NUREG/CR-1367
SAND 80-0403

AEROSOL USERS MANUAL

Fred Gelbard

Sandia National Laboratories
Albuquerque, NM 87185
operated by
Sandia Corporation
for the
U. S. Department of Energy

Prepared for

Division of Reactor Safety Research
Office of Nuclear Regulatory Research
U. S. Nuclear Regulatory Commission
Washington, DC 20555

Under Memorandum of Understanding DOE 40-550-75
NRC FIN No. A1218

ABSTRACT

This report documents the AEROSOL computer code as of February 25, 1980. AEROSOL is a general purpose code for solving the conservation equations of a spatially homogenous single chemical component aerosol. The mechanisms which may be incorporated to determine the evolving particle size distribution are: (1) coagulation; (2) deposition; (3) aerosol generation; (4) nucleation; and (5) particle growth by condensation. Detailed instructions for implementing the code and nine example problems are discussed.

The potential user should note that work is still in progress. Specifically, although not available at the time of this report, a new code will be available which not only computes the particle size distribution, but also the distribution of chemical species with respect to particle size. The new code, which is significantly smaller and faster than AEROSOL (if implemented properly), will be called MAEROS and is based on work reported in references (7) and (8) of this manual. In addition, the numerical technique used in MAEROS completely avoids the curve fitting problem associated with the numerical technique used in AEROSOL, as discussed on pages 17 and 37 of this manual.

TABLE OF CONTENTS

		Page
	ABSTRACT	1
I.	INTRODUCTION	4
II.	GENERAL DYNAMIC EQUATION	6
	A. Discrete General Dynamic Equation	6
	B. Discrete-Continuous General Dynamic Equation	7
	C. Continuous General Dynamic Equation	11
III.	NUMERICAL TECHNIQUES	12
	A. Continuous Regime	12
	B. Discrete Regime	13
	C. Temporal Integration	14
	D. Integrations with Respect to Particle Size	14
	E. Units for Dimensional Quantities	15
	F. Possible Numerical Difficulties	15
IV.	SUBROUTINES	18
	A. Coefficient Specification Routines	19
	1. BETA	19
	2. BOUNDR	20
	3. CSOURC	21
	4. DSOURC	23
	5. EVAP	24
	6. GROWTH	25
	7. XINTL	26
	B. Interpolation, Integration and Output Routines	26
	1. DISTW	26
	2. SETUP	28

3. DERIVT 28

4. BLOCK DATA 29

5. OUTPUT 30

C. Convenience Routines 33

1. CHECK 33

2. TOTAL 34

3. CHKTBL 34

4. STEADY 35

D. Central Routines 35

E. Ordinary Differential Equation Package 36

V. INPUT DATA 37

A. Detailed Discussion 37

B. Condensed Reference List 40

VI. COMMON BLOCKS AND ARRAY STORAGE 43

VII. EXAMPLES 47

A. Discussion 47

B. Listing 50

FIGURES 96

FOOTNOTES 108

REFERENCES 109

NOTATION 110

I. INTRODUCTION

This manual is written for the first-time user of the AEROSOL program, which simulates the size distribution dynamics of a spatially homogenous aerosol. The program is a general purpose code for solving population balance equations and is therefore not restricted to only aerosol problems. However, because the development of the code was motivated by the need to simulate atmospheric aerosols, all variables will be discussed as pertaining to an aerosol. Since it is assumed that the potential user is familiar with the derivation of population balance equations or what is often called the General Dynamic Equation (GDE) for aerosols, the details of the theory and numerics have been omitted, but ample references are provided to guide the interested reader. Complete and detailed instructions are given on all aspects of the code and additional references are not required to implement the code. Because this manual is intended to also serve as a reference manual, many suggestions and warnings are repeated throughout the manual.

The manual is divided into seven sections. In Section II a discussion of the forms of the GDE that can be solved by AEROSOL is given. To familiarize the user with the terminology used in the manual, a brief discussion of the numerical techniques is given in Section III. The code consists of five types of subroutines and each type is discussed Section IV. The proper format for input data is given in Section V. Array dimensions for altering storage requirments are given in Section

VI. Finally, in Section VII we present the nine example problems which are referred to throughout the manual.

II. GENERAL DYNAMIC EQUATION

II.A. Discrete General Dynamic Equation

In the most fundamental form of the GDE, particles are represented as consisting of integer multiples of a single structural unit, typically a molecule. In these discrete equations, particles differ only in the number of monomers they contain. For a spatially homogenous aerosol the concentration of monomers is governed by (1),

$$\frac{dN_1}{dt} = -N_1 \sum_{j=1}^{\infty} \beta_{1,j} N_j + \sum_{j=2}^{\infty} (1 + \delta_{2,j}) E_j N_j + \bar{S}_1(t) + \bar{R}_1(N_1, t) \qquad [1]$$

and the concentration of i-mers where $i > 1$ is governed by

$$\frac{dN_i}{dt} = \frac{1}{2} \sum_{j=1}^{i-1} \beta_{i-j,j} N_{i-j} N_j - N_i \sum_{j=1}^{\infty} \beta_{i,j} N_j + E_{i+1} N_{i+1} - E_i N_i + \bar{S}_i(t) + \bar{R}_i(N_i, t) \qquad [2]$$

where

$$\delta_{2,j} = \begin{cases} 0 & j \neq 2 \\ 1 & j = 2 \end{cases} \qquad [3]$$

t is time, N_i is the concentration of i-mers, $\beta_{i,j}$ is the coagulation coefficient between i-mers and j-mers, E_i is the monomer evaporation coefficient of an i-mer, $\overline{S}_i$ is the generation rate of i-mers, and $\overline{R}_i$ represents the removal rate of i-mers and is therefore usually a negative quantity.

II.B. Discrete-Continuous General Dynamic Equation

Although the discrete GDE as given in Eqs. [1] and [2] is an accurate description of aerosol dynamics, the number of equations needed to simulate actual aerosols can be immense. For large particles, the difference in size between an i-mer and an (i+1)-mer is relatively small. Thus if N_k approaches N_{k+1} where $k >> 1$, the discrete concentrations may be represented by $n(x_i,t)$, which is a continuous function in the limit as $x_1/x_k \to 0$, where $i \geq k + 1$, x_i is the mass (or any conserved quantity) of an i-mer and $n(x,t)$ is the size distribution function defined by,[1]

$$N_i = \int_{x_i}^{x_i+x_1} n(x,t)\, dx \qquad i > k \qquad [4]$$

Since the range of integration of Eq.[4] is so small, $n(x,t)$ may be approximated as a constant in that range and therefore

$$n(x_i,t) = \frac{N_i}{x_1} \qquad i > k \qquad [5]$$

By dividing the particle size domain into a discrete regime for particles smaller than or equal to a k-mer and a continuous regime for particles larger than or equal to a (k+1)-mer, and assuming mass (or any quantity represented by x) is conserved by coagulation, Eq. [1] may be expressed as

$$\frac{dN_1}{dt} = -N_1\left[\sum_{j=1}^{k} \beta_{1,j}N_j + \int_{x_{k+1}}^{\infty} \beta(x_1,y)n(y,t)\,dy\right]$$

$$+ \sum_{j=2}^{k} (1+\delta_{2,j})E_jN_j + \int_{x_{k+1}}^{\infty} E(y)n(y,t)\,dy \qquad [6]$$

$$+ \bar{S}_1(t) + \bar{R}_1(N_1,t)$$

where $\beta(x_i,x_j) = \beta_{i,j}$ and $E(x_i) = E_i$. For $2 \leq i \leq k$, Eq. [2] becomes

$$\frac{dN_i}{dt} = \frac{1}{2}\sum_{j=1}^{i-1} \beta_{i-j,j}N_{i-j}N_j - N_i\left[\sum_{j=1}^{k} \beta_{i,j}N_j + \int_{x_{k+1}}^{\infty} \beta(x_i,y)n(y,t)\,dy\right]$$

$$-E_iN_i + \bar{S}_i(t) + \bar{R}_i(N_i,t) \qquad [7]$$

$$+ \begin{cases} E_{i+1}N_{i+1} & 2 \leq i \leq k-1 \\ \displaystyle\int_{x_{k+1}}^{x_{k+2}} E(y)n(y,t)\,dy & i = k \end{cases}$$

For $x_{k+1} \leq x_i \leq x_{2k}$, Eq. [2] becomes

$$\frac{dn(x_i,t)}{dt} = \sum_{j=1}^{i-k-1} \beta_{i-j,j} N_j n(x_i - x_j, t) \qquad i \geq k+2$$

$$\frac{1}{2} \sum_{j=i-k}^{k} \frac{\beta_{i-j,j} N_{i-j} N_j}{x_1} \qquad [8]$$

$$-n(x_i,t) \left[\sum_{j=1}^{k} \beta_{i,j} N_j + \int_{x_{k+1}}^{\infty} \beta(x_i,y) n(y,t)\, dy \right]$$

$$+ E_{i+1} n(x_{i+1},t) - E_i n(x_i,t)$$

$$+ S_i(t) + R_i[n(x_i,t),t]$$

where $S_i = \bar{S}_i/x_1$ and $R_i = \bar{R}_i(N_i,t)/x_1$.

Finally, for $x > x_{2k}$, Eq. [2] becomes

$$\frac{\partial n(x,t)}{\partial t} = \sum_{j=1}^{k} \beta(x-x_j,x_j)N_j n(x-x_j,t) + \frac{1}{2}\int_{x_{k+1}}^{x-x_{k+1}} \beta(x-y,y)n(x-y,t)n(y,t)\,dy \qquad [9]$$

$$-n(x,t)\left[\sum_{j=1}^{k} \beta(x,x_j)N_j + \int_{x_{k+1}}^{\infty} \beta(x,y)n(y,t)\,dy\right]$$

$$+ E(x+x_1)n(x+x_1,t) - E(x)n(x,t)$$

$$+ S(x,t) + R[n(x,t),x,t]$$

where $S(x_i,t) = S_i(t)$ and $R[n(x_i,t),x_i,t] = R_i[n(x_i,t),t]$. Combined with the appropriate initial conditions, Eqs. [6-9] constitute the discrete-continuous GDE.

II.C. Continuous General Dynamic Equation

If only the continuous regime is of interest, Eqs. [8] and [9] may be reduced to the continuous GDE (1),

$$\frac{\partial n(x,t)}{\partial t} + \frac{\partial [I(x,t)n(x,t)]}{\partial x} = \frac{1}{2}\int_{x_{k+1}}^{x-x_{k+1}} \beta(x-y,y)n(x-y,t)n(y,t)\ dy$$

$$-n(x,t)\int_{x_{k+1}}^{\infty} \beta(x,y)n(y,t)\ dy + S(x,t) \qquad [10]$$

$$+ R[n(x,t),x,t]$$

where $I(x,t) = [\beta(x,x_1)N_1 - E(x)]x_1$ is the net growth rate of a particle of size x due to condensation of monomer and the second term on the left hand side of Eq. [10] is called the condensation term.

Both the discrete-continuous GDE and the continuous GDE can be solved by the code. If the dynamics of molecular cluster is of interest, the discrete-continuous GDE is recommended, otherwise the simpler continuous GDE should be used. Since the code can solve the GDE for arbitrary functional forms of the coefficients β, I, S, R and E, and initial conditions, it is in the specification of these functions which determines the solution to a specific problem. In essence, the AEROSOL code is not a model but a general algorithm for solving the conservation equations for aerosols. The modeling aspects of the problem are introduced through the assumed functional forms of the coefficients.

III. NUMERICAL TECHNIQUES

III.A. Continuous Regime

In the continuous regime, $n(x,t)$ is assumed to be a continuous function of x. Therefore, by determining the temporal variations of $n(x,t)$ at a finite number of grid points in x, the values of $n(x,t)$ at intermediate points can be determined by interpolation. Because x typically varies over many orders of magnitude, the following logarithmic rescaling is used in the code,

$$w = \frac{\ln(x/x_a)}{\ln(x_b/x_a)} \qquad [11]$$

$$m(w,t) = xn(x,t)\ln(x_b/x_a) \qquad [12]$$

Where x_a and x_b are the lower and upper particle mass limits of interest, respectively. Thus the rescaled independent variable is w, which varies from 0 to 1 and $m(w,t)$ is the rescaled distribution function. By substituting Eqs. [11] and [12] into the GDE, the rescaled equations can be obtained (2). Since the code will automatically rescale user input, the user need not be concerned with the details of logarithmically transforming the GDE. However, in the internal documentation of the code, the "M" distribution of "W" refers to the quantities given in Eqs. [11] and [12].

The maximum number of grid points has been internally set to 40 but can be changed as discussed in Section VI.

The first and last grid points are automatically set at x_a (i.e., w=0), and x_b (i.e., w=1), respectively. The code automatically places the user specified number of grid points logarithmically spaced in x (i.e., linearly spaced in w). However, the user has the option of positioning grid points anywhere in the region $[x_a, x_b]$.

Interpolations are performed on m(w,t) over the domain $0 \leq w \leq 1$, and the user has the option of using virtually any desired interpolation formula. For convenience, the code is supplied with the option of using cubic, linear or logarithmic spline interpolation formulas. One also has the option of switching formulas during a simulation. The routines DISTW, SETUP and DERIVT as discussed in Section IV.B., determine the interpolation formula.

III.B. Discrete Regime

In the discrete regime N_i, (i=1,...,k) is governed by solving a set of k ordinary differential equations (i.e., Eqs. [6] and [7]). The user need only specify the number of discrete sizes and based on the size of the monomer, the code will determine all the cluster sizes and the smallest particle size in the continuous regime. One must also specify grid point locations in the continuous regime in the range $x_{k+1} \leq x \leq x_{2k}$. Since the distribution is assumed to be continuous in this range, not all k points have to be specified. These points are called multiplets and are discussed in Section V.A.

It is often assumed that the concentration of molecular clusters has reached steady state. Therefore, the user has the option of 1) doing a transient calculation of the cluster concentrations and not invoking the steady state approximation, 2) using a transient calculation up to a user specified time and then invoking the steady state approximation, 3) doing a transient calculation and simultaneously computing the steady state concentrations or 4) initially doing a transient calculation but having the code switch to the steady state approximation once the transients have decayed.

III.C. Temporal Integration

The temporal variations of $n(x,t)$ at the grid points and $N_i,(i=1,\ldots,k)$ are governed by a set of first order ordinary differential equations. Sandia's O.D.E. package is supplied with AEROSOL to integrate the system in time and a detailed discussion of the package is given in (3).

III.D. Integrations with Respect to Particle Size

Gaussian-Legendre quadrature is used to evaluate the coagulation integrals over particle size (4). Any even point formula may be used by supplying the appropriate quadrature points and weighting factors in the subroutine BLOCK DATA which is discussed in Section IV.B. For convenience, the code is supplied with 12-, 24- and 40-point formulas and is set up for a 24-point formula. To date, a 24-point formula has been more than adequate for all problems and the author would be interested in any problem which required a 40-point formula to

obtain an accurate solution. Quadrature points and weighting factors are tabulated in (5) and the array dimensions needed to accommodate various formulas are discussed in Section VI. If integrations over particle size are to be performed in any of the user supplied routines, one can access the quadrature points and weight factors through the common blocks AERSL3 and AERSL8 as discussed in Sections IV.B.4 and IV.B.5.

III.E. Units for Dimensional Quantities

Since the coefficients and initial distribution are supplied by the user, they may be dimensionless or in any consistent set of units. However, if the OUTPUT routine as supplied with the code is used, cgs units (i.e., centimeter-gram-second) will be printed adjacent to the numerical results regardless of the units used on input. Clearly if one is using dimensionless variables or is not using cgs units, the printed units should be disregarded or the user may replace the OUTPUT routine supplied with the code. To remind the user of the units consistent with OUTPUT, the appropriate units for subroutines will be given in Section IV.

III.F. Possible Numerical Difficulties

Although the numerical techniques discussed above should work for nearly all problems of practical interest, the user should be aware of two situations which may cause some difficulty.

Due to coagulation and growth, particles larger than those of the initial distribution are formed. However, by using a finite computational domain, these newly formed particles are artifically removed from the computations. Therefore, the interactions of particles within the computational domain and those outside the domain are neglected. This "finite domain error" (2) must be distinguished from errors usually associated with numerical solutions, in that this error is present for any method which uses a finite computational domain, regardless of its accuracy. Therefore, one is advised to use a computational domain large enough to include nearly all the particles of the aerosol, yet small enough so that the particle size domain of interest is not obscured. Fortunately, for atmospheric aerosols, removal mechanisms such as gravitational settling effectively bounds the particle size domain. One may check for finite domain errors by checking that the total aerosol mass is conserved. Although the suspended, generated and removed aerosol mass is printed for this purpose, one should be aware that the removed aerosol mass is computed by using a mass balance and thus no checks can be made if removal mechanisms are incorporated in the G.D.E.

The second problem that the user may encounter involves interpolation of the distribution. Although one may choose virtually any interpolation formula, an inaccurate formula may result in negative values of the distribution. A high order formula may be "smoother" than a low order formula but

is generally more likely to yield negative values in regions where the distribution varies rapidly with particle size[2]. Therefore, the code has been constructed such that grid points may be automatically added to regions in which interpolation may be difficult.

IV. SUBROUTINES

Excluding the O.D.E. package, AEROSOL consists of 20 routines which can be divided into four basic categories. Routines in the first category specify the kinetic coefficients and initial conditions. Since these routines define the problem, the user will undoubtedly want to write these routines for his own particular applications. Routines in the second category determine the interpolation and integration methods, and the form of the output. Although one may be interested in quantities other than those produced by the OUTPUT routine supplied with code, the interpolation and integration methods should be adequate for most applications. For the user's convenience, there are four routines in the third category which 1) diagnose user input for possible errors or inconsistencies, 2) check the interpolation method and add grid points as needed, 3) compute variations in the first moment of the distribution and 4) compute the steady state approximate solution for the discrete regime. Although checking user input and the interpolation method is not essential for a simulation, experience has shown that if these routines are implemented, errors which might have gone undetected are usually found before a simulation begins. Routines in the fourth category control the code by reading the input data, calculating coefficients, storing values of the kinetic coefficients and providing the derivatives needed for the O.D.E. package. Except for possibly using a package

other than O.D.E., the user will hardly ever need to be concerned with these routines.

In general, the calling sequence of the routines are such that input quantities are given before output quantities and integers are before real variables which are before arrays. Until experience has been gained with the code, one is advised to use the routines supplied with the code except for coefficient specification routines.

IV.A. Coefficients Specification Routines

IV.A.1. BETA(NPTS,TIME,X,YARRAY,COEF)

This routine calculates NPTS coagulation coefficients for two particles of sizes X and YARRAY(I), I=1,...,NPTS, at time TIME. The coefficient for a particle of size X and that of size YARRAY(I) are stored in COEF(I). NPTS is an integer and TIME, X, YARRAY and COEF are real variables. YARRAY and COEF must be dimensioned to 1. COEF must be positive and symmetric with respect to the sizes of the coagulating particles. Although the coagulation coefficient is generally not a function of time, one has the option of using a time dependent coefficient. An improper coefficient will result in an error message as shown by the last two error messages in Example 1 of Section VII. In units consistent with OUTPUT, TIME is in seconds, X and YARRAY are in cubic centimeters and COEF is in cubic centimeters per second. Thus, conservation of particle volume is assumed for the problem.

As part of the input data discussed in Section V, the user may pass 10 integer values to any subroutine through the common block AERSL1 (which is discussed in Section VI). A listing of the BETA routine supplied with the code is given in Figure 1. Notice from Figure 1 that the third integer of the common block AERSL1 is used to determine which function will be used for the coagulation coefficient. For IPARM(3) not equal to 1 or 2, Fuch's (6) coefficient is used. For IPARM(3) equal to 1 or 2, an improper coagulation coefficient will be generated which will result in the error messages discussed above.

IV.A.2 BOUNDR(TIME)

If condensation is included in the continuous GDE, particles in the range about x_a are often assumed to be forming due to nucleation. The BOUNDR function enables the user to specify the source rate at $x = x_a$ as a function of time. Therefore, for non-zero values of BOUNDR, one is adding

$$\int_0^\infty \delta(x - x_a)\,\text{BOUNDR(TIME)}\; dx \qquad [13]$$

to the right hand side of Eq. [10] where δ is the Dirac delta function. BOUNDR should be a continuous non-negative function of time, and both BOUNDR and TIME are real variables. In units consistent with OUTPUT, BOUNDR is in cm^{-6} sec^{-1} and TIME is in seconds. A listing of the BOUNDR routine supplied with the code is given in Figure 2. Notice that the the common block AERSL6 (which is discussed in Section VI), is used to

access the output times, TOUT(I) (which are specified as part of the input data discussed in Section V), to modify BOUNDR past the second output time. This routine is used only if condensation is requested.

IV.A.3. CSOURC(NPTS,TIME,X,DISTX,SOURCE,REMOVE)

This routine calculates the continuous source function $S(x_i,t)$ and the continuous removal function $R[n(x_i,t),x_i,t]$ at x_i = X(I), i=I=1,...,NPTS and t = TIME for $n(x_i,t)$ = DISTX(I), i=I=1,...,NPTS. Since source and removal mechanisms are often represented as sums of distinct source or removal mechanisms respectively, the code has been developed such that the effect of each distinct mechanism on the first moment of the distribution can be determined. Therefore, the values of $S(x_i,t)$ and $R[n(x_i,t),x_i,t]$ are to be stored in the two-dimensional arrays SOURCE and REMOVE, respectively. The first subscript of SOURCE or REMOVE corresponds to the specific source or removal mechanism respectively and the second subscript corresponds to the points in the X array. For example, if the source term is due to two linearly independent mechanisms, then

$$S(x,t) = S_1(x,t) + S_2(x,t)$$

where $S_j(x_i,t)$ corresponds to the source at the i-th point in the X array for the j-th source mechanism. Therefore, $S_1(x_i,t)$ would be stored in SOURCE(1,I), I=1,...,NPTS and $S_2(x_i,t)$ would be stored in SOURCE(2,I), I=1,...,NPTS. Although the same results would be obtained if the sum of $S_1(x,t)$ and $S_2(x,t)$ are stored in SOURCE(1,I), the effect

of each mechanism could not be individually determined. Decomposing S and R will add a relatively small amount of computer time but is very useful for sensitivity studies of the components of S and R. However, if condensation is included in the GDE, the code will compute n(x,t) with all the components of S and R but will not determine the individual effects of the component of S and R on the first moment of the distribution. Therefore the user should not bother to decompose S or R if condensation is included.

SOURCE and REMOVE may be positive or negative and should be continuous functions of n(x,t), x and t. All variables are real except NPTS which is an integer. The code is currently dimensioned for a maximum of 3 source and 3 removal mechanisms, but can be changed as discussed in Section VI. If the user specifies more than 3 mechanisms for either the source of removal functions, an error message will be printed, as shown in Example 1 of Section VII. Therefore, SOURCE and REMOVE must each be dimensioned to (3,1). X and DISTX must each be dimensioned to 1. In units consistent with OUTPUT, X is in cm^3, DISTX is in cm^{-6} and SOURCE and REMOVE are in cm^{-6} sec^{-1}.

One may place the components of S(x,t) in the REMOVE array or the components of R[n(x,t),x,t] in the SOURCE array, without affecting the computed evolution of n(x,t). However, it is better to use the SOURCE array for functions which are independent of n(x,t) and the remove array for functions dependent of n(x,t). This grouping is generally clearer to

the user and more accurate for determining the effects of source mechanisms on the first moment of the the distribution. WARNING: Although the computed evolution of n(x,t) is not affected by placing functions dependent on n(x,t) in the SOURCE array, the computed effect on the first moment of n(x,t) (as determined in the routine TOTAL) will be wrong. A listing of the CSOURC routine supplied with code is given in Figure 3. A method for computing integrals as part of the REMOVE array is discussed in Sections IV.B.4 and IV.B.5.

IV.A.4. DSOURC(NPTS,TIME,CLSCON,SOURCE,REMOVE)

This routine is the discrete analogue of CSOURC. The routine calculates the discrete source function $\bar{S}_i(t)$ and the discrete removal function $\bar{R}_i(N_i,t)$ for i=1,...,NPTS. Since source and removal mechanisms are often represented as sums of distinct source or removal mechanisms respectively, the code has been developed such that the effect of each distinct mechanism on the first moment of the distribution can be determined. Therefore, the values of $\bar{S}_i(t)$ and $\bar{R}_i(N_i,t)$ are to be stored in the two-dimensional arrays SOURCE and REMOVE, respectively. The first subscript of SOURCE or REMOVE corresponds to the specific source or removal mechanism respectively and the second subscript corresponds to the cluster size, (i.e., 1 for monomer, 2 for dimer, etc.).

SOURCE and REMOVE should be continuous functions of N_i and t, and may be positive or negative. In general, SOURCE is positive and REMOVE is negative. WARNING: If the steady

state approximation is used for the discrete regime, each removal mechanism must be a proportional to N_i.

All variables are real except the number of discrete sizes NPTS, which is an integer. The code is currently dimensioned for a maximum of 3 source and 3 removal mechanisms but can be changed as discussed in Section VI. If the user specifies more than 3 mechanisms for either the source or removal functions, an error message will be generated. Therefore, SOURCE and REMOVE must each be dimensioned to (3,1). The cluster concentrations, which are stored in the third argument of the call list, must be dimensioned to 1. In units consistent with OUTPUT, TIME is in seconds, CLSCON is in cm^{-3}, and SOURCE and REMOVE are in cm^{-3} sec^{-1}.

One may place the components of $\bar{S}_i(t)$ in the REMOVE array or the components of $R_i(N_i,t)$ in the SOURCE array, without affecting the computed evolution of $n(x,t)$ or N_j, (j=1,...,k). However, as explained in Section IV.A.4, it is better to use the SOURCE array for functions which are independent of N_j, j=1,...,k and the REMOVE array for functions dependent on N_j. A listing of the DSOURC routine supplied with the code is given in Figure 4.

IV.A.5. EVAP(NPTS,TIME,X,COEF)

This routine calculates the evaporation coefficient $E(x_i)$, at x_i = X(I), i=I=1,...,NPTS at time TIME and stores the result in COEF(I), I=1,...,NPTS. NPTS is an integer and TIME, X and COEF are real variables. X and COEF must be dimensioned to 1 and COEF must be non-negative. A listing

of the EVAP routine supplied with the code is given in Figure 5. Notice from Figure 5 that the first integer of the common block AERSL1 (which is discussed in Section VI), is used to determine if evaporation is to be included in the simulation. Also notice that EVAP uses the BETA routine. The user should not forget that AEROSOL is a collection of FORTRAN subroutines and therefore any routine can call another routine as long as the rules of FORTRAN are followed. The user can also add subroutines as long as they do not conflict with existing subroutine names. This routine is used only if the discrete-continuous GDE is requested and the user has the option of using a time dependent evaporation coefficient. In units consistent with OUTPUT, TIME is in seconds, X is in cm^3 and COEF is in sec^{-1}.

IV.A.6. GROWTH(NPTS,TIME,X,RATE,PARTAL)

This subroutine calculates the growth rate $I(x_i,t)$, the partial of the growth rate with respect to particle size $\partial I(x_i,t)/\partial x$, at x_i, i=1,...,NPTS, and stores the result in RATE(I) and PARTAL(I), I=1,...,NPTS, respectively. NPTS is an integer, TIME, X, RATE and PARTAL are real variables and RATE must be positive. In units consistent with OUTPUT, TIME is in seconds, X is in cm^3, RATE is in cm^3 sec^{-1} and PARTAL is in sec^{-1}. The arrays X, RATE and PARTAL must be dimensioned to 1. A listing of the GROWTH routine supplied with the code is given in Figure 6. Notice from Figure 6 that the third integer in the common block AERSL1 (which is discussed in Section VI) is used to determine the functional form of RATE. For IPARM(3)

equal to 1, an illegal growth rate will be computed which will generate the seventh and eighth error messages given in Example 1 of Section VII. This routine is used only if condensation is requested.

IV.A.7. XINTL(NPTS,X,DISTX)

This routine calculates the initial distribution $n(x_i,0)$, at x_i=X(I) for i=I=1,...,NPTS. X and DISTX are real arrays which must be dimensioned to 1 and NPTS is an integer. X(I) are in ascending order and DISTX must be non-negative. In units consistent with OUTPUT, X is in cm^3 and DISTX is in cm^{-6}. A listing of the XINTL routine supplied with the code is given in Figure 7. Notice from Figure 7 that the second integer in the common block AERSL1 is used to determine the functional form of the initial distribution.

IV.B Interpolation, Integration and Output Routines

IV.B.1. DISTW(ISPLIN,NVARM1,NPTS,WX,ZDIST)

This routine determines the values of the "M" distribution given in Eq. [12], by interpolating from the values of the "M" distribution at the grid points. The array WX contains the "W" values given in Eq. [11] at which the values of the "M" distribution are to be computed and stored in the ZDIST array. ISPLIN is the user specified flag which determines the interpolation method, NVARM1 is the number of grid points minus 1 and NPTS is the number of points in the WX array. The values of the "M" distribution, the conserved quantity of the aerosol x, and the "W" values at the grid points are stored in the common block

AERSL2, respectively. A listing of the DISTW routine supplied with the code is given in Figure 8. Notice from Figure 8 that "A" is used to represent the value of the "M" distribution at the grid points. Note that the elements of WX are given in ascending order to minimize the computer time required to find the adjacent grid points.

For ISPLIN equal to 0, 1 or 2, cubic, linear or logarithmic splines will be used, respectively. Additional interpolation formulae may be incorporated or the user may wish to replace an existing formula. If an interpolation formula requires a preliminary calculation, the routine SETUP will be called prior to calling DISTW to perform preliminary calculations. For example, to obtain the second derivatives of the distribution at the grid points, cubic splines requires the solution to a system of linear algebraic equations with a tri-diagonal coefficient matrix. Therefore, as part of the input data, the user would specify that SETUP should be called prior to interpolation such that the second derivatives are computed for a cubic spline interpolation. The common block AERSL7 is reserved for passing variables from SETUP to DISTW and DERIVT. For cubic splines, the P array contains the second derivatives at the grid points and is passed to DISTW by the common block AERSL7. No preliminary calculations are required for linear or logarithmic splines and therefore SETUP need not be called for ISPLIN equal to 1 or 2. Since DISTW is called several times for the same values of the distribution at the grid points, all preliminary calculations should be performed in SETUP to eliminate redundant calculations.

IV.B.2. SETUP(ISPLIN,NEWSET,NVAR)

This routine performs any preliminary calculations needed for interpolating the value of the "M" distribution as discussed in Section IV.B.1. ISPLIN is the flag which specifies the interpolation formula, NEWSET is the flag which indicates if a new set of grid points has been specified since the last call to SETUP and NVAR is the number of grid points. As shown in Figure 9, the SETUP routine supplied with the code is used for determining the second derivatives of the distribution at the grid points for a cubic spline interpolation. A detailed discussion of this routine is contained in Section IV.B.1 and a listing of the SETUP routine supplied with the code is given in Figure 9.

IV.B.3. DERIVT(ISPLIN,NVAR,DERIV)

This routine calculates the derivatives of the "M" distribution with respect to "W" at NVAR grid points and stores the derivatives in the array DERIV. ISPLIN is the flag used to determine the spline interpolation formula, which must correspond to the appropriate expression used for the derivatives. For the DERIVT routine given in Figure 10, a cubic spline interpolation formula is assumed as discussed in Section IV.B.1. Notice that DERIV must be dimensioned to 1 and that the second derivatives at the grid points are passed through the common block AERSL7. This routine is used only if condensation is requested.

IV.B.4. BLOCK DATA

This data block stores the quadrature points and weight factors needed for an even point Gaussian-Legendre quadrature formula (3). WARNING: The number of quadrature points must be specified as an even number in this routine. Note that for an NQUADP point formula, only NQUADP/2 points and weight factors have to be stored. The routine supplied with the code uses a 24-point formula (i.e. NQUADP=24), but any other even point formula of lower order (i.e. NQUADP < 24) can be used by replacing the values of NQUADP, WQUAD and WTQUAD with the appropriate values given in (5). For the user's convenience, the numbers needed for a 12-point formula are supplied with the routine in the form of comment cards. Although to date a 24-point formula was more than adequate, the user may also use a higher order formula (i.e. NQUADP > 24) but array dimensions must be changed as given in Section VI. The numbers needed for a 40-point formula are also supplied in the form of comment cards. If a lower order formula is used (i.e. NQUADP < 24), one may reduce storage requirements by reducing array dimensions (as given in Section VI) but that is not necessary. In general, a high order quadrature formula is more accurate than a lower order formula but requires a greater computational effort. Therefore, unless the user has reason to believe that either the coagulation coefficient or the size distribution is such that greater accuracy is required to evaluate the coagulation integrals, it is easier and simpler to use the existing routine. Because the quadrature points

and weight factors are stored in the common block AERSL8, one can access these numbers in any routine for which integrals are to be numerically evaluated. However, if the integral is over the entire particle size domain, it is better to use the existing rescaled quadrature points and weight factors stored in the common block AERSL3, as discussed in Section IV.B.5.

IV.B.5. OUTPUT(ISTOP)

This routine prints the results of a simulation at the times specified by the user in the input data cards. ISTOP is an error flag with the following meaning,

0 = No error

1 = Error found, program will skip this case

-1 = Correctable error found, program will revert to previous output time

-2 = Print only discrete regime which has been determined by using the steady state approximation

Although n(x,0) is given by the routine XINTL, OUTPUT prints the number, surface area and volume distributions based on the logarithm of particle diameter, i.e. $n(x,t)dx/d(\log_{10}D)$, $\pi D^2 n(x,t)dx/d(\log_{10}D)$ and $(\pi D^3/6)n(x,t)dx/d(\log_{10}D)$, respectively where D is particle diameter. Since the quantities printed by the routine supplied with the code correspond to those of interest to the author, the user may wish to replace this routine for other applications. Recall that the values of the "M" distribution of "W" as given in Eqs. [11] and [12] are stored in the common block AERSL2. Therefore, if 25 grid

points are used, the following may be used as a simplified routine to print n(x,t).

```
      SUBROUTINE OUTPUT(ISTOP)
      DIMENSION ANSWR(25)
      COMMON/AERSL2/A(60),X(60),W(40)
      ALGXBA = ALOG(X(25)/X(1))
      DO 1 I = 1,25
    1 ANSWR(I) = A(I)/(X(I)*ALGXBA)
      WRITE(6,2)(I,X(I),ANSWR(I),I=1,25)
    2 FORMAT(I5,2E14.4)
      RETURN
      END
```

If 15 discrete sizes were used in the discrete regime, one would add the following line to print the cluster concentrations,

```
     WRITE(6,2)(I-25,X(I),A(I),I=26,40)
```

Since the simplified OUTPUT routine given above prints a complete set independent variables, virtually all properties of the aerosol size distribution can be determined from that set. In essence, the OUTPUT routine supplied with code is merely a much embellished form of the simplified routine given above. However, since the user may be interested in some of the features available in the existing OUTPUT routine, a discussion of these features will be given.

The number of grid points, the number of discrete sizes and the sum of these numbers are stored in the common block AERSL5 as NVAR, NDISCR and NVART, respectively. Therefore,

the limits of DO LOOPS can be set using these variables. ALGXBA as used in the simplified routine and the current output time TIME, are stored in the common block AERSL6 and therefore can also be easily accessed. The values of the "M" distribution at points other than at the grid points can be obtained by calling DISTW as discussed in Section IV.B.1. The spline interpolation flag, ISPLIN and the number of grid points minus 1, NVARM1 which are needed for the call are stored in the common block AERSL5. If a call to SETUP is required (as discussed in Sections IV.B.1 and IV.B.2), the call to SETUP will have already been made prior to executing OUTPUT and the user need not be concerned with calling SETUP. To compute integrals of the form

$$\int_{x_b}^{x_b} f(x)n(x,t)\,dx \qquad [14]$$

the following may be used to store the result in AREA.

```
      DIMENSION ZDIST(24)
      COMMON/AERSL3/Z(24),FACTOR(24),ZVOLM(24)
      CALL DISTW(ISPLIN,NVARM1,NQUADP,Z,ZDIST)
      AREA=0.
      DO 10 I=1,NQUADP
   10 AREA=AREA+F(ZVOLM(I))*FACTOR(I)*ZDIST(I)
```

where NQUADP is stored in the common block AERSL5 and is equal to the number of quadrature points and F represents the function f(x) given in Eq. [14].

By calling the routine TOTAL(TOTALQ) for ISTOP=0, the effect of source and removal mechanisms on the first moment of the distribution, i.e.

$$\int_{x_a}^{x_b} xn(x,t)\ dx + \sum_{i=1}^{NDISCR} x_i N_i \qquad [15]$$

can be obtained. The cumulative effect of each continuous source, continuous removal, discrete source and discrete removal mechanism on [15] are stored in the array TOTALQ, respectively. Negative values of TOTALQ indicate removal mechanisms and TOTALQ must be dimensioned in OUTPUT. The routine TOTAL may be deleted from the code if a new OUTPUT routine is used, since OUTPUT is the only routine which calls TOTAL.

IV.C. Convenience Routines

IV.C.1. CHECK(ISTOP)

This routine checks the user input for obvious errors and prints an error message if any errors are encountered. The routine will go through all error checks regardless of the number of errors and ISTOP will be set to 0 if no errors are found or to 1 if an error is found. If array dimensions are changed, the following variables in the data statement should be changed to the new dimensions

MAXVAR = maximum number of grid points

MAXDIS = maximum number of discrete sizes

MAXTIM = maximum number of output times

MAXSRC = maximum number of continuous source mechanisms

MAXRMV = maximum number of continuous removal mechanisms

MAXDSR = maximum number of discrete source mechanisms

MAXDRM = maximum number of discrete removal mechanisms

If the user is confident that no errors have been made in writing the coefficient specification routines or in the input data, this routine may be replaced with the following to reduce storage requirements for the code.

```
      SUBROUTINE CHECK(ISTOP)
      ISTOP = 0
      RETURN
      END
```

Error messages from CHECK are preceded by "---" as shown in examples 1 and 7 of Section VII. The user should be aware that not all possible errors are checked and it is possible to provide improper or inconsistent input which will not be detected by CHECK.

IV.C.2. TOTAL(TOTALQ)

This routine computes the effect of source and removal mechanisms on the first moment of the distribution. The methods used are discussed in the documentation of the routine in the form of comment cards. The procedure for using the routine is discussed in Section IV.B.5. WARNING: TOTAL should only be called if ISTOP=0 in the OUTPUT routine.

IV.C.3. CHKTBL(ISTOP)

This routine checks the interpolation for negative values of the distribution. If a negative value is found, the routine

will set ISTOP to -1, regress to the previous output time and add grid points at those locations where the interpolated distribution is negative. A detailed discussion of the routine is given in the form of comment cards as supplied with the code. Since no essential calculations are performed in this routine, to reduce storage it may be replaced with the following.

```
      SUBROUTINE CHKTBL(ISTOP)
      ISTOP = 0
      RETURN
      END
```

IV.C.4. STEADY(TIME)

This routine computes the steady state cluster concentrations in the discrete regime. WARNING: If there are removal mechanisms in the discrete regime for the i-th cluster, they must be proportional to N_i. If the steady state concentrations cannot be determined, an error message will be printed and execution will be terminated. This routine is required only if the steady state approximation is used.

IV.D. Central Routines

The following routines form the basic structure of the code and the user should seldom need to be concerned with these routines.

Name	Function
MAIN	controls program
INPUT	reads input data

COAGCF	stores values of coagulation and evaporation coefficients
DIFFUN	calculates derivatives needed for O.D.E. package

IV.E. Ordinary Differential Equation Package

Sandia's O.D.E. package is used to integrate the set of ordinary differential equations which govern N_i, (i=1,...,k) and n(x,t) at the grid points. A detailed discussion of the package is given in (3). WARNING: The machine unit round-off error must be reset in the routines DE and STEP1 if one is not using a Control Data Corporation computer with a 60-bit word length.

V. INPUT DATA

Part A of this Section discusses in detail some of the input data quantities. A complete but condensed reference list of the input data is given in Part B and examples are given in Section VII. For the user's convenience, all input data are printed in exactly the same order they are read. Therefore, after the user has gained some familiarity with the code, the condensed reference list may not be required since the first page of any run should be sufficient for modifying input data for future runs.

V.A. Detailed Discussion

Particle Size Domain

The particle size domain as determined by x_a and x_b should cover the domain of interest and be large enough to minimize "finite domain errors" (2). However, because of possible interpolation problems, m(w,t) as given in Eq. [12] should not vary over too many orders of magnitude within the domain. As a rough guide, at least 2 or 3 grid points may be required for each order of magnitude variation in m(w,t). If the distribution cannot be accurately interpolated without an excessive number of grid points, one should consider using a "sectional technique" (7), for which no interpolations are performed.

Error Tolerances

The relative and absolute error tolerances are discussed in detail elsewhere (3). As a rough guide, m(w,t) given in Eq. [12] is accurate to within m(w,t) times the relative error plus the absolute error. The cluster concentrations N_i, are

also computed to the same accuracy for both the transient and steady state calculations. Typical values of the relative error are 10^{-2} to 10^{-3} and 10^{-25} for the absolute error.

Number of Grid Points

The number of permissible grid points must be between 10 and 40, inclusive. The code will automatically increase the number of grid points to 10, if less than 10 are specified. If the user would like the code to automatically add grid points at the interpolated values for which the distribution is negative, the maximum number of grid points must be greater than the number of grid points. If the maximum is set less than the number of grid points, the code will automatically reset the maximum. Typically, 25-30 grid points are sufficient and 35-40 may be used for a maximum.

Number of Discrete Multiplets

The number of discrete multiplets must be between 2 and twice the number of discrete sizes minus 1. It should be set to the number of multiplets which are to be read in the fourth card(s)/case as discussed in Section V.B. Since the distribution is assumed to be continuous throughout the continuous regime, not all possible multiplets have to be specified.

Number of Outputs

This is the number of output times for which the program is to call OUTPUT to print the distribution. It should correspond to the number of output times given in the third card(s)/case as discussed in Section V.B.

Steady State Approximation

WARNING: If the steady state approximation is used, the discrete removal mechanism for the i-th cluster must be proportional to the concentration of the i-th cluster.

Grid Point Selection

Unless the user has some reason to specify grid point locations, it is advisable to let the code select the grid points.

Number of Interpolating Points

This is the number of points between adjacent grid points for which the distribution is to be checked. If no interpolating points are specified, the code will not add grid points if the distribution becomes negative.

Interpolation Method

As discussed in Section IV.B.1, the code currently can use cubic, linear or logarithmic splines (i.e. spline numbers of 0, 1 and 2, respectively). Cubic splines were generally found to be the most accurate and it is the only method of the ones supplied with the code which requires a call to SETUP. Linear splines requires the least computational effort and is less likely to cause interpolation problems. Although the author has found cubic splines to be the best of the available methods, one may decide to use another method or change the routine DISTW for other applications. One may also switch interpolation methods during a simulation, as shown in Example 9 of Section VII.

V. B. Condensed Reference List

```
C*-------------------------------------------------------------------
C*
C*    NOTE: IN THE WRITEUP THE I-TH CLUSTER MEANS A PARTICLE IN THE
C*          DISCRETE REGIME CONTAINING I MONOMERS.  THE I-TH MULTIPLET
C*          IS THE I-TH GRID POINT, WHICH IS SPECIFIED BY THE NUMBER OF
C*          MONOMERS CONTAINED IN THE PARTICLE.
C*
C*    R.J.=RIGHT JUSTIFIED INTEGER
C*    GDE=GENERAL DYNAMIC EQUATION
C*    DC GDE=DISCRETE-CONTINUOUS GENERAL DYNAMIC EQUATION
C*
C*                    FIRST CARD OF INPUT
C*
C*    IN COLUMNS 1-2 THE NUMBER OF CASES TO BE COMPUTED     R.J.
C*    IF A BLANK CARD IS USED ONE CASE WILL BE ASSUMED.
C*
C*
C*    FOR EACH CASE THE FOLLOWING SET OF CARDS MUST FOLLOW.  UNLESS
C*    SPECIFIED AS RIGHT JUSTIFIED (R.J.) INTEGER, ALL INPUT DATA IS TO
C*    BE IN REAL FORMAT.
C*
C*                    FIRST CARD/CASE
C*
C*    COLUMNS
C*     1-40     TITLE TO BE PRINTED AS A HEADING FOR THIS CASE
C*    41-50     SMALLEST PARTICLE SIZE  (IF THE DC GDE IS USED THIS IS
C                 THE MONOMER SIZE. THE PROGRAM AUTOMATICALLY DETERMINES
C*                THE SMALLEST SIZE IN THE CONTINUOUS REGIME)
C*    51-60     LARGEST PARTICLE SIZE
C*    61-70     RELATIVE INTEGRATION ERROR
C*    71-80     ABSOLUTE INTEGRATION ERROR
C*
C*                    SECOND CARD/CASE
C*
C*     1- 2     NUMBER OF GRID POINTS   (10-40)                R.J.
C*     3- 4     MAXIMUM NUMBER OF GRID POINTS  (10-40)         R.J.
C*     5- 6     NUMBER OF DISCRETE SIZES (0,2-20)              R.J.
C*                (IF SET TO ZERO THE CONTINUOUS GDE IS ASSUMED)
C*     7- 8     NUMBER OF DISCRETE MULTIPLETS IN THE CONTINUOUS REGIME
C*                (0,2-19)                                     R.J.
C*                (MUST BE SET TO ZERO FOR THE CONTINUOUS GDE)
C*     9-10     NUMBER OF OUTPUTS (1-16)                       R.J.
C*    11-12     THE OUTPUT TIME NUMBER WHEN THE PROGRAM IS TO SWITCH
C*              INTERPOLATION METHODS.  FOR EXAMPLE, IF 5 OUTPUT TIMES
C*              ARE GIVEN AS 0.0,60.,300.,600., AND 1200. SECONDS, THEN
C*              A 2 WOULD RESULT IN THE PROGRAM SWITCHING FROM THE
C*              INITIALLY SPECIFIED INTERPOLATION METHOD TO THE OTHER
C*              ALTERNATIVE IMMEDIATELY AFTER 60. SECONDS OF SIMULATION.
C*              A NUMBER LESS THAN OR EQUAL TO 1 OR GREATER THAN THE
C*              NUMBER OF OUTPUTS WOULD RESULT IN NO CHANGE OF
```

```
C*              INTERPOLATING METHODS.                                R.J.
C*    13-14     THE OUTPUT TIME NUMBER WHEN THE PROGRAM IS TO
C*              SWITCH TO THE STEADY STATE APPROXIMATION FOR THE
C*              DISCRETE REGIME.  IF SET TO -1 THE PROGRAM WILL PRINT
C*              THE STEADY STATE CONCENTRATIONS IN ADDITION TO THE
C*              TRANSIENT CONCENTRATIONS.  IF SET TO -2 THE PROGRAM
C*              WILL SWITCH TO THE STEADY STATE APPROXIMATION WHEN
C*              THE SPECIFIED ERROR TOLERANCES ARE MET.  (SET TO
C*              ZERO FOR THE CONTINUOUS GDE)                          R.J.
C*    15-20           *** BLANK ***
C*    21-23     GRID POINT SELECTION:
C*                0=PROGRAM WILL SELECT LOGARITHMICALLY SPACED GRID
C*                  IN PARTICLE SIZE
C*                FOR VALUES GREATER THAN OR EQUAL TO ONE, THIS IS THE
C*                  NUMBER OF GRID POINTS TO BE READ IN ASCENDING ORDER.
C*                  FOR NEGATIVE VALUES, READ IN DESCENDING ORDER THE
C*                  ABSOLUTE VALUE NUMBER OF GRID POINTS. THE PROGRAM
C*                  WILL SELECT THE REMAINING GRID POINTS AS
C*                  LOGARITHMICALLY SPACED IN PARTICLE SIZE.  R.J.
C*    24-30           *** BLANK ***
C*       31     NUMBER OF INTERPOLATING POINTS BETWEEN GRID POINTS
C*       32     NUMBER OF CONTINUOUS SOURCE TERMS  (0-3)
C*       33     NUMBER OF CONTINUOUS REMOVAL TERMS (0-3)
C*       34     NUMBER OF DISCRETE SOURCE TERMS    (0-3)
C*       35     NUMBER OF DISCRETE REMOVAL TERMS   (0-3)
C*       36     0=NO CONDENSATION, 1=WITH CONDENSATION
C*       37     INITIAL SPLINE INTERPOLATION METHOD NUMBER
C*                0=CUBIC SPLINES
C*                1=LINEAR SPLINES
C*                2=LOGARITHMIC SPLINES (POWER LAW FIT)
C*       38     CALL SETUP BEFORE INTERPOLATING  (0=NO/1=YES)
C*       39     SPLINE INTERPOLATION METHOD NUMBER FOR SWITCHED METHOD
C*       40     CALL SETUP FOR SWITCHED METHOD  (0=NO/1=YES)
C*       41     TIME DEPENDENT COAGULATION AND EVAPORATION
C*                COEFFICIENTS (0=NO/1=YES)
C*    42-50           *** BLANK ***
C*    51-60     USER DEFINED PARAMETER SWITCHES
C*
C*                    THIRD CARD(S)/CASE
C*
C*    IN FORMAT 8E10.4 THE TIMES FOR WHICH OUTPUT IS DESIRED IN
C*    ASCENDING ORDER. IF THE FIRST OUTPUT TIME IS GREATER
C*    THAN ZERO THE SEVENTH CARD(S)/CASE MUST BE INCLUDED.  THE NUMBER
C*    OF OUTPUT TIMES IS SPECIFIED IN COLUMNS 9-10 OF THE SECOND
C*    CARD/CASE
C*
C*                    FOURTH CARD(S)/CASE
C*
C*    IN FORMAT 16I5 THE DISCRETE MULTIPLET NUMBERS IN ASCENDING
C*    ORDER.  THE FIRST AND LAST GRID POINTS ARE AUTOMATICALLY FIXED AT
```

```
C*    THE SMALLEST AND LARGEST PARTICLE SIZE RESPECTIVELY, IN THE
C*    CONTINUOUS REGIME.  THEREFORE THE FIRST MULTIPLET NUMBER GIVEN
C*    MUST BE GREATER THAN OR EQUAL TO THE NUMBER OF DISCRETE SIZES PLUS
C*    TWO. THE MULTIPLET NUMBERS MUST RANGE FROM THE NUMBER OF DISCRETE
C*    SIZES PLUS TWO TO TWICE THE NUMBER OF DISCRETE SIZES. THE LAST
C*    MULTIPLET NUMBER MUST BE EQUAL TO TWICE THE NUMBER OF DISCRETE
C*    SIZES. THE NUMBER OF MULTIPLETS GIVEN SHOULD CORRESPOND TO
C*    COLUMNS 7-8 IN THE SECOND CARD/CASE
C*    (IF NO DISCRETE SIZES ARE USED DO NOT PUT IN ANY CARD)
C*
C*                    FIFTH CARD(S)/CASE
C*
C*    IN FORMAT 8E10.4 THE INITIAL NUMBER CONCENTRATION IN THE
C*    DISCRETE REGIME IN ASCENDING PARTICLE SIZE ORDER. IF THE
C*    STEADY STATE APPROXIMATION IS REQUESTED FOR THE FIRST
C*    OUTPUT TIME (I.E. COLUMNS 13-14 IN THE SECOND CARD/CASE
C*    IS 01), THEN THESE VALUES WILL BE USED AS THE INITIAL
C*    GUESS FOR DETERMINING THE STEADY STATE CONCENTRATIONS
C*    (IF NO DISCRETE SIZES ARE USED DO NOT PUT IN ANY CARD)
C*
C*                    SIXTH CARD(S)/CASE
C*
C*    IN FORMAT 8E10.4 THE PARTICLE SIZES OF THE GRID POINTS IN
C*    ASCENDING OR DESCENDING ORDER AS SPECIFIED IN CARD COLUMNS 21-23
C*    IN THE SECOND CARD/CASE.
C*    THE NUMBER OF GRID POINTS GIVEN MUST BE LESS THAN THE NUMBER OF
C*    GRID POINTS MINUS THE NUMBER OF MULTIPLETS GIVEN MINUS TWO.
C*    FOR THE CONTINUOUS GDE THE FIRST GRID POINT IS AUTOMATICALLY SET
C*    AT THE SMALLEST PARTICLE SIZE.  THEREFORE GRID POINTS ARE READ
C*    IN STARTING WITH THE SECOND ACTUAL GRID POINT. FOR THE DISCRETE -
C*    CONTINUOUS GDE GRID POINTS ARE READ IN STARTING AFTER THE LAST
C*    MULTIPLET SIZE AND SHOULD THEREFORE HAVE A SIZE GREATER THAN
C*    THE MONOMER SIZE TIMES TWICE THE NUMBER OF DISCRETE SIZES. IF
C*    GRID POINTS ARE READ IN DECSENDING ORDER THE FIRST GRID POINT
C*    SIZE READ IN MUST BE LESS THAN THE LARGEST SIZE SPECIFIED IN
C*    COLUMNS 51-60 OF THE FIRST CARD/CASE.
C*    (NOT TO BE USED IF LOGARITHMICALLY SPACED GRID POINTS ARE
C*     REQUESTED, I.E. COLUMNS 21-23 IN THE SECOND CARD/CASE IS ZERO)
C*
C*                    SEVENTH CARD(S)/CASE
C*
C*    IN FORMAT 8E10.4 THE DISTRIBUTION VALUES FOR INITIAL TIMES
C*    GREATER THAN ZERO.  EVEN IF GRID POINTS ARE READ IN
C*    DESCENDING ORDER, THE DISTRIBUTION VALUES ARE READ IN
C*    ASCENDING ORDER. THIS CARD(S) IS REQUIRED ONLY IF THE FIRST
C*    OUTPUT TIME IS GREATER THAN ZERO
C*
C*    TO TRANSFER PARAMETER VALUES TO THE ROUTINES
C*    INSERT THE FOLLOWING: COMMON/AERSL1/IPARM(10)
```

VI. COMMON BLOCKS AND ARRAY STORAGE

Array dimensions can be determined from the following values. WARNING: NCSORC must equal NDSORC and NCREMV must equal NDREMV for dimensioning arrays.

NCREMV = number of continuous removals

NCSORC = number of continuous sources

NDISCR = number of discrete sizes

NDREMV = number of discrete removals

NDSORC = number of discrete sources

NMULP1 = number of multiplets (equal to the number of multiplets supplied by the user plus 1)

NQUADP = number of quadrature points

NVAR = number of grid points

If a calculated dimension is zero, the dimension should be set to one. The quantities represented by the variable are given in the code in the form of comment cards.

/AERSL1/IPARM(10)

IPARM stores the ten user-defined switches which are read in as input data.

/AERSL2/A(NVAR+NDISCR),X(NVAR+NDISCR),W(NVAR)

The values of the "M" distribution at the grid points are stored in A(I),I=1,...,NVAR and the cluster concentrations are stored in A(I+NVAR),I=1,...,NDISCR. The particle sizes at the grid points and for the cluster sizes are stored in X. W(I), I=1,...,NVAR contains the "W" values at the grid points.

/AERSL3/Z(NQUADP),FACTOR(NQUADP),ZVOLM(NQUADP)

Z(I),FACTOR(I) and ZVOLM(I) are the "W" value, quadrative weight factor and particle size, respectively, at the I-th quadrature point. Using these values, integrals over the entire particle size domain may be computed as shown in Section IV.B.5.

/AERSL4/BESR(NDISCR,NVAR or 1 if NDISCR=0),
BETADC(NDISCR,NVAR or 1 if NDISCR=0),
BETDSR((NDISCR*(NDISCR+1))/2),
EVPPLS(NVAR or 1 if NDISCR=0),
EVPSTR(NVAR+NDISCR or 1 if NDISCR=0),
FACT1A(NQUADP,NVAR),FACT1B(NQUADP,NVAR),
FACT2(NQUADP,NVAR+NDISCR),FACTE1(NQUADP),
FACTE3(4),MAXGDC(NVAR or 1 if NDISCR=0),
WD(NDISCR,NVAR or 1 if NDISCR=0),
WMZ1(NQUADP,NVAR),WPLS(NVAR or 1 if NDISCR=0),
Z1(NQUADP,NVAR),Z2(NQUADP,NVAR),ZE(4)

/AERSL5/...NDCLS(NDISCR)...

/AERSL6/TOUT(16),TIME,XA,XB,XBOXA,ALGXBA,RELERR,ABSERR

TOUT(I) = the I-th output time
TIME = current time

/AERSL7/P(NVAR-2)

See Sections IV.B.1. and IV.B.2.

/AERSL8/WQUAD(NQUADP/2),WTQUAD(NQUADP/2)

See Sections IV.B.4 and IV.B.5.

ROUTINE	DIMENSIONS
MAIN	WORK(21*(NDISCR+NVAR)+100),IWORK(5), ASAVE(NDISCR)
OUTPUT	TOTALQ(NCREMV+NCSORC+NDREMV+NDSORC), ZDIST(NQUADP)
CHKTBL	ASAVE(NVAR+NDISCR),XTROUB(NVAR+ NDISCR),WTROUB(NVAR),WBET(9),ZDIST(9)
TOTAL	TOTALQ(1),SUM2(MAX0(NCSORC,NCREMV +NDREMV)),AVRGX(NCSORC),SOURCE(NCSORC, NQUADP),REMOVE(NCREMV,NQUADP), ZDIST(NQUADP),TFACT(4),ZTIME(4), AVRGR(NCREMV+NDREMV)
DIFFUN	ADIF(NVAR+NDISCR),DADT(NVAR+NDISCR) ZDIST(MAX0(NVAR,NQUADP)), SOURCE(NCSORC,MAX0(NVAR,NQUADP)), REMOVE(NCREMV,MAX0(NVAR,NQUADP)), ZDIST2(MAX0(NVAR,NQUADP))
STEADY	SCAVNG(NDISCR),ATRY(NDISCR), STRSRC(NDISCR),SOURCE(NDSORC,NDISCR), REMOVE(NDREMV,NDISCR),ZDIST(NQUADP)
SETUP	G(NVAR-2),DIAG(NVAR-2), SUB(NVAR-2),SUPER(NVAR-2)

```
CSOURC          SOURCE(NCSORC,1),REMOVE(NCREMV,1)

DSOURC          SOURCE(NDSORC,1),REMOVE(NDREMV,1)
```

VII. EXAMPLES

To demonstrate many of the features of the code, nine examples arranged in order of increasing complexity are given in this section. The user is advised to review the example problems before proceeding with his own problems. The printed output is determined by the OUTPUT routine discussed in Section IV.B.5.

VII.A. Discussion

Example No. 1: Input Errors

This example demonstrates some of the checking capabilities of the code and the error messages should be self-explanatory. Notice that although 7 and 6 were specified as the number of grid points and the maximum number of grid points respectively, the code automatically uses 10 grid points if less than 10 points are specified. Since no discrete sizes were specified, the code automatically forces the following to zero, 1) the number multiplets given, 2) the number of discrete sources, 3) the number of discrete removals and 4) the output time number to use the steady state approximation. Since the user may add interpolation formulae, no check is made on the spline number used to specify the interpolation formula. Therefore, although 0, 1 and 2 are the only values of the spline numbers for which an interpolation method is supplied with the code, no error message was generated for a specified spline number of 5 for the switched spline interpolation method.

If no errors are encountered by the routine CHECK, "NO DATA INPUT ERRORS FOUND" will be printed, as shown in examples 2, 3, 4, 5, 6, 8 and 9.

Example No. 2: Simple Coagulation

Notice that since coagulation is the only mechanism affecting the size distribution, total aerosol volume is conserved. Because cubic splines require a preliminary calculation, SETUP is called before interpolating.

Example No. 3: Coagulation and Sources

If no interpolating points are specified, no checks are made for negative values of the distribution. The cumulative volume of aerosol added by each mechanism at each output time may be used to monitor "finite domain errors"(2). Notice that the initial aerosol volume plus that added, is equal to the aerosol volume after ten minutes of simulation.

Example No. 4: Add Grid Points

The first and last grid points are specified by the smallest and largest particle size, respectively. Therefore, the number of grid point locations the user may specify is equal to the number of grid points minus 2. The cumulative aerosol volume removed is listed as a negative value after the cumulative aerosol volume added by source mechanisms.