 0, 0.0D0, 0.0D0)                    LS01228(
 290  CONTINUE                                                         LS01229(
C-----------------------------------------------------------------------LS01230(
C     CALL STODE(NEQ,Y,YH,NYH,YH,EWT,SAVF,ACOR,WM,IWM,F,JAC,PREPJ,SOLSY)LS01231(
C-----------------------------------------------------------------------LS01232(
      CALL STODE (NEQ, Y, RWORK(LYH), NYH, RWORK(LYH), RWORK(LEWT),    LS01233(
     1   RWORK(LSAVF), RWORK(LACOR), RWORK(LWM), IWORK(LIWM),          LS01234(
     2   F, JAC, PREPJ, SOLSY)                                         LS01235(
      KGO = 1 - KFLAG                                                  LS01236(
      GO TO (300, 530, 540), KGO                                       LS01237(
C-----------------------------------------------------------------------LS01238(
C BLOCK F.                                                             LS01239(
C THE FOLLOWING BLOCK HANDLES THE CASE OF A SUCCESSFUL RETURN FROM THE LS01240(
C CORE INTEGRATOR (KFLAG = 0).  TEST FOR STOP CONDITIONS.              LS01241(
C-----------------------------------------------------------------------LS01242(
 300  INIT = 1                                                         LS01243(
      GO TO (310, 400, 330, 340, 350), ITASK                           LS01244(
C ITASK = 1.  IF TOUT HAS BEEN REACHED, INTERPOLATE. -------------------LS01245(
 310  IF ((TN - TOUT)*H .LT. 0.0D0) GO TO 250                          LS01246(
      CALL INTDY (TOUT, 0, RWORK(LYH), NYH, Y, IFLAG)                  LS01247(
      T = TOUT                                                         LS01248(
      GO TO 420                                                        LS01249(
```

```
C ITASK = 3.  JUMP TO EXIT IF TOUT WAS REACHED. ----------------------LS0125
 330  IF ((TN - TOUT)*H .GE. 0.0D0) GO TO 400                          LS0125
      GO TO 250                                                        LS0125
C ITASK = 4.  SEE IF TOUT OR TCRIT WAS REACHED.  ADJUST H IF NECESSARY. LS0125
 340  IF ((TN - TOUT)*H .LT. 0.0D0) GO TO 345                          LS0125
      CALL INTDY (TOUT, 0, RWORK(LYH), NYH, Y, IFLAG)                  LS0125
      T = TOUT                                                         LS0125
      GO TO 420                                                        LS0125
 345  HMX = DABS(TN) + DABS(H)                                         LS0125
      IHIT = DABS(TN - TCRIT) .LE. 100.0D0*UROUND*HMX                  LS0125
      IF (IHIT) GO TO 400                                              LS0126
      TNEXT = TN + H*(1.0D0 + 4.0D0*UROUND)                            LS0126
      IF ((TNEXT - TCRIT)*H .LE. 0.0D0) GO TO 250                      LS0126
      H = (TCRIT - TN)*(1.0D0 - 4.0D0*UROUND)                          LS0126
      JSTART = -2                                                      LS0126
      GO TO 250                                                        LS0126
C ITASK = 5.  SEE IF TCRIT WAS REACHED AND JUMP TO EXIT. --------------LS0126
 350  HMX = DABS(TN) + DABS(H)                                         LS0126
      IHIT = DABS(TN - TCRIT) .LE. 100.0D0*UROUND*HMX                  LS0126
C-----------------------------------------------------------------------LS0126
C BLOCK G.                                                             LS0127
C THE FOLLOWING BLOCK HANDLES ALL SUCCESSFUL RETURNS FROM LSODE.       LS0127
C IF ITASK .NE. 1, Y IS LOADED FROM YH AND T IS SET ACCORDINGLY.       LS0127
C ISTATE IS SET TO 2, THE ILLEGAL INPUT COUNTER IS ZEROED, AND THE     LS0127
C OPTIONAL OUTPUTS ARE LOADED INTO THE WORK ARRAYS BEFORE RETURNING.   LS0127
C IF ISTATE = 1 AND TOUT = T, THERE IS A RETURN WITH NO ACTION TAKEN,  LS0127
C EXCEPT THAT IF THIS HAS HAPPENED REPEATEDLY, THE RUN IS TERMINATED.  LS0127
C-----------------------------------------------------------------------LS0127
 400  DO 410 I = 1,N                                                   LS0127
 410    Y(I) = RWORK(I+LYH-1)                                          LS0127
      T = TN                                                           LS0128
      IF (ITASK .NE. 4 .AND. ITASK .NE. 5) GO TO 420                   LS0128
      IF (IHIT) T = TCRIT                                              LS0128
 420  ISTATE = 2                                                       LS0128
      ILLIN = 0                                                        LS0128
      RWORK(11) = HU                                                   LS0128
      RWORK(12) = H                                                    LS0128
      RWORK(13) = TN                                                   LS0128
      IWORK(11) = NST                                                  LS0128
      IWORK(12) = NFE                                                  LS0128
      IWORK(13) = NJE                                                  LS0129
      IWORK(14) = NQU                                                  LS0129
      IWORK(15) = NQ                                                   LS0129
      RETURN                                                           LS0129
C                                                                      LS0129
 430  NTREP = NTREP + 1                                                LS0129
      IF (NTREP .LT. 5) RETURN                                         LS0129
      CALL XERRWV(                                                     LS0129
     1  60HLSODE--  REPEATED CALLS WITH ISTATE = 1 AND TOUT = T (=R1)  ,LS0129
     1   60, 301, 1, 0, 0, 0, 1, T, 0.0D0)                            LS0129
      GO TO 800                                                        LS0130
C-----------------------------------------------------------------------LS0130
C BLOCK H.                                                             LS0130
C THE FOLLOWING BLOCK HANDLES ALL UNSUCCESSFUL RETURNS OTHER THAN      LS0130
C THOSE FOR ILLEGAL INPUT.  FIRST THE ERROR MESSAGE ROUTINE IS CALLED. LS0130
C IF THERE WAS AN ERROR TEST OR CONVERGENCE TEST FAILURE, IMXER IS SET. LS0130
C THEN Y IS LOADED FROM YHIS, T IS SET TO TN, AND THE ILLEGAL INPUT    LS0130
C COUNTER ILLIN IS SET TO 0.  THE OPTIONAL OUTPUTS ARE LOADED INTO     LS0130
C THE WORK ARRAYS BEFORE RETURNING.                                    LS0130
C-----------------------------------------------------------------------LS0130
```

```
C THE MAXIMUM NUMBER OF STEPS WAS TAKEN BEFORE REACHING TOUT. ----------LS0131
 500  CALL XERRWV(50HLSODE--  AT CURRENT T (=R1), MXSTEP (=I1) STEPS   ,LS0131
     1   50, 201, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                          LS0131
      CALL XERRWV(50H      TAKEN ON THIS CALL BEFORE REACHING TOUT     ,LS0131
     1   50, 201, 1, 1, MXSTEP, 0, 1, TN, 0.0D0)                        LS0131
      ISTATE = -1                                                       LS0131
      GO TO 580                                                         LS0131
C EWT(I) .LE. 0.0 FOR SOME I (NOT AT START OF PROBLEM). ----------------LS0131
 510  EWTI = RWORK(LEWT+I-1)                                            LS0131
      CALL XERRWV(50HLSODE--  AT T (=R1), EWT(I1) HAS BECOME R2 .LE. 0.,LS0131
     1   50, 202, 1, 1, I, 0, 2, TN, EWTI)                             LS0132
      ISTATE = -6                                                       LS0132
      GO TO 580                                                         LS0132
C TOO MUCH ACCURACY REQUESTED FOR MACHINE PRECISION. -------------------LS0132
 520  CALL XERRWV(50HLSODE--  AT T (=R1), TOO MUCH ACCURACY REQUESTED  ,LS0132
     1   50, 203, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                          LS0132
      CALL XERRWV(50H      FOR PRECISION OF MACHINE..  SEE TOLSF (=R2) ,LS0132
     1   50, 203, 1, 0, 0, 0, 2, TN, TOLSF)                            LS0132
      RWORK(14) = TOLSF                                                 LS0132
      ISTATE = -2                                                       LS0132
      GO TO 580                                                         LS0133
C KFLAG = -1.  ERROR TEST FAILED REPEATEDLY OR WITH ABS(H) = HMIN. -----LS0133
 530  CALL XERRWV(50HLSODE--  AT T(=R1) AND STEP SIZE H(=R2), THE ERROR,LS0133
     1   50, 204, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                          LS0133
      CALL XERRWV(50H      TEST FAILED REPEATEDLY OR WITH ABS(H) = HMIN,LS0133
     1   50, 204, 1, 0, 0, 0, 2, TN, H)                                LS0133
      ISTATE = -4                                                       LS0133
      GO TO 560                                                         LS0133
C KFLAG = -2.  CONVERGENCE FAILED REPEATEDLY OR WITH ABS(H) = HMIN. ----LS0133
 540  CALL XERRWV(50HLSODE--  AT T (=R1) AND STEP SIZE H (=R2), THE    ,LS0133
     1   50, 205, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                          LS0134
      CALL XERRWV(50H      CORRECTOR CONVERGENCE FAILED REPEATEDLY     ,LS0134
     1   50, 205, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                          LS0134
      CALL XERRWV(30H      OR WITH ABS(H) = HMIN    ,                  LS0134
     1   30, 205, 1, 0, 0, 0, 2, TN, H)                                LS0134
      ISTATE = -5                                                       LS0134
C COMPUTE IMXER IF RELEVANT. -------------------------------------------LS0134
 560  BIG = 0.0D0                                                       LS0134
      IMXER = 1                                                         LS0134
      DO 570 I = 1,N                                                    LS0134
        SIZE = DABS(RWORK(I+LACOR-1)/RWORK(I+LEWT-1))                   LS0135
        IF (BIG .GE. SIZE) GO TO 570                                    LS0135
        BIG = SIZE                                                      LS0135
        IMXER = I                                                       LS0135
 570    CONTINUE                                                        LS0135
      IWORK(16) = IMXER                                                 LS0135
C SET Y VECTOR, T, ILLIN, AND OPTIONAL OUTPUTS. ------------------------LS0135
 580  DO 590 I = 1,N                                                    LS0135
 590    Y(I) = RWORK(I+LYH-1)                                           LS0135
      T = TN                                                            LS0135
      ILLIN = 0                                                         LS0136
      RWORK(11) = HU                                                    LS0136
      RWORK(12) = H                                                     LS0136
      RWORK(13) = TN                                                    LS0136
      IWORK(11) = NST                                                   LS0136
      IWORK(12) = NFE                                                   LS0136
      IWORK(13) = NJE                                                   LS0136
      IWORK(14) = NQU                                                   LS0136
      IWORK(15) = NQ                                                    LS0136
      RETURN                                                            LS0136
```

```
C-----------------------------------------------------------------------LS0137
C BLOCK I.                                                               LS0137
C THE FOLLOWING BLOCK HANDLES ALL ERROR RETURNS DUE TO ILLEGAL INPUT     LS0137
C (ISTATE = -3), AS DETECTED BEFORE CALLING THE CORE INTEGRATOR.         LS0137
C FIRST THE ERROR MESSAGE ROUTINE IS CALLED.  THEN IF THERE HAVE BEEN    LS0137
C 5 CONSECUTIVE SUCH RETURNS JUST BEFORE THIS CALL TO THE SOLVER,        LS0137
C THE RUN IS HALTED.                                                     LS0137
C-----------------------------------------------------------------------LS0137
 601  CALL XERRWV(30HLSODE--  ISTATE (=I1) ILLEGAL ,                     LS0137
     1    30, 1, 1, 1, ISTATE, 0, 0, 0.0D0, 0.0D0)                      LS0137
      GO TO 700                                                          LS0138
 602  CALL XERRWV(30HLSODE--  ITASK (=I1) ILLEGAL  ,                     LS0138
     1    30, 2, 1, 1, ITASK, 0, 0, 0.0D0, 0.0D0)                       LS0138
      GO TO 700                                                          LS0138
 603  CALL XERRWV(50HLSODE--  ISTATE .GT. 1 BUT LSODE NOT INITIALIZED  ,LS0138
     1    50, 3, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                           LS0138
      GO TO 700                                                          LS0138
 604  CALL XERRWV(30HLSODE--  NEQ (=I1) .LT. 0     ,                    LS0138
     1    30, 4, 1, 1, NEQ(1), 0, 0, 0.0D0, 0.0D0)                      LS0138
      GO TO 700                                                          LS0138
 605  CALL XERRWV(50HLSODE--  ISTATE = 3 AND NEQ INCREASED (I1 TO I2)  ,LS0139
     1    50, 5, 1, 2, N, NEQ(1), 0, 0.0D0, 0.0D0)                      LS0139
      GO TO 700                                                          LS0139
 606  CALL XERRWV(30HLSODE--  ITOL (=I1) ILLEGAL   ,                     LS0139
     1    30, 6, 1, 1, ITOL, 0, 0, 0.0D0, 0.0D0)                        LS0139
      GO TO 700                                                          LS0139
 607  CALL XERRWV(30HLSODE--  IOPT (=I1) ILLEGAL   ,                     LS0139
     1    30, 7, 1, 1, IOPT, 0, 0, 0.0D0, 0.0D0)                        LS0139
      GO TO 700                                                          LS0139
 608  CALL XERRWV(30HLSODE--  MF (=I1) ILLEGAL     ,                     LS0139
     1    30, 8, 1, 1, MF, 0, 0, 0.0D0, 0.0D0)                          LS0140
      GO TO 700                                                          LS0140
 609  CALL XERRWV(50HLSODE--  ML (=I1) ILLEGAL.. .LT.0 OR .GE.NEQ (=I2),LS0140
     1    50, 9, 1, 2, ML, NEQ(1), 0, 0.0D0, 0.0D0)                     LS0140
      GO TO 700                                                          LS0140
 610  CALL XERRWV(50HLSODE--  MU (=I1) ILLEGAL.. .LT.0 OR .GE.NEQ (=I2),LS0140
     1    50, 10, 1, 2, MU, NEQ(1), 0, 0.0D0, 0.0D0)                    LS0140
      GO TO 700                                                          LS0140
 611  CALL XERRWV(30HLSODE--  MAXORD (=I1) .LT. 0  ,                     LS0140
     1    30, 11, 1, 1, MAXORD, 0, 0, 0.0D0, 0.0D0)                     LS0140
      GO TO 700                                                          LS0141
 612  CALL XERRWV(30HLSODE--  MXSTEP (=I1) .LT. 0  ,                     LS0141
     1    30, 12, 1, 1, MXSTEP, 0, 0, 0.0D0, 0.0D0)                     LS0141
      GO TO 700                                                          LS0141
 613  CALL XERRWV(30HLSODE--  MXHNIL (=I1) .LT. 0  ,                     LS0141
     1    30, 13, 1, 1, MXHNIL, 0, 0, 0.0D0, 0.0D0)                     LS0141
      GO TO 700                                                          LS0141
 614  CALL XERRWV(40HLSODE--  TOUT (=R1) BEHIND T (=R2)      ,           LS0141
     1    40, 14, 1, 0, 0, 0, 2, TOUT, T)                               LS0141
      CALL XERRWV(50H      INTEGRATION DIRECTION IS GIVEN BY H0 (=R1)  ,LS0141
     1    50, 14, 1, 0, 0, 0, 1, H0, 0.0D0)                             LS0142
      GO TO 700                                                          LS0142
 615  CALL XERRWV(30HLSODE--  HMAX (=R1) .LT. 0.0  ,                     LS0142
     1    30, 15, 1, 0, 0, 0, 1, HMAX, 0.0D0)                           LS0142
      GO TO 700                                                          LS0142
 616  CALL XERRWV(30HLSODE--  HMIN (=R1) .LT. 0.0  ,                     LS0142
     1    30, 16, 1, 0, 0, 0, 1, HMIN, 0.0D0)                           LS0142
      GO TO 700                                                          LS0142
 617  CALL XERRWV(                                                       LS0142
     1  60HLSODE--  RWORK LENGTH NEEDED, LENRW (=I1), EXCEEDS LRW (=I2),LS0142
```

```
     1   60, 17, 1, 2, LENRW, LRW, 0, 0.0D0, 0.0D0)                        LS0143
      GO TO 700                                                            LS0143
 618  CALL XERRWV(                                                         LS0143
     1  60HLSODE--  IWORK LENGTH NEEDED, LENIW (=I1), EXCEEDS LIW (=I2),LS0143
     1   60, 18, 1, 2, LENIW, LIW, 0, 0.0D0, 0.0D0)                        LS0143
      GO TO 700                                                            LS0143
 619  CALL XERRWV(40HLSODE--  RTOL(I1) IS R1 .LT. 0.0        ,            LS0143
     1   40, 19, 1, 1, I, 0, 1, RTOLI, 0.0D0)                              LS0143
      GO TO 700                                                            LS0143
 620  CALL XERRWV(40HLSODE--  ATOL(I1) IS R1 .LT. 0.0        ,            LS0143
     1   40, 20, 1, 1, I, 0, 1, ATOLI, 0.0D0)                              LS0144
      GO TO 700                                                            LS0144
 621  EWTI = RWORK(LEWT+I-1)                                               LS0144
      CALL XERRWV(40HLSODE--  EWT(I1) IS R1 .LE. 0.0         ,            LS0144
     1   40, 21, 1, 1, I, 0, 1, EWTI, 0.0D0)                               LS0144
      GO TO 700                                                            LS0144
 622  CALL XERRWV(                                                         LS0144
     1  60HLSODE--  TOUT (=R1) TOO CLOSE TO T(=R2) TO START INTEGRATION,LS0144
     1   60, 22, 1, 0, 0, 0, 2, TOUT, T)                                   LS0144
      GO TO 700                                                            LS0144
 623  CALL XERRWV(                                                         LS0145
     1  60HLSODE--  ITASK = I1 AND TOUT (=R1) BEHIND TCUR - HU (= R2)   ,LS0145
     1   60, 23, 1, 1, ITASK, 0, 2, TOUT, TP)                              LS0145
      GO TO 700                                                            LS0145
 624  CALL XERRWV(                                                         LS0145
     1  60HLSODE--  ITASK = 4 OR 5 AND TCRIT (=R1) BEHIND TCUR (=R2)    ,LS0145
     1   60, 24, 1, 0, 0, 0, 2, TCRIT, TN)                                 LS0145
      GO TO 700                                                            LS0145
 625  CALL XERRWV(                                                         LS0145
     1  60HLSODE--  ITASK = 4 OR 5 AND TCRIT (=R1) BEHIND TOUT (=R2)    ,LS0145
     1   60, 25, 1, 0, 0, 0, 2, TCRIT, TOUT)                               LS0146
      GO TO 700                                                            LS0146
 626  CALL XERRWV(50HLSODE--  AT START OF PROBLEM, TOO MUCH ACCURACY    ,LS0146
     1   50, 26, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                              LS0146
      CALL XERRWV(                                                         LS0146
     1  60H      REQUESTED FOR PRECISION OF MACHINE..  SEE TOLSF (=R1) ,LS0146
     1   60, 26, 1, 0, 0, 0, 1, TOLSF, 0.0D0)                              LS0146
      RWORK(14) = TOLSF                                                    LS0146
      GO TO 700                                                            LS0146
 627  CALL XERRWV(50HLSODE--  TROUBLE FROM INTDY. ITASK = I1, TOUT = R1,LS0146
     1   50, 27, 1, 1, ITASK, 0, 1, TOUT, 0.0D0)                           LS0147
C                                                                          LS0147
 700  IF (ILLIN .EQ. 5) GO TO 710                                          LS0147
      ILLIN = ILLIN + 1                                                    LS0147
      ISTATE = -3                                                          LS0147
      RETURN                                                               LS0147
 710  CALL XERRWV(50HLSODE--  REPEATED OCCURRENCES OF ILLEGAL INPUT    ,LS0147
     1   50, 302, 1, 0, 0, 0, 0, 0.0D0, 0.0D0)                             LS0147
C                                                                          LS0147
 800  CALL XERRWV(50HLSODE--  RUN ABORTED.. APPARENT INFINITE LOOP     ,LS0147
     1   50, 303, 2, 0, 0, 0, 0, 0.0D0, 0.0D0)                             LS0148
      RETURN                                                               LS0148
C----------------------- END OF SUBROUTINE LSODE ----------------------LS0148
      END                                                                  LS0148
      SUBROUTINE INTDY (T, K, YH, NYH, DKY, IFLAG)                         LS0148
      INTEGER K, NYH, IFLAG, I, IC, IER, IOWND, IOWNS, J, JB, JB2,       LS0148
     1   JJ, JJ1, JP1, JSTART, KFLAG, L, MAXORD, METH, MITER, N, NFE,    LS0148
     2   NJE, NQ, NQU, NST                                                LS0148
      DOUBLE PRECISION T, YH, DKY,                                         LS0148
     1   ROWND, ROWNS, EL0, H, HMIN, HMXI, HU, TN, UROUND,                LS0148
```

```
      2   C, R, S, TP                                                       LS0149
       DIMENSION YH(NYH,1), DKY(1)                                          LS0149
       COMMON /LS0001/ ROWND, ROWNS(210),                                   LS0149
      1   EL0, H, HMIN, HMXI, HU, TN, UROUND, IOWND(14), IOWNS(6),          LS0149
      4   IER, JSTART, KFLAG, L, METH, MITER, MAXORD, N, NQ, NST, NFE,      LS0149
      5   NJE, NQU                                                          LS0149
C-----------------------------------------------------------------------LS0149
C INTDY COMPUTES INTERPOLATED VALUES OF THE K-TH DERIVATIVE OF THE       LS0149
C DEPENDENT VARIABLE VECTOR Y, AND STORES IT IN DKY.                     LS0149
C THIS ROUTINE IS CALLED BY LSODE WITH K = 0 AND T = TOUT, BUT MAY       LS0149
C ALSO BE CALLED BY THE USER FOR ANY K UP TO THE CURRENT ORDER.          LS0150
C (SEE DETAILED INSTRUCTIONS IN LSODE USAGE DOCUMENTATION.)              LS0150
C-----------------------------------------------------------------------LS0150
C THE COMPUTED VALUES IN DKY ARE GOTTEN BY INTERPOLATION USING THE       LS0150
C NORDSIECK HISTORY ARRAY YH.  THIS ARRAY CORRESPONDS UNIQUELY TO A      LS0150
C VECTOR-VALUED POLYNOMIAL OF DEGREE NQCUR OR LESS, AND DKY IS SET       LS0150
C TO THE K-TH DERIVATIVE OF THIS POLYNOMIAL AT T.                        LS0150
C THE FORMULA FOR DKY IS..                                               LS0150
C              Q                                                         LS0150
C  DKY(I)  =  SUM  C(J,K) * (T - TN)**(J-K) * H**(-J) * YH(I,J+1)        LS0150
C             J=K                                                        LS0151
C WHERE  C(J,K) = J*(J-1)*...*(J-K+1), Q = NQCUR, TN = TCUR, H = HCUR.   LS0151
C THE QUANTITIES  NQ = NQCUR, L = NQ+1, N = NEQ, TN, AND H ARE           LS0151
C COMMUNICATED BY COMMON.  THE ABOVE SUM IS DONE IN REVERSE ORDER.       LS0151
C IFLAG IS RETURNED NEGATIVE IF EITHER K OR T IS OUT OF BOUNDS.          LS0151
C-----------------------------------------------------------------------LS0151
      IFLAG = 0                                                          LS0151
      IF (K .LT. 0 .OR. K .GT. NQ) GO TO 80                              LS0151
      TP = TN - HU*(1.0D0 + 100.0D0*UROUND)                              LS0151
      IF ((T-TP)*(T-TN) .GT. 0.0D0) GO TO 90                             LS0151
C                                                                        LS0152
      S = (T - TN)/H                                                     LS0152
      IC = 1                                                             LS0152
      IF (K .EQ. 0) GO TO 15                                             LS0152
      JJ1 = L - K                                                        LS0152
      DO 10 JJ = JJ1,NQ                                                  LS0152
 10     IC = IC*JJ                                                       LS0152
 15   C = DFLOAT(IC)                                                     LS0152
      DO 20 I = 1,N                                                      LS0152
 20     DKY(I) = C*YH(I,L)                                               LS0152
      IF (K .EQ. NQ) GO TO 55                                            LS0153
      JB2 = NQ - K                                                       LS0153
      DO 50 JB = 1,JB2                                                   LS0153
        J = NQ - JB                                                      LS0153
        JP1 = J + 1                                                      LS0153
        IC = 1                                                           LS0153
        IF (K .EQ. 0) GO TO 35                                           LS0153
        JJ1 = JP1 - K                                                    LS0153
        DO 30 JJ = JJ1,J                                                 LS0153
 30       IC = IC*JJ                                                     LS0153
 35     C = DFLOAT(IC)                                                   LS0154
        DO 40 I = 1,N                                                    LS0154
 40       DKY(I) = C*YH(I,JP1) + S*DKY(I)                                LS0154
 50     CONTINUE                                                         LS0154
      IF (K .EQ. 0) RETURN                                               LS0154
 55   R = H**(-K)                                                        LS0154
      DO 60 I = 1,N                                                      LS0154
 60     DKY(I) = R*DKY(I)                                                LS0154
      RETURN                                                             LS0154
C                                                                        LS0154
```

```
  80   CALL XERRWV(30HINTDY--  K (=I1) ILLEGAL      ,                    LS01550
     1   30, 51, 1, 1, K, 0, 0, 0.0D0, 0.0D0)                            LS01551
       IFLAG = -1                                                        LS01552
       RETURN                                                            LS01553
  90   CALL XERRWV(30HINTDY--  T (=R1) ILLEGAL      ,                    LS01554
     1   30, 52, 1, 0, 0, 0, 1, T, 0.0D0)                                LS01555
       CALL XERRWV(                                                      LS01556
     1  60H      T NOT IN INTERVAL TCUR - HU (= R1) TO TCUR (=R2)      ,LS01557
     1   60, 52, 1, 0, 0, 0, 2, TP, TN)                                  LS01558
       IFLAG = -2                                                        LS01559
       RETURN                                                            LS01560
C----------------------- END OF SUBROUTINE INTDY ----------------------LS01561
       END                                                               LS01562
       SUBROUTINE STODE (NEQ, Y, YH, NYH, YH1, EWT, SAVF, ACOR,          LS01563
     1   WM, IWM, F, JAC, PJAC, SLVS)                                    LS01564
       EXTERNAL F, JAC, PJAC, SLVS                                       LS01565
       INTEGER NEQ, NYH, IWM, I, I1, IALTH, IER, IOWND, IREDO, IRET,     LS01566
     1   IPUP, J, JB, JSTART, KFLAG, L, LMAX, M, MAXORD, MEO, METH,      LS01567
     2   MITER, N, NCF, NEWQ, NFE, NJE, NQ, NQNYH, NQU, NST, NSTEPJ      LS01568
       DOUBLE PRECISION Y, YH, YH1, EWT, SAVF, ACOR, WM,                 LS01569
     1   ROWND, CONIT, CRATE, EL, ELCO, HOLD, RC, RMAX, TESCO,           LS01570
     2   EL0, H, HMIN, HMXI, HU, TN, UROUND,                             LS01571
     3   DCON, DDN, DEL, DELP, DSM, DUP, EXDN, EXSM, EXUP,               LS01572
     4   R, RH, RHDN, RHSM, RHUP, TOLD, VNORM                            LS01573
       DIMENSION NEQ(1), Y(1), YH(NYH,1), YH1(1), EWT(1), SAVF(1),       LS01574
     1   ACOR(1), WM(1), IWM(1)                                          LS01575
       COMMON /LS0001/ ROWND, CONIT, CRATE, EL(13), ELCO(13,12),         LS01576
     1   HOLD, RC, RMAX, TESCO(3,12),                                    LS01577
     2   EL0, H, HMIN, HMXI, HU, TN, UROUND, IOWND(14),                  LS01578
     3   IALTH, IPUP, LMAX, MEO, NQNYH, NSTEPJ,                          LS01579
     4   IER, JSTART, KFLAG, L, METH, MITER, MAXORD, N, NQ, NST, NFE,    LS01580
     5   NJE, NQU                                                        LS01581
C-----------------------------------------------------------------------LS01582
C STODE PERFORMS ONE STEP OF THE INTEGRATION OF AN INITIAL VALUE        LS01583
C PROBLEM FOR A SYSTEM OF ORDINARY DIFFERENTIAL EQUATIONS.              LS01584
C NOTE.. STODE IS INDEPENDENT OF THE VALUE OF THE ITERATION METHOD      LS01585
C INDICATOR MITER, WHEN THIS IS .NE. 0, AND HENCE IS INDEPENDENT        LS01586
C OF THE TYPE OF CHORD METHOD USED, OR THE JACOBIAN STRUCTURE.          LS01587
C COMMUNICATION WITH STODE IS DONE WITH THE FOLLOWING VARIABLES..       LS01588
C                                                                       LS01589
C NEQ    = INTEGER ARRAY CONTAINING PROBLEM SIZE IN NEQ(1), AND         LS01590
C          PASSED AS THE NEQ ARGUMENT IN ALL CALLS TO F AND JAC.        LS01591
C Y      = AN ARRAY OF LENGTH .GE. N USED AS THE Y ARGUMENT IN          LS01592
C          ALL CALLS TO F AND JAC.                                      LS01593
C YH     = AN NYH BY LMAX ARRAY CONTAINING THE DEPENDENT VARIABLES      LS01594
C          AND THEIR APPROXIMATE SCALED DERIVATIVES, WHERE              LS01595
C          LMAX = MAXORD + 1.  YH(I,J+1) CONTAINS THE APPROXIMATE       LS01596
C          J-TH DERIVATIVE OF Y(I), SCALED BY H**J/FACTORIAL(J)         LS01597
C          (J = 0,1,...,NQ).  ON ENTRY FOR THE FIRST STEP, THE FIRST    LS01598
C          TWO COLUMNS OF YH MUST BE SET FROM THE INITIAL VALUES.       LS01599
C NYH    = A CONSTANT INTEGER .GE. N, THE FIRST DIMENSION OF YH.        LS01600
C YH1    = A ONE-DIMENSIONAL ARRAY OCCUPYING THE SAME SPACE AS YH.      LS01601
C EWT    = AN ARRAY OF N ELEMENTS WITH WHICH THE ESTIMATED LOCAL        LS01602
C          ERRORS IN YH ARE COMPARED.                                   LS01603
C SAVF   = AN ARRAY OF WORKING STORAGE, OF LENGTH N.                    LS01604
C          ALSO USED FOR INPUT OF YH(*,MAXORD+2) WHEN JSTART = -1       LS01605
C          AND MAXORD .LT. THE CURRENT ORDER NQ.                        LS01606
C ACOR   = A WORK ARRAY OF LENGTH N, USED FOR THE ACCUMULATED           LS01607
C          CORRECTIONS.  ON A SUCCESSFUL RETURN, ACOR(I) CONTAINS       LS01608
C          THE ESTIMATED ONE-STEP LOCAL ERROR IN Y(I).                  LS01609
```

```
C WM,IWM = REAL AND INTEGER WORK ARRAYS ASSOCIATED WITH MATRIX          LS0161(
C          OPERATIONS IN CHORD ITERATION (MITER .NE. 0).                LS0161)
C PJAC   = NAME OF ROUTINE TO EVALUATE AND PREPROCESS JACOBIAN MATRIX   LS0161*
C          IF A CHORD METHOD IS BEING USED.                             LS0161+
C SLVS   = NAME OF ROUTINE TO SOLVE LINEAR SYSTEM IN CHORD ITERATION.   LS0161,
C H      = THE STEP SIZE TO BE ATTEMPTED ON THE NEXT STEP.              LS0161-
C          H IS ALTERED BY THE ERROR CONTROL ALGORITHM DURING THE       LS0161.
C          PROBLEM.  H CAN BE EITHER POSITIVE OR NEGATIVE, BUT ITS      LS0161/
C          SIGN MUST REMAIN CONSTANT THROUGHOUT THE PROBLEM.            LS0161(
C HMIN   = THE MINIMUM ABSOLUTE VALUE OF THE STEP SIZE H TO BE USED.    LS0161)
C HMXI   = INVERSE OF THE MAXIMUM ABSOLUTE VALUE OF H TO BE USED.       LS0162(
C          HMXI = 0.0 IS ALLOWED AND CORRESPONDS TO AN INFINITE HMAX.   LS0162)
C          HMIN AND HMXI MAY BE CHANGED AT ANY TIME, BUT WILL NOT       LS0162*
C          TAKE EFFECT UNTIL THE NEXT CHANGE OF H IS CONSIDERED.        LS0162+
C TN     = THE INDEPENDENT VARIABLE. TN IS UPDATED ON EACH STEP TAKEN.  LS0162,
C JSTART = AN INTEGER USED FOR INPUT ONLY, WITH THE FOLLOWING           LS0162-
C          VALUES AND MEANINGS..                                        LS0162.
C               0  PERFORM THE FIRST STEP.                              LS0162/
C            .GT.0  TAKE A NEW STEP CONTINUING FROM THE LAST.           LS0162(
C              -1  TAKE THE NEXT STEP WITH A NEW VALUE OF H, MAXORD,    LS0162)
C                    N, METH, MITER, AND/OR MATRIX PARAMETERS.          LS0163(
C              -2  TAKE THE NEXT STEP WITH A NEW VALUE OF H,            LS0163)
C                    BUT WITH OTHER INPUTS UNCHANGED.                   LS0163*
C          ON RETURN, JSTART IS SET TO 1 TO FACILITATE CONTINUATION.    LS0163+
C KFLAG  = A COMPLETION CODE WITH THE FOLLOWING MEANINGS..              LS0163,
C               0  THE STEP WAS SUCCESFUL.                              LS0163-
C              -1  THE REQUESTED ERROR COULD NOT BE ACHIEVED.           LS0163.
C              -2  CORRECTOR CONVERGENCE COULD NOT BE ACHIEVED.         LS0163/
C          A RETURN WITH KFLAG = -1 OR -2 MEANS EITHER                  LS0163(
C          ABS(H) = HMIN OR 10 CONSECUTIVE FAILURES OCCURRED.           LS0163)
C          ON A RETURN WITH KFLAG NEGATIVE, THE VALUES OF TN AND        LS0164(
C          THE YH ARRAY ARE AS OF THE BEGINNING OF THE LAST             LS0164)
C          STEP, AND H IS THE LAST STEP SIZE ATTEMPTED.                 LS0164*
C MAXORD = THE MAXIMUM ORDER OF INTEGRATION METHOD TO BE ALLOWED.       LS0164+
C METH/MITER = THE METHOD FLAGS.  SEE DESCRIPTION IN DRIVER.            LS0164,
C N      = THE NUMBER OF FIRST-ORDER DIFFERENTIAL EQUATIONS.            LS0164-
C-----------------------------------------------------------------------LS0164.
      KFLAG = 0                                                         LS0164/
      TOLD = TN                                                         LS0164(
      NCF = 0                                                           LS0164)
      IF (JSTART .GT. 0) GO TO 200                                      LS0165(
      IF (JSTART .EQ. -1) GO TO 100                                     LS0165)
      IF (JSTART .EQ. -2) GO TO 160                                     LS0165*
C-----------------------------------------------------------------------LS0165+
C ON THE FIRST CALL, THE ORDER IS SET TO 1, AND OTHER VARIABLES ARE     LS0165,
C INITIALIZED.  RMAX IS THE MAXIMUM RATIO BY WHICH H CAN BE INCREASED   LS0165-
C IN A SINGLE STEP.  IT IS INITIALLY 1.E4 TO COMPENSATE FOR THE SMALL   LS0165.
C INITIAL H, BUT THEN IS NORMALLY EQUAL TO 10.  IF A FAILURE            LS0165/
C OCCURS (IN CORRECTOR CONVERGENCE OR ERROR TEST), RMAX IS SET AT 2     LS0165(
C FOR THE NEXT INCREASE.                                                LS0165)
C-----------------------------------------------------------------------LS0166(
      LMAX = MAXORD + 1                                                 LS0166)
      NQ = 1                                                            LS0166*
      L = 2                                                             LS0166+
      IALTH = 2                                                         LS0166,
      RMAX = 10000.0D0                                                  LS0166-
      RC = 0.0D0                                                        LS0166.
      EL0 = 1.0D0                                                       LS0166/
      CRATE = 0.7D0                                                     LS0166(
      DELP = 0.0D0                                                      LS0166)
```

```
      HOLD = H                                                          LS0167
      MEO = METH                                                        LS0167
      NSTEPJ = 0                                                        LS0167
      IRET = 3                                                          LS0167
      GO TO 140                                                         LS0167
C-----------------------------------------------------------------------LS0167
C THE FOLLOWING BLOCK HANDLES PRELIMINARIES NEEDED WHEN JSTART = -1.    LS0167
C IPUP IS SET TO MITER TO FORCE A MATRIX UPDATE.                        LS0167
C IF AN ORDER INCREASE IS ABOUT TO BE CONSIDERED (IALTH = 1),           LS0167
C IALTH IS RESET TO 2 TO POSTPONE CONSIDERATION ONE MORE STEP.          LS0167
C IF THE CALLER HAS CHANGED METH, CFODE IS CALLED TO RESET              LS0168
C THE COEFFICIENTS OF THE METHOD.                                       LS0168
C IF THE CALLER HAS CHANGED MAXORD TO A VALUE LESS THAN THE CURRENT     LS0168
C ORDER NQ, NQ IS REDUCED TO MAXORD, AND A NEW H CHOSEN ACCORDINGLY.    LS0168
C IF H IS TO BE CHANGED, YH MUST BE RESCALED.                           LS0168
C IF H OR METH IS BEING CHANGED, IALTH IS RESET TO L = NQ + 1           LS0168
C TO PREVENT FURTHER CHANGES IN H FOR THAT MANY STEPS.                  LS0168
C-----------------------------------------------------------------------LS0168
 100  IPUP = MITER                                                     LS0168
      LMAX = MAXORD + 1                                                 LS0168
      IF (IALTH .EQ. 1) IALTH = 2                                       LS0169
      IF (METH .EQ. MEO) GO TO 110                                      LS0169
      CALL CFODE (METH, ELCO, TESCO)                                    LS0169
      MEO = METH                                                        LS0169
      IF (NQ .GT. MAXORD) GO TO 120                                     LS0169
      IALTH = L                                                         LS0169
      IRET = 1                                                          LS0169
      GO TO 150                                                         LS0169
 110  IF (NQ .LE. MAXORD) GO TO 160                                     LS0169
 120  NQ = MAXORD                                                       LS0169
      L = LMAX                                                          LS0170
      DO 125 I = 1,L                                                    LS0170
 125    EL(I) = ELCO(I,NQ)                                             LS0170
      NQNYH = NQ*NYH                                                    LS0170
      RC = RC*EL(1)/EL0                                                 LS0170
      EL0 = EL(1)                                                       LS0170
      CONIT = 0.5D0/DFLOAT(NQ+2)                                        LS0170
      DDN = VNORM (N, SAVF, EWT)/TESCO(1,L)                             LS0170
      EXDN = 1.0D0/DFLOAT(L)                                            LS0170
      RHDN = 1.0D0/(1.3D0*DDN**EXDN + 0.0000013D0)                      LS0170
      RH = DMIN1(RHDN,1.0D0)                                            LS0171
      IREDO = 3                                                         LS0171
      IF (H .EQ. HOLD) GO TO 170                                        LS0171
      RH = DMIN1(RH,DABS(H/HOLD))                                       LS0171
      H = HOLD                                                          LS0171
      GO TO 175                                                         LS0171
C-----------------------------------------------------------------------LS0171
C CFODE IS CALLED TO GET ALL THE INTEGRATION COEFFICIENTS FOR THE       LS0171
C CURRENT METH.  THEN THE EL VECTOR AND RELATED CONSTANTS ARE RESET     LS0171
C WHENEVER THE ORDER NQ IS CHANGED, OR AT THE START OF THE PROBLEM.     LS0171
C-----------------------------------------------------------------------LS0172
 140  CALL CFODE (METH, ELCO, TESCO)                                    LS0172
 150  DO 155 I = 1,L                                                    LS0172
 155    EL(I) = ELCO(I,NQ)                                              LS0172
      NQNYH = NQ*NYH                                                    LS0172
      RC = RC*EL(1)/EL0                                                 LS0172
      EL0 = EL(1)                                                       LS0172
      CONIT = 0.5D0/DFLOAT(NQ+2)                                        LS0172
      GO TO (160, 170, 200), IRET                                       LS0172
C-----------------------------------------------------------------------LS0172
```

```
C                                                                       DG0008
            DO 30 J = KP1, N                                            DG0008
               T = A(L,J)                                               DG0008
               IF (L .EQ. K) GO TO 20                                   DG0008
                  A(L,J) = A(K,J)                                       DG0008
                  A(K,J) = T                                            DG0009
   20             CONTINUE                                             DG0009
                  CALL DAXPY(N-K,T,A(K+1,K),1,A(K+1,J),1)               DG0009
   30          CONTINUE                                                DG0009
            GO TO 50                                                    DG0009
   40       CONTINUE                                                    DG0009
               INFO = K                                                 DG0009
   50       CONTINUE                                                    DG0009
   60    CONTINUE                                                       DG0009
   70    CONTINUE                                                       DG0009
         IPVT(N) = N                                                    DG0010
         IF (A(N,N) .EQ. 0.0D0) INFO = N                                DG0010
         RETURN                                                         DG0010
         END                                                            DG0010
         SUBROUTINE DGESL(A,LDA,N,IPVT,B,JOB)                           DG0010
         INTEGER LDA,N,IPVT(1),JOB                                      DG0010
         DOUBLE PRECISION A(LDA,1),B(1)                                 DG0010
C                                                                       DG0010
C        DGESL SOLVES THE DOUBLE PRECISION SYSTEM                       DG0010
C        A * X = B  OR  TRANS(A) * X = B                                DG0010
C        USING THE FACTORS COMPUTED BY DGECO OR DGEFA.                  DG0011
C                                                                       DG0011
C        ON ENTRY                                                       DG0011
C                                                                       DG0011
C           A        DOUBLE PRECISION(LDA, N)                          DG0011
C                    THE OUTPUT FROM DGECO OR DGEFA.                   DG0011
C                                                                       DG0011
C           LDA      INTEGER                                           DG0011
C                    THE LEADING DIMENSION OF THE ARRAY  A .           DG0011
C                                                                       DG0011
C           N        INTEGER                                           DG0012
C                    THE ORDER OF THE MATRIX  A .                      DG0012
C                                                                       DG0012
C           IPVT     INTEGER(N)                                        DG0012
C                    THE PIVOT VECTOR FROM DGECO OR DGEFA.             DG0012
C                                                                       DG0012
C           B        DOUBLE PRECISION(N)                               DG0012
C                    THE RIGHT HAND SIDE VECTOR.                       DG0012
C                                                                       DG0012
C           JOB      INTEGER                                           DG0012
C                    = 0         TO SOLVE  A*X = B ,                   DG0013
C                    = NONZERO   TO SOLVE  TRANS(A)*X = B  WHERE       DG0013
C                                TRANS(A)  IS THE TRANSPOSE.           DG0013
C                                                                       DG0013
C        ON RETURN                                                      DG0013
C                                                                       DG0013
C           B        THE SOLUTION VECTOR  X .                          DG0013
C                                                                       DG0013
C        ERROR CONDITION                                                DG0013
C                                                                       DG0013
C           A DIVISION BY ZERO WILL OCCUR IF THE INPUT FACTOR CONTAINS A DG0014
C           ZERO ON THE DIAGONAL.  TECHNICALLY THIS INDICATES SINGULARITY DG0014
C           BUT IT IS OFTEN CAUSED BY IMPROPER ARGUMENTS OR IMPROPER     DG0014
C           SETTING OF LDA .  IT WILL NOT OCCUR IF THE SUBROUTINES ARE   DG0014
C           CALLED CORRECTLY AND IF DGECO HAS SET RCOND .GT. 0.0         DG0014
```

```
C IF H IS BEING CHANGED, THE H RATIO RH IS CHECKED AGAINST                LS0173
C RMAX, HMIN, AND HMXI, AND THE YH ARRAY RESCALED.  IALTH IS SET TO       LS0173
C L = NQ + 1 TO PREVENT A CHANGE OF H FOR THAT MANY STEPS, UNLESS         LS0173
C FORCED BY A CONVERGENCE OR ERROR TEST FAILURE.                          LS0173
C-----------------------------------------------------------------------LS0173
 160  IF (H .EQ. HOLD) GO TO 200                                          LS0173
      RH = H/HOLD                                                         LS0173
      H = HOLD                                                            LS0173
      IREDO = 3                                                           LS0173
      GO TO 175                                                           LS0173
 170  RH = DMAX1(RH,HMIN/DABS(H))                                         LS0174
 175  RH = DMIN1(RH,RMAX)                                                 LS0174
      RH = RH/DMAX1(1.0D0,DABS(H)*HMXI*RH)                                LS0174
      R = 1.0D0                                                           LS0174
      DO 180 J = 2,L                                                      LS0174
        R = R*RH                                                          LS0174
        DO 180 I = 1,N                                                    LS0174
 180      YH(I,J) = YH(I,J)*R                                             LS0174
      H = H*RH                                                            LS0174
      RC = RC*RH                                                          LS0174
      IALTH = L                                                           LS0175
      IF (IREDO .EQ. 0) GO TO 680                                         LS0175
C-----------------------------------------------------------------------LS0175
C THIS SECTION COMPUTES THE PREDICTED VALUES BY EFFECTIVELY               LS0175
C MULTIPLYING THE YH ARRAY BY THE PASCAL TRIANGLE MATRIX.                 LS0175
C RC IS THE RATIO OF NEW TO OLD VALUES OF THE COEFFICIENT  H*EL(1).       LS0175
C WHEN RC DIFFERS FROM 1 BY MORE THAN 30 PERCENT, IPUP IS SET TO MITER   LS0175
C TO FORCE PJAC TO BE CALLED, IF A JACOBIAN IS INVOLVED.                  LS0175
C IN ANY CASE, PJAC IS CALLED AT LEAST EVERY 20-TH STEP.                  LS0175
C-----------------------------------------------------------------------LS0175
 200  IF (DABS(RC-1.0D0) .GT. 0.3D0) IPUP = MITER                         LS0176
      IF (NST .GE. NSTEPJ+20) IPUP = MITER                                LS0176
      TN = TN + H                                                         LS0176
      I1 = NQNYH + 1                                                      LS0176
      DO 215 JB = 1,NQ                                                    LS0176
        I1 = I1 - NYH                                                     LS0176
        DO 210 I = I1,NQNYH                                               LS0176
 210      YH1(I) = YH1(I) + YH1(I+NYH)                                    LS0176
 215    CONTINUE                                                          LS0176
C-----------------------------------------------------------------------LS0176
C UP TO 3 CORRECTOR ITERATIONS ARE TAKEN.  A CONVERGENCE TEST IS          LS0177
C MADE ON THE R.M.S. NORM OF EACH CORRECTION, WEIGHTED BY THE ERROR       LS0177
C WEIGHT VECTOR EWT.  THE SUM OF THE CORRECTIONS IS ACCUMULATED IN THE    LS0177
C VECTOR ACOR(I).  THE YH ARRAY IS NOT ALTERED IN THE CORRECTOR LOOP.     LS0177
C-----------------------------------------------------------------------LS0177
 220  M = 0                                                               LS0177
      DO 230 I = 1,N                                                      LS0177
 230    Y(I) = YH(I,1)                                                    LS0177
      CALL F (NEQ, TN, Y, SAVF)                                           LS0177
      NFE = NFE + 1                                                       LS0177
      IF (IPUP .LE. 0) GO TO 250                                          LS0178
C-----------------------------------------------------------------------LS0178
C IF INDICATED, THE MATRIX P = I - H*EL(1)*J IS REEVALUATED AND           LS0178
C PREPROCESSED BEFORE STARTING THE CORRECTOR ITERATION.  IPUP IS SET      LS0178
C TO 0 AS AN INDICATOR THAT THIS HAS BEEN DONE.                           LS0178
C-----------------------------------------------------------------------LS0178
      IPUP = 0                                                            LS0178
      RC = 1.0D0                                                          LS0178
      NSTEPJ = NST                                                        LS0178
      CRATE = 0.7D0                                                       LS0178
```

```
      CALL PJAC (NEQ, Y, YH, NYH, EWT, ACOR, SAVF, WM, IWM, F, JAC)     LS01790
      IF (IER .NE. 0) GO TO 430                                         LS01791
 250  DO 260 I = 1,N                                                    LS01792
 260    ACOR(I) = 0.0D0                                                 LS01793
 270  IF (MITER .NE. 0) GO TO 350                                       LS01794
C-----------------------------------------------------------------------LS01795
C IN THE CASE OF FUNCTIONAL ITERATION, UPDATE Y DIRECTLY FROM           LS01796
C THE RESULT OF THE LAST FUNCTION EVALUATION.                           LS01797
C-----------------------------------------------------------------------LS01798
      DO 290 I = 1,N                                                    LS01799
        SAVF(I) = H*SAVF(I) - YH(I,2)                                   LS01800
 290    Y(I) = SAVF(I) - ACOR(I)                                        LS01801
      DEL = VNORM (N, Y, EWT)                                           LS01802
      DO 300 I = 1,N                                                    LS01803
        Y(I) = YH(I,1) + EL(1)*SAVF(I)                                  LS01804
 300    ACOR(I) = SAVF(I)                                               LS01805
      GO TO 400                                                         LS01806
C-----------------------------------------------------------------------LS01807
C IN THE CASE OF THE CHORD METHOD, COMPUTE THE CORRECTOR ERROR,         LS01808
C AND SOLVE THE LINEAR SYSTEM WITH THAT AS RIGHT-HAND SIDE AND          LS01809
C P AS COEFFICIENT MATRIX.                                              LS01810
C-----------------------------------------------------------------------LS01811
 350  DO 360 I = 1,N                                                    LS01812
 360    Y(I) = H*SAVF(I) - (YH(I,2) + ACOR(I))                          LS01813
      CALL SLVS (WM, IWM, Y, SAVF)                                      LS01814
      IF (IER .NE. 0) GO TO 410                                         LS01815
      DEL = VNORM (N, Y, EWT)                                           LS01816
      DO 380 I = 1,N                                                    LS01817
        ACOR(I) = ACOR(I) + Y(I)                                        LS01818
 380    Y(I) = YH(I,1) + EL(1)*ACOR(I)                                  LS01819
C-----------------------------------------------------------------------LS01820
C TEST FOR CONVERGENCE.  IF M.GT.0, AN ESTIMATE OF THE CONVERGENCE      LS01821
C RATE CONSTANT IS STORED IN CRATE, AND THIS IS USED IN THE TEST.       LS01822
C-----------------------------------------------------------------------LS01823
 400  IF (M .NE. 0) CRATE = DMAX1(0.2D0*CRATE,DEL/DELP)                 LS01824
      DCON = DEL*DMIN1(1.0D0,1.5D0*CRATE)/(TESCO(2,NQ)*CONIT)           LS01825
      IF (DCON .LE. 1.0D0) GO TO 450                                    LS01826
      M = M + 1                                                         LS01827
      IF (M .EQ. 3) GO TO 410                                           LS01828
      IF (M .GE. 2 .AND. DEL .GT. 2.0D0*DELP) GO TO 410                 LS01829
      DELP = DEL                                                        LS01830
      CALL F (NEQ, TN, Y, SAVF)                                         LS01831
      NFE = NFE + 1                                                     LS01832
      GO TO 270                                                         LS01833
C-----------------------------------------------------------------------LS01834
C THE CORRECTOR ITERATION FAILED TO CONVERGE IN 3 TRIES.                LS01835
C IF MITER .NE. 0 AND THE JACOBIAN IS OUT OF DATE, PJAC IS CALLED FOR   LS01836
C THE NEXT TRY.  OTHERWISE THE YH ARRAY IS RETRACTED TO ITS VALUES      LS01837
C BEFORE PREDICTION, AND H IS REDUCED, IF POSSIBLE.  IF H CANNOT BE     LS01838
C REDUCED OR 10 FAILURES HAVE OCCURRED, EXIT WITH KFLAG = -2.           LS01839
C-----------------------------------------------------------------------LS01840
 410  IF (IPUP .EQ. 0) GO TO 430                                        LS01841
      IPUP = MITER                                                      LS01842
      GO TO 220                                                         LS01843
 430  TN = TOLD                                                         LS01844
      NCF = NCF + 1                                                     LS01845
      RMAX = 2.0D0                                                      LS01846
      I1 = NQNYH + 1                                                    LS01847
      DO 445 JB = 1,NQ                                                  LS01848
        I1 = I1 - NYH                                                   LS01849
```

```
        DO 440 I = I1,NQNYH                                              LS0185(
 440      YH1(I) = YH1(I) - YH1(I+NYH)                                   LS0185:
 445    CONTINUE                                                         LS0185:
      IF (DABS(H) .LE. HMIN*1.00001D0) GO TO 670                         LS0185:
      IF (NCF .EQ. 10) GO TO 670                                         LS0185<
      RH = 0.25D0                                                        LS0185=
      IPUP = MITER                                                       LS0185>
      IREDO = 1                                                          LS0185?
      GO TO 170                                                          LS0185@
C-----------------------------------------------------------------------LS0185A
C THE CORRECTOR HAS CONVERGED.  IPUP IS SET TO -1 IF MITER .NE. 0,       LS0186(
C TO SIGNAL THAT THE JACOBIAN INVOLVED MAY NEED UPDATING LATER.          LS01861
C THE LOCAL ERROR TEST IS MADE AND CONTROL PASSES TO STATEMENT 500       LS01862
C IF IT FAILS.                                                           LS01863
C-----------------------------------------------------------------------LS01864
 450  IF (MITER .NE. 0) IPUP = -1                                        LS01865
      IF (M .EQ. 0) DSM = DEL/TESCO(2,NQ)                                LS01866
      IF (M .GT. 0) DSM = VNORM (N, ACOR, EWT)/TESCO(2,NQ)               LS01867
      IF (DSM .GT. 1.0D0) GO TO 500                                      LS01868
C-----------------------------------------------------------------------LS01869
C AFTER A SUCCESSFUL STEP, UPDATE THE YH ARRAY.                          LS01870
C CONSIDER CHANGING H IF IALTH = 1.  OTHERWISE DECREASE IALTH BY 1.      LS01871
C IF IALTH IS THEN 1 AND NQ .LT. MAXORD, THEN ACOR IS SAVED FOR          LS01872
C USE IN A POSSIBLE ORDER INCREASE ON THE NEXT STEP.                     LS01873
C IF A CHANGE IN H IS CONSIDERED, AN INCREASE OR DECREASE IN ORDER       LS01874
C BY ONE IS CONSIDERED ALSO.  A CHANGE IN H IS MADE ONLY IF IT IS BY A   LS01875
C FACTOR OF AT LEAST 1.1.  IF NOT, IALTH IS SET TO 3 TO PREVENT          LS01876
C TESTING FOR THAT MANY STEPS.                                           LS01877(
C-----------------------------------------------------------------------LS01878(
      KFLAG = 0                                                          LS01879(
      IREDO = 0                                                          LS01880(
      NST = NST + 1                                                      LS01881(
      HU = H                                                             LS01882(
      NQU = NQ                                                           LS01883(
      DO 470 J = 1,L                                                     LS01884(
        DO 470 I = 1,N                                                   LS01885(
 470      YH(I,J) = YH(I,J) + EL(J)*ACOR(I)                             LS01886(
      IALTH = IALTH - 1                                                  LS01887(
      IF (IALTH .EQ. 0) GO TO 520                                        LS01888(
      IF (IALTH .GT. 1) GO TO 690                                        LS01889(
      IF (L .EQ. LMAX) GO TO 690                                         LS01890(
      DO 490 I = 1,N                                                     LS01891(
 490    YH(I,LMAX) = ACOR(I)                                            LS01892(
      GO TO 690                                                          LS01893(
C-----------------------------------------------------------------------LS01894(
C THE ERROR TEST FAILED.  KFLAG KEEPS TRACK OF MULTIPLE FAILURES.        LS01895(
C RESTORE TN AND THE YH ARRAY TO THEIR PREVIOUS VALUES, AND PREPARE      LS01896(
C TO TRY THE STEP AGAIN.  COMPUTE THE OPTIMUM STEP SIZE FOR THIS OR      LS01897(
C ONE LOWER ORDER.  AFTER 2 OR MORE FAILURES, H IS FORCED TO DECREASE    LS01898(
C BY A FACTOR OF 0.2 OR LESS.                                            LS01899(
C-----------------------------------------------------------------------LS01900(
 500  KFLAG = KFLAG - 1                                                  LS01901(
      TN = TOLD                                                          LS01902(
      I1 = NQNYH + 1                                                     LS01903(
      DO 515 JB = 1,NQ                                                   LS01904(
        I1 = I1 - NYH                                                    LS01905(
        DO 510 I = I1,NQNYH                                              LS01906(
 510      YH1(I) = YH1(I) - YH1(I+NYH)                                   LS01907(
 515    CONTINUE                                                         LS01908(
      RMAX = 2.0D0                                                       LS01909(
```

```
      IF (DABS(H) .LE. HMIN*1.00001D0) GO TO 660                        LS0191
      IF (KFLAG .LE. -3) GO TO 640                                      LS0191
      IREDO = 2                                                         LS0191
      RHUP = 0.0D0                                                      LS0191
      GO TO 540                                                         LS0191
C-----------------------------------------------------------------------LS0191
C REGARDLESS OF THE SUCCESS OR FAILURE OF THE STEP, FACTORS             LS0191
C RHDN, RHSM, AND RHUP ARE COMPUTED, BY WHICH H COULD BE MULTIPLIED     LS0191
C AT ORDER NQ - 1, ORDER NQ, OR ORDER NQ + 1, RESPECTIVELY.             LS0191
C IN THE CASE OF FAILURE, RHUP = 0.0 TO AVOID AN ORDER INCREASE.        LS0191
C THE LARGEST OF THESE IS DETERMINED AND THE NEW ORDER CHOSEN           LS0192
C ACCORDINGLY.  IF THE ORDER IS TO BE INCREASED, WE COMPUTE ONE         LS0192
C ADDITIONAL SCALED DERIVATIVE.                                         LS0192
C-----------------------------------------------------------------------LS0192
 520  RHUP = 0.0D0                                                      LS0192
      IF (L .EQ. LMAX) GO TO 540                                        LS0192
      DO 530 I = 1,N                                                    LS0192
 530    SAVF(I) = ACOR(I) - YH(I,LMAX)                                  LS0192
      DUP = VNORM (N, SAVF, EWT)/TESCO(3,NQ)                            LS0192
      EXUP = 1.0D0/DFLOAT(L+1)                                          LS0192
      RHUP = 1.0D0/(1.4D0*DUP**EXUP + 0.0000014D0)                      LS0193
 540  EXSM = 1.0D0/DFLOAT(L)                                            LS0193
      RHSM = 1.0D0/(1.2D0*DSM**EXSM + 0.0000012D0)                      LS0193
      RHDN = 0.0D0                                                      LS0193
      IF (NQ .EQ. 1) GO TO 560                                          LS0193
      DDN = VNORM (N, YH(1,L), EWT)/TESCO(1,NQ)                         LS0193
      EXDN = 1.0D0/DFLOAT(NQ)                                           LS0193
      RHDN = 1.0D0/(1.3D0*DDN**EXDN + 0.0000013D0)                      LS0193
 560  IF (RHSM .GE. RHUP) GO TO 570                                     LS0193
      IF (RHUP .GT. RHDN) GO TO 590                                     LS0193
      GO TO 580                                                         LS0194
 570  IF (RHSM .LT. RHDN) GO TO 580                                     LS0194
      NEWQ = NQ                                                         LS0194
      RH = RHSM                                                         LS0194
      GO TO 620                                                         LS0194
 580  NEWQ = NQ - 1                                                     LS0194
      RH = RHDN                                                         LS0194
      IF (KFLAG .LT. 0 .AND. RH .GT. 1.0D0) RH = 1.0D0                  LS0194
      GO TO 620                                                         LS0194
 590  NEWQ = L                                                          LS0194
      RH = RHUP                                                         LS0195
      IF (RH .LT. 1.1D0) GO TO 610                                      LS0195
      R = EL(L)/DFLOAT(L)                                               LS0195
      DO 600 I = 1,N                                                    LS0195
 600    YH(I,NEWQ+1) = ACOR(I)*R                                        LS0195
      GO TO 630                                                         LS0195
 610  IALTH = 3                                                         LS0195
      GO TO 690                                                         LS0195
 620  IF ((KFLAG .EQ. 0) .AND. (RH .LT. 1.1D0)) GO TO 610               LS0195
      IF (KFLAG .LE. -2) RH = DMIN1(RH,0.2D0)                           LS0195
C-----------------------------------------------------------------------LS0196
C IF THERE IS A CHANGE OF ORDER, RESET NQ, L, AND THE COEFFICIENTS.     LS0196
C IN ANY CASE H IS RESET ACCORDING TO RH AND THE YH ARRAY IS RESCALED.  LS0196
C THEN EXIT FROM 680 IF THE STEP WAS OK, OR REDO THE STEP OTHERWISE.    LS0196
C-----------------------------------------------------------------------LS0196
      IF (NEWQ .EQ. NQ) GO TO 170                                       LS0196
 630  NQ = NEWQ                                                         LS0196
      L = NQ + 1                                                        LS0196
      IRET = 2                                                          LS0196
      GO TO 150                                                         LS0196
```

```
C-----------------------------------------------------------------------LS0197
C CONTROL REACHES THIS SECTION IF 3 OR MORE FAILURES HAVE OCCURED.       LS0197
C IF 10 FAILURES HAVE OCCURRED, EXIT WITH KFLAG = -1.                    LS0197
C IT IS ASSUMED THAT THE DERIVATIVES THAT HAVE ACCUMULATED IN THE        LS0197
C YH ARRAY HAVE ERRORS OF THE WRONG ORDER.  HENCE THE FIRST              LS0197
C DERIVATIVE IS RECOMPUTED, AND THE ORDER IS SET TO 1.  THEN             LS0197
C H IS REDUCED BY A FACTOR OF 10, AND THE STEP IS RETRIED,               LS0197
C UNTIL IT SUCCEEDS OR H REACHES HMIN.                                   LS0197
C-----------------------------------------------------------------------LS0197
 640  IF (KFLAG .EQ. -10) GO TO 660                                      LS0197
      RH = 0.1D0                                                         LS0198
      RH = DMAX1(HMIN/DABS(H),RH)                                        LS0198
      H = H*RH                                                           LS0198
      DO 645 I = 1,N                                                     LS0198
 645    Y(I) = YH(I,1)                                                   LS0198
      CALL F (NEQ, TN, Y, SAVF)                                          LS0198
      NFE = NFE + 1                                                      LS0198
      DO 650 I = 1,N                                                     LS0198
 650    YH(I,2) = H*SAVF(I)                                              LS0198
      IPUP = MITER                                                       LS0198
      IALTH = 5                                                          LS0199
      IF (NQ .EQ. 1) GO TO 200                                           LS0199
      NQ = 1                                                             LS0199
      L = 2                                                              LS0199
      IRET = 3                                                           LS0199
      GO TO 150                                                          LS0199
C-----------------------------------------------------------------------LS0199
C ALL RETURNS ARE MADE THROUGH THIS SECTION.  H IS SAVED IN HOLD         LS0199
C TO ALLOW THE CALLER TO CHANGE H ON THE NEXT STEP.                      LS0199
C-----------------------------------------------------------------------LS0199
 660  KFLAG = -1                                                         LS0200
      GO TO 700                                                          LS0200
 670  KFLAG = -2                                                         LS0200
      GO TO 700                                                          LS0200
 680  RMAX = 10.0D0                                                      LS0200
 690  R = 1.0D0/TESCO(2,NQU)                                             LS0200
      DO 695 I = 1,N                                                     LS0200
 695    ACOR(I) = ACOR(I)*R                                              LS0200
 700  HOLD = H                                                           LS0200
      JSTART = 1                                                         LS0200
      RETURN                                                             LS0201
C---------------------- END OF SUBROUTINE STODE ----------------------LS0201
      END                                                                LS0201
      SUBROUTINE CFODE (METH, ELCO, TESCO)                               LS0201
      INTEGER METH, I, IB, NQ, NQM1, NQP1                                LS0201
      DOUBLE PRECISION ELCO, TESCO, AGAMQ, FNQ, FNQM1, PC, PINT, RAGQ,   LS0201
     1   RQFAC, RQ1FAC, TSIGN, XPIN                                      LS0201
      DIMENSION ELCO(13,12), TESCO(3,12)                                 LS0201
C-----------------------------------------------------------------------LS0201
C CFODE IS CALLED BY THE INTEGRATOR ROUTINE TO SET COEFFICIENTS          LS0201
C NEEDED THERE.  THE COEFFICIENTS FOR THE CURRENT METHOD, AS             LS0202
C GIVEN BY THE VALUE OF METH, ARE SET FOR ALL ORDERS AND SAVED.          LS0202
C THE MAXIMUM ORDER ASSUMED HERE IS 12 IF METH = 1 AND 5 IF METH = 2.    LS0202
C (A SMALLER VALUE OF THE MAXIMUM ORDER IS ALSO ALLOWED.)                LS0202
C CFODE IS CALLED ONCE AT THE BEGINNING OF THE PROBLEM,                  LS0202
C AND IS NOT CALLED AGAIN UNLESS AND UNTIL METH IS CHANGED.              LS0202
C                                                                        LS0202
C THE ELCO ARRAY CONTAINS THE BASIC METHOD COEFFICIENTS.                 LS0202
C THE COEFFICIENTS EL(I), 1 .LE. I .LE. NQ+1, FOR THE METHOD OF          LS0202
C ORDER NQ ARE STORED IN ELCO(I,NQ).  THEY ARE GIVEN BY A GENETRATING    LS0202
```

```
C POLYNOMIAL, I.E.,                                                    LS02030
C      L(X) = EL(1) + EL(2)*X + ... + EL(NQ+1)*X**NQ.                  LS02031
C FOR THE IMPLICIT ADAMS METHODS, L(X) IS GIVEN BY                     LS02032
C      DL/DX = (X+1)*(X+2)*...*(X+NQ-1)/FACTORIAL(NQ-1),    L(-1) = 0.  LS02033
C FOR THE BDF METHODS, L(X) IS GIVEN BY                                LS02034
C      L(X) = (X+1)*(X+2)* ... *(X+NQ)/K,                              LS02035
C WHERE          K = FACTORIAL(NQ)*(1 + 1/2 + ... + 1/NQ).             LS02036
C                                                                      LS02037
C THE TESCO ARRAY CONTAINS TEST CONSTANTS USED FOR THE                 LS02038
C LOCAL ERROR TEST AND THE SELECTION OF STEP SIZE AND/OR ORDER.        LS02039
C AT ORDER NQ, TESCO(K,NQ) IS USED FOR THE SELECTION OF STEP           LS02040
C SIZE AT ORDER NQ - 1 IF K = 1, AT ORDER NQ IF K = 2, AND AT ORDER    LS02041
C NQ + 1 IF K = 3.                                                     LS02042
C-----------------------------------------------------------------------LS02043
      DIMENSION PC(12)                                                 LS02044
C                                                                      LS02045
      GO TO (100, 200), METH                                           LS02046
C                                                                      LS02047
 100  ELCO(1,1) = 1.0D0                                                LS02048
      ELCO(2,1) = 1.0D0                                                LS02049
      TESCO(1,1) = 0.0D0                                               LS02050
      TESCO(2,1) = 2.0D0                                               LS02051
      TESCO(1,2) = 1.0D0                                               LS02052
      TESCO(3,12) = 0.0D0                                              LS02053
      PC(1) = 1.0D0                                                    LS02054
      RQFAC = 1.0D0                                                    LS02055
      DO 140 NQ = 2,12                                                 LS02056
C-----------------------------------------------------------------------LS02057
C THE PC ARRAY WILL CONTAIN THE COEFFICIENTS OF THE POLYNOMIAL         LS02058
C      P(X) = (X+1)*(X+2)*...*(X+NQ-1).                                LS02059
C INITIALLY, P(X) = 1.                                                 LS02060
C-----------------------------------------------------------------------LS02061
        RQ1FAC = RQFAC                                                 LS02062
        RQFAC = RQFAC/DFLOAT(NQ)                                       LS02063
        NQM1 = NQ - 1                                                  LS02064
        FNQM1 = DFLOAT(NQM1)                                           LS02065
        NQP1 = NQ + 1                                                  LS02066
C FORM COEFFICIENTS OF P(X)*(X+NQ-1). ---------------------------------LS02067
        PC(NQ) = 0.0D0                                                 LS02068
        DO 110 IB = 1,NQM1                                             LS02069
          I = NQP1 - IB                                                LS02070
 110      PC(I) = PC(I-1) + FNQM1*PC(I)                                LS02071
        PC(1) = FNQM1*PC(1)                                            LS02072
C COMPUTE INTEGRAL, -1 TO 0, OF P(X) AND X*P(X). ----------------------LS02073
        PINT = PC(1)                                                   LS02074
        XPIN = PC(1)/2.0D0                                             LS02075
        TSIGN = 1.0D0                                                  LS02076
        DO 120 I = 2,NQ                                                LS02077
          TSIGN = -TSIGN                                               LS02078
          PINT = PINT + TSIGN*PC(I)/DFLOAT(I)                          LS02079
 120      XPIN = XPIN + TSIGN*PC(I)/DFLOAT(I+1)                        LS02080
C STORE COEFFICIENTS IN ELCO AND TESCO. -------------------------------LS02081
        ELCO(1,NQ) = PINT*RQ1FAC                                       LS02082
        ELCO(2,NQ) = 1.0D0                                             LS02083
        DO 130 I = 2,NQ                                                LS02084
 130      ELCO(I+1,NQ) = RQ1FAC*PC(I)/DFLOAT(I)                        LS02085
        AGAMQ = RQFAC*XPIN                                             LS02086
        RAGQ = 1.0D0/AGAMQ                                             LS02087
        TESCO(2,NQ) = RAGQ                                             LS02088
        IF (NQ .LT. 12) TESCO(1,NQP1) = RAGQ*RQFAC/DFLOAT(NQP1)        LS02089
```

```
        TESCO(3,NQM1) = RAGQ                                              LS02090
 140    CONTINUE                                                          LS02091
      RETURN                                                              LS02092
C                                                                         LS02093
 200  PC(1) = 1.0D0                                                       LS02094
      RQ1FAC = 1.0D0                                                      LS02095
      DO 230 NQ = 1,5                                                     LS02096
C-----------------------------------------------------------------------LS02097
C THE PC ARRAY WILL CONTAIN THE COEFFICIENTS OF THE POLYNOMIAL            LS02098
C     P(X) = (X+1)*(X+2)*...*(X+NQ).                                      LS02099
C INITIALLY, P(X) = 1.                                                    LS02100
C-----------------------------------------------------------------------LS02101
        FNQ = DFLOAT(NQ)                                                  LS02102
        NQP1 = NQ + 1                                                     LS02103
C FORM COEFFICIENTS OF P(X)*(X+NQ). -------------------------------------LS02104
        PC(NQP1) = 0.0D0                                                  LS02105
        DO 210 IB = 1,NQ                                                  LS02106
          I = NQ + 2 - IB                                                 LS02107
 210      PC(I) = PC(I-1) + FNQ*PC(I)                                     LS02108
        PC(1) = FNQ*PC(1)                                                 LS02109
C STORE COEFFICIENTS IN ELCO AND TESCO. ---------------------------------LS02110
        DO 220 I = 1,NQP1                                                 LS02111
 220      ELCO(I,NQ) = PC(I)/PC(2)                                        LS02112
        ELCO(2,NQ) = 1.0D0                                                LS02113
        TESCO(1,NQ) = RQ1FAC                                              LS02114
        TESCO(2,NQ) = DFLOAT(NQP1)/ELCO(1,NQ)                             LS02115
        TESCO(3,NQ) = DFLOAT(NQ+2)/ELCO(1,NQ)                             LS02116
        RQ1FAC = RQ1FAC/FNQ                                               LS02117
 230    CONTINUE                                                          LS02118
      RETURN                                                              LS02119
C---------------------- END OF SUBROUTINE CFODE -----------------------LS02120
      END                                                                 LS02121
      SUBROUTINE PREPJ (NEQ, Y, YH, NYH, EWT, FTEM, SAVF, WM, IWM,        LS02122
     1   F, JAC)                                                          LS02123
      EXTERNAL F, JAC                                                     LS02124
      INTEGER NEQ, NYH, IWM, I, I1, I2, IER, II, IOWND, IOWNS, J, J1,     LS02125
     1   JJ, JSTART, KFLAG, L, LENP, MAXORD, MBA, MBAND, MEB1, MEBAND,    LS02126
     2   METH, MITER, ML, ML3, MU, N, NFE, NJE, NQ, NQU, NST              LS02127
      DOUBLE PRECISION Y, YH, EWT, FTEM, SAVF, WM,                        LS02128
     1   ROWND, ROWNS, EL0, H, HMIN, HMXI, HU, TN, UROUND,                LS02129
     2   CON, DI, FAC, HL0, R, R0, SRUR, YI, YJ, YJJ, VNORM               LS02130
      DIMENSION NEQ(1), Y(1), YH(NYH,1), EWT(1), FTEM(1), SAVF(1),        LS02131
     1   WM(1), IWM(1)                                                    LS02132
      COMMON /LS0001/ ROWND, ROWNS(210),                                  LS02133
     1   EL0, H, HMIN, HMXI, HU, TN, UROUND, IOWND(14), IOWNS(6),         LS02134
     4   IER, JSTART, KFLAG, L, METH, MITER, MAXORD, N, NQ, NST, NFE,     LS02135
     5   NJE, NQU                                                         LS02136
C-----------------------------------------------------------------------LS02137
C PREPJ IS CALLED BY STODE TO COMPUTE AND PROCESS THE MATRIX              LS02138
C P = I - H*EL(1)*J , WHERE J IS AN APPROXIMATION TO THE JACOBIAN.        LS02139
C HERE J IS COMPUTED BY THE USER-SUPPLIED ROUTINE JAC IF                  LS02140
C MITER = 1 OR 4, OR BY FINITE DIFFERENCING IF MITER = 2, 3, OR 5.        LS02141
C IF MITER = 3, A DIAGONAL APPROXIMATION TO J IS USED.                    LS02142
C J IS STORED IN WM AND REPLACED BY P.  IF MITER .NE. 3, P IS THEN        LS02143
C SUBJECTED TO LU DECOMPOSITION IN PREPARATION FOR LATER SOLUTION         LS02144
C OF LINEAR SYSTEMS WITH P AS COEFFICIENT MATRIX. THIS IS DONE            LS02145
C BY DGEFA IF MITER = 1 OR 2, AND BY DGBFA IF MITER = 4 OR 5.             LS02146
C                                                                         LS02147
C IN ADDITION TO VARIABLES DESCRIBED PREVIOUSLY, COMMUNICATION            LS02148
C WITH PREPJ USES THE FOLLOWING..                                         LS02149
```

```
C Y    = ARRAY CONTAINING PREDICTED VALUES ON ENTRY.                        LS02150
C FTEM = WORK ARRAY OF LENGTH N (ACOR IN STODE).                            LS02151
C SAVF = ARRAY CONTAINING F EVALUATED AT PREDICTED Y.                       LS02152
C WM   = REAL WORK SPACE FOR MATRICES.  ON OUTPUT IT CONTAINS THE           LS02153
C          INVERSE DIAGONAL MATRIX IF MITER = 3 AND THE LU DECOMPOSITION    LS02154
C          OF P IF MITER IS 1, 2 , 4, OR 5.                                 LS02155
C          STORAGE OF MATRIX ELEMENTS STARTS AT WM(3).                      LS02156
C          WM ALSO CONTAINS THE FOLLOWING MATRIX-RELATED DATA..             LS02157
C          WM(1) = SQRT(UROUND), USED IN NUMERICAL JACOBIAN INCREMENTS.     LS02158
C          WM(2) = H*EL0, SAVED FOR LATER USE IF MITER = 3.                 LS02159
C IWM  = INTEGER WORK SPACE CONTAINING PIVOT INFORMATION, STARTING AT       LS02160
C          IWM(21), IF MITER IS 1, 2, 4, OR 5.  IWM ALSO CONTAINS THE       LS02161
C          BAND PARAMETERS ML = IWM(1) AND MU = IWM(2) IF MITER IS 4 OR 5.LS02162
C EL0  = EL(1) (INPUT).                                                     LS02163
C IER  = OUTPUT ERROR FLAG,  = 0 IF NO TROUBLE, .NE. 0 IF                   LS02164
C          P MATRIX FOUND TO BE SINGULAR.                                   LS02165
C THIS ROUTINE ALSO USES THE COMMON VARIABLES EL0, H, TN, UROUND,           LS02166
C MITER, N, NFE, AND NJE.                                                   LS02167
C-----------------------------------------------------------------------LS02168
      NJE = NJE + 1                                                         LS02169
      HL0 = H*EL0                                                           LS02170
      GO TO (100, 200, 300, 400, 500), MITER                                LS02171
C IF MITER = 1, CALL JAC AND MULTIPLY BY SCALAR. ----------------------LS02172
 100  LENP = N*N                                                            LS02173
      DO 110 I = 1,LENP                                                     LS02174
 110    WM(I+2) = 0.0D0                                                     LS02175
      CALL JAC (NEQ, TN, Y, 0, 0, WM(3), N)                                 LS02176
      CON = -HL0                                                            LS02177
      DO 120 I = 1,LENP                                                     LS02178
 120    WM(I+2) = WM(I+2)*CON                                               LS02179
      GO TO 240                                                             LS02180
C IF MITER = 2, MAKE N CALLS TO F TO APPROXIMATE J. --------------------LS02181
 200  FAC = VNORM (N, SAVF, EWT)                                            LS02182
      R0 = 1000.0D0*DABS(H)*UROUND*DFLOAT(N)*FAC                            LS02183
      IF (R0 .EQ. 0.0D0) R0 = 1.0D0                                         LS02184
      SRUR = WM(1)                                                          LS02185
      J1 = 2                                                                LS02186
      DO 230 J = 1,N                                                        LS02187
        YJ = Y(J)                                                           LS02188
        R = DMAX1(SRUR*DABS(YJ),R0*EWT(J))                                  LS02189
        Y(J) = Y(J) + R                                                     LS02190
        FAC = -HL0/R                                                        LS02191
        CALL F (NEQ, TN, Y, FTEM)                                           LS02192
        DO 220 I = 1,N                                                      LS02193
 220      WM(I+J1) = (FTEM(I) - SAVF(I))*FAC                                LS02194
        Y(J) = YJ                                                           LS02195
        J1 = J1 + N                                                         LS02196
 230    CONTINUE                                                            LS02197
      NFE = NFE + N                                                         LS02198
C ADD IDENTITY MATRIX. -------------------------------------------------LS02199
 240  J = 3                                                                 LS02200
      DO 250 I = 1,N                                                        LS02201
        WM(J) = WM(J) + 1.0D0                                               LS02202
 250    J = J + (N + 1)                                                     LS02203
C DO LU DECOMPOSITION ON P. --------------------------------------------LS02204
      CALL DGEFA (WM(3), N, N, IWM(21), IER)                                LS02205
      RETURN                                                                LS02206
C IF MITER = 3, CONSTRUCT A DIAGONAL APPROXIMATION TO J AND P. ---------LS02207
 300  WM(2) = HL0                                                           LS02208
      IER = 0                                                               LS02209
```

```
      R = EL0*0.1D0                                                     LS0221(
      DO 310 I = 1,N                                                    LS0221
  310   Y(I) = Y(I) + R*(H*SAVF(I) - YH(I,2))                          LS0221
      CALL F (NEQ, TN, Y, WM(3))                                        LS0221
      NFE = NFE + 1                                                     LS0221
      DO 320 I = 1,N                                                    LS0221
        R0 = H*SAVF(I) - YH(I,2)                                        LS0221
        DI = 0.1D0*R0 - H*(WM(I+2) - SAVF(I))                           LS0221
        WM(I+2) = 1.0D0                                                 LS0221
        IF (DABS(R0) .LT. UROUND*EWT(I)) GO TO 320                      LS0221
        IF (DABS(DI) .EQ. 0.0D0) GO TO 330                              LS0222(
        WM(I+2) = 0.1D0*R0/DI                                           LS0222
  320   CONTINUE                                                        LS0222
      RETURN                                                            LS0222
  330 IER = -1                                                          LS0222
      RETURN                                                            LS0222
C IF MITER = 4, CALL JAC AND MULTIPLY BY SCALAR. ----------------------LS0222
  400 ML = IWM(1)                                                       LS0222
      MU = IWM(2)                                                       LS0222
      ML3 = ML + 3                                                      LS0222
      MBAND = ML + MU + 1                                               LS0223(
      MEBAND = MBAND + ML                                               LS0223
      LENP = MEBAND*N                                                   LS0223
      DO 410 I = 1,LENP                                                 LS0223
  410   WM(I+2) = 0.0D0                                                 LS0223
      CALL JAC (NEQ, TN, Y, ML, MU, WM(ML3), MEBAND)                    LS0223
      CON = -HL0                                                        LS0223
      DO 420 I = 1,LENP                                                 LS0223
  420   WM(I+2) = WM(I+2)*CON                                           LS0223
      GO TO 570                                                         LS0223
C IF MITER = 5, MAKE MBAND CALLS TO F TO APPROXIMATE J. ---------------LS0224(
  500 ML = IWM(1)                                                       LS0224
      MU = IWM(2)                                                       LS0224
      MBAND = ML + MU + 1                                               LS0224
      MBA = MIN0(MBAND,N)                                               LS0224
      MEBAND = MBAND + ML                                               LS0224
      MEB1 = MEBAND - 1                                                 LS0224
      SRUR = WM(1)                                                      LS0224
      FAC = VNORM (N, SAVF, EWT)                                        LS0224
      R0 = 1000.0D0*DABS(H)*UROUND*DFLOAT(N)*FAC                        LS0224
      IF (R0 .EQ. 0.0D0) R0 = 1.0D0                                     LS0225(
      DO 560 J = 1,MBA                                                  LS0225
        DO 530 I = J,N,MBAND                                            LS0225
          YI = Y(I)                                                     LS0225
          R = DMAX1(SRUR*DABS(YI),R0*EWT(I))                            LS0225
  530     Y(I) = Y(I) + R                                               LS0225
        CALL F (NEQ, TN, Y, FTEM)                                       LS0225
        DO 550 JJ = J,N,MBAND                                           LS0225
          Y(JJ) = YH(JJ,1)                                              LS0225
          YJJ = Y(JJ)                                                   LS0225
          R = DMAX1(SRUR*DABS(YJJ),R0*EWT(JJ))                          LS0226(
          FAC = -HL0/R                                                  LS0226
          I1 = MAX0(JJ-MU,1)                                            LS0226
          I2 = MIN0(JJ+ML,N)                                            LS0226
          II = JJ*MEB1 - ML + 2                                         LS0226
          DO 540 I = I1,I2                                              LS0226
  540       WM(II+I) = (FTEM(I) - SAVF(I))*FAC                          LS0226
  550     CONTINUE                                                      LS0226
  560   CONTINUE                                                        LS0226
      NFE = NFE + MBA                                                   LS0226
```

```
C ADD IDENTITY MATRIX. ---------------------------------------------LS0227(
 570  II = MBAND + 2                                                 LS02271
      DO 580 I = 1,N                                                 LS02272
        WM(II) = WM(II) + 1.0D0                                      LS02273
 580    II = II + MEBAND                                             LS02274
C DO LU DECOMPOSITION OF P. ----------------------------------------LS02275
      CALL DGBFA (WM(3), MEBAND, N, ML, MU, IWM(21), IER)            LS02276
      RETURN                                                         LS02277
C---------------------- END OF SUBROUTINE PREPJ ---------------------LS02278
      END                                                            LS02279
      SUBROUTINE SOLSY (WM, IWM, X, TEM)                             LS02280
      INTEGER IWM, I, IER, IOWND, IOWNS, JSTART, KFLAG, L, MAXORD,   LS02281
     1   MEBAND, METH, MITER, ML, MU, N, NFE, NJE, NQ, NQU, NST      LS02282
      DOUBLE PRECISION WM, X, TEM,                                   LS02283
     1   ROWND, ROWNS, EL0, H, HMIN, HMXI, HU, TN, UROUND,           LS02284
     2   DI, HL0, PHL0, R                                            LS02285
      DIMENSION WM(1), IWM(1), X(1), TEM(1)                          LS02286
      COMMON /LS0001/ ROWND, ROWNS(210),                             LS02287
     1   EL0, H, HMIN, HMXI, HU, TN, UROUND, IOWND(14), IOWNS(6),    LS02288
     4   IER, JSTART, KFLAG, L, METH, MITER, MAXORD, N, NQ, NST, NFE, LS02289
     5   NJE, NQU                                                    LS02290
C-----------------------------------------------------------------------LS02291
C THIS ROUTINE MANAGES THE SOLUTION OF THE LINEAR SYSTEM ARISING FROM  LS02292
C A CHORD ITERATION.  IT IS CALLED BY STODE IF MITER .NE. 0.           LS02293
C IF MITER IS 1 OR 2, IT CALLS DGESL TO ACCOMPLISH THIS.               LS02294
C IF MITER = 3 IT UPDATES THE COEFFICIENT H*EL0 IN THE DIAGONAL        LS02295
C MATRIX, AND THEN COMPUTES THE SOLUTION.                              LS02296
C IF MITER IS 4 OR 5, IT CALLS DGBSL.                                  LS02297
C COMMUNICATION WITH SOLSY USES THE FOLLOWING VARIABLES..              LS02298
C WM  = REAL WORK SPACE CONTAINING THE INVERSE DIAGONAL MATRIX IF MITER LS02299
C       IS 3 AND THE LU DECOMPOSITION OF THE MATRIX OTHERWISE.         LS02300
C       STORAGE OF MATRIX ELEMENTS STARTS AT WM(3).                    LS02301
C       WM ALSO CONTAINS THE FOLLOWING MATRIX-RELATED DATA..           LS02302
C       WM(1) = SQRT(UROUND) (NOT USED HERE),                          LS02303
C       WM(2) = HL0, THE PREVIOUS VALUE OF H*EL0, USED IF MITER = 3.   LS02304
C IWM = INTEGER WORK SPACE CONTAINING PIVOT INFORMATION, STARTING AT    LS02305
C       IWM(21), IF MITER IS 1, 2, 4, OR 5.  IWM ALSO CONTAINS THE     LS02306
C       BAND PARAMETERS ML = IWM(1) AND MU = IWM(2) IF MITER IS 4 OR 5. LS02307
C X   = THE RIGHT-HAND SIDE VECTOR ON INPUT, AND THE SOLUTION VECTOR   LS02308
C       ON OUTPUT, OF LENGTH N.                                        LS02309
C TEM = VECTOR OF WORK SPACE OF LENGTH N, NOT USED IN THIS VERSION.    LS02310
C IER = OUTPUT FLAG (IN COMMON).  IER = 0 IF NO TROUBLE OCCURRED.      LS02311
C       IER = -1 IF A SINGULAR MATRIX AROSE WITH MITER = 3.            LS02312
C THIS ROUTINE ALSO USES THE COMMON VARIABLES EL0, H, MITER, AND N.    LS02313
C-----------------------------------------------------------------------LS02314
      IER = 0                                                        LS02315
      GO TO (100, 100, 300, 400, 400), MITER                         LS02316
 100  CALL DGESL (WM(3), N, N, IWM(21), X, 0)                        LS02317
      RETURN                                                         LS02318
C                                                                    LS02319
 300  PHL0 = WM(2)                                                   LS02320
      HL0 = H*EL0                                                    LS02321
      WM(2) = HL0                                                    LS02322
      IF (HL0 .EQ. PHL0) GO TO 330                                   LS02323
      R = HL0/PHL0                                                   LS02324
      DO 320 I = 1,N                                                 LS02325
        DI = 1.0D0 - R*(1.0D0 - 1.0D0/WM(I+2))                       LS02326
        IF (DABS(DI) .EQ. 0.0D0) GO TO 390                           LS02327
 320    WM(I+2) = 1.0D0/DI                                           LS02328
 330  DO 340 I = 1,N                                                 LS02329
```

```
 340    X(I) = WM(I+2)*X(I)                                              LS02330
        RETURN                                                           LS02331
 390    IER = -1                                                         LS02332
        RETURN                                                           LS02333
C                                                                        LS02334
 400    ML = IWM(1)                                                      LS02335
        MU = IWM(2)                                                      LS02336
        MEBAND = 2*ML + MU + 1                                           LS02337
        CALL DGBSL (WM(3), MEBAND, N, ML, MU, IWM(21), X, 0)             LS02338
        RETURN                                                           LS02339
C----------------------- END OF SUBROUTINE SOLSY ----------------------LS02340
        END                                                              LS02341
        SUBROUTINE EWSET (N, ITOL, RTOL, ATOL, YCUR, EWT)                LS02342
C-----------------------------------------------------------------------LS02343
C THIS SUBROUTINE SETS THE ERROR WEIGHT VECTOR EWT ACCORDING TO          LS02344
C      EWT(I) = RTOL(I)*DABS(YCUR(I)) + ATOL(I),  I = 1,...,N,           LS02345
C WITH THE SUBSCRIPT ON RTOL AND/OR ATOL POSSIBLY REPLACED BY 1 ABOVE,   LS02346
C DEPENDING ON THE VALUE OF ITOL.                                        LS02347
C-----------------------------------------------------------------------LS02348
        INTEGER N, ITOL, I                                               LS02349
        DOUBLE PRECISION RTOL, ATOL, YCUR, EWT, ATOLI, RTOLI             LS02350
        DIMENSION RTOL(1), ATOL(1), YCUR(N), EWT(N)                      LS02351
        RTOLI = RTOL(1)                                                  LS02352
        ATOLI = ATOL(1)                                                  LS02353
        DO 10 I = 1,N                                                    LS02354
          IF (ITOL .GE. 3) RTOLI = RTOL(I)                               LS02355
          IF (ITOL .EQ. 2 .OR. ITOL .EQ. 4) ATOLI = ATOL(I)              LS02356
          EWT(I) = RTOLI*DABS(YCUR(I)) + ATOLI                           LS02357
 10       CONTINUE                                                       LS02358
        RETURN                                                           LS02359
C----------------------- END OF SUBROUTINE EWSET ----------------------LS02360
        END                                                              LS02361
        DOUBLE PRECISION FUNCTION VNORM (N, V, W)                        LS02362
C-----------------------------------------------------------------------LS02363
C THIS FUNCTION ROUTINE COMPUTES THE WEIGHTED ROOT-MEAN-SQUARE NORM      LS02364
C OF THE VECTOR OF LENGTH N CONTAINED IN THE ARRAY V, WITH WEIGHTS       LS02365
C CONTAINED IN THE ARRAY W OF LENGTH N..                                 LS02366
C   VNORM = SQRT( (1/N) * SUM( V(I)/W(I) )**2 )                         LS02367
C-----------------------------------------------------------------------LS02368
        INTEGER N, I                                                     LS02369
        DOUBLE PRECISION V, W, SUM                                       LS02370
        DIMENSION V(N), W(N)                                             LS02371
        SUM = 0.0D0                                                      LS02372
        DO 10 I = 1,N                                                    LS02373
 10       SUM = SUM + (V(I)/W(I))**2                                     LS02374
        VNORM = DSQRT(SUM/DFLOAT(N))                                     LS02375
        RETURN                                                           LS02376
C----------------------- END OF FUNCTION VNORM ------------------------LS02377
        END                                                              LS02378
        SUBROUTINE SVCOM (RSAV, ISAV)                                    LS02379
C-----------------------------------------------------------------------LS02380
C THIS ROUTINE STORES IN RSAV AND ISAV THE CONTENTS OF COMMON BLOCKS     LS02381
C LS0001 AND EH0001, WHICH ARE USED INTERNALLY IN THE LSODE PACKAGE.     LS02382
C                                                                        LS02383
C RSAV = REAL ARRAY OF LENGTH 218 OR MORE.                               LS02384
C ISAV = INTEGER ARRAY OF LENGTH 35 OR MORE.                             LS02385
C-----------------------------------------------------------------------LS02386
        INTEGER ISAV, I, IEH, ILS, LENILS, LENRLS                        LS02387
        DOUBLE PRECISION RSAV, RLS                                       LS02388
        DIMENSION RSAV(1), ISAV(1)                                       LS02389
```

```
      COMMON /LS0001/ RLS(218), ILS(33)                                 LS02390
      COMMON /EH0001/ IEH(2)                                            LS02391
      DATA LENRLS/218/, LENILS/33/                                      LS02392
C                                                                       LS02393
      DO 10 I = 1,LENRLS                                                LS02394
 10     RSAV(I) = RLS(I)                                                LS02395
      DO 20 I = 1,LENILS                                                LS02396
 20     ISAV(I) = ILS(I)                                                LS02397
      ISAV(LENILS+1) = IEH(1)                                           LS02398
      ISAV(LENILS+2) = IEH(2)                                           LS02399
      RETURN                                                            LS02400
C----------------------- END OF SUBROUTINE SVCOM ----------------------LS02401
      END                                                               LS02402
      SUBROUTINE RSCOM (RSAV, ISAV)                                     LS02403
C-----------------------------------------------------------------------LS02404
C THIS ROUTINE RESTORES FROM RSAV AND ISAV THE CONTENTS OF COMMON       LS02405
C BLOCKS LS0001 AND EH0001, WHICH ARE USED INTERNALLY IN THE LSODE      LS02406
C PACKAGE.  THIS PRESUMES THAT RSAV AND ISAV WERE LOADED BY MEANS       LS02407
C OF SUBROUTINE SVCOM OR THE EQUIVALENT.                                LS02408
C-----------------------------------------------------------------------LS02409
      INTEGER ISAV, I, IEH, ILS, LENILS, LENRLS                         LS02410
      DOUBLE PRECISION RSAV, RLS                                        LS02411
      DIMENSION RSAV(1), ISAV(1)                                        LS02412
      COMMON /LS0001/ RLS(218), ILS(33)                                 LS02413
      COMMON /EH0001/ IEH(2)                                            LS02414
      DATA LENRLS/218/, LENILS/33/                                      LS02415
C                                                                       LS02416
      DO 10 I = 1,LENRLS                                                LS02417
 10     RLS(I) = RSAV(I)                                                LS02418
      DO 20 I = 1,LENILS                                                LS02419
 20     ILS(I) = ISAV(I)                                                LS02420
      IEH(1) = ISAV(LENILS+1)                                           LS02421
      IEH(2) = ISAV(LENILS+2)                                           LS02422
      RETURN                                                            LS02423
C----------------------- END OF SUBROUTINE RSCOM ----------------------LS02424
      END                                                               LS02425
      SUBROUTINE DGEFA(A,LDA,N,IPVT,INFO)                               DG00001
      INTEGER LDA,N,IPVT(1),INFO                                        DG00002
      DOUBLE PRECISION A(LDA,1)                                         DG00003
C                                                                       DG00004
C     DGEFA FACTORS A DOUBLE PRECISION MATRIX BY GAUSSIAN ELIMINATION.  DG00005
C                                                                       DG00006
C     DGEFA IS USUALLY CALLED BY DGECO, BUT IT CAN BE CALLED            DG00007
C     DIRECTLY WITH A SAVING IN TIME IF  RCOND  IS NOT NEEDED.          DG00008
C     (TIME FOR DGECO) = (1 + 9/N)*(TIME FOR DGEFA) .                   DG00009
C                                                                       DG00010
C     ON ENTRY                                                          DG00011
C                                                                       DG00012
C        A       DOUBLE PRECISION(LDA, N)                             DG00013
C                THE MATRIX TO BE FACTORED.                           DG00014
C                                                                       DG00015
C        LDA     INTEGER                                              DG00016
C                THE LEADING DIMENSION OF THE ARRAY  A .              DG00017
C                                                                       DG00018
C        N       INTEGER                                              DG00019
C                THE ORDER OF THE MATRIX  A .                         DG00020
C                                                                       DG00021
C     ON RETURN                                                         DG00022
C                                                                       DG00023
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS       DG00024
```

```
C                 WHICH WERE USED TO OBTAIN IT.                          DG00025
C                 THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE       DG00026
C                 L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER          DG00027
C                 TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.       DG00028
C                                                                        DG00029
C         IPVT    INTEGER(N)                                             DG00030
C                 AN INTEGER VECTOR OF PIVOT INDICES.                    DG00031
C                                                                        DG00032
C         INFO    INTEGER                                                DG00033
C                 = 0  NORMAL VALUE.                                     DG00034
C                 = K  IF  U(K,K) .EQ. 0.0 .  THIS IS NOT AN ERROR       DG00035
C                      CONDITION FOR THIS SUBROUTINE, BUT IT DOES        DG00036
C                      INDICATE THAT DGESL OR DGEDI WILL DIVIDE BY ZERO  DG00037
C                      IF CALLED.  USE  RCOND  IN DGECO FOR A RELIABLE   DG00038
C                      INDICATION OF SINGULARITY.                        DG00039
C                                                                        DG00040
C     LINPACK. THIS VERSION DATED 08/14/78 .                             DG00041
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.       DG00042
C                                                                        DG00043
C     SUBROUTINES AND FUNCTIONS                                          DG00044
C                                                                        DG00045
C     BLAS DAXPY,DSCAL,IDAMAX                                            DG00046
C                                                                        DG00047
C     INTERNAL VARIABLES                                                 DG00048
C                                                                        DG00049
      DOUBLE PRECISION T                                                 DG00050
      INTEGER IDAMAX,J,K,KP1,L,NM1                                       DG00051
C                                                                        DG00052
C                                                                        DG00053
C     GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING                         DG00054
C                                                                        DG00055
      INFO = 0                                                           DG00056
      NM1 = N - 1                                                        DG00057
      IF (NM1 .LT. 1) GO TO 70                                           DG00058
      DO 60 K = 1, NM1                                                   DG00059
         KP1 = K + 1                                                     DG00060
C                                                                        DG00061
C        FIND L = PIVOT INDEX                                            DG00062
C                                                                        DG00063
         L = IDAMAX(N-K+1,A(K,K),1) + K - 1                              DG00064
         IPVT(K) = L                                                     DG00065
C                                                                        DG00066
C        ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED           DG00067
C                                                                        DG00068
         IF (A(L,K) .EQ. 0.0D0) GO TO 40                                 DG00069
C                                                                        DG00070
C           INTERCHANGE IF NECESSARY                                     DG00071
C                                                                        DG00072
            IF (L .EQ. K) GO TO 10                                       DG00073
               T = A(L,K)                                                DG00074
               A(L,K) = A(K,K)                                           DG00075
               A(K,K) = T                                                DG00076
   10       CONTINUE                                                     DG00077
C                                                                        DG00078
C           COMPUTE MULTIPLIERS                                          DG00079
C                                                                        DG00080
            T = -1.0D0/A(K,K)                                            DG00081
            CALL DSCAL(N-K,T,A(K+1,K),1)                                 DG00082
C                                                                        DG00083
C           ROW ELIMINATION WITH COLUMN INDEXING                         DG00084
```

```
C        OR DGEFA HAS SET INFO .EQ. 0 .                                  DG0014
C                                                                        DG0014
C     TO COMPUTE  INVERSE(A) * C  WHERE  C  IS A MATRIX                  DG0014
C     WITH  P  COLUMNS                                                   DG0014
C           CALL DGECO(A,LDA,N,IPVT,RCOND,Z)                             DG0014
C           IF (RCOND IS TOO SMALL) GO TO ...                            DG0015
C           DO 10 J = 1, P                                               DG0015
C              CALL DGESL(A,LDA,N,IPVT,C(1,J),0)                         DG0015
C        10 CONTINUE                                                     DG0015
C                                                                        DG0015
C     LINPACK. THIS VERSION DATED 08/14/78 .                             DG0015
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.       DG0015
C                                                                        DG0015
C     SUBROUTINES AND FUNCTIONS                                          DG0015
C                                                                        DG0015
C     BLAS DAXPY,DDOT                                                    DG0016
C                                                                        DG0016
C     INTERNAL VARIABLES                                                 DG0016
C                                                                        DG0016
      DOUBLE PRECISION DDOT,T                                            DG0016
      INTEGER K,KB,L,NM1                                                 DG0016
C                                                                        DG0016
      NM1 = N - 1                                                        DG0016
      IF (JOB .NE. 0) GO TO 50                                           DG0016
C                                                                        DG0016
C        JOB = 0 , SOLVE  A * X = B                                      DG0017
C        FIRST SOLVE  L*Y = B                                            DG0017
C                                                                        DG0017
         IF (NM1 .LT. 1) GO TO 30                                        DG0017
         DO 20 K = 1, NM1                                                DG0017
            L = IPVT(K)                                                  DG0017
            T = B(L)                                                     DG0017
            IF (L .EQ. K) GO TO 10                                       DG0017
               B(L) = B(K)                                               DG0017
               B(K) = T                                                  DG0017
   10       CONTINUE                                                     DG0018
            CALL DAXPY(N-K,T,A(K+1,K),1,B(K+1),1)                        DG0018
   20    CONTINUE                                                        DG0018
   30    CONTINUE                                                        DG0018
C                                                                        DG0018
C        NOW SOLVE  U*X = Y                                              DG0018
C                                                                        DG0018
         DO 40 KB = 1, N                                                 DG0018
            K = N + 1 - KB                                               DG0018
            B(K) = B(K)/A(K,K)                                           DG0018
            T = -B(K)                                                    DG0019
            CALL DAXPY(K-1,T,A(1,K),1,B(1),1)                            DG0019
   40    CONTINUE                                                        DG0019
      GO TO 100                                                          DG0019
   50 CONTINUE                                                           DG0019
C                                                                        DG0019
C        JOB = NONZERO, SOLVE  TRANS(A) * X = B                          DG0019
C        FIRST SOLVE  TRANS(U)*Y = B                                     DG0019
C                                                                        DG0019
         DO 60 K = 1, N                                                  DG0019
            T = DDOT(K-1,A(1,K),1,B(1),1)                                DG0020
            B(K) = (B(K) - T)/A(K,K)                                     DG0020
   60    CONTINUE                                                        DG0020
C                                                                        DG0020
C        NOW SOLVE TRANS(L)*X = Y                                        DG0020
```

```
C                                                                       DG00205
      IF (NM1 .LT. 1) GO TO 90                                          DG00206
      DO 80 KB = 1, NM1                                                 DG00207
         K = N - KB                                                     DG00208
         B(K) = B(K) + DDOT(N-K,A(K+1,K),1,B(K+1),1)                    DG00209
         L = IPVT(K)                                                    DG00210
         IF (L .EQ. K) GO TO 70                                         DG00211
            T = B(L)                                                    DG00212
            B(L) = B(K)                                                 DG00213
            B(K) = T                                                    DG00214
   70    CONTINUE                                                       DG00215
   80    CONTINUE                                                       DG00216
   90    CONTINUE                                                       DG00217
  100 CONTINUE                                                          DG00218
      RETURN                                                            DG00219
      END                                                               DG00220
      SUBROUTINE DGBFA(ABD,LDA,N,ML,MU,IPVT,INFO)                       DG00221
      INTEGER LDA,N,ML,MU,IPVT(1),INFO                                  DG00222
      DOUBLE PRECISION ABD(LDA,1)                                       DG00223
C                                                                       DG00224
C     DGBFA FACTORS A DOUBLE PRECISION BAND MATRIX BY ELIMINATION.      DG00225
C                                                                       DG00226
C     DGBFA IS USUALLY CALLED BY DGBCO, BUT IT CAN BE CALLED            DG00227
C     DIRECTLY WITH A SAVING IN TIME IF  RCOND  IS NOT NEEDED.          DG00228
C                                                                       DG00229
C     ON ENTRY                                                          DG00230
C                                                                       DG00231
C        ABD      DOUBLE PRECISION(LDA, N)                            DG00232
C                 CONTAINS THE MATRIX IN BAND STORAGE.  THE COLUMNS   DG00233
C                 OF THE MATRIX ARE STORED IN THE COLUMNS OF  ABD  AND DG00234
C                 THE DIAGONALS OF THE MATRIX ARE STORED IN ROWS      DG00235
C                 ML+1 THROUGH 2*ML+MU+1 OF  ABD .                    DG00236
C                 SEE THE COMMENTS BELOW FOR DETAILS.                 DG00237
C                                                                     DG00238
C        LDA      INTEGER                                             DG00239
C                 THE LEADING DIMENSION OF THE ARRAY  ABD .           DG00240
C                 LDA MUST BE .GE. 2*ML + MU + 1 .                    DG00241
C                                                                     DG00242
C        N        INTEGER                                             DG00243
C                 THE ORDER OF THE ORIGINAL MATRIX.                   DG00244
C                                                                     DG00245
C        ML       INTEGER                                             DG00246
C                 NUMBER OF DIAGONALS BELOW THE MAIN DIAGONAL.        DG00247
C                 0 .LE. ML .LT. N .                                  DG00248
C                                                                     DG00249
C        MU       INTEGER                                             DG00250
C                 NUMBER OF DIAGONALS ABOVE THE MAIN DIAGONAL.        DG00251
C                 0 .LE. MU .LT. N .                                  DG00252
C                 MORE EFFICIENT IF  ML .LE. MU .                     DG00253
C     ON RETURN                                                       DG00254
C                                                                     DG00255
C        ABD      AN UPPER TRIANGULAR MATRIX IN BAND STORAGE AND      DG00256
C                 THE MULTIPLIERS WHICH WERE USED TO OBTAIN IT.       DG00257
C                 THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE    DG00258
C                 L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER       DG00259
C                 TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.    DG00260
C                                                                     DG00261
C        IPVT     INTEGER(N)                                          DG00262
C                 AN INTEGER VECTOR OF PIVOT INDICES.                 DG00263
C                                                                     DG00264
```

```
C         INFO    INTEGER                                                DG00265
C                 = 0  NORMAL VALUE.                                     DG00266
C                 = K  IF  U(K,K) .EQ. 0.0 .  THIS IS NOT AN ERROR       DG00267
C                      CONDITION FOR THIS SUBROUTINE, BUT IT DOES        DG00268
C                      INDICATE THAT DGBSL WILL DIVIDE BY ZERO IF        DG00269
C                      CALLED.  USE  RCOND  IN DGBCO FOR A RELIABLE      DG00270
C                      INDICATE OF SINGULARITY.                          DG00271
C                                                                        DG00272
C     BAND STORAGE                                                       DG00273
C                                                                        DG00274
C           IF  A  IS A BAND MATRIX, THE FOLLOWING PROGRAM SEGMENT       DG00275
C           WILL SET UP THE INPUT.                                       DG00276
C                                                                        DG00277
C                   ML = (BAND WIDTH BELOW THE DIAGONAL)                 DG00278
C                   MU = (BAND WIDTH ABOVE THE DIAGONAL)                 DG00279
C                   M = ML + MU + 1                                      DG00280
C                   DO 20 J = 1, N                                       DG00281
C                      I1 = MAX0(1, J-MU)                                DG00282
C                      I2 = MIN0(N, J+ML)                                DG00283
C                      DO 10 I = I1, I2                                  DG00284
C                         K = I - J + M                                  DG00285
C                         ABD(K,J) = A(I,J)                              DG00286
C                10     CONTINUE                                         DG00287
C                20 CONTINUE                                             DG00288
C                                                                        DG00289
C           THIS USES ROWS  ML+1  THROUGH  2*ML+MU+1  OF  ABD .          DG00290
C           IN ADDITION, THE FIRST  ML  ROWS IN  ABD  ARE USED FOR       DG00291
C           ELEMENTS GENERATED DURING THE TRIANGULARIZATION.             DG00292
C           THE TOTAL NUMBER OF ROWS NEEDED IN  ABD  IS  2*ML+MU+1 .     DG00293
C           THE  ML+MU BY ML+MU  UPPER LEFT TRIANGLE AND THE             DG00294
C           ML BY ML  LOWER RIGHT TRIANGLE ARE NOT REFERENCED.           DG00295
C                                                                        DG00296
C     LINPACK. THIS VERSION DATED 08/14/78 .                             DG00297
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.       DG00298
C                                                                        DG00299
C     SUBROUTINES AND FUNCTIONS                                          DG00300
C                                                                        DG00301
C     BLAS DAXPY,DSCAL,IDAMAX                                            DG00302
C     FORTRAN MAX0,MIN0                                                  DG00303
C                                                                        DG00304
C     INTERNAL VARIABLES                                                 DG00305
C                                                                        DG00306
      DOUBLE PRECISION T                                                 DG00307
      INTEGER I,IDAMAX,I0,J,JU,JZ,J0,J1,K,KP1,L,LM,M,MM,NM1              DG00308
C                                                                        DG00309
C                                                                        DG00310
      M = ML + MU + 1                                                    DG00311
      INFO = 0                                                           DG00312
C                                                                        DG00313
C     ZERO INITIAL FILL-IN COLUMNS                                       DG00314
C                                                                        DG00315
      J0 = MU + 2                                                        DG00316
      J1 = MIN0(N,M) - 1                                                 DG00317
      IF (J1 .LT. J0) GO TO 30                                           DG00318
      DO 20 JZ = J0, J1                                                  DG00319
         I0 = M + 1 - JZ                                                 DG00320
         DO 10 I = I0, ML                                                DG00321
            ABD(I,JZ) = 0.0D0                                            DG00322
   10    CONTINUE                                                        DG00323
   20 CONTINUE                                                           DG00324
```

```
   30 CONTINUE                                                          DG0032
      JZ = J1                                                           DG0032
      JU = 0                                                            DG0032
C                                                                       DG0032
C     GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING                        DG0032
C                                                                       DG0033
      NM1 = N - 1                                                       DG0033
      IF (NM1 .LT. 1) GO TO 130                                         DG0033
      DO 120 K = 1, NM1                                                 DG0033
         KP1 = K + 1                                                    DG0033
C                                                                       DG0033
C        ZERO NEXT FILL-IN COLUMN                                       DG0033
C                                                                       DG0033
         JZ = JZ + 1                                                    DG0033
         IF (JZ .GT. N) GO TO 50                                        DG0033
         IF (ML .LT. 1) GO TO 50                                        DG0034
            DO 40 I = 1, ML                                             DG0034
               ABD(I,JZ) = 0.0D0                                        DG0034
   40       CONTINUE                                                    DG0034
   50    CONTINUE                                                       DG0034
C                                                                       DG0034
C        FIND L = PIVOT INDEX                                           DG0034
C                                                                       DG0034
         LM = MIN0(ML,N-K)                                              DG0034
         L = IDAMAX(LM+1,ABD(M,K),1) + M - 1                            DG0034
         IPVT(K) = L + K - M                                            DG0035
C                                                                       DG0035
C        ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED          DG0035
C                                                                       DG0035
         IF (ABD(L,K) .EQ. 0.0D0) GO TO 100                             DG0035
C                                                                       DG0035
C           INTERCHANGE IF NECESSARY                                    DG0035
C                                                                       DG0035
            IF (L .EQ. M) GO TO 60                                      DG0035
               T = ABD(L,K)                                             DG0035
               ABD(L,K) = ABD(M,K)                                      DG0036
               ABD(M,K) = T                                             DG0036
   60       CONTINUE                                                    DG0036
C                                                                       DG0036
C           COMPUTE MULTIPLIERS                                         DG0036
C                                                                       DG0036
            T = -1.0D0/ABD(M,K)                                         DG0036
            CALL DSCAL(LM,T,ABD(M+1,K),1)                               DG0036
C                                                                       DG0036
C           ROW ELIMINATION WITH COLUMN INDEXING                        DG0036
C                                                                       DG0037
            JU = MIN0(MAX0(JU,MU+IPVT(K)),N)                            DG0037
            MM = M                                                      DG0037
            IF (JU .LT. KP1) GO TO 90                                   DG0037
            DO 80 J = KP1, JU                                           DG0037
               L = L - 1                                                DG0037
               MM = MM - 1                                              DG0037
               T = ABD(L,J)                                             DG0037
               IF (L .EQ. MM) GO TO 70                                  DG0037
                  ABD(L,J) = ABD(MM,J)                                  DG0037
                  ABD(MM,J) = T                                         DG0038
   70          CONTINUE                                                 DG0038
               CALL DAXPY(LM,T,ABD(M+1,K),1,ABD(MM+1,J),1)              DG0038
   80       CONTINUE                                                    DG0038
   90       CONTINUE                                                    DG0038
```

```
            GO TO 110                                                   DG0038
  100       CONTINUE                                                    DG0038
               INFO = K                                                 DG0038
  110       CONTINUE                                                    DG0038
  120    CONTINUE                                                       DG0038
  130    CONTINUE                                                       DG0039
         IPVT(N) = N                                                    DG0039
         IF (ABD(M,N) .EQ. 0.0D0) INFO = N                              DG0039
         RETURN                                                         DG0039
         END                                                            DG0039
         SUBROUTINE DGBSL(ABD,LDA,N,ML,MU,IPVT,B,JOB)                   DG0039
         INTEGER LDA,N,ML,MU,IPVT(1),JOB                                DG0039
         DOUBLE PRECISION ABD(LDA,1),B(1)                               DG0039
C                                                                       DG0039
C        DGBSL SOLVES THE DOUBLE PRECISION BAND SYSTEM                  DG0039
C        A * X = B  OR  TRANS(A) * X = B                                DG0040
C        USING THE FACTORS COMPUTED BY DGBCO OR DGBFA.                  DG0040
C                                                                       DG0040
C        ON ENTRY                                                       DG0040
C                                                                       DG0040
C           ABD      DOUBLE PRECISION(LDA, N)                          DG0040
C                    THE OUTPUT FROM DGBCO OR DGBFA.                   DG0040
C                                                                       DG0040
C           LDA      INTEGER                                           DG0040
C                    THE LEADING DIMENSION OF THE ARRAY  ABD .         DG0040
C                                                                       DG0041
C           N        INTEGER                                           DG0041
C                    THE ORDER OF THE ORIGINAL MATRIX.                 DG0041
C                                                                       DG0041
C           ML       INTEGER                                           DG0041
C                    NUMBER OF DIAGONALS BELOW THE MAIN DIAGONAL.      DG0041
C                                                                       DG0041
C           MU       INTEGER                                           DG0041
C                    NUMBER OF DIAGONALS ABOVE THE MAIN DIAGONAL.      DG0041
C                                                                       DG0041
C           IPVT     INTEGER(N)                                        DG0042
C                    THE PIVOT VECTOR FROM DGBCO OR DGBFA.             DG0042
C                                                                       DG0042
C           B        DOUBLE PRECISION(N)                               DG0042
C                    THE RIGHT HAND SIDE VECTOR.                       DG0042
C                                                                       DG0042
C           JOB      INTEGER                                           DG0042
C                    = 0          TO SOLVE  A*X = B ,                  DG0042
C                    = NONZERO    TO SOLVE  TRANS(A)*X = B , WHERE     DG0042
C                                 TRANS(A)  IS THE TRANSPOSE.          DG0042
C                                                                       DG0043
C        ON RETURN                                                      DG0043
C                                                                       DG0043
C           B        THE SOLUTION VECTOR  X .                          DG0043
C                                                                       DG0043
C        ERROR CONDITION                                                DG0043
C                                                                       DG0043
C           A DIVISION BY ZERO WILL OCCUR IF THE INPUT FACTOR CONTAINS A DG0043
C           ZERO ON THE DIAGONAL.  TECHNICALLY THIS INDICATES SINGULARITY DG0043
C           BUT IT IS OFTEN CAUSED BY IMPROPER ARGUMENTS OR IMPROPER     DG0043
C           SETTING OF LDA .  IT WILL NOT OCCUR IF THE SUBROUTINES ARE   DG0044
C           CALLED CORRECTLY AND IF DGBCO HAS SET RCOND .GT. 0.0         DG0044
C           OR DGBFA HAS SET INFO .EQ. 0 .                               DG0044
C                                                                        DG0044
C        TO COMPUTE  INVERSE(A) * C  WHERE  C  IS A MATRIX               DG0044
```

```
C     WITH  P  COLUMNS                                                        DG00445
C           CALL DGBCO(ABD,LDA,N,ML,MU,IPVT,RCOND,Z)                          DG00446
C           IF (RCOND IS TOO SMALL) GO TO ...                                 DG00447
C           DO 10 J = 1, P                                                    DG00448
C              CALL DGBSL(ABD,LDA,N,ML,MU,IPVT,C(1,J),0)                      DG00449
C        10 CONTINUE                                                         DG00450
C                                                                            DG00451
C     LINPACK. THIS VERSION DATED 08/14/78 .                                 DG00452
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.           DG00453
C                                                                            DG00454
C     SUBROUTINES AND FUNCTIONS                                              DG00455
C                                                                            DG00456
C     BLAS DAXPY,DDOT                                                        DG00457
C     FORTRAN MIN0                                                           DG00458
C                                                                            DG00459
C     INTERNAL VARIABLES                                                     DG00460
C                                                                            DG00461
      DOUBLE PRECISION DDOT,T                                                DG00462
      INTEGER K,KB,L,LA,LB,LM,M,NM1                                          DG00463
C                                                                            DG00464
      M = MU + ML + 1                                                        DG00465
      NM1 = N - 1                                                            DG00466
      IF (JOB .NE. 0) GO TO 50                                               DG00467
C                                                                            DG00468
C        JOB = 0 , SOLVE  A * X = B                                          DG00469
C        FIRST SOLVE L*Y = B                                                 DG00470
C                                                                            DG00471
         IF (ML .EQ. 0) GO TO 30                                             DG00472
         IF (NM1 .LT. 1) GO TO 30                                            DG00473
            DO 20 K = 1, NM1                                                 DG00474
               LM = MIN0(ML,N-K)                                             DG00475
               L = IPVT(K)                                                   DG00476
               T = B(L)                                                      DG00477
               IF (L .EQ. K) GO TO 10                                        DG00478
                  B(L) = B(K)                                                DG00479
                  B(K) = T                                                   DG00480
   10          CONTINUE                                                      DG00481
               CALL DAXPY(LM,T,ABD(M+1,K),1,B(K+1),1)                        DG00482
   20       CONTINUE                                                         DG00483
   30    CONTINUE                                                            DG00484
C                                                                            DG00485
C        NOW SOLVE  U*X = Y                                                  DG00486
C                                                                            DG00487
         DO 40 KB = 1, N                                                     DG00488
            K = N + 1 - KB                                                   DG00489
            B(K) = B(K)/ABD(M,K)                                             DG00490
            LM = MIN0(K,M) - 1                                               DG00491
            LA = M - LM                                                      DG00492
            LB = K - LM                                                      DG00493
            T = -B(K)                                                        DG00494
            CALL DAXPY(LM,T,ABD(LA,K),1,B(LB),1)                             DG00495
   40    CONTINUE                                                            DG00496
      GO TO 100                                                              DG00497
   50 CONTINUE                                                               DG00498
C                                                                            DG00499
C        JOB = NONZERO, SOLVE  TRANS(A) * X = B                              DG00500
C        FIRST SOLVE  TRANS(U)*Y = B                                         DG00501
C                                                                            DG00502
         DO 60 K = 1, N                                                      DG00503
            LM = MIN0(K,M) - 1                                               DG00504
```

```
            LA = M - LM                                                    DG00505
            LB = K - LM                                                    DG00506
            T = DDOT(LM,ABD(LA,K),1,B(LB),1)                               DG00507
            B(K) = (B(K) - T)/ABD(M,K)                                     DG00508
   60     CONTINUE                                                         DG00509
C                                                                         DG00510
C        NOW SOLVE TRANS(L)*X = Y                                         DG00511
C                                                                         DG00512
         IF (ML .EQ. 0) GO TO 90                                          DG00513
         IF (NM1 .LT. 1) GO TO 90                                         DG00514
            DO 80 KB = 1, NM1                                             DG00515
               K = N - KB                                                 DG00516
               LM = MIN0(ML,N-K)                                          DG00517
               B(K) = B(K) + DDOT(LM,ABD(M+1,K),1,B(K+1),1)               DG00518
               L = IPVT(K)                                                DG00519
               IF (L .EQ. K) GO TO 70                                     DG00520
                  T = B(L)                                                DG00521
                  B(L) = B(K)                                             DG00522
                  B(K) = T                                                DG00523
   70          CONTINUE                                                   DG00524
   80       CONTINUE                                                      DG00525
   90    CONTINUE                                                         DG00526
  100 CONTINUE                                                            DG00527
      RETURN                                                              DG00528
      END                                                                 DG00529
      SUBROUTINE DAXPY(N,DA,DX,INCX,DY,INCY)                              DG00530
C                                                                         DG00531
C     CONSTANT TIMES A VECTOR PLUS A VECTOR.                              DG00532
C     USES UNROLLED LOOPS FOR INCREMENTS EQUAL TO ONE.                    DG00533
C     JACK DONGARRA, LINPACK, 3/11/78.                                    DG00534
C                                                                         DG00535
      DOUBLE PRECISION DX(1),DY(1),DA                                     DG00536
      INTEGER I,INCX,INCY,IX,IY,M,MP1,N                                   DG00537
C                                                                         DG00538
      IF(N.LE.0)RETURN                                                    DG00539
      IF (DA .EQ. 0.0D0) RETURN                                           DG00540
      IF(INCX.EQ.1.AND.INCY.EQ.1)GO TO 20                                 DG00541
C                                                                         DG00542
C        CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS                  DG00543
C          NOT EQUAL TO 1                                                 DG00544
C                                                                         DG00545
      IX = 1                                                              DG00546
      IY = 1                                                              DG00547
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1                                   DG00548
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1                                   DG00549
      DO 10 I = 1,N                                                       DG00550
        DY(IY) = DY(IY) + DA*DX(IX)                                       DG00551
        IX = IX + INCX                                                    DG00552
        IY = IY + INCY                                                    DG00553
   10 CONTINUE                                                            DG00554
      RETURN                                                              DG00555
C                                                                         DG00556
C        CODE FOR BOTH INCREMENTS EQUAL TO 1                              DG00557
C                                                                         DG00558
C                                                                         DG00559
C        CLEAN-UP LOOP                                                    DG00560
C                                                                         DG00561
   20 M = MOD(N,4)                                                        DG00562
      IF( M .EQ. 0 ) GO TO 40                                             DG00563
      DO 30 I = 1,M                                                       DG00564
```

```
        DY(I) = DY(I) + DA*DX(I)                                        DG00565
   30 CONTINUE                                                          DG00566
      IF( N .LT. 4 ) RETURN                                             DG00567
   40 MP1 = M + 1                                                       DG00568
      DO 50 I = MP1,N,4                                                 DG00569
        DY(I) = DY(I) + DA*DX(I)                                        DG00570
        DY(I + 1) = DY(I + 1) + DA*DX(I + 1)                            DG00571
        DY(I + 2) = DY(I + 2) + DA*DX(I + 2)                            DG00572
        DY(I + 3) = DY(I + 3) + DA*DX(I + 3)                            DG00573
   50 CONTINUE                                                          DG00574
      RETURN                                                            DG00575
      END                                                               DG00576
      SUBROUTINE DSCAL(N,DA,DX,INCX)                                    DG00577
C                                                                       DG00578
C     SCALES A VECTOR BY A CONSTANT.                                    DG00579
C     USES UNROLLED LOOPS FOR INCREMENT EQUAL TO ONE.                   DG00580
C     JACK DONGARRA, LINPACK, 3/11/78.                                  DG00581
C                                                                       DG00582
      DOUBLE PRECISION DA,DX(1)                                         DG00583
      INTEGER I,INCX,M,MP1,N,NINCX                                      DG00584
C                                                                       DG00585
      IF(N.LE.0)RETURN                                                  DG00586
      IF(INCX.EQ.1)GO TO 20                                             DG00587
C                                                                       DG00588
C        CODE FOR INCREMENT NOT EQUAL TO 1                              DG00589
C                                                                       DG00590
      NINCX = N*INCX                                                    DG00591
      DO 10 I = 1,NINCX,INCX                                            DG00592
        DX(I) = DA*DX(I)                                                DG00593
   10 CONTINUE                                                          DG00594
      RETURN                                                            DG00595
C                                                                       DG00596
C        CODE FOR INCREMENT EQUAL TO 1                                  DG00597
C                                                                       DG00598
C                                                                       DG00599
C        CLEAN-UP LOOP                                                  DG00600
C                                                                       DG00601
   20 M = MOD(N,5)                                                      DG00602
      IF( M .EQ. 0 ) GO TO 40                                           DG00603
      DO 30 I = 1,M                                                     DG00604
        DX(I) = DA*DX(I)                                                DG00605
   30 CONTINUE                                                          DG00606
      IF( N .LT. 5 ) RETURN                                             DG00607
   40 MP1 = M + 1                                                       DG00608
      DO 50 I = MP1,N,5                                                 DG00609
        DX(I) = DA*DX(I)                                                DG00610
        DX(I + 1) = DA*DX(I + 1)                                        DG00611
        DX(I + 2) = DA*DX(I + 2)                                        DG00612
        DX(I + 3) = DA*DX(I + 3)                                        DG00613
        DX(I + 4) = DA*DX(I + 4)                                        DG00614
   50 CONTINUE                                                          DG00615
      RETURN                                                            DG00616
      END                                                               DG00617
      DOUBLE PRECISION FUNCTION DDOT(N,DX,INCX,DY,INCY)                 DG00618
C                                                                       DG00619
C     FORMS THE DOT PRODUCT OF TWO VECTORS.                             DG00620
C     USES UNROLLED LOOPS FOR INCREMENTS EQUAL TO ONE.                  DG00621
C     JACK DONGARRA, LINPACK, 3/11/78.                                  DG00622
C                                                                       DG00623
      DOUBLE PRECISION DX(1),DY(1),DTEMP                                DG00624
```

```
      INTEGER I,INCX,INCY,IX,IY,M,MP1,N                                 DG00625
C                                                                       DG00626
      DDOT = 0.0D0                                                      DG00627
      DTEMP = 0.0D0                                                     DG00628
      IF(N.LE.0)RETURN                                                  DG00629
      IF(INCX.EQ.1.AND.INCY.EQ.1)GO TO 20                               DG00630
C                                                                       DG00631
C        CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS                DG00632
C          NOT EQUAL TO 1                                               DG00633
C                                                                       DG00634
      IX = 1                                                            DG00635
      IY = 1                                                            DG00636
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1                                 DG00637
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1                                 DG00638
      DO 10 I = 1,N                                                     DG00639
        DTEMP = DTEMP + DX(IX)*DY(IY)                                   DG00640
        IX = IX + INCX                                                  DG00641
        IY = IY + INCY                                                  DG00642
   10 CONTINUE                                                          DG00643
      DDOT = DTEMP                                                      DG00644
      RETURN                                                            DG00645
C                                                                       DG00646
C        CODE FOR BOTH INCREMENTS EQUAL TO 1                            DG00647
C                                                                       DG00648
C                                                                       DG00649
C        CLEAN-UP LOOP                                                  DG00650
C                                                                       DG00651
   20 M = MOD(N,5)                                                      DG00652
      IF( M .EQ. 0 ) GO TO 40                                           DG00653
      DO 30 I = 1,M                                                     DG00654
        DTEMP = DTEMP + DX(I)*DY(I)                                     DG00655
   30 CONTINUE                                                          DG00656
      IF( N .LT. 5 ) GO TO 60                                           DG00657
   40 MP1 = M + 1                                                       DG00658
      DO 50 I = MP1,N,5                                                 DG00659
        DTEMP = DTEMP + DX(I)*DY(I) + DX(I + 1)*DY(I + 1) +             DG00660
     *   DX(I + 2)*DY(I + 2) + DX(I + 3)*DY(I + 3) + DX(I + 4)*DY(I + 4)DG00661
   50 CONTINUE                                                          DG00662
   60 DDOT = DTEMP                                                      DG00663
      RETURN                                                            DG00664
      END                                                               DG00665
      INTEGER FUNCTION IDAMAX(N,DX,INCX)                                DG00666
C                                                                       DG00667
C     FINDS THE INDEX OF ELEMENT HAVING MAX. ABSOLUTE VALUE.            DG00668
C     JACK DONGARRA, LINPACK, 3/11/78.                                  DG00669
C                                                                       DG00670
      DOUBLE PRECISION DX(1),DMAX                                       DG00671
      INTEGER I,INCX,IX,N                                               DG00672
C                                                                       DG00673
      IDAMAX = 0                                                        DG00674
      IF( N .LT. 1 ) RETURN                                             DG00675
      IDAMAX = 1                                                        DG00676
      IF(N.EQ.1)RETURN                                                  DG00677
      IF(INCX.EQ.1)GO TO 20                                             DG00678
C                                                                       DG00679
C        CODE FOR INCREMENT NOT EQUAL TO 1                              DG00680
C                                                                       DG00681
      IX = 1                                                            DG00682
      DMAX = DABS(DX(1))                                                DG00683
      IX = IX + INCX                                                    DG00684
```

```
      DO 10 I = 2,N                                                     DG0068
         IF(DABS(DX(IX)).LE.DMAX) GO TO 5                               DG0068
         IDAMAX = I                                                     DG0068
         DMAX = DABS(DX(IX))                                            DG0068
    5    IX = IX + INCX                                                 DG0068
   10 CONTINUE                                                          DG0069
      RETURN                                                            DG0069
C                                                                       DG0069
C        CODE FOR INCREMENT EQUAL TO 1                                  DG0069
C                                                                       DG0069
   20 DMAX = DABS(DX(1))                                                DG0069
      DO 30 I = 2,N                                                     DG0069
         IF(DABS(DX(I)).LE.DMAX) GO TO 30                               DG0069
         IDAMAX = I                                                     DG0069
         DMAX = DABS(DX(I))                                             DG0069
   30 CONTINUE                                                          DG0070
      RETURN                                                            DG0070
      END                                                               DG0070
      DOUBLE PRECISION FUNCTION D1MACH (IDUM)                           D1MA001
      INTEGER IDUM                                                      D1MA002
C-----------------------------------------------------------------------D1MA003
C THIS ROUTINE COMPUTES THE UNIT ROUNDOFF OF THE MACHINE IN DOUBLE      D1MA004
C PRECISION.  THIS IS DEFINED AS THE SMALLEST POSITIVE MACHINE NUMBER   D1MA005
C U SUCH THAT  1.0D0 + U .NE. 1.0D0 (IN DOUBLE PRECISION).              D1MA006
C-----------------------------------------------------------------------D1MA007
      DOUBLE PRECISION U, COMP                                          D1MA008
      U = 1.0D0                                                         D1MA009
 10   U = U*0.5D0                                                       D1MA010
      COMP = 1.0D0 + U                                                  D1MA011
      IF (COMP .NE. 1.0D0) GO TO 10                                     D1MA012
      D1MACH = U*2.0D0                                                  D1MA013
      RETURN                                                            D1MA014
C----------------------- END OF FUNCTION D1MACH ------------------------D1MA015
      END                                                               D1MA016
      SUBROUTINE XERRWV (MSG, NMES, NERR, IERT, NI, I1, I2, NR, R1, R2) EHP0001
      INTEGER MSG, NMES, NERR, IERT, NI, I1, I2, NR,                    EHP0002
     1   I, LUN, LUNIT, MESFLG, NWDS                                    EHP0003
      DOUBLE PRECISION R1, R2                                           EHP0004
      DIMENSION MSG(NMES)                                               EHP0005
C-----------------------------------------------------------------------EHP0006
C SUBROUTINES XERRWV, XSETF, AND XSETUN, AS GIVEN HERE, CONSTITUTE      EHP0007
C A SIMPLIFIED VERSION OF THE SLATEC ERROR HANDLING PACKAGE.            EHP0008
C WRITTEN BY A. C. HINDMARSH AT LLL.  VERSION OF JANUARY 23, 1980.      EHP0009
C THIS VERSION IS IN DOUBLE PRECISION.                                  EHP0010
C                                                                       EHP0011
C ALL ARGUMENTS ARE INPUT ARGUMENTS.                                    EHP0012
C                                                                       EHP0013
C MSG    = THE MESSAGE (HOLLERITH LITTERAL OR INTEGER ARRAY).           EHP0014
C NMES   = THE LENGTH OF MSG (NUMBER OF CHARACTERS).                    EHP0015
C NERR   = THE ERROR NUMBER (NOT USED).                                 EHP0016
C IERT   = THE ERROR TYPE..                                             EHP0017
C          1 MEANS RECOVERABLE (CONTROL RETURNS TO CALLER).             EHP0018
C          2 MEANS FATAL (RUN IS ABORTED--SEE NOTE BELOW).              EHP0019
C NI     = NUMBER OF INTEGERS (0, 1, OR 2) TO BE PRINTED WITH MESSAGE.  EHP0020
C I1,I2  = INTEGERS TO BE PRINTED, DEPENDING ON NI.                     EHP0021
C NR     = NUMBER OF REALS (0, 1, OR 2) TO BE PRINTED WITH MESSAGE.     EHP0022
C R1,R2  = REALS TO BE PRINTED, DEPENDING ON NI.                        EHP0023
C                                                                       EHP0024
C NOTE..  THIS ROUTINE IS MACHINE-DEPENDENT AND SPECIALIZED FOR USE     EHP0025
C IN LIMITED CONTEXT, IN THE FOLLOWING WAYS..                           EHP0026
```

```
C 1. THE NUMBER OF HOLLERITH CHARACTERS STORED PER WORD IS ASSUMED      EHP0027
C    TO BE 10.                                                         EHP0028
C 2. THE VALUE OF NMES IS ASSUMED TO BE A MULTIPLE OF 10 AND AT MOST 60.EHP0029
C    (MULTI-LINE MESSAGES ARE GENERATED BY REPEATED CALLS.)            EHP0030
C 3. IF IERT = 2, CONTROL PASSES TO THE STATEMENT   STOP               EHP0031
C    TO ABORT THE RUN.  THIS STATEMENT MAY BE MACHINE-DEPENDENT.       EHP0032
C 4. R1 AND R2 ARE ASSUMED TO BE IN DOUBLE PRECISION AND ARE PRINTED   EHP0033
C    IN D21.13 FORMAT.                                                 EHP0034
C 5. THE COMMON BLOCK /EH0001/ BELOW IS DATA-LOADED (A MACHINE-        EHP0035
C    DEPENDENT FEATURE) WITH DEFAULT VALUES.                           EHP0036
C    THIS BLOCK IS NEEDED FOR PROPER RETENTION OF PARAMETERS USED BY   EHP0037
C    THIS ROUTINE WHICH THE USER CAN RESET BY CALLING XSETF OR XSETUN. EHP0038
C    THE VARIABLES IN THIS BLOCK ARE AS FOLLOWS..                      EHP0039
C       MESFLG = PRINT CONTROL FLAG..                                  EHP0040
C                1 MEANS PRINT ALL MESSAGES (THE DEFAULT).             EHP0041
C                0 MEANS NO PRINTING.                                  EHP0042
C       LUNIT  = LOGICAL UNIT NUMBER FOR MESSAGES.                     EHP0043
C                THE DEFAULT IS 6 (MACHINE-DEPENDENT).                 EHP0044
C-----------------------------------------------------------------------EHP0045
C THE FOLLOWING ARE INSTRUCTIONS FOR INSTALLING THIS ROUTINE           EHP0046
C IN DIFFERENT MACHINE ENVIRONMENTS.                                   EHP0047
C                                                                      EHP0048
C TO CHANGE THE DEFAULT OUTPUT UNIT, CHANGE THE DATA STATEMENT         EHP0049
C IN THE BLOCK DATA SUBPROGRAM BELOW.                                  EHP0050
C                                                                      EHP0051
C FOR A DIFFERENT NUMBER OF CHARACTERS PER WORD, CHANGE THE            EHP0052
C STATEMENTS SETTING NWDS BELOW, AND FORMAT 10.                        EHP0053
C                                                                      EHP0054
C FOR A DIFFERENT RUN-ABORT COMMAND, CHANGE THE STATEMENT FOLLOWING    EHP0055
C STATEMENT 100 AT THE END.                                            EHP0056
C-----------------------------------------------------------------------EHP0057
      COMMON /EH0001/ MESFLG, LUNIT                                    EHP0058
C                                                                      EHP0059
      IF (MESFLG .EQ. 0) GO TO 100                                     EHP0060
C GET LOGICAL UNIT NUMBER. -------------------------------------------EHP0061
      LUN = LUNIT                                                      EHP0062
C GET NUMBER OF WORDS IN MESSAGE. ------------------------------------EHP0063
      NWDS = NMES/10                                                   EHP0064
      NWDS = MIN0(NWDS,6)                                              EHP0065
C WRITE THE MESSAGE. -------------------------------------------------EHP0066
      WRITE (LUN, 10) (MSG(I),I=1,NWDS)                                EHP0067
 10   FORMAT(6A10)                                                     EHP0068
      IF (NI .EQ. 1) WRITE (LUN, 20) I1                                EHP0069
 20   FORMAT(6X,23HIN ABOVE MESSAGE,  I1 =,I10)                        EHP0070
      IF (NI .EQ. 2) WRITE (LUN, 30) I1,I2                             EHP0071
 30   FORMAT(6X,23HIN ABOVE MESSAGE,  I1 =,I10,3X,4HI2 =,I10)          EHP0072
      IF (NR .EQ. 1) WRITE (LUN, 40) R1                                EHP0073
 40   FORMAT(6X,23HIN ABOVE MESSAGE,  R1 =,D21.13)                     EHP0074
      IF (NR .EQ. 2) WRITE (LUN, 50) R1,R2                             EHP0075
 50   FORMAT(6X,15HIN ABOVE,  R1 =,D21.13,3X,4HR2 =,D21.13)            EHP0076
C ABORT THE RUN IF IERT = 2. -----------------------------------------EHP0077
 100  IF (IERT .NE. 2) RETURN                                          EHP0078
      CALL EXIT
      STOP                                                             EHP0079
C---------------------- END OF SUBROUTINE XERRWV ----------------------EHP0080
      END                                                              EHP0081
      SUBROUTINE XSETF (MFLAG)                                         EHP0082
C                                                                      EHP0083
C THIS ROUTINE RESETS THE PRINT CONTROL FLAG MFLAG.                    EHP0084
C                                                                      EHP0085
```

```
      INTEGER MFLAG, MESFLG, LUNIT                                      EHP008
      COMMON /EH0001/ MESFLG, LUNIT                                     EHP008
C                                                                       EHP008
      IF (MFLAG .EQ. 0 .OR. MFLAG .EQ. 1) MESFLG = MFLAG                EHP008
      RETURN                                                            EHP009
C---------------------- END OF SUBROUTINE XSETF ----------------------EHP009
      END                                                               EHP009
      SUBROUTINE XSETUN (LUN)                                           EHP009
C                                                                       EHP009
C THIS ROUTINE RESETS THE LOGICAL UNIT NUMBER FOR MESSAGES.             EHP009
C                                                                       EHP009
      INTEGER LUN, MESFLG, LUNIT                                        EHP009
      COMMON /EH0001/ MESFLG, LUNIT                                     EHP009
C                                                                       EHP009
      IF (LUN .GT. 0) LUNIT = LUN                                       EHP010
      RETURN                                                            EHP010
C---------------------- END OF SUBROUTINE XSETUN ---------------------EHP010
      END                                                               EHP010
      BLOCK DATA                                                        BLDA00
C-----------------------------------------------------------------------BLDA00
C THE FOLLOWING DATA STATEMENT LOADS VARIABLES INTO THE INTERNAL COMMON BLDA00
C BLOCKS USED BY LSODE.  THE VARIABLES ARE DEFINED AS FOLLOWS..         BLDA00
C   ILLIN  = COUNTER FOR THE NUMBER OF CONSECUTIVE TIMES LSODE WAS      BLDA00
C            CALLED WITH ILLEGAL INPUT.  THE RUN IS STOPPED WHEN        BLDA00
C            ILLIN REACHES 5.                                           BLDA00
C   NTREP  = COUNTER FOR THE NUMBER OF CONSECUTIVE TIMES LSODE WAS      BLDA00
C            CALLED WITH ISTATE = 1 AND TOUT = T.  THE RUN IS STOPPED   BLDA00
C            WHEN NTREP REACHES 5.                                      BLDA01
C   MESFLG = FLAG TO CONTROL PRINTING OF ERROR MESSAGES.  1 MEANS PRINT,BLDA01
C            0 MEANS NO PRINTING.                                       BLDA01
C   LUNIT  = DEFAULT VALUE OF LOGICAL UNIT NUMBER FOR PRINTING OF ERROR BLDA01
C            MESSAGES.                                                  BLDA01
C-----------------------------------------------------------------------BLDA01
      INTEGER ILLIN, IDUMA, NTREP, IDUMB, IOWNS, ICOMM, MESFLG, LUNIT   BLDA01
      DOUBLE PRECISION ROWND, ROWNS, RCOMM                              BLDA01
      COMMON /LS0001/ ROWND, ROWNS(210), RCOMM(7),                      BLDA01
     1   ILLIN, IDUMA(10), NTREP, IDUMB(2), IOWNS(6), ICOMM(13)         BLDA01
      COMMON /EH0001/ MESFLG, LUNIT                                     BLDA02
      DATA ILLIN/0/, NTREP/0/                                           BLDA02
      DATA MESFLG/1/, LUNIT/6/                                          BLDA02
C                                                                       BLDA02
C---------------------- END OF BLOCK DATA ----------------------------BLDA02
      END                                                               BLDA02
      SUBROUTINE WIPE ( A, IMAX )
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C     WIPE ... ZEROS OUT THE ARRAY A(*)
C
      DIMENSION A(1)
      DO 10 I=1,IMAX
   10 A(I) = 0.0
      RETURN
      END
C
      SUBROUTINE IWIPE ( IA, IMAX )
C     WIPE ... ZEROS OUT THE ARRAY IA(*)
C
      DIMENSION IA(1)
      DO 10 I=1,IMAX
   10 IA(I) = 0
```

```
      RETURN
      END
      FUNCTION ISIMEQ  ( B, M, N, N1 )
      IMPLICIT REAL*8(A-H,O-Z)
C
C     DEFINITION OF TERMS
C        B  = MATRIX PLUS RHS
C        M  = SIZE OF MATRIX, M => N
C        N  = NUMBER OF EQUATIONS IN "B"
C        N1 = NUMBER OF RHS'S
C        NOTE:  MATRIX AND RHS DO NOT HAVE TO BE REARRANGED
C               TO ELIMINATE VOID SPACE
C
      REAL          B(1),     COL(200), PIVOT,    RATIO
      INTEGER       ROW(200), PR,       PC,       ISIMEQ
C
C     PR = ROW OF PIVOT.
C     PC = COLUMN OF PIVOT.
C     PIVOT = VALUE OF THE MATRIX PIVOT.
C
C
C     ZERO ROW AND COL, AND SET CONTROLS.
C
C     ISIMEQ = COMPUTATION CODE
C        0     INITIAL VALUE ... NO MEANING
C        1     COMPUTATION SUCCESSFUL
C        2     ERROR ... PIVOT IS EQUAL TO 0.0
C
      ISIMEQ = 0
C
      DO 10 I=1,N
      ROW(I) = 0
      COL(I) = 0.0
   10 CONTINUE
      N2 = N + 1
      N3 = N + N1
C
C     ITERATE LOOP.
C
      DO 200 NN = 1,N
C
C     FIND MAXIMUM PIVOT, REGARDLESS OF SIGN.
C
      PIVOT = 0.0
C
C     J ... COLUMN INDEX
C     I ... ROW INDEX
C
C     K+1 ... INDEX IN "B" FOR THE BEGINNING OF THE SPECIFIC
C             COLUMN BEING SEARCHED, HENCE THE INITIAL VALUE
C             OF ZERO (0)
C
      K = 0
C
      DO 50 J=1,N
      IF ( COL(J) .NE. 0.0 ) GO TO 40
      DO 40 I=1,N
      IF ( ROW(I) .NE. 0 ) GO TO 20
      IF ( ABS (B(I + K)) .LE. ABS (PIVOT) ) GO TO 20
      PR = I
```

```
      PC = J
      PIVOT = B (I + K)
C
   20 CONTINUE
C
C     INDEX K BY THE NUMBER OF ROWS/COLUMNS IN "B" ... M
   40 K = K + M
   50 CONTINUE
C
C     SEE IF ERROR.
C
      IF ( PIVOT .NE. 0.0 ) GO TO 70
   60 ISIMEQ = 2
      RETURN
C
C     MARK KEY ROW AND COLUMN.
C
   70 ROW(PR) = PC
      COL(PC) = 1.0
      L = M*(PC-1)
C
C     ITERATE ON MATRIX AND RHS.
C
      DO 190 I=1,N
      IF ( I .EQ. PR  .OR.  B(I+L) .EQ. 0.0 ) GO TO 190
C
C     ITERATE ON MATRIX.
C
      RATIO = - B(I+L) / PIVOT
      K = 0
      DO 100 J=1,N
      B(I+K) = B(I+K) + RATIO*B(K+PR)
C
C     INDEX K BY THE NUMBER OF ROWS/COLUMNS IN "B" ... M
  100 K = K + M
C
C     ITERATE ON RHS.
C
      DO 120 J=N2,N3
      B(I+K) = B(I+K) + RATIO*B(K+PR)
C     INDEX K BY THE NUMBER OF ROWS IN "B's RHS" ... M
  120 K = K + M
C
  190 CONTINUE
C
  200 CONTINUE
C
C     SOLVE AND STORE ANSWERS.
C
      K = M*N
      DO 250 J=N2,N3
      DO 210 I=1,N
  210 COL(ROW(I)) = B(I+K) / B(M*ROW(I) - M + I)
      DO 220 I=1,N
  220 B(I+K) = COL(I)
  250 K = K + M
C
      ISIMEQ = 1
      RETURN
C
```

```
      END
C                                                                       MINV 0(
C     .................................................................MINV 0(
C                                                                       MINV 0(
C        SUBROUTINE MINV                                                MINV 0(
C                                                                       MINV 0(
C        PURPOSE                                                        MINV 0(
C           INVERT A MATRIX                                             MINV 0(
C                                                                       MINV 0(
C        USAGE                                                          MINV 0(
C           CALL MINV(A,N,D,L,M)                                        MINV 01
C                                                                       MINV 01
C        DESCRIPTION OF PARAMETERS                                      MINV 01
C           A -  INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY MINV 01
C                RESULTANT INVERSE.                                     MINV 01
C           N -  ORDER OF MATRIX A                                      MINV 01
C           D -  RESULTANT DETERMINANT                                  MINV 01
C           L -  WORK VECTOR OF LENGTH N                                MINV 01
C           M -  WORK VECTOR OF LENGTH N                                MINV 01
C                                                                       MINV 01
C        REMARKS                                                        MINV 02
C           MATRIX A MUST BE A GENERAL MATRIX                           MINV 02
C                                                                       MINV 02
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MINV 02
C           NONE                                                        MINV 02
C                                                                       MINV 02
C        METHOD                                                         MINV 02
C           THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT   MINV 02
C           IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT    MINV 02
C           THE MATRIX IS SINGULAR.                                     MINV 02
C                                                                       MINV 03
C     .................................................................MINV 03
C                                                                       MINV 03
      SUBROUTINE MINV (A,N,D,L,M)                                       MINV 03
      DIMENSION A(1),L(1),M(1)                                          MINV 03
C                                                                       MINV 03
C        .............................................................MINV 03
C                                                                       MINV 03
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  MINV 03
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      MINV 03
C        STATEMENT WHICH FOLLOWS.                                       MINV 04
C                                                                       MINV 04
      DOUBLE PRECISION A,D,BIGA,HOLD                                    MINV 04
C                                                                       MINV 04
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    MINV 04
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      MINV 04
C        ROUTINE.                                                       MINV 04
C                                                                       MINV 04
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      MINV 04
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENT  MINV 04
C        10 MUST BE CHANGED TO DABS.                                    MINV 05
C                                                                       MINV 05
C        .............................................................MINV 05
C                                                                       MINV 05
C        SEARCH FOR LARGEST ELEMENT                                     MINV 05
C                                                                       MINV 05
      D=1.0                                                             MINV 05
      NK=-N                                                             MINV 05
      DO 80 K=1,N                                                       MINV 05
      NK=NK+N                                                           MINV 05
```

```
      L(K)=K                                                            MINV 0
      M(K)=K                                                            MINV 0
      KK=NK+K                                                           MINV 0
      BIGA=A(KK)                                                        MINV 0
      DO 20 J=K,N                                                       MINV 0
      IZ=N*(J-1)                                                        MINV 0
      DO 20 I=K,N                                                       MINV 0
      IJ=IZ+I                                                           MINV 0
   10 IF( DABS(BIGA)- DABS(A(IJ))) 15,20,20                             MINV 0
   15 BIGA=A(IJ)                                                        MINV 0
      L(K)=I                                                            MINV 0
      M(K)=J                                                            MINV 0
   20 CONTINUE                                                          MINV 0
C                                                                       MINV 0
C        INTERCHANGE ROWS                                               MINV 0
C                                                                       MINV 0
      J=L(K)                                                            MINV 0
      IF(J-K) 35,35,25                                                  MINV 0
   25 KI=K-N                                                            MINV 0
      DO 30 I=1,N                                                       MINV 0
      KI=KI+N                                                           MINV 0
      HOLD=-A(KI)                                                       MINV 0
      JI=KI-K+J                                                         MINV 0
      A(KI)=A(JI)                                                       MINV 0
   30 A(JI) =HOLD                                                       MINV 0
C                                                                       MINV 0
C        INTERCHANGE COLUMNS                                            MINV 0
C                                                                       MINV 0
   35 I=M(K)                                                            MINV 0
      IF(I-K) 45,45,38                                                  MINV 0
   38 JP=N*(I-1)                                                        MINV 0
      DO 40 J=1,N                                                       MINV 0
      JK=NK+J                                                           MINV 0
      JI=JP+J                                                           MINV 0
      HOLD=-A(JK)                                                       MINV 0
      A(JK)=A(JI)                                                       MINV 0
   40 A(JI) =HOLD                                                       MINV 0
C                                                                       MINV 0
C        DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS        MINV 0
C        CONTAINED IN BIGA)                                             MINV 0
C                                                                       MINV 1
   45 IF(BIGA) 48,46,48                                                 MINV 1
   46 D=0.0                                                             MINV 1
      RETURN                                                            MINV 1
   48 DO 55 I=1,N                                                       MINV 1
      IF(I-K) 50,55,50                                                  MINV 1
   50 IK=NK+I                                                           MINV 1
      A(IK)=A(IK)/(-BIGA)                                               MINV 1
   55 CONTINUE                                                          MINV 1
C                                                                       MINV 1
C        REDUCE MATRIX                                                  MINV 1
C                                                                       MINV 1
      DO 65 I=1,N                                                       MINV 1
      IK=NK+I                                                           MINV 1
      IJ=I-N                                                            MINV 1
      DO 65 J=1,N                                                       MINV 1
      IJ=IJ+N                                                           MINV 1
      IF(I-K) 60,65,60                                                  MINV 1
   60 IF(J-K) 62,65,62                                                  MINV 1
   62 KJ=IJ-I+K                                                         MINV 1
```

```
      A(IJ)=A(IK)*A(KJ)+A(IJ)                                          MINV 12
   65 CONTINUE                                                         MINV 12
C                                                                      MINV 12
C        DIVIDE ROW BY PIVOT                                           MINV 12
C                                                                      MINV 12
      KJ=K-N                                                           MINV 12
      DO 75 J=1,N                                                      MINV 12
      KJ=KJ+N                                                          MINV 12
      IF(J-K) 70,75,70                                                 MINV 12
   70 A(KJ)=A(KJ)/BIGA                                                 MINV 12
   75 CONTINUE                                                         MINV 13
C                                                                      MINV 13
C        PRODUCT OF PIVOTS                                             MINV 13
C                                                                      MINV 13
      D=D*BIGA                                                         MINV 13
C                                                                      MINV 13
C        REPLACE PIVOT BY RECIPROCAL                                   MINV 13
C                                                                      MINV 13
      A(KK)=1.0/BIGA                                                   MINV 13
   80 CONTINUE                                                         MINV 13
C                                                                      MINV 14
C        FINAL ROW AND COLUMN INTERCHANGE                              MINV 14
C                                                                      MINV 14
      K=N                                                              MINV 14
  100 K=(K-1)                                                          MINV 14
      IF(K) 150,150,105                                                MINV 14
  105 I=L(K)                                                           MINV 14
      IF(I-K) 120,120,108                                              MINV 14
  108 JQ=N*(K-1)                                                       MINV 14
      JR=N*(I-1)                                                       MINV 14
      DO 110 J=1,N                                                     MINV 15
      JK=JQ+J                                                          MINV 15
      HOLD=A(JK)                                                       MINV 15
      JI=JR+J                                                          MINV 15
      A(JK)=-A(JI)                                                     MINV 15
  110 A(JI) =HOLD                                                      MINV 15
  120 J=M(K)                                                           MINV 15
      IF(J-K) 100,100,125                                              MINV 15
  125 KI=K-N                                                           MINV 15
      DO 130 I=1,N                                                     MINV 15
      KI=KI+N                                                          MINV 16
      HOLD=A(KI)                                                       MINV 16
      JI=KI-K+J                                                        MINV 16
      A(KI)=-A(JI)                                                     MINV 16
  130 A(JI) =HOLD                                                      MINV 16
      GO TO 100                                                        MINV 16
  150 RETURN                                                           MINV 16
      END                                                              MINV 16
```

Appendix: B

Input and Output for the first column

```
COMPONENTS        6
COMPONENT   1WATER          212.0       18.0          8.35
COMPONENT   2TOLUENE        231.2       92.14         7.24
COMPONENT   3o-NT           431.6       137.14        9.72
COMPONENT   4m-NT           451.4       137.14        9.68
COMPONENT   5p-NT           460.4       137.14        9.22
COMPONENT   6DNT            572.0       182.14        11.03
TEMP        104.0     176.0     248.0     320.0     392.0     464.0     536.0     608.0
VP       1  55.31     355.20    1489.     4635.4    11662.    25110.    48150.    84668.
VP       2  56.53     280.43    957.5     2528.6    5568.8    10750.    18817.    31305.
VP       3  0.3505    4.3187    29.43     130.90    431.9     1141.32   2556.2    5052.
VP       4  0.2054    2.78      20.22     95.17     327.1     892.13    2048.     4127.
VP       5  0.1926    2.549     18.306    85.466    292.29    794.98    1822.     3668.
VP       6  0.0017    0.050     0.672     5.123     25.88     96.077    282.04    690.8
HV       1  1.11      1.14      1.163     1.185     1.200     1.205     1.195     1 102
HV       2  24.42     26.26     28.47     31.14     34.36     36.67     39.62     42.94
HV       3  21.09     23.26     25.42     27.59     29.75     31.91     34.09     36.24
HV       4  22.27     24.44     26.60     28.77     30.93     33.09     35.26     37.42
HV       5  22.39     24.56     26.72     28.88     31.05     33.21     35.37     37.54
HV       6  33.25     35.41     37.56     39.72     41.88     44.03     46.18     48.34
HL       1  0.074     0.144     0.217     0.290     0.366     0.446     0.532     0.629
HL       2  8.48      11.51     14.56     17.78     21.92     25.34     29.48     35.66
HL       3  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       4  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       5  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       6  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
VM       1  0.1207    0.1233    0.127     0.1321    0.1386    0.1473    0-159     0.179
VM       2  13.02     13.64     14.36     15.23     16.34     17.90     20.49     43.13
VM       3  20.82     20.72     21.32     22.04     22.94     24.09     25.61     27.71
VM       4  22.63     23.17     23.81     24.60     25.57     26.81     28.44     30.69
VM       5  23.31     23.87     24.51     25.30     26.27     27.49     29.07     31.23
VM       6  35.22     35.82     36.81     37.37     38.37     39.58     41.07     42.95
CHARGE      1 17.314
COMP        1 0.002
COMP        2 1.316
COMP        3 9.486
COMP        4 0.643
COMP        5 5.865
COMP        6 0.002
STAGES          7
HOLDDUTYEM    0.6         0.2         0.0         0.7
PRESDPRRDR    760.0       0.0         0.0         4.0         4.76
CUTTEMPS    240.436.456   0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
CONVERGENCE  0.001        50.0           15      0      1

( LAPSED CPU TIME =      0.00 SEC        ENTRY COUNT =     0 )
```

C O M P O N E N T P R O P E R T I E S

NO	NAME	NBP	MOLE WT	DENSITY
1	WATER	212.0	18.00	8.350
2	TOLUENE	231.2	92.14	7.240
3	o-NT	431.6	137.1	9.720
4	m-NT	451.4	137.1	9.680
5	p-NT	460.4	137.1	9.220
6	DNT	572.0	182.1	11.03

VAPOR PRESSURE DATA

1 WATER

TEMP	EXP	CALC	%DEV
104.0	55.31	60.29	8.998
176.0	355.2	300.2	-15.49
248.0	1489.	1390.	-6.648
320.0	4635.	4960.	6.998
392.0	0.1166E+05	0.1338E+05	14.71
464.0	0.2511E+05	0.2818E+05	12.21
536.0	0.4815E+05	0.4841E+05	0.5428
608.0	0.8467E+05	0.7070E+05	-16.50

2 TOLUENE

TEMP	EXP	CALC	%DEV
104.0	56.53	61.26	8.362
176.0	280.4	240.0	-14.43
248.0	957.5	897.0	-6.324
320.0	2529.	2688.	6.291
392.0	5569.	6323.	13.55
464.0	0.1075E+05	0.1200E+05	11.61
536.0	0.1882E+05	0.1906E+05	1.290
608.0	0.3131E+05	0.2627E+05	-16.08

3 o-NT

TEMP	EXP	CALC	%DEV
104.0	0.3505	0.3763	7.372
176.0	4.319	3.761	-12.91
248.0	29.43	27.75	-5.723
320.0	130.9	138.6	5.874
392.0	431.9	484.0	12.07
464.0	1141.	1255.	9.999
536.0	2556.	2567.	0.4140
608.0	5052.	4352.	-13.86

4 m-NT

TEMP	EXP	CALC	%DEV
104.0	0.2054	0.2204	7.323
176.0	2.780	2.421	-12.93
248.0	20.22	19.13	-5.410
320.0	95.17	100.6	5.728
392.0	327.1	365.9	11.86
464.0	892.1	980.0	9.847
536.0	2048.	2056.	0.4126
608.0	4127.	3562.	-13.68

5 p-NT

TEMP	EXP	CALC	%DEV
104.0	0.1926	0.2066	7.274
176.0	2.549	2.222	-12.84
248.0	18.31	17.32	-5.400
320.0	85.47	90.33	5.691
392.0	292.3	326.8	11.80
464.0	795.0	872.8	9.788
536.0	1822.	1829.	0.4056
608.0	3668.	3169.	-13.61

6 DNT

TEMP	EXP	CALC	%DEV
104.0	0.1700E-02	0.1816E-02	6.806
176.0	0.5000E-01	0.4394E-01	-12.12
248.0	0.6720	0.6379	-5.076
320.0	5.123	5.405	5.509
392.0	25.88	28.74	11.03
464.0	96.08	104.7	8.931

536.0	282.0	282.6	0.2046
608.0	690.8	603.7	-12.61

VAPOR ENTHALPY DATA

1 WATER

TEMP	EXP	CALC	%DEV
104.0	1.110	1.098	-1.038
176.0	1.140	1.144	0.3437
248.0	1.163	1.174	0.9771
320.0	1.185	1.192	0.5740
392.0	1.200	1.198	-0.1483
464.0	1.205	1.196	-0.7794
536.0	1.195	1.186	-0.7573
608.0	1.162	1.171	0.7947

2 TOLUENE

TEMP	EXP	CALC	%DEV
104.0	24.42	24.09	-1.343
176.0	26.26	26.45	0.7351
248.0	28.47	28.83	1.279
320.0	31.14	31.29	0.4845
392.0	34.36	33.88	-1.399
464.0	36.67	36.66	-0.4021E-01
536.0	39.62	39.67	0.1370
608.0	42.94	42.99	0.1216

3 o-NT

TEMP	EXP	CALC	%DEV
104.0	21.09	21.05	-0.1805
176.0	23.26	23.28	0.7393E-01
248.0	25.42	25.46	0.1403
320.0	27.59	27.61	0.5493E-01
392.0	29.75	29.74	-0.2201E-01
464.0	31.91	31.89	-0.6780E-01
536.0	34.09	34.06	-0.9479E-01
608.0	36.24	36.27	0.8064E-01

4 m-NT

TEMP	EXP	CALC	%DEV
104.0	22.27	22.22	-0.2181
176.0	24.44	24.46	0.9522E-01
248.0	26.60	26.64	0.1690
320.0	28.77	28.79	0.6749E-01
392.0	30.93	30.92	-0.3380E-01
464.0	33.09	33.06	-0.9595E-01
536.0	35.26	35.23	-0.8949E-01
608.0	37.42	37.45	0.8829E-01

5 p-NT

TEMP	EXP	CALC	%DEV
104.0	22.39	22.34	-0.2191
176.0	24.56	24.58	0.9361E-01
248.0	26.72	26.76	0.1634
320.0	28.88	28.91	0.9195E-01
392.0	31.05	31.04	-0.4762E-01
464.0	33.21	33.17	-0.1107
536.0	35.37	35.34	-0.7307E-01
608.0	37.54	37.57	0.8414E-01

6 DNT

TEMP	EXP	CALC	%DEV
104.0	33.25	33.09	-0.4664
176.0	35.41	35.49	0.2198
248.0	37.56	37.70	0.3626
320.0	39.72	39.79	0.1776
392.0	41.88	41.84	-0.9198E-01
464.0	44.03	43.98	-0.2509

536.0	46.18	46.10	-0.1836
608.0	48.34	48.44	0.2051

LIQUID ENTHALPY DATA

1 WATER

TEMP	EXP	CALC	%DEV
104.0	0.7400E-01	0.8005E-01	8.176
176.0	0.1440	0.1419	-1.489
248.0	0.2170	0.2111	-2.710
320.0	0.2900	0.2866	-1.189
392.0	0.3660	0.3669	0.2343
464.0	0.4460	0.4507	1.064
536.0	0.5320	0.5369	0.9265
608.0	0.6290	0.6241	-0.7773

2 TOLUENE

TEMP	EXP	CALC	%DEV
104.0	8.480	8.787	3.617
176.0	11.51	11.39	-1.030
248.0	14.56	14.37	-1.324
320.0	17.78	17.72	-0.3125
392.0	21.92	21.47	-2.035
464.0	25.34	25.63	1.129
536.0	29.48	30.19	2.415
608.0	35.66	35.18	-1.342

3 o-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

4 m-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

5 p-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

6 DNT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332

536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

MOLAR VOLUME DATA

1 WATER

TEMP	EXP	CALC	%DEV
104.0	0.1207	0.1206	-0.8606E-01
176.0	0.1233	0.1238	0.3779
248.0	0.1270	0.1271	0.9470E-01
320.0	0.1321	0.1316	-0.4078
392.0	0.1386	0.1380	-0.4392
464.0	0.1473	0.1473	0.8270E-02
536.0	0.1590	0.1604	0.8971
608.0	0.1790	0.1782	-0.4268

2 TOLUENE

TEMP	EXP	CALC	%DEV
104.0	13.02	15.58	19.63
176.0	13.64	13.52	-0.8612
248.0	14.36	12.34	-14.04
320.0	15.23	12.58	-17.38
392.0	16.34	14.79	-9.495
464.0	17.90	19.50	8.958
536.0	20.49	27.27	33.11
608.0	43.13	38.64	-10.40

3 o-NT

TEMP	EXP	CALC	%DEV
104.0	20.22	20.14	-0.4041
176.0	20.72	20.79	0.3194
248.0	21.32	21.38	0.2976
320.0	22.04	22.05	0.3522E-01
392.0	22.94	22.90	-0.1886
464.0	24.09	24.05	-0.1742
536.0	25.61	25.62	0.3630E-01
608.0	27.71	27.73	0.6546E-01

4 m-NT

TEMP	EXP	CALC	%DEV
104.0	22.63	22.53	-0.4435
176.0	23.17	23.24	0.3228
248.0	23.81	23.90	0.3576
320.0	24.60	24.61	0.4262E-01
392.0	25.57	25.52	-0.1933
464.0	26.81	26.76	-0.2043
536.0	28.44	28.44	0.1473E-01
608.0	30.69	30.72	0.8881E-01

5 p-NT

TEMP	EXP	CALC	%DEV
104.0	23.31	23.20	-0.4774
176.0	23.87	23.94	0.3075
248.0	24.51	24.61	0.4030
320.0	25.30	25.32	0.9531E-01
392.0	26.27	26.22	-0.1956
464.0	27.49	27.42	-0.2490
536.0	29.07	29.06	-0.2716E-01
608.0	31.23	31.27	0.1266

6 DNT

TEMP	EXP	CALC	%DEV
104.0	35.22	35.01	-0.6052
176.0	35.82	36.03	0.5753
248.0	36.81	36.81	0.2920E-02
320.0	37.37	37.53	0.4257
392.0	38.37	38.35	-0.5901E-01
464.0	39.58	39.43	-0.3710

536.0	41.07	40.95	-0.2830
608.0	42.95	43.08	0.2944

VAPOR PRESSURE COEFFICIENTS

1	WATER	-0.57833E-02	35553.	-0.32480E+08	0.77471E+10
2	TOLUENE	-0.54543E-02	31947.	-0.28993E+08	0.69304E+10
3	o-NT	-0.47888E-02	31082.	-0.30116E+08	0.69259E+10
4	m-NT	-0.47305E-02	30963.	-0.30215E+08	0.69236E+10
5	p-NT	-0.47027E-02	30697.	-0.30040E+08	0.68983E+10
6	DNT	-0.43852E-02	30460.	-0.31977E+08	0.72171E+10

VAPOR ENTHALPY COEFFICIENTS

1	WATER	-0.42378E-03	0.34366E-02	-0.31384E-05	0.88736E-09
2	TOLUENE	-0.87834E-02	0.60517E-01	-0.45541E-04	0.24898E-07
3	o-NT	-0.13173E-02	0.45912E-01	-0.19670E-04	0.79481E-08
4	m-NT	-0.16365E-02	0.50441E-01	-0.25308E-04	0.10229E-07
5	p-NT	-0.16632E-02	0.50945E-01	-0.25994E-04	0.10527E-07
6	DNT	-0.51662E-02	0.92612E-01	-0.77881E-04	0.31505E-07

LIQUID ENTHALPY COEFFICIENTS

1	WATER	0.22154E-03	-0.70065E-03	0.18194E-05	-0.57682E-09
2	TOLUENE	0.13152E-01	-0.10146E-02	0.26743E-04	0.47385E-08
3	o-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
4	m-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
5	p-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
6	DNT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07

MOLAR VOLUME COEFFICIENTS

1	WATER	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
2	TOLUENE	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
3	o-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
4	m-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
5	p-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
6	DNT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00

F E E D C H A R G E

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.2000E-02	0.1155E-01	0.3600E-01	0.1555E-02	0.4311E-02	0.1746E-02
2	TOLUENE	1.316	7.601	121.3	5.238	16.75	6.781
3	o-NT	9.486	54.79	1301.	56.19	133.8	54.19
4	m-NT	0.6430	3.714	88.18	3.809	9.110	3.689
5	p-NT	5.865	33.87	804.3	34.74	87.24	35.32
6	DNT	0.2000E-02	0.1155E-01	0.3643	0.1574E-01	0.3303E-01	0.1337E-01
	TOTAL	17.31	100.0	2315.	100.0	247.0	100.0

(LAPSED CPU TIME = 2.20 SEC ENTRY COUNT = 1)

ITERATION NUMBER 0

S T E A D Y - S T A T E P R O F I L E

STAGE	DEL TEMP	TEMPERATURE	VAPOR RATE	LIQUID RATE	DEL HOLDUP	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE	SUMDX
1	0.0000E+00	238.2	0.0000E+00	0.0000E+00	0.0000E+00	0.6000	0.6000	0.0000E+00	760.0	0.0000E+00
2	0.0000E+00	259.5	0.0000E+00	0.0000E+00	0.0000E+00	0.2000	0.2000	0.0000E+00	760.0	0.0000E+00
3	0.0000E+00	259.5	0.0000E+00	0.0000E+00	0.0000E+00	0.2000	0.2000	0.0000E+00	760.0	0.0000E+00
4	0.0000E+00	259.5	0.0000E+00	0.0000E+00	0.0000E+00	0.2000	0.2000	0.0000E+00	760.0	0.0000E+00
5	0.0000E+00	291.3	0.0000E+00	0.0000E+00	0.0000E+00	0.2000	0.2000	0.0000E+00	760.0	0.0000E+00
6	0.0000E+00	451.6	0.0000E+00	0.0000E+00	0.0000E+00	0.2000	0.2000	0.0000E+00	760.0	0.0000E+00
7	0.0000E+00	433.8	0.0000E+00	0.0000E+00	0.0000E+00	15.71	0.0000E+00	0.0000E+00	760.0	0.0000E+00

SCALE	CONV	ICONV
0.0000E+00	0.0000E+00	0

TOTAL TIME, HOURS	0.2971E-04
CUMULATIVE TIME, HOURS	0.2971E-04
CUT TIME, HOURS	0.2971E-04
TIME STEP, HOURS	0.2971E-04
OVERHEAD TEMPERATURE	239.6
AT 760 MM HG	239.8
K = 12	239.8

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1026E-01	0.2156	0.1847	0.4203E-01	0.2212E-01	0.3655E-01
2	TOLUENE	4.715	99.05	434.4	98.87	60.00	99.14
3	o-NT	0.2615E-01	0.5493	3.586	0.8161	0.3689	0.6095
4	m-NT	0.1041E-02	0.2187E-01	0.1428	0.3249E-01	0.1475E-01	0.2437E-01
5	p-NT	0.7807E-02	0.1640	1.071	0.2437	0.1161	0.1919
6	DNT	0.1219E-06	0.2560E-05	0.2220E-04	0.5052E-05	0.2012E-05	0.3325E-05
	TOTAL	4.760	100.0	439.4	100.0	60.52	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.3049E-06	0.2156	0.5488E-05	0.4203E-01	0.6572E-06	0.3655E-01
2	TOLUENE	0.1401E-03	99.05	0.1291E-01	98.87	0.1783E-02	99.14
3	o-NT	0.7757E-06	0.5484	0.1064E-03	0.8148	0.1094E-04	0.6086
4	m-NT	0.3084E-07	0.2181E-01	0.4230E-05	0.3240E-01	0.4370E-06	0.2430E-01
5	p-NT	0.2312E-06	0.1635	0.3171E-04	0.2429	0.3439E-05	0.1912
6	DNT	0.3444E-11	0.2435E-05	0.6272E-09	0.4804E-05	0.5687E-10	0.3162E-05
	TOTAL	0.1414E-03	100.0	0.1306E-01	100.0	0.1798E-02	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.3049E-06	0.2156	0.5488E-05	0.4203E-01	0.6572E-06	0.3655E-01
2	TOLUENE	0.1401E-03	99.05	0.1291E-01	98.87	0.1783E-02	99.14
3	o-NT	0.7757E-06	0.5484	0.1064E-03	0.8148	0.1094E-04	0.6086
4	m-NT	0.3084E-07	0.2181E-01	0.4230E-05	0.3240E-01	0.4370E-06	0.2430E-01
5	p-NT	0.2312E-06	0.1635	0.3171E-04	0.2429	0.3439E-05	0.1912
6	DNT	0.3444E-11	0.2435E-05	0.6272E-09	0.4804E-05	0.5687E-10	0.3162E-05
	TOTAL	0.1414E-03	100.0	0.1306E-01	100.0	0.1798E-02	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.3049E-06	0.2156	0.5488E-05	0.4203E-01	0.6572E-06	0.3655E-01
2	TOLUENE	0.1401E-03	99.05	0.1291E-01	98.87	0.1783E-02	99.14
3	o-NT	0.7757E-06	0.5484	0.1064E-03	0.8148	0.1094E-04	0.6086
4	m-NT	0.3084E-07	0.2181E-01	0.4230E-05	0.3240E-01	0.4370E-06	0.2430E-01
5	p-NT	0.2312E-06	0.1635	0.3171E-04	0.2429	0.3439E-05	0.1912
6	DNT	0.3444E-11	0.2435E-05	0.6272E-09	0.4804E-05	0.5687E-10	0.3162E-05
	TOTAL	0.1414E-03	100.0	0.1306E-01	100.0	0.1798E-02	100.0

ITERATION NUMBER 5

S T E A D Y - S T A T E P R O F I L E

STAGE	DEL TEMP	TEMPERATURE	VAPOR RATE	LIQUID RATE	DEL HOLDUP	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE	SUMDX
1	0.6865E-08	238.7	0.0000E+00	23.80	-0.2991E-16	0.6000	0.6000	-347.7	760.0	0.2722E-09
2	0.2500E-07	239.6	23.80	23.54	-0.3698E-17	0.2000	0.2000	0.0000E+00	760.0	0.7579E-09
3	0.1063E-06	242.5	23.54	22.74	0.8161E-17	0.2000	0.2000	0.0000E+00	760.0	0.1998E-08
4	0.4892E-06	252.5	22.74	20.28	-0.4441E-17	0.2000	0.2000	0.0000E+00	760.0	0.9305E-09
5	-0.9336E-06	292.5	20.28	16.56	0.4669E-18	0.2000	0.2000	0.0000E+00	760.0	0.1393E-08
6	0.8268E-07	383.7	16.56	16.19	0.4368E-17	0.2000	0.2000	0.0000E+00	760.0	0.3489E-09
7	-0.2061E-07	428.7	16.19	0.0000E+00	0.2505E-16	15.71	15.71	347.7	760.0	0.6867E-10

SCALE	CONV	ICONV
1.000	1.000	0

(LAPSED CPU TIME = 2.89 SEC ENTRY COUNT = 2)

```
NUMBER OF STEPS                 1
NUMBER OF FUNCTION CALLS        5
NUMBER OF JACOBIAN CALLS        2
```

TIME, HOURS 0.2971E-04

DISTILLATE RATE 4.760

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	238.7	0.0000E+00	19.04	0.6000	0.6000	-347.7	760.0
2	239.6	23.80	18.78	0.2000	0.2000	0.0000E+00	760.0
3	242.5	23.54	17.98	0.2000	0.2000	0.0000E+00	760.0
4	252.5	22.74	15.53	0.2000	0.2000	0.0000E+00	760.0
5	292.5	20.29	11.91	0.2000	0.2000	0.0000E+00	760.0
6	383.8	16.67	12.06	0.2000	0.2000	0.0000E+00	760.0
7	428.7	16.82	0.0000E+00	15.71	15.71	362.3	760.0

TOTAL TIME, HOURS	0.1388E-01
CUMULATIVE TIME, HOURS	0.1388E-01
CUT TIME, HOURS	0.1388E-01
TIME STEP, HOURS	0.1138E-02
OVERHEAD TEMPERATURE	240.0
AT 760 MM HG	240.2
K = 12	240.2

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1004E-01	0.2108	0.1806	0.4109E-01	0.2163E-01	0.3574E-01
2	TOLUENE	4.712	98.98	434.1	98.76	59.96	99.06
3	o-NT	0.2878E-01	0.6047	3.947	0.8979	0.4061	0.6709
4	m-NT	0.1140E-02	0.2396E-01	0.1564	0.3557E-01	0.1615E-01	0.2669E-01
5	p-NT	0.8522E-02	0.1790	1.169	0.2659	0.1268	0.2094
6	DNT	0.1244E-06	0.2613E-05	0.2265E-04	0.5153E-05	0.2054E-05	0.3393E-05
	TOTAL	4.760	100.0	439.6	100.0	60.53	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1412E-03	0.2138	0.2542E-02	0.4169E-01	0.3044E-03	0.3625E-01
2	TOLUENE	0.6540E-01	99.02	6.026	98.83	0.8324	99.11
3	o-NT	0.3760E-03	0.5692	0.5156E-01	0.8455	0.5304E-02	0.6316
4	m-NT	0.1495E-04	0.2264E-01	0.2050E-02	0.3362E-01	0.2118E-03	0.2522E-01
5	p-NT	0.1120E-03	0.1696	0.1536E-01	0.2519	0.1666E-02	0.1984
6	DNT	0.1710E-08	0.2590E-05	0.3115E-06	0.5109E-05	0.2824E-07	0.3363E-05
	TOTAL	0.6605E-01	100.0	6.098	100.0	0.8398	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1412E-03	0.2138	0.2542E-02	0.4169E-01	0.3044E-03	0.3625E-01
2	TOLUENE	0.6540E-01	99.02	6.026	98.83	0.8324	99.11
3	o-NT	0.3760E-03	0.5692	0.5156E-01	0.8455	0.5304E-02	0.6316
4	m-NT	0.1495E-04	0.2264E-01	0.2050E-02	0.3362E-01	0.2118E-03	0.2522E-01
5	p-NT	0.1120E-03	0.1696	0.1536E-01	0.2519	0.1666E-02	0.1984
6	DNT	0.1710E-08	0.2590E-05	0.3115E-06	0.5109E-05	0.2824E-07	0.3363E-05
	TOTAL	0.6605E-01	100.0	6.098	100.0	0.8398	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1412E-03	0.2138	0.2542E-02	0.4169E-01	0.3044E-03	0.3625E-01
2	TOLUENE	0.6540E-01	99.02	6.026	98.83	0.8324	99.11
3	o-NT	0.3760E-03	0.5692	0.5156E-01	0.8455	0.5304E-02	0.6316
4	m-NT	0.1495E-04	0.2264E-01	0.2050E-02	0.3362E-01	0.2118E-03	0.2522E-01
5	p-NT	0.1120E-03	0.1696	0.1536E-01	0.2519	0.1666E-02	0.1984
6	DNT	0.1710E-08	0.2590E-05	0.3115E-06	0.5109E-05	0.2824E-07	0.3363E-05
	TOTAL	0.6605E-01	100.0	6.098	100.0	0.8398	100.0

```
NUMBER OF STEPS                35
NUMBER OF FUNCTION CALLS       44
NUMBER OF JACOBIAN CALLS        8
```

TIME, HOURS 0.1388E-01

DISTILLATE RATE 4.760

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	238.7	0.0000E+00	19.04	0.6000	0.6000	-348.3	760.0
2	240.0	23.80	18.68	0.2000	0.2000	0.0000E+00	760.0
3	244.0	23.44	17.59	0.2000	0.2000	0.0000E+00	760.0
4	258.0	22.35	14.25	0.2000	0.2000	0.0000E+00	760.0
5	319.8	19.01	11.51	0.2000	0.2000	0.0000E+00	760.0
6	407.3	16.27	11.89	0.2000	0.2000	0.0000E+00	760.0
7	429.1	16.65	0.0000E+00	15.65	15.65	359.8	760.0

TOTAL TIME, HOURS	2.492
CUMULATIVE TIME, HOURS	2.492
CUT TIME, HOURS	2.478
TIME STEP, HOURS	0.3738E-01
OVERHEAD TEMPERATURE	436.3
AT 760 MM HG	436.5
K = 12	436.5

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.7282E-10	0.1530E-08	0.1311E-08	0.2008E-09	0.1570E-09	0.2289E-09
2	TOLUENE	0.8391E-07	0.1763E-05	0.7732E-05	0.1184E-05	0.1068E-05	0.1557E-05
3	o-NT	2.704	56.81	370.8	56.81	38.15	55.64
4	m-NT	0.2248	4.723	30.83	4.723	3.185	4.644
5	p-NT	1.831	38.47	251.1	38.47	27.24	39.72
6	DNT	0.1768E-05	0.3714E-04	0.3220E-03	0.4932E-04	0.2919E-04	0.4257E-04
	TOTAL	4.760	100.0	652.8	100.0	68.57	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.1859E-02	0.1576E-01	0.3346E-01	0.2143E-02	0.4007E-02	0.2406E-02
2	TOLUENE	1.251	10.60	115.2	7.381	15.92	9.556
3	o-NT	7.769	65.86	1065.	68.25	109.6	65.81
4	m-NT	0.3478	2.949	47.70	3.055	4.927	2.958
5	p-NT	2.426	20.57	332.8	21.32	36.09	21.67
6	DNT	0.1503E-05	0.1275E-04	0.2738E-03	0.1754E-04	0.2483E-04	0.1491E-04
	TOTAL	11.80	100.0	1561.	100.0	166.5	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.2000E-02	0.1686E-01	0.3600E-01	0.2297E-02	0.4311E-02	0.2576E-02
2	TOLUENE	1.316	11.09	121.3	7.737	16.75	10.01
3	o-NT	7.769	65.50	1065.	67.98	109.6	65.49
4	m-NT	0.3478	2.932	47.70	3.043	4.927	2.944
5	p-NT	2.426	20.46	332.8	21.23	36.09	21.56
6	DNT	0.1505E-05	0.1269E-04	0.2742E-03	0.1749E-04	0.2486E-04	0.1485E-04
	TOTAL	11.86	100.0	1567.	100.0	167.4	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.2000E-02	0.1686E-01	0.3600E-01	0.2297E-02	0.4311E-02	0.2576E-02
2	TOLUENE	1.316	11.09	121.3	7.737	16.75	10.01
3	o-NT	7.769	65.50	1065.	67.98	109.6	65.49
4	m-NT	0.3478	2.932	47.70	3.043	4.927	2.944
5	p-NT	2.426	20.46	332.8	21.23	36.09	21.56
6	DNT	0.1505E-05	0.1269E-04	0.2742E-03	0.1749E-04	0.2486E-04	0.1485E-04
	TOTAL	11.86	100.0	1567.	100.0	167.4	100.0

```
NUMBER OF STEPS              355
NUMBER OF FUNCTION CALLS     563
NUMBER OF JACOBIAN CALLS      52
```

TIME, HOURS 2.492

DISTILLATE RATE 4.760

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	434.3	0.0000E+00	19.04	0.6000	0.6000	-494.8	760.0
2	436.3	23.80	18.95	0.2000	0.2000	0.0000E+00	760.0
3	438.1	23.71	18.87	0.2000	0.2000	0.0000E+00	760.0
4	439.6	23.63	18.81	0.2000	0.2000	0.0000E+00	760.0
5	441.0	23.57	18.75	0.2000	0.2000	0.0000E+00	760.0
6	442.2	23.51	18.70	0.2000	0.2000	0.0000E+00	760.0
7	443.4	23.46	0.0000E+00	3.853	3.853	494.9	760.0

TOTAL TIME, HOURS	3.124
CUMULATIVE TIME, HOURS	3.124
CUT TIME, HOURS	0.6323
TIME STEP, HOURS	0.8672E-02
OVERHEAD TEMPERATURE	444.4
AT 760 MM HG	444.7
K = 12	444.7

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.5119E-12	0.1075E-10	0.9215E-11	0.1412E-11	0.1104E-11	0.1587E-11
2	TOLUENE	0.6452E-09	0.1356E-07	0.5945E-07	0.9107E-08	0.8212E-08	0.1181E-07
3	o-NT	1.364	28.66	187.1	28.66	19.25	27.67
4	m-NT	0.2842	5.970	38.97	5.970	4.026	5.788
5	p-NT	3.112	65.37	426.7	65.37	46.28	66.54
6	DNT	0.8528E-05	0.1792E-03	0.1553E-02	0.2379E-03	0.1408E-03	0.2025E-03
	TOTAL	4.760	100.0	652.8	100.0	69.56	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.9240E-11	0.3070E-09	0.1663E-09	0.4029E-10	0.1992E-10	0.4565E-10
2	TOLUENE	0.1084E-07	0.3601E-06	0.9985E-06	0.2419E-06	0.1379E-06	0.3161E-06
3	o-NT	1.339	44.47	183.6	44.47	18.89	43.29
4	m-NT	0.1625	5.399	22.29	5.399	2.302	5.277
5	p-NT	1.509	50.13	206.9	50.13	22.44	51.44
6	DNT	0.1692E-05	0.5621E-04	0.3081E-03	0.7465E-04	0.2794E-04	0.6403E-04
	TOTAL	3.010	100.0	412.7	100.0	43.63	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.2000E-02	0.1345E-01	0.3600E-01	0.1818E-02	0.4311E-02	0.2043E-02
2	TOLUENE	1.316	8.849	121.3	6.124	16.75	7.937
3	o-NT	9.108	61.24	1249.	63.08	128.5	60.90
4	m-NT	0.5103	3.432	69.98	3.535	7.230	3.426
5	p-NT	3.935	26.46	539.7	27.26	58.53	27.74
6	DNT	0.3197E-05	0.2150E-04	0.5823E-03	0.2941E-04	0.5279E-04	0.2502E-04
	TOTAL	14.87	100.0	1980.	100.0	211.0	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.2000E-02	0.1345E-01	0.3600E-01	0.1818E-02	0.4311E-02	0.2043E-02
2	TOLUENE	1.316	8.849	121.3	6.124	16.75	7.937
3	o-NT	9.108	61.24	1249.	63.08	128.5	60.90
4	m-NT	0.5103	3.432	69.98	3.535	7.230	3.426
5	p-NT	3.935	26.46	539.7	27.26	58.53	27.74
6	DNT	0.3197E-05	0.2150E-04	0.5823E-03	0.2941E-04	0.5279E-04	0.2502E-04
	TOTAL	14.87	100.0	1980.	100.0	211.0	100.0

```
NUMBER OF STEPS              414
NUMBER OF FUNCTION CALLS     680
NUMBER OF JACOBIAN CALLS      87
```

TIME, HOURS 3.124

DISTILLATE RATE 4.760

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	442.4	0.0000E+00	19.04	0.6000	0.6000	-503.2	760.0
2	444.4	23.80	18.97	0.2000	0.2000	0.0000E+00	760.0
3	446.0	23.73	18.91	0.2000	0.2000	0.0000E+00	760.0
4	447.2	23.67	18.87	0.2000	0.2000	0.0000E+00	760.0
5	448.2	23.63	18.84	0.2000	0.2000	0.0000E+00	760.0
6	448.9	23.60	18.80	0.2000	0.2000	0.0000E+00	760.0
7	449.8	23.56	0.0000E+00	0.8430	0.8430	503.2	760.0

(LAPSED CPU TIME = 22.88 SEC ENTRY COUNT = 3)

F E E D C H A R G E (REMAINING IN STILL)

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	WATER	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
2	TOLUENE	0.8379E-10	0.3430E-08	0.7720E-08	0.2304E-08	0.1066E-08	0.2965E-08
3	o-NT	0.3785	15.49	51.90	15.49	5.340	14.85
4	m-NT	0.1327	5.432	18.20	5.430	1.880	5.228
5	p-NT	1.930	79.00	264.7	78.97	28.71	79.83
6	DNT	0.1997E-02	0.8174E-01	0.3637	0.1085	0.3297E-01	0.9170E-01
	TOTAL	2.443	100.0	335.1	100.0	35.96	100.0

(LAPSED CPU TIME = 26.05 SEC ENTRY COUNT = 4)

BATCH TERMINATING

Appendix: C

Input and Output for the second column

```
COMPONENTS        4
COMPONENT   1o-NT          431.6       137.14        9.72
COMPONENT   2m-NT          451.4       137.14        9.68
COMPONENT   3p-NT          460.4       137.14        9.22
COMPONENT   4DNT           572.0       182.14       11.03
TEMP        104.0     176.0     248.0     320.0     392.0     464.0     536.0     608.0
VP       1  0.3505    4.3187    29.43     130.90    431.9     1141.32   2556.2    5052.
VP       2  0.2054    2.78      20.22     95.17     327.1     892.13    2048.     4127.
VP       3  0.1926    2.549     18.306    85.466    292.29    794.98    1822.     3668.
VP       4  0.0017    0.050     0.672     5.123     25.88     96.077    282.04    690.8
HV       1  21.09     23.26     25.42     27.59     29.75     31.91     34.09     36.24
HV       2  22.27     24.44     26.60     28.77     30.93     33.09     35.26     37.42
HV       3  22.39     24.56     26.72     28.88     31.05     33.21     35.37     37.54
HV       4  33.25     35.41     37.56     39.72     41.88     44.03     46.18     48.34
HL       1  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       2  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       3  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
HL       4  0.808     2.974     5.137     7.300     9.467     11.630    13.79     15.96
VM       1  20.22     20.72     21.32     22.04     22.94     24.09     25.61     27.71
VM       2  22.63     23.17     23.81     24.60     25.57     26.81     28.44     30.69
VM       3  23.31     23.87     24.51     25.30     26.27     27.49     29.07     31.23
VM       4  35.22     35.82     36.81     37.37     38.37     39.58     41.07     42.95
CHARGE      1 15.996
COMP        1 9.486
COMP        2 0.643
COMP        3 5.865
COMP        4 0.002
STAGES           14
HOLDDUTYEM    0.05          0.01          0.0           0.7
PRESDPRRDR    5.0           0.0           0.0           10.0          1.5332
CUTTEMPS      186.          199.2         205.          0.            0.
CONVERGENCE   0.001         50.0             15         0      1

( LAPSED CPU TIME =      0.00 SEC         ENTRY COUNT =     0 )
```

C O M P O N E N T P R O P E R T I E S

NO	NAME	NBP	MOLE WT	DENSITY
1	o-NT	431.6	137.1	9.720
2	m-NT	451.4	137.1	9.680
3	p-NT	460.4	137.1	9.220
4	DNT	572.0	182.1	11.03

VAPOR PRESSURE DATA

1 o-NT

TEMP	EXP	CALC	%DEV
104.0	0.3505	0.3763	7.372
176.0	4.319	3.761	-12.91
248.0	29.43	27.75	-5.723
320.0	130.9	138.6	5.874
392.0	431.9	484.0	12.07
464.0	1141.	1255.	9.999
536.0	2556.	2567.	0.4140
608.0	5052.	4352.	-13.86

2 m-NT

TEMP	EXP	CALC	%DEV
104.0	0.2054	0.2204	7.323
176.0	2.780	2.421	-12.93
248.0	20.22	19.13	-5.410
320.0	95.17	100.6	5.728
392.0	327.1	365.9	11.86
464.0	892.1	980.0	9.847
536.0	2048.	2056.	0.4126
608.0	4127.	3562.	-13.68

3 p-NT

TEMP	EXP	CALC	%DEV
104.0	0.1926	0.2066	7.274
176.0	2.549	2.222	-12.84
248.0	18.31	17.32	-5.400
320.0	85.47	90.33	5.691
392.0	292.3	326.8	11.80
464.0	795.0	872.8	9.788
536.0	1822.	1829.	0.4056
608.0	3668.	3169.	-13.61

4 DNT

TEMP	EXP	CALC	%DEV
104.0	0.1700E-02	0.1816E-02	6.806
176.0	0.5000E-01	0.4394E-01	-12.12
248.0	0.6720	0.6379	-5.076
320.0	5.123	5.405	5.509
392.0	25.88	28.74	11.03
464.0	96.08	104.7	8.931
536.0	282.0	282.6	0.2046
608.0	690.8	603.7	-12.61

VAPOR PRESSURE DATA

1 o-NT

TEMP	EXP	CALC	%DEV
104.0	0.3505	0.3763	7.372
176.0	4.319	3.761	-12.91
248.0	29.43	27.75	-5.723
320.0	130.9	138.6	5.874
392.0	431.9	484.0	12.07
464.0	1141.	1255.	9.999
536.0	2556.	2567.	0.4140
608.0	5052.	4352.	-13.86

2 m-NT

TEMP	EXP	CALC	%DEV
104.0	0.2054	0.2204	7.323
176.0	2.780	2.421	-12.93
248.0	20.22	19.13	-5.410
320.0	95.17	100.6	5.728
392.0	327.1	365.9	11.86
464.0	892.1	980.0	9.847
536.0	2048.	2056.	0.4126
608.0	4127.	3562.	-13.68

3 p-NT

TEMP	EXP	CALC	%DEV
104.0	0.1926	0.2066	7.274
176.0	2.549	2.222	-12.84
248.0	18.31	17.32	-5.400
320.0	85.47	90.33	5.691
392.0	292.3	326.8	11.80
464.0	795.0	872.8	9.788
536.0	1822.	1829.	0.4056
608.0	3668.	3169.	-13.61

4 DNT

TEMP	EXP	CALC	%DEV
104.0	0.1700E-02	0.1816E-02	6.806
176.0	0.5000E-01	0.4394E-01	-12.12
248.0	0.6720	0.6379	-5.076
320.0	5.123	5.405	5.509
392.0	25.88	28.74	11.03
464.0	96.08	104.7	8.931
536.0	282.0	282.6	0.2046
608.0	690.8	603.7	-12.61

VAPOR ENTHALPY DATA

1 o-NT

TEMP	EXP	CALC	%DEV
104.0	21.09	21.05	-0.1805
176.0	23.26	23.28	0.7393E-01
248.0	25.42	25.46	0.1403
320.0	27.59	27.61	0.5493E-01
392.0	29.75	29.74	-0.2201E-01
464.0	31.91	31.89	-0.6780E-01
536.0	34.09	34.06	-0.9479E-01
608.0	36.24	36.27	0.8064E-01

2 m-NT

TEMP	EXP	CALC	%DEV
104.0	22.27	22.22	-0.2181
176.0	24.44	24.46	0.9522E-01
248.0	26.60	26.64	0.1690
320.0	28.77	28.79	0.6749E-01
392.0	30.93	30.92	-0.3380E-01
464.0	33.09	33.06	-0.9595E-01
536.0	35.26	35.23	-0.8949E-01
608.0	37.42	37.45	0.8829E-01

3 p-NT

TEMP	EXP	CALC	%DEV
104.0	22.39	22.34	-0.2191
176.0	24.56	24.58	0.9361E-01
248.0	26.72	26.76	0.1634
320.0	28.88	28.91	0.9195E-01
392.0	31.05	31.04	-0.4762E-01
464.0	33.21	33.17	-0.1107
536.0	35.37	35.34	-0.7307E-01
608.0	37.54	37.57	0.8414E-01

4 DNT

TEMP	EXP	CALC	%DEV
104.0	33.25	33.09	-0.4664
176.0	35.41	35.49	0.2198
248.0	37.56	37.70	0.3626
320.0	39.72	39.79	0.1776
392.0	41.88	41.84	-0.9198E-01
464.0	44.03	43.92	-0.2509
536.0	46.18	46.10	-0.1836
608.0	48.34	48.44	0.2051

LIQUID ENTHALPY DATA

1 o-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

2 m-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

3 p-NT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

4 DNT

TEMP	EXP	CALC	%DEV
104.0	0.8080	0.9626	19.13
176.0	2.974	2.894	-2.700
248.0	5.137	5.006	-2.540
320.0	7.300	7.231	-0.9408
392.0	9.467	9.499	0.3331
464.0	11.63	11.74	0.9332
536.0	13.79	13.88	0.6647
608.0	15.96	15.86	-0.6370

MOLAR VOLUME DATA

1 o-NT

TEMP	EXP	CALC	%DEV
104.0	20.22	20.14	-0.4041
176.0	20.72	20.79	0.3194
248.0	21.32	21.38	0.2976
320.0	22.04	22.05	0.3522E-01
392.0	22.94	22.90	-0.1886
464.0	24.09	24.05	-0.1742
536.0	25.61	25.62	0.3630E-01
608.0	27.71	27.73	0.6546E-01

2 m-NT

TEMP	EXP	CALC	%DEV
104.0	22.63	22.53	-0.4435
176.0	23.17	23.24	0.3228
248.0	23.81	23.90	0.3576
320.0	24.60	24.61	0.4262E-01
392.0	25.57	25.52	-0.1933
464.0	26.81	26.76	-0.2043
536.0	28.44	28.44	0.1473E-01
608.0	30.69	30.72	0.8881E-01

3 p-NT

TEMP	EXP	CALC	%DEV
104.0	23.31	23.20	-0.4774
176.0	23.87	23.94	0.3075
248.0	24.51	24.61	0.4030
320.0	25.30	25.32	0.9531E-01
392.0	26.27	26.22	-0.1956
464.0	27.49	27.42	-0.2490
536.0	29.07	29.06	-0.2716E-01
608.0	31.23	31.27	0.1266

4 DNT

TEMP	EXP	CALC	%DEV
104.0	35.22	35.01	-0.6052
176.0	35.82	36.03	0.5753
248.0	36.81	36.81	0.2920E-02
320.0	37.37	37.53	0.4257
392.0	38.37	38.35	-0.5901E-01
464.0	39.58	39.43	-0.3710
536.0	41.07	40.95	-0.2830
608.0	42.95	43.08	0.2944

VAPOR PRESSURE COEFFICIENTS

1	o-NT	-0.47888E-02	31082.	-0.30116E+08	0.69259E+10
2	m-NT	-0.47305E-02	30963.	-0.30215E+08	0.69236E+10
3	p-NT	-0.47027E-02	30697.	-0.30040E+08	0.68983E+10
4	DNT	-0.43852E-02	30460.	-0.31977E+08	0.72171E+10

VAPOR ENTHALPY COEFFICIENTS

1	o-NT	-0.13173E-02	0.45912E-01	-0.19670E-04	0.79481E-08
2	m-NT	-0.16365E-02	0.50441E-01	-0.25308E-04	0.10229E-07
3	p-NT	-0.16632E-02	0.50945E-01	-0.25994E-04	0.10527E-07
4	DNT	-0.51662E-02	0.92612E-01	-0.77881E-04	0.31505E-07

LIQUID ENTHALPY COEFFICIENTS

1	o-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
2	m-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
3	p-NT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07
4	DNT	0.51447E-02	-0.31719E-01	0.76809E-04	-0.31090E-07

MOLAR VOLUME COEFFICIENTS

1	o-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
2	m-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
3	p-NT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00
4	DNT	1.0000	0.00000E+00	0.00000E+00	0.00000E+00

F E E D C H A R G E

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	9.486	59.30	1301.	59.30	133.8	58.14
2	m-NT	0.6430	4.020	88.18	4.020	9.110	3.957
3	p-NT	5.865	36.67	804.3	36.66	87.24	37.89
4	DNT	0.2000E-02	0.1250E-01	0.3643	0.1661E-01	0.3303E-01	0.1435E-01
	TOTAL	16.00	100.0	2194.	100.0	230.2	100.0

(LAPSED CPU TIME = 0.58 SEC ENTRY COUNT = 1)

ITERATION NUMBER 0

S T E A D Y - S T A T E P R O F I L E

STAGE	DEL TEMP	TEMPERATURE	VAPOR RATE	LIQUID RATE	DEL HOLDUP	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE	SUMDX
1	0.0000E+00	185.5	0.0000E+00	0.0000E+00	0.0000E+00	0.5000E-01	0.5000E-01	0.0000E+00	5.000	0.0000E+00
2	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
3	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
4	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
5	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
6	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
7	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
8	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
9	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
10	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
11	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
12	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
13	0.0000E+00	197.7	0.0000E+00	0.0000E+00	0.0000E+00	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.0000E+00
14	0.0000E+00	191.5	0.0000E+00	0.0000E+00	0.0000E+00	15.83	0.0000E+00	0.0000E+00	5.000	0.0000E+00

SCALE	CONV	ICONV
0.0000E+00	0.0000E+00	0

ITERATION NUMBER 5

S T E A D Y - S T A T E P R O F I L E

STAGE	DEL TEMP	TEMPERATURE	VAPOR RATE	LIQUID RATE	DEL HOLDUP	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE	SUMDX
1	-0.1666E-06	185.6	0.0000E+00	16.87	-0.7305E-18	0.5000E-01	0.5000E-01	-344.3	5.000	0.2798E-08
2	-0.1272E-06	185.6	16.87	16.86	-0.1160E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.4419E-08
3	-0.2792E-07	185.7	16.86	16.86	0.7912E-19	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1396E-07
4	0.1037E-06	185.8	16.86	16.85	0.2347E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.8307E-08
5	0.3655E-07	185.9	16.85	16.84	0.1517E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.3291E-08
6	0.8992E-08	186.0	16.84	16.83	-0.1031E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.9360E-09
7	-0.6053E-10	186.2	16.83	16.81	-0.5865E-19	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1582E-09
8	-0.9619E-09	186.5	16.81	16.79	-0.4468E-22	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.5852E-11
9	-0.3396E-09	186.8	16.79	16.76	0.2298E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1006E-10
10	-0.1091E-09	187.3	16.76	16.72	-0.1128E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1286E-10
11	-0.1101E-09	187.9	16.72	16.67	0.3036E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1604E-10
12	-0.1384E-09	188.7	16.67	16.60	0.3464E-19	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.1943E-10
13	-0.1670E-09	189.7	16.60	16.51	-0.4164E-18	0.1000E-01	0.1000E-01	0.0000E+00	5.000	0.2273E-10
14	-0.1939E-09	191.5	16.51	0.0000E+00	0.5040E-18	15.83	15.83	344.3	5.000	0.2574E-10

SCALE	CONV	ICONV
1.000	1.000	0

(LAPSED CPU TIME = 1.59 SEC ENTRY COUNT = 2)

TOTAL TIME, HOURS	0.3584E-04
CUMULATIVE TIME, HOURS	0.3584E-04
CUT TIME, HOURS	0.3584E-04
TIME STEP, HOURS	0.3584E-04
OVERHEAD TEMPERATURE	185.6
AT 760 MM HG	460.3
K = 12	440.1

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	1.521	99.20	208.6	99.20	21.46	99.16
2	m-NT	0.2271E-02	0.1481	0.3114	0.1481	0.3217E-01	0.1487
3	p-NT	0.1002E-01	0.6536	1.374	0.6536	0.1491	0.6888
4	DNT	0.2458E-11	0.1603E-09	0.4477E-09	0.2129E-09	0.4059E-10	0.1876E-09
	TOTAL	1.533	100.0	210.3	100.0	21.64	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.5450E-04	99.20	0.7474E-02	99.20	0.7690E-03	99.16
2	m-NT	0.8138E-07	0.1481	0.1116E-04	0.1481	0.1153E-05	0.1487
3	p-NT	0.3592E-06	0.6537	0.4925E-04	0.6537	0.5342E-05	0.6889
4	DNT	0.2380E-13	0.4331E-07	0.4335E-11	0.5753E-07	0.3930E-12	0.5068E-07
	TOTAL	0.5494E-04	100.0	0.7535E-02	100.0	0.7755E-03	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.5450E-04	99.20	0.7474E-02	99.20	0.7690E-03	99.16
2	m-NT	0.8138E-07	0.1481	0.1116E-04	0.1481	0.1153E-05	0.1487
3	p-NT	0.3592E-06	0.6537	0.4925E-04	0.6537	0.5342E-05	0.6889
4	DNT	0.2380E-13	0.4331E-07	0.4335E-11	0.5753E-07	0.3930E-12	0.5068E-07
	TOTAL	0.5494E-04	100.0	0.7535E-02	100.0	0.7755E-03	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.5450E-04	99.20	0.7474E-02	99.20	0.7690E-03	99.16
2	m-NT	0.8138E-07	0.1481	0.1116E-04	0.1481	0.1153E-05	0.1487
3	p-NT	0.3592E-06	0.6537	0.4925E-04	0.6537	0.5342E-05	0.6889
4	DNT	0.2380E-13	0.4331E-07	0.4335E-11	0.5753E-07	0.3930E-12	0.5068E-07
	TOTAL	0.5494E-04	100.0	0.7535E-02	100.0	0.7755E-03	100.0

NUMBER OF STEPS	1
NUMBER OF FUNCTION CALLS	4
NUMBER OF JACOBIAN CALLS	1

TIME, HOURS 0.3584E-04

DISTILLATE RATE 1.533

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	185.6	0.0000E+00	15.33	0.5000E-01	0.5000E-01	-344.3	5.000
2	185.6	16.87	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
3	185.7	16.86	15.32	0.1000E-01	0.1000E-01	0.0000E+00	5.000
4	185.8	16.86	15.32	0.1000E-01	0.1000E-01	0.0000E+00	5.000
5	185.9	16.85	15.31	0.1000E-01	0.1000E-01	0.0000E+00	5.000
6	186.0	16.84	15.29	0.1000E-01	0.1000E-01	0.0000E+00	5.000
7	186.2	16.83	15.28	0.1000E-01	0.1000E-01	0.0000E+00	5.000
8	186.5	16.81	15.25	0.1000E-01	0.1000E-01	0.0000E+00	5.000
9	186.8	16.79	15.22	0.1000E-01	0.1000E-01	0.0000E+00	5.000
10	187.3	16.76	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
11	187.9	16.72	15.13	0.1000E-01	0.1000E-01	0.0000E+00	5.000
12	188.7	16.67	15.07	0.1000E-01	0.1000E-01	0.0000E+00	5.000
13	189.7	16.60	14.98	0.1000E-01	0.1000E-01	0.0000E+00	5.000
14	191.5	16.51	0.0000E+00	15.83	15.83	344.3	5.000

TOTAL TIME, HOURS	2.770
CUMULATIVE TIME, HOURS	2.770
CUT TIME, HOURS	2.770
TIME STEP, HOURS	0.1456E-01
OVERHEAD TEMPERATURE	186.0
AT 760 MM HG	460.8
K = 12	440.5

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	1.491	97.23	204.4	97.23	21.03	97.11
2	m-NT	0.7683E-02	0.5011	1.054	0.5011	0.1089	0.5026
3	p-NT	0.3473E-01	2.265	4.763	2.265	0.5166	2.385
4	DNT	0.3922E-11	0.2558E-09	0.7143E-09	0.3397E-09	0.6476E-10	0.2990E-09
	TOTAL	1.533	100.0	210.3	100.0	21.66	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	4.163	98.01	570.9	98.01	58.73	97.92
2	m-NT	0.1535E-01	0.3615	2.105	0.3615	0.2175	0.3626
3	p-NT	0.6913E-01	1.628	9.480	1.628	1.028	1.714
4	DNT	-0.2781E-07	-0.6548E-06	-0.5066E-05	-0.8697E-06	-0.4593E-06	-0.7657E-06
	TOTAL	4.247	100.0	582.5	100.0	59.98	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	4.163	98.01	570.9	98.01	58.73	97.92
2	m-NT	0.1535E-01	0.3615	2.105	0.3615	0.2175	0.3626
3	p-NT	0.6913E-01	1.628	9.480	1.628	1.028	1.714
4	DNT	-0.2781E-07	-0.6548E-06	-0.5066E-05	-0.8697E-06	-0.4593E-06	-0.7657E-06
	TOTAL	4.247	100.0	582.5	100.0	59.98	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	4.163	98.01	570.9	98.01	58.73	97.92
2	m-NT	0.1535E-01	0.3615	2.105	0.3615	0.2175	0.3626
3	p-NT	0.6913E-01	1.628	9.480	1.628	1.028	1.714
4	DNT	-0.2781E-07	-0.6548E-06	-0.5066E-05	-0.8697E-06	-0.4593E-06	-0.7657E-06
	TOTAL	4.247	100.0	582.5	100.0	59.98	100.0

NUMBER OF STEPS	498
NUMBER OF FUNCTION CALLS	909
NUMBER OF JACOBIAN CALLS	209

TIME. HOURS 2.770

DISTILLATE RATE 1.533

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	185.9	0.0000E+00	15.33	0.5000E-01	0.5000E-01	-344.8	5.000
2	186.0	16.87	15.32	0.1000E-01	0.1000E-01	0.0000E+00	5.000
3	186.2	16.85	15.31	0.1000E-01	0.1000E-01	0.0000E+00	5.000
4	186.4	16.84	15.29	0.1000E-01	0.1000E-01	0.0000E+00	5.000
5	186.7	16.82	15.26	0.1000E-01	0.1000E-01	0.0000E+00	5.000
6	187.1	16.80	15.23	0.1000E-01	0.1000E-01	0.0000E+00	5.000
7	187.6	16.77	15.20	0.1000E-01	0.1000E-01	0.0000E+00	5.000
8	188.1	16.73	15.15	0.1000E-01	0.1000E-01	0.0000E+00	5.000
9	188.8	16.69	15.11	0.1000E-01	0.1000E-01	0.0000E+00	5.000
10	189.6	16.64	15.05	0.1000E-01	0.1000E-01	0.0000E+00	5.000
11	190.5	16.58	14.99	0.1000E-01	0.1000E-01	0.0000E+00	5.000
12	191.4	16.53	14.93	0.1000E-01	0.1000E-01	0.0000E+00	5.000
13	192.4	16.47	14.86	0.1000E-01	0.1000E-01	0.0000E+00	5.000
14	193.9	16.40	0.0000E+00	11.58	11.58	344.8	5.000

TOTAL TIME, HOURS	6.955
CUMULATIVE TIME, HOURS	6.955
CUT TIME, HOURS	4.185
TIME STEP, HOURS	0.9802E-02
OVERHEAD TEMPERATURE	199.2
AT 760 MM HG	475.6
K = 12	457.3

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.3184	20.77	43.67	20.77	4.493	20.02
2	m-NT	0.1709	11.15	23.44	11.15	2.421	10.79
3	p-NT	1.044	68.08	143.2	68.08	15.53	69.19
4	DNT	0.1587E-10	0.1035E-08	0.2890E-08	0.1374E-08	0.2620E-09	0.1168E-08
	TOTAL	1.533	100.0	210.3	100.0	22.44	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	5.177	80.69	709.9	80.69	73.04	79.98
2	m-NT	0.1986	3.096	27.24	3.096	2.814	3.082
3	p-NT	1.040	16.22	142.7	16.22	15.47	16.94
4	DNT	-0.3182E-07	-0.4960E-06	-0.5796E-05	-0.6588E-06	-0.5255E-06	-0.5754E-06
	TOTAL	6.416	100.0	879.9	100.0	91.33	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	9.340	87.59	1281.	87.59	131.8	87.09
2	m-NT	0.2140	2.007	29.35	2.007	3.032	2.004
3	p-NT	1.109	10.40	152.1	10.40	16.50	10.91
4	DNT	-0.5964E-07	-0.5593E-06	-0.1086E-04	-0.7428E-06	-0.9848E-06	-0.6509E-06
	TOTAL	10.66	100.0	1462.	100.0	151.3	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	9.340	87.59	1281.	87.59	131.8	87.09
2	m-NT	0.2140	2.007	29.35	2.007	3.032	2.004
3	p-NT	1.109	10.40	152.1	10.40	16.50	10.91
4	DNT	-0.5964E-07	-0.5593E-06	-0.1086E-04	-0.7428E-06	-0.9848E-06	-0.6509E-06
	TOTAL	10.66	100.0	1462.	100.0	151.3	100.0

NUMBER OF STEPS	1286
NUMBER OF FUNCTION CALLS	2370
NUMBER OF JACOBIAN CALLS	564

TIME, HOURS 6.955

DISTILLATE RATE 1.533

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	198.1	0.0000E+00	15.33	0.5000E-01	0.5000E-01	-362.3	5.000
2	199.2	16.87	15.28	0.1000E-01	0.1000E-01	0.0000E+00	5.000
3	200.0	16.82	15.25	0.1000E-01	0.1000E-01	0.0000E+00	5.000
4	200.6	16.78	15.22	0.1000E-01	0.1000E-01	0.0000E+00	5.000
5	201.1	16.76	15.21	0.1000E-01	0.1000E-01	0.0000E+00	5.000
6	201.4	16.74	15.20	0.1000E-01	0.1000E-01	0.0000E+00	5.000
7	201.6	16.73	15.19	0.1000E-01	0.1000E-01	0.0000E+00	5.000
8	201.7	16.72	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
9	201.8	16.72	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
10	201.9	16.71	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
11	201.9	16.71	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
12	202.0	16.71	15.18	0.1000E-01	0.1000E-01	0.0000E+00	5.000
13	202.0	16.71	15.17	0.1000E-01	0.1000E-01	0.0000E+00	5.000
14	202.0	16.71	0.0000E+00	5.163	5.163	362.3	5.000

TOTAL TIME, HOURS	9.805
CUMULATIVE TIME, HOURS	9.805
CUT TIME, HOURS	2.850
TIME STEP, HOURS	0.4317E-02
OVERHEAD TEMPERATURE	202.7
AT 760 MM HG	479.2
K = 12	461.7

INSTANTANEOUS DISTILLATE DRAWOFF PER HOUR

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.1880E-05	0.1226E-03	0.2578E-03	0.1226E-03	0.2652E-04	0.1165E-03
2	m-NT	0.6742E-01	4.398	9.246	4.398	0.9552	4.197
3	p-NT	1.466	95.60	201.0	95.60	21.80	95.80
4	DNT	0.1100E-09	0.7174E-08	0.2003E-07	0.9528E-08	0.1816E-08	0.7981E-08
	TOTAL	1.533	100.0	210.3	100.0	22.76	100.0

CUT DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.1464	3.350	20.07	3.350	2.065	3.197
2	m-NT	0.4040	9.245	55.40	9.245	5.723	8.859
3	p-NT	3.819	87.41	523.8	87.41	56.81	87.94
4	DNT	-0.5508E-07	-0.1260E-05	-0.1003E-04	-0.1674E-05	-0.9095E-06	-0.1408E-05
	TOTAL	4.370	100.0	599.3	100.0	64.60	100.0

CUMULATIVE DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	9.486	63.10	1301.	63.10	133.8	61.99
2	m-NT	0.6180	4.111	84.75	4.111	8.755	4.055
3	p-NT	4.929	32.79	675.9	32.79	73.31	33.96
4	DNT	-0.1147E-06	-0.7631E-06	-0.2089E-04	-0.1013E-05	-0.1894E-05	-0.8774E-06
	TOTAL	15.03	100.0	2062.	100.0	215.9	100.0

TOTAL DISTILLATE DRAWOFF

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	9.486	63.10	1301.	63.10	133.8	61.99
2	m-NT	0.6180	4.111	84.75	4.111	8.755	4.055
3	p-NT	4.929	32.79	675.9	32.79	73.31	33.96
4	DNT	-0.1147E-06	-0.7631E-06	-0.2089E-04	-0.1013E-05	-0.1894E-05	-0.8774E-06
	TOTAL	15.03	100.0	2062.	100.0	215.9	100.0

NUMBER OF STEPS	2353
NUMBER OF FUNCTION CALLS	4378
NUMBER OF JACOBIAN CALLS	1022

TIME, HOURS 9.805

DISTILLATE RATE 1.533

U N S T E A D Y - S T A T E P R O F I L E

STAGE	TEMPERATURE	VAPOR RATE	LIQUID RATE	MOL HOLDUP	VOL HOLDUP	HEAT DUTY	PRESSURE
1	202.7	0.0000E+00	15.33	0.5000E-01	0.5000E-01	-366.5	5.000
2	202.7	16.87	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
3	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
4	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
5	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
6	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
7	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
8	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
9	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
10	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
11	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
12	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
13	202.7	16.86	15.33	0.1000E-01	0.1000E-01	0.0000E+00	5.000
14	202.8	16.86	0.0000E+00	0.7932	0.7932	366.5	5.000

(LAPSED CPU TIME = 179.21 SEC ENTRY COUNT = 3)

F E E D C H A R G E (REMAINING IN STILL)

NO	NAME	LBMOLES	MOLE %	POUNDS	WEIGHT %	GALLONS	VOLUME %
1	o-NT	0.1474E-06	0.1530E-04	0.2021E-04	0.1529E-04	0.2080E-05	0.1453E-04
2	m-NT	0.2504E-01	2.600	3.434	2.598	0.3548	2.479
3	p-NT	0.9362	97.19	128.4	97.13	13.92	97.29
4	DNT	0.2000E-02	0.2076	0.3643	0.2756	0.3303E-01	0.2308
	TOTAL	0.9632	100.0	132.2	100.0	14.31	100.0

(LAPSED CPU TIME = 180.76 SEC ENTRY COUNT = 4)

BATCH TERMINATING

```
$!
$!
$! COMMAND FILES USED TO EXECUTE THE MAIN SOURCE CODE
$!
$! SUBMIT NITS.COM
$!  --------------
$!
$  SUBMIT/NOTIFY/QUEUE=MEDIUM   NITS_SUB.COM
$!
$! NITS_SUB.COM
$! ------------
$!
$!  BATCH dISTILLATION
$!  BY NITIN ...
$! ---------------------------------
$!
$ ASSIGN   NITS.DAT   FOR005
$ ASSIGN   NITS.OUT   FOR006
$!
$ RUN   NITS.EXE
$!
$ EXIT
```

BIBLIOGRAPHY

(1) Gillespie, R. J. and Millen, D. J., *Quart. Revs.*, **2** (1948), 277.

(2) Haun, J. K. and Kobe, K. A., *Ind. Eng. Chem.*, **32** (1940), 1095.

(3) Albright, L. F., *Kirk-Othmer's "Encyclopedia of Chemical Tech" 2nd ed.*, **13** (1967), 784.

(4) Albright, L. F. and Hanson, C., *Chem. Eng. Prog. Tech. Manual "Loss prevention"*, **3** (1969), 26.

(5) Hanson, C., Marsland, J. G. and Wilson, G., *Chem. Eng. Sci.*, **26** (1971), 1513.

(6) Hanson, C., Marsland, J. G. and Naz, M. A., *Chem. Eng. Sci.*, **29** (1974), 297.

(7) Strachan, A. N. and Cox, P. R., *Chem. Eng. Sci.*, **26** (1971), 1113.

(8) Strachan, A. N. and Cox, P. R., *Chem. Eng. Sci.*, **4** (1972), 253.

(9) Strachan, A. N., Cox, P. R. and Chapman, J. W., *Chem. Eng. Sci.*, **29** (1974), 1247.

(10) Lewis, W. K. and Suen, T. J., *Ind. Eng. Chem.*, **32** (1940), 1095.

(11) Barduhn, A. J. and Kobe, K. A., *Ind. Eng. Chem.*, **48** (1956), 1305.

(12) Sohrabi, M., *PhD Thesis, Uni. of Bradford, UK.*, (1972).

(13) Astarita, E. G., " *Mass Transfer with Chemical Reaction* ", *Elsevier.*, (1967).

(14) Danckwert, P. V., " *Gas Liquid Reactions* ", *McGraw Hill*, (1970).

(15) McKinley, C. and White, R. R., *Trans. AIChE.*, **40** (1944), 143.

(16) Biggs, R. D. and White, R. R., *AIChE.*, **2** (1956), 26.