LA-11560-MS

KIVA-II: A Computer Program for Chemically Reactive Flows with Sprays

Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

Prepared by Adrienne Rosen, Group T-3

This work was supported by the US Department of Energy, Division of Energy Conservation and Utilization Technologies.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither The Regents of the University of California, the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by The Regents of the University of California, the United States Government, or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of The Regents of the University of California, the United States Government, or any agency thereof. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

LA-11560-MS

UC-96
Issued: May 1989

KIVA-II: A Computer Program for Chemically Reactive Flows with Sprays

A. A. Amsden
P. J. O'Rourke
T. D. Butler

Los Alamos
NATIONAL LABORATORY

Los Alamos, New Mexico 87545

TABLE OF CONTENTS

KIVA-II: A COMPUTER PROGRAM FOR CHEMICALLY REACTIVE FLOWS WITH SPRAYS

by

A. A. Amsden, P. J. O'Rourke, and T. D. Butler

ABSTRACT

This report documents the KIVA-II computer program for the numerical calculation of transient, two- and three-dimensional, chemically reactive fluid flows with sprays. KIVA-II extends and enhances the earlier KIVA code, improving its computational accuracy and efficiency and its ease-of-use. The KIVA-II equations and numerical solution procedure are very general and can be applied to laminar or turbulent flows, subsonic or supersonic flows, and single-phase or dispersed two-phase flows. Arbitrary numbers of species and chemical reactions are allowed. A stochastic particle method is used to calculate evaporating liquid sprays, including the effects of droplet collisions and aerodynamic breakups. Although the initial and boundary conditions and mesh generation have been written for internal combustion engine calculations, the logic for these specifications can be easily modified for a variety of other applications. Following an overview of the principal features of the KIVA-II program, we describe in detail the equations solved, the numerical solution procedure, and the structure of the computer program. Sixteen appendices provide additional details concerning the numerical solution procedure.

I. INTRODUCTION AND BACKGROUND

The in-cylinder dynamics of advanced internal combustion engines, such as the direct-injection stratified-charge (DISC) engine, involve a number of complex, closely coupled physical and chemical processes. These include the transient three-dimensional dynamics of evaporating fuel sprays interacting with flowing multicomponent gases undergoing mixing, ignition, chemical reactions, and heat transfer. The KIVA code[1-3] has the ability to calculate such flows in engine cylinders with arbitrarily shaped piston geometries, including the effects of turbulence and wall heat transfer. In response to the needs of a large user community and to recent developments in the fields of numerical fluid dynamics and internal combustion engine modeling, we have implemented many improvements to KIVA since its public release in 1985. The changes are incorporated in a new version of the code, called KIVA-II, that is documented in this report. KIVA-II builds on the capabilities of KIVA and is quite similar in structure. Current users of KIVA will find the transition to KIVA-II to be straightforward.

An excerpt from Ref. 1 explains the basis under which KIVA was written: "Since KIVA was developed with applications to internal combustion engines in mind, it contains several features designed to facilitate such applications. However, the basic code structure is modular and quite general, and most of the major options (chemical reactions, sprays, etc.) can be individually activated or deactivated by setting appropriate values for the associated input switches. The code is therefore applicable to a wide variety of multidimensional problems in fluid dynamics, with or without chemical reactions or sprays." Indeed, KIVA has been used for numerous studies besides internal combustion engines, including cold flow analyses in complicated geometries, continuous spray combustors, Bunsen burner flames, nonreacting sprays, and hydrogen-oxygen flames propagating in long tubes, to name just a few. It is impractical to cite all such studies here because of the widespread distribution and use of the code in industry and universities. For internal combustion engines, besides the studies of the DISC engine that have been carried on at General Motors Research Laboratories, Princeton University, and Los Alamos National Laboratory,[4-7] it has been used as the basis for numerical investigations of diesel engines[8-10] and of coal-fired diesels[11] as well.

From a historical perspective, KIVA-II is the latest in a series of multidimensional codes that we have produced since we began work on numerical simulations of internal combustion engines 12 years ago, under the sponsorship of what has become the Department of Energy's Energy Conversion and Utilization Technologies (ECUT) program. All of them are multidimensional finite-difference codes that solve the transient equations of motion. The first of these was the RICE code.[12] RICE was a two-dimensional Eulerian code that utilized rectangular computing zones for its mesh, eddy diffusivity to model the turbulence, Arrhenius kinetics with an arbitrary number of reactions and species to represent the chemical kinetics, and a partially implicit finite difference formulation to efficiently treat the acoustic terms for low Mach number flows. Bracco et al. at Princeton modified RICE and produced the REC code,[13] which included the effect of piston motion in the unresolved third dimension of the calculations. Another two-dimensional Eulerian code, APACHE,[14] followed RICE. This had the capabilities of RICE and the generality of arbitrarily shaped cells. CONCHAS[15] followed APACHE, and it likewise utilized arbitrarily shaped cells but offered the feature of an arbitrary Lagrangian-Eulerian formulation that allowed the computing zones to follow the piston motion. In addition the turbulence effects were included in the calculations by use of a subgrid scale model. CONCHAS-SPRAY[16] replaced CONCHAS. As its name implies, it included a model for the spray dynamics, a statistical representation that accounted for a spectrum of droplet sizes and the effects of evaporation. The turbulence was calculated by means of a subgrid scale model that used a transport equation for turbulent kinetic energy and a law-of-the-wall treatment for turbulent boundary layers. The chemistry was generalized to include both kinetic and equilibrium reactions. KIVA[1,2] then followed. In addition to retaining the capabilities of CONCHAS-SPRAY, it featured the ability to do either two- or three-dimensional problems with the same code. Furthermore, it had an expanded spray model that treated collisions and coalescence. An acoustic subcycling method was adopted to permit the efficient computation of low-Mach number flows.

Table I gives the ways in which KIVA-II differs from KIVA. These fall into four general categories: computational efficiency improvements, numerical accuracy improvements, new or improved physical submodels, and improvements in ease-of-use and versatility.

While some familiarity with KIVA[1-3] or the CONCHAS-SPRAY code[16] would be helpful, it is not necessary for understanding this report or using KIVA-II. This report is intended to fully document the KIVA-II code. Accordingly we now summarize the equations solved, the numerical solution procedure, and some special features designed to facilitate internal combustion engine applications.

KIVA-II solves the unsteady equations of motion of a turbulent, chemically reactive mixture of ideal gases, coupled to the equations for a single-component vaporizing fuel spray. The gas-phase solution procedure is based on a finite volume method called the ALE (arbitrary Lagrangian-Eulerian) method.[17,18] Spatial differences are formed on a finite-difference mesh that subdivides the computational region into a number of small cells that are hexahedrons. The corners of the cells are called vertices, and the positions of the vertices may be arbitrarily specified functions of time, thereby allowing a Lagrangian, Eulerian, or mixed description. The arbitrary mesh can conform to curved boundaries and can move to follow changes in combustion chamber geometry. A strength of the method is that the mesh need not be orthogonal. The spatial differencing is made conservative wherever possible. The procedure used is to difference the basic equations in integral form, with the volume of a typical cell used as the control volume, and with divergence terms transformed to surface integrals using the divergence theorem.[17]

The Cartesian components of the velocity vector are stored at cell vertices, and the momentum equations are differenced in a strictly conservative fashion. In contrast to the original ALE method,[17,18] however, cell-faced velocities are used during a portion of the computational cycle.[2] Their use greatly reduces the tendency of the ALE method to parasitic velocity modes, thereby largely eliminating the need for node coupler.

The transient solution is marched out in a sequence of finite time increments called cycles or timesteps. On each cycle the values of the dependent variables are calculated

TABLE I

KIVA-II FEATURES

1. Computational Efficiency Improvements
   - Coupled, implicit differencing of diffusion terms and terms associated with pressure wave propagation
   - Subcycled calculation of convection
   - Stochastic spray particle injector
   - 2-D to 3-D converter
2. Numerical Accuracy Improvements
   - Optional quasi-second-order upwind convection scheme
   - Generalized mesh diffusion algorithm
   - Method for computing turbulent droplet dispersion when Δt exceeds turbulent correlation time
   - Convection of length scale in place of the turbulence dissipation rate ε
3. New or Improved Physical Submodels
   - k–ε turbulence model
   - Model for droplet aerodynamic breakup
4. Improvements in Ease-of-Use and Versatility
   - Nonflat cylinder head option
   - Inflow/outflow boundaries
   - Simplified velocity boundary conditions
   - Alphabetized epilogue listing FORTRAN variables and their definitions
   - Gravitational terms
   - Eulerian and Lagrangian options
   - Library of thermophysical properties of common hydrocarbons
   - Initial Bessel function swirl profile
   - Optional tabular input of spray injection velocity

from those on the previous cycle. As in the original ALE method,[17,18] each cycle is divided into two phases – a Lagrangian phase and a rezone phase. In the Lagrangian phase the vertices move with the fluid velocity, and there is no convection across cell boundaries. In the rezone phase, the flow field is frozen, the vertices are moved to new user-specified positions, and the flow field is remapped or rezoned onto the new computational mesh. This remapping is accomplished by convecting material across the boundaries of the computational cells, which are regarded as moving relative to the flow field.

In contrast to KIVA, the temporal difference scheme in KIVA-II is largely implicit. Because of this, the timesteps used by KIVA-II are calculated based on accuracy, not stability, criteria and can be considerably larger than the timesteps used by KIVA. This has resulted in considerable savings of computational time in many problems. In the Lagrangian phase, implicit differencing is used for all the diffusion terms and the terms associated with pressure wave propagation. The coupled implicit equations are solved by a method similar to the SIMPLE[19] algorithm, with individual equations being solved by the conjugate residual method.[20]

Explicit methods are used to calculate convection in the rezone phase; but the convection calculation can be subcycled an arbitrary number of times, and thus the main computational timestep is not restricted by the Courant stability condition of explicit methods.[21] The convection timestep is a submultiple of the main computational timestep and does satisfy the Courant condition. In addition to the partial donor cell differencing in KIVA,[1] KIVA-II can use a quasi-second-order upwind (QSOU) scheme for convection. Based on the ideas of van Leer,[22] this scheme is monotone and approaches second-order accuracy when convecting smooth profiles. While more accurate than partial donor cell differencing, QSOU is also more time-consuming, and thus it is included as an option.

The number of species and chemical reactions that can be accounted for in KIVA-II are arbitrary; they are limited only by computer time and storage considerations. The code distinguishes between slow reactions, which proceed kinetically, and fast reactions, which are assumed to be in equilibrium.[23] Chemical rate expressions for the kinetic reactions, which are Arrhenius in form, are evaluated by a partially implicit procedure. Two implicit equation solvers are available to compute chemical equilibria-- a fast algebraic solver for hydrocarbon/air combustion[24] and an iterative solver for more general circumstances.[25]

Two models are available to represent the effects of turbulence. The user has the option to use a standard version of the $k-\varepsilon$ turbulence model,[26] modified to include volumetric expansion effects[27] and spray/turbulence interactions,[4] or to use a modified version of the subgrid scale (SGS) turbulence model of KIVA.[1] The SGS model reduces to the $k-\varepsilon$ model near walls where all turbulence length scales are too small to be resolved by the computational mesh. Boundary layer drag and wall heat transfer are calculated by matching to a modified turbulent law of the wall. KIVA-II does not have a model for the effects of turbulence on the mean chemical reaction rates, but the user can easily modify the code to include a mixing-controlled chemistry model.[28-31]

Evaporating liquid sprays are represented by a discrete-particle technique,[32] in which each computational particle represents a number of droplets of identical size, velocity, and temperature. Probability distributions often govern the assignment of droplet properties at injection or the changes in drop properties at downstream locations. When this is the case, droplet properties are determined by using a Monte Carlo sampling technique. The particles and fluid interact by exchanging mass, momentum, and energy. The momentum exchange is treated by implicit coupling procedures to avoid the prohibitively small timesteps that would otherwise be necessary. Accurate calculation of mass and energy exchange is ensured by automatic reductions in the timestep when the exchange rates become large. Turbulence effects on the droplets are accounted for in one of two ways. When the timestep is smaller than the droplet turbulence correlation time, a fluctuating component is added to the local mean gas velocity when calculating each particle's mass, momentum, and energy exchange with the gas.[32] When the timestep exceeds the turbulence correlation time, turbulent changes in droplet position and velocity are chosen randomly from analytically derived probability distributions for these changes.[33] Droplet collisions and coalescences are accounted for,[34] and a new model for droplet aerodynamic breakup has been installed.[35] Volume displacement[32] and thick spray effects on the exchange rates[34] are neglected.

Because of improvements to the code's ease-of-use and versatility, for many applications, all required geometrical specifications, initial conditions, and boundary conditions may be specified using the standard input alone. This is particularly true for internal combustion engine applications. The mesh generation logic allows the computational region to include cupped pistons and domed cylinder heads and to offset these relative to the axis of the cylinder. In addition to two- and three-dimensional Cartesian and cylindrical meshes, the code allows the calculation of the flow in a single "sector" of certain three-dimensional cylindrical configurations in which there is an n-fold symmetry about the axis of the cylinder. This symmetry is often found in engine cylinders with multihole injectors. For initial conditions, one can specify an axisymmetric swirl-velocity field with a Bessel function profile and a specified swirl ratio. Standard boundary conditions and rezone logic allow the mesh to follow the motion of a piston.

In response to many users outside the automotive engine design community, a number of other features have been incorporated in KIVA-II. These include gravitational terms, the options to calculate with purely Eulerian or Lagrangian meshes, and inflow and outflow boundaries. The latter are included only for the special case of inflow at the bottom and outflow at the right or top of the mesh, but it is hoped that using these as examples the user can easily modify the code for other inflow/outflow conditions. The user is aided in the task of code modification by the modular structure of the program and a new alphabetized epilogue that allows one to easily find the definitions and uses of FORTRAN variables within the code.

II. THE GOVERNING EQUATIONS

In this section we give the equations of motion for the fluid phase, followed by those for the spray droplets, and finally the boundary conditions. For compactness these are written in vector notation with bold symbols representing vector and tensor quantities. The unit vectors in the x-, y-, and z-directions are denoted by **i**, **j**, and **k** respectively. The position vector **x** is defined by

$$\mathbf{x} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \,,$$

the vector operator ∇ is given by

$$\nabla = \mathbf{i}\frac{\partial}{\partial x} + \mathbf{j}\frac{\partial}{\partial y} + \mathbf{k}\frac{\partial}{\partial z} \,,$$

and the fluid velocity vector **u** is given by

$$\mathbf{u} = u(x,y,z,t)\mathbf{i} + v(x,y,z,t)\mathbf{j} + w(x,y,z,t)\mathbf{k} \,,$$

where t is time.

A. The Fluid Phase

The KIVA-II equations can be used to solve for both laminar and turbulent flows. The mass, momentum, and energy equations for the two cases differ primarily in the form and magnitude of the transport coefficients (i.e., viscosity, thermal conductivity, and species diffusivity), which are much larger in the turbulent case because of the additional transport caused by turbulent fluctuations. In the turbulent case the transport coefficients are derived from a turbulent diffusivity that depends on the turbulent kinetic energy and its dissipation rate.

The continuity equation for species m is

$$\frac{\partial \rho_m}{\partial t} + \nabla \cdot (\rho_m \mathbf{u}) = \nabla \cdot \left[\rho D \nabla \left(\frac{\rho_m}{\rho}\right)\right] + \dot{\rho}^c_m + \dot{\rho}^s \delta_{m1} \,, \tag{1}$$

where ρ_m is the mass density of species m, ρ the total mass density, and **u** the fluid velocity. We assume Fick's Law diffusion with a single diffusion coefficient D. Equations for D and source terms due to chemistry $\dot{\rho}^c{}_m$ and the spray $\dot{\rho}^s$ will be given later. Species 1 is the species of which the spray droplets are composed, and δ is the Dirac delta function. By summing Eq. (1) over all species we obtain the total fluid density equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = \dot{\rho}^s \ , \tag{2}$$

since mass is conserved in chemical reactions.

The momentum equation for the fluid mixture is

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\frac{1}{\alpha^2} \nabla p - A_o \nabla (2/3\ \rho k) + \nabla \cdot \sigma + \mathbf{F}^s + \rho \mathbf{g} \ , \tag{3}$$

where p is the fluid pressure. The dimensionless quantity α is used in conjunction with the Pressure Gradient Scaling (PGS) Method.[36] This is a method for enhancing computational efficiency in low Mach number flows, where the pressure is nearly uniform. The user may opt not to use the PGS method, in which case $\alpha \equiv 1$. If the PGS method is used, then α, which varies only in time, is determined in a manner described in Appendix A.

In Eq. (3) the quantity A_o is zero in laminar calculations and unity when one of the turbulence models is used. The viscous stress tensor is Newtonian in form:

$$\sigma = \mu \left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right] + \lambda \nabla \cdot \mathbf{u} \ \mathbf{I} \ . \tag{4}$$

The first and second coefficients of viscosity, μ and λ, are defined later. The superscript T denotes the transpose and $\mathbf{I}$ is the unit dyadic. $\mathbf{F}^s$ is the rate of momentum gain per unit volume due to the spray, to be defined later. The specific body force $\mathbf{g}$ is assumed constant.

The internal energy equation is

$$\frac{\partial(\rho I)}{\partial t} + \nabla \cdot (\rho \mathbf{u} I) = -p \nabla \cdot \mathbf{u} + (1 - A_o)\sigma : \nabla \mathbf{u} - \nabla \cdot \mathbf{J} + A_o \rho \varepsilon + \dot{Q}^c + \dot{Q}^s \ , \tag{5}$$

where I is the specific internal energy, exclusive of chemical energy. The heat flux vector $\mathbf{J}$ is the sum of contributions due to heat conduction and enthalpy diffusion:

$$\mathbf{J} = -K \nabla T - \rho D \sum_m h_m \nabla (\rho_m / \rho) \ , \tag{6}$$

where T is the fluid temperature and h_m the specific enthalpy of species m. The source terms due to chemical heat release $\dot{Q}^c$ and spray interactions $\dot{Q}^s$ will be defined later.

When one of the turbulence models are in use ($A_o = 1$), two additional transport equations are solved for the turbulent kinetic energy k and its dissipation rate ε:

$$\frac{\partial \rho k}{\partial t} + \nabla\cdot(\rho \mathbf{u} k) = -\frac{2}{3}\rho k \nabla\cdot\mathbf{u} + \sigma:\nabla\mathbf{u} + \nabla\cdot\left[\left(\frac{\mu}{Pr_k}\right)\nabla k\right] - \rho\varepsilon + \dot{W}^s , \tag{7}$$

and

$$\frac{\partial\rho\varepsilon}{\partial t} + \nabla\cdot(\rho\mathbf{u}\varepsilon) = -\left(\tfrac{2}{3}c_{\varepsilon_1} - c_{\varepsilon_3}\right)\rho\varepsilon\nabla\cdot\mathbf{u} + \nabla\cdot\left[\left(\frac{\mu}{Pr_\varepsilon}\right)\nabla\varepsilon\right] + \frac{\varepsilon}{k}\left[c_{\varepsilon_1}\sigma:\nabla\mathbf{u} - c_{\varepsilon_2}\rho\varepsilon + c_s\dot{W}^s\right] . \tag{8}$$

These are standard $k-\varepsilon$ equations[26] with some added terms. The source term $(c_{\varepsilon_3} - \frac{2}{3}c_{\varepsilon_1})$ $\nabla\cdot\mathbf{u}$ in the ε-equation accounts for length scale changes when there is velocity dilatation. Source terms involving the quantity $\dot{W}^s$ arise due to interaction with the spray. Later we will define $\dot{W}^s$ and give its physical significance.

The quantities c_{ε_1}, c_{ε_2}, c_{ε_3}, Pr_k, and Pr_ε are constants whose values are determined from experiments and some theoretical considerations. Standard values of these constants are often used in engine calculations, and these are given in Table II below. A value of c_s equal to 1.50 has been suggested,[37] based on the postulate of length scale conservation in spray/turbulence interactions, and has been found to give good agreement with measurements of diesel sprays.[4]

When the SGS turbulence model is used, the value of ε is constrained to satisfy the inequality

$$\varepsilon \geq \left[\frac{c_\mu}{Pr_\varepsilon(c_{\varepsilon_2} - c_{\varepsilon_1})}\right]^{\frac{1}{2}} \frac{k^{3/2}}{L_{SGS}} . \tag{9}$$

L_{SGS} is an input SGS length scale whose value is typically taken to be $4\delta x$, where δx is a representative computational cell dimension. Inequality (9) is enforced by integrating Eqs. (7) and (8) in time at all points and then setting ε equal to the right-hand side of Eq. (9) at points where the inequality is violated. Since $k^{3/2}/\varepsilon$ is proportional to the k–ε length scale, Eq. (9) is a constraint that the turbulent length scale be less than or equal L_{SGS}.

TABLE II

STANDARD VALUES OF k – ε TURBULENCE MODEL CONSTANTS

$c_{\varepsilon_1} = \ 1.44$	$Pr_k = 1.0$
$c_{\varepsilon_2} = \ 1.92$	$Pr_\varepsilon = 1.3$
$c_{\varepsilon_3} = -1.0$	

Near rigid walls this constraint is always satisfied (see the Boundary Conditions section), and thus the standard k–ε equations are solved near walls. In regions where the length scale is L_{SGS}, the model reduces to a one-equation SGS model similar to that of KIVA.[1]

The state relations are assumed to be those of an ideal gas mixture. Therefore,

$$p = R_0 T \sum_m (\rho_m / W_m) \ , \tag{10}$$

$$I(T) = \sum_m (\rho_m / \rho) I_m(T) \ , \tag{11}$$

$$c_p(T) = \sum_m (\rho_m / \rho) c_{pm}(T) \ , \tag{12}$$

and

$$h_m(T) = I_m(T) + R_0 T / W_m \ , \tag{13}$$

where R_0 is the universal gas constant; W_m, the molecular weight of species m; $I_m(T)$, the specific internal energy of species m; and c_{pm}, the specific heat at constant pressure of species m. The values of $h_m(T)$ and $c_{pm}(T)$ are taken from the JANAF tables.[38]

The chemical reactions occurring in the system are symbolized by

$$\sum_m a_{mr} \chi_m \leftrightarrows \sum_m b_{mr} \chi_m \ , \tag{14}$$

where χ_m represents one mole of species m and a_{mr} and b_{mr} are integral stoichiometric coefficients for reaction r. The stoichiometric coefficients must satisfy

$$\sum_m (a_{mr} - b_{mr}) W_m = 0 \ , \tag{15}$$

so that mass is conserved in chemical reactions. Chemical reactions are divided into two classes: those that proceed kinetically and those that are assumed to be in equilibrium. Kinetic reaction r proceeds at a rate $\dot{\omega}_r$ given by

$$\dot{\omega}_r = k_{fr} \prod_m (\rho_m / W_m)^{a'_{mr}} - k_{br} \prod_m (\rho_m / W_m)^{b'_{mr}} \ . \tag{16}$$

Here the reaction orders a'_{mr} and b'_{mr} need not equal a_{mr} and b_{mr}, so that empirical reaction orders can be used. The coefficients k_{fr} and k_{br} are assumed to be of a generalized Arrhenius form:

$$k_{fr} = A_{fr} T^{\zeta_{fr}} exp\{-E_{fr}/T\} ,$$

and

$$k_{br} = A_{br} T^{\zeta_{br}} exp\{-E_{br}/T\} , \tag{17}$$

where E_{fr} and E_{br} are activation temperatures.

The rates of equilibrium reactions are implicitly determined by the constraint conditions

$$\prod_m (\rho_m/W_m)^{b_{mr}-a_{mr}} = K_c^r(T) , \tag{18}$$

where $K_c{}^r(T)$, the concentration equilibrium constant, is assumed to be of the form

$$K_c^r = exp\{A_r \ell n T_A + B_r/T_A + C_r + D_r T_A + E_r T_A^2\} , \tag{19}$$

where $T_A = T/1000$ K.

With the reactions rates $\dot{\omega}_r$ determined by Eqs. (16) or (18), the chemical source term in the species continuity equation is given by

$$\dot{\rho}_m^c = W_m \sum_r (b_{mr} - a_{mr}) \dot{\omega}_r , \tag{20}$$

and the chemical heat release term in the energy equation is given by

$$\dot{Q}^c = \sum_r Q_r \dot{\omega}_r , \tag{21}$$

where Q_r is the negative of the heat of reaction at absolute zero,

$$Q_r = \sum_m (a_{mr} - b_{mr})(\Delta h_f^o)_m , \tag{22}$$

and $(\Delta h_f^o)_m$ is the heat of formation of species m at absolute zero.

The transport coefficients in KIVA are taken to be

$$\mu = (1.0 - A_o)\rho\upsilon_o + \mu_{air} + A_o c_\mu k^2/\varepsilon \ ,$$

$$\lambda = A_3\mu \ ,$$

$$K = \frac{\mu c_p}{Pr} \ ,$$

and

$$D = \frac{\mu}{\rho Sc} \ . \tag{23}$$

The diffusivity υ_o is an input constant, and c_μ is an empirical constant with a standard value of 0.09. A Sutherland formula is used for μ_{air}:

$$\mu_{air} = \frac{A_1 T^{3/2}}{T + A_2} \ , \tag{24}$$

where A_1 and A_2 are constants. The constant A_3 is taken to be $-\frac{2}{3}$ in calculations of turbulent flow but can be arbitrarily specified in laminar flows. The Prandtl and Schmidt numbers, Pr and Sc, are input constants.

B. The Spray Droplets

Solving for the essential dynamics of a spray and its interactions with a gas is an extremely complicated problem. To calculate the mass, momentum, and energy exchange between the spray and the gas, one must account for a distribution of drop sizes, velocities, and temperatures. In many sprays, drop Weber numbers[39] are larger than unity, and drop oscillations, distortions, and breakup must be considered. Drop collisions and coalescences have also been found to be important in many engine sprays.[34,40-42] A mathematical formulation that is capable of representing these complex physical processes is the spray equation formulation.[43] In this formulation we solve for a droplet probability distribution function f, and in KIVA-II f has ten independent variables in addition to time. These are the three droplet position components **x**, three velocity components **v**, equilibrium radius r (the radius the droplet would have if it were spherical), temperature T_d (assumed to be uniform within the drop), distortion from sphericity y, and the time rate of change $dy/dt = \dot{y}$. We keep track of the fundamental mode of oscillation corresponding to the lowest order spherical zonal harmonic[44] with axis aligned with the relative velocity vector between the droplet and gas. The dimensionless quantity y is proportional to the displacement of the droplet surface from its equilibrium position divided by the droplet radius r. Droplets break up if and only if $y > 1.0$.[35]

The droplet distribution function f is defined in such a way that

$$f(\mathbf{x}, \mathbf{v}, r, T_d, y, \dot{y}, t)\, d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y}$$

is the probable number of droplets per unit volume at position x and time t with velocities in the interval $(\mathbf{v}, \mathbf{v} + d\mathbf{v})$, radii in the interval $(r, r + dr)$, temperatures in the interval $(T_d, T_d + dT_d)$, and displacement parameters in the intervals $(y, y + dy)$ and $(\dot{y}, \dot{y} + d\dot{y})$. Two moments of f have important physical significance. The liquid volume fraction θ, given by

$$\theta = \int f 4/3\, \pi\, r^3\, d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y}\ ,$$

is assumed to be small compared to unity in our equations. The liquid macroscopic density ρ'_ℓ, given by

$$\rho'_\ell = \rho_d \theta\ ,$$

where ρ_d is the liquid microscopic density, can nevertheless be comparable to or larger than the gas density ρ because of the large ratio of ρ_d to ρ. The density ρ_d is assumed constant.

The time evolution of f is obtained by solving a form of the spray equation,

$$\frac{\partial f}{\partial t} + \nabla_{\mathbf{x}} \cdot (f\mathbf{v}) + \nabla_{\mathbf{v}} \cdot (f\mathbf{F}) + \frac{\partial}{\partial r}(fR) + \frac{\partial}{\partial T_d}(f\dot{T}_d) + \frac{\partial}{\partial y}(f\dot{y}) + \frac{\partial}{\partial \dot{y}}(f\ddot{y}) = \dot{f}_{coll} + \dot{f}_{bu}\ . \qquad (25)$$

In Eq. (25), the quantities $\mathbf{F}$, R, $\dot{T}_d$, and $\ddot{y}$ are the time rates of change, following an individual drop, of its velocity, radius, temperature, and oscillation velocity $\dot{y}$. Expressions for these will be given later. The terms $\dot{f}_{coll}$ and $\dot{f}_{bu}$ are sources due to droplet collisions and breakups, and we now define these.

The collision source term $\dot{f}_{coll}$ is given by

$$\dot{f}_{coll} = \frac{1}{2} \int\int f(\mathbf{x}, \mathbf{v}_1, r_1, T_{d_1}, y_1, \dot{y}_1, t)\, f(\mathbf{x}, \mathbf{v}_2, r_2, T_{d_2}, y_2, \dot{y}_2, t)\, \pi\, (r_1 + r_2)^2 |\mathbf{v}_1 - \mathbf{v}_2|$$

$$\{\sigma(\mathrm{v}, r, T_d, y, \dot{y}, \mathrm{v}_1, r_1, T_{d_1}, y_1, \dot{y}_1, \mathrm{v}_2, r_2, T_{d_2}, y_2, \dot{y}_2)$$

$$-\,\delta(\mathrm{v}-\mathrm{v}_1)\,\delta(r-r_1)\,\delta(T_d-T_{d_1})\,\delta(y-y_1)\,\delta(\dot{y}-\dot{y}_1)\}$$

$$-\,\delta(\mathrm{v}-\mathrm{v}_2)\,\delta(r-r_2)\,\delta(T_d-T_{d_2})\,\delta(y-y_2)\,\delta(\dot{y}-\dot{y}_2)$$

$$d\mathrm{v}_1\, dr_1\, dT_{d_1}\, dy_1 d\dot{y}_1\, d\mathrm{v}_2\, dr_2\, dT_{d_2}\, dy_2\, d\dot{y}_2\ . \qquad (26)$$

The collision transition probability function σ is defined so that $\sigma d\mathrm{v}\ dr\ dT_d\ dy\ d\dot{y}$ is the probable number of drops with properties in the implied intervals that result from a collision between a droplet with subscript 1 properties and one with subscript 2 properties. Two types of collisions are accounted for. If the collision impact parameter b is less than a critical value b_{cr} the droplets coalesce, and if b exceeds b_{cr} the droplets maintain their sizes and temperatures but undergo velocity changes. The critical impact parameter b_{cr} is given by

$$b_{cr}^2 = (r_1 + r_2)^2\, min\,(1.0, 2.4\, f(\gamma)/We_L)\ ,$$

$$f(\gamma) = \gamma^3 - 2.4\,\gamma^2 + 2.7\,\gamma\ ,$$

$$\gamma = r_2/r_1 \quad \text{where} \quad r_1 \le r_2\ ,$$

$$We_L = \rho_d |\mathrm{v}_1 - \mathrm{v}_2|\, r_1 / \alpha(\overline{T}_d)\ ,$$

and

$$\overline{T}_d = \frac{r_1^3 T_{d_1} + r_2^3 T_{d_2}}{r_1^3 + r_2^3}\ . \qquad (27)$$

The quantity α is the liquid surface tension coefficient, which is assumed to vary linearly between reference value α_0 at reference temperature T_0 and zero at the fuel species critical temperature T_{cr}. The precise form for σ is

$$\sigma = \frac{b_{cr}^2}{(r_1+r_2)^2}\,\delta[r-(r_1^3+r_2^3)^{\frac{1}{3}}]\,\delta\left[\mathrm{v}-\frac{r_1^3\mathrm{v}_1+r_2^3\mathrm{v}_2}{r_1^3+r_2^3}\right]\delta\left[T_d-\frac{r_1^3T_{d_1}+r_2^3T_{d_2}}{r_1^3+r_2^3}\right]\delta(y-y_2)\,\delta(\dot y-\dot y_2)$$

$$+\frac{2}{(r_1+r_2)^2}\int_{b_{cr}}^{r_1+r_2}[\delta(r-r_1)\,\delta(\mathrm{v}-\mathrm{v}_1')\,\delta(T_d-T_{d_1})\,\delta(y-y_1)\,\delta(\dot y-\dot y_1)$$

$$+\,\delta(r-r_2)\,\delta(\mathrm{v}-\mathrm{v}_2')\,\delta(T_d-T_{d_2})\,\delta(y-y_2)\,\delta(\dot y-\dot y_2)]\,b\,db\ , \tag{28}$$

where

$$\mathrm{v}_1' = \frac{r_1^3\mathrm{v}_1+r_2^3\mathrm{v}_2+r_2^3(\mathrm{v}_1-\mathrm{v}_2)\,\dfrac{b-b_{cr}}{r_1+r_2-b_{cr}}}{r_1^3+r_2^3}$$

and

$$\mathrm{v}_2' = \frac{r_1^3\mathrm{v}_1+r_2^3\mathrm{v}_2+r_1^3(\mathrm{v}_2-\mathrm{v}_1)\,\dfrac{b-b_{cr}}{r_1+r_2-b_{cr}}}{r_1^3+r_2^3}\ .$$

Justification for Eqs. (26)-(28) is given in Ref. 34.

The breakup source term $\dot f_{bu}$ is given by

$$\dot f_{bu} = \int f(\mathbf{x},\mathrm{v}_1,r_1,T_{d_1},1,\dot y_1,t)\,\dot y_1\,B(\mathrm{v},r,T_d,y,\dot y,\mathrm{v}_1,r_1,T_{d_1},\dot y_1,\mathbf{x},t)\,d\mathrm{v}_1\,dr_1\,dT_{d_1}\,d\dot y_1\ . \tag{29}$$

The breakup transition probability function B is defined so that $B d\mathrm{v}\ dr\ dT_d\ dy\ d\dot y$ is the probable number of droplets with properties in the implied intervals that are produced by the breakup of a droplet with subscript 1 properties. The meaning of Eq. (29) is the following: when a droplet's distortion y exceeds unity, it breaks up into a distribution of smaller drops given by B. We obtain the total source to f by multiplying the local flux of droplets through the surface $y = 1$ by B and integrating over the entire surface $y = 1$.

After breakup we assume the droplet radii follow a χ-squared distribution:

$$g(r) = \frac{1}{\bar r}e^{-r/\bar r}\ , \tag{30}$$

where the Sauter mean radius r_{32} is given by

$$r_{32} = 3\bar{r} = \frac{r_1}{\frac{7}{3} + \frac{1}{8}\,\frac{\rho_d r_1^3}{\alpha(T_{d_1})}\,\dot{y}_1^2} \quad . \tag{31}$$

The product droplet velocities also differ from that of the parent droplet by a velocity with magnitude w and with direction randomly distributed in a plane normal to the relative velocity vector between the parent drop and gas. The quantity w is given by

$$w = \tfrac{1}{2}\, r_1 \dot{y}_1 \; . \tag{32}$$

The precise form for B is

$$B = g\,(r)\,\delta\,(T_d - T_{d_1})\,\delta\,(y)\,\delta\,(\dot{y})\,\frac{1}{2\pi}\int \delta[\mathbf{v} - (\mathbf{v}_1 + w\mathbf{n})]\,d\mathbf{n} \;\; , \tag{33}$$

where the integral is over normal directions to the relative velocity vector. Justification for Eqs. (30) - (33) is given in Ref. 35.

We now define the functions **F**, R, $\dot{T}_d$, and $\ddot{y}$ that determine the trajectories of individual droplets. The droplet acceleration **F** has contributions due to aerodynamic drag and gravitational force:

$$\mathbf{F} = \frac{3}{8}\,\frac{\rho}{\rho_d}\,\frac{|\mathbf{u} + \mathbf{u}' - \mathbf{v}|}{r}\,(\mathbf{u} + \mathbf{u}' - \mathbf{v})\,C_D + \mathbf{g} \; . \tag{34}$$

The drag coefficient C_D is given by

$$C_D = \begin{cases} \dfrac{24}{Re_d}\,(1 + 1/6\,Re_d^{2/3}) & Re_d \; < \; 1000 \\[2ex] 0.424 & Re_d \; > \; 1000 \end{cases} \tag{35}$$

where

$$Re_d = \frac{2\rho\,|\mathbf{u} + \mathbf{u}' - \mathbf{v}|\,r}{\mu_{air}(\hat{T})} \quad ,$$

$$\hat{T} = \frac{T + 2T_d}{3} ,$$

and μ_{air} is given by Eq. (24). The gas turbulence velocity $\mathbf{u}'$ is added to the local mean gas velocity when calculating a droplet's drag and vaporization rate. It is assumed that each component $\mathbf{u}'$ follows a Gaussian distribution with mean square deviation 2/3 k. Thus we assume

$$G(\mathbf{u}') = (4/3\, \pi\, k)^{-3/2}\, exp\, \{-3|\mathbf{u}'|^2/4k\} . \tag{36}$$

The value of $\mathbf{u}'$ is chosen once every turbulence correlation time t_{turb} and is otherwise held constant. The droplet correlation time is given by

$$t_{turb} = min\left(\frac{k}{\varepsilon} , c_{ps} \frac{k^{3/2}}{\varepsilon} \frac{1}{|\mathbf{u} + \mathbf{u}' - \mathbf{v}|} \right) , \tag{37}$$

where c_{ps} is an empirical constant with value 0.16432. Thus t_{turb} is the minimum of an eddy breakup time and a time for the droplet to traverse an eddy.

The rate of droplet radius change R is given by the Frossling correlation,[45]

$$R = - \frac{(\rho D)_{air}(\hat{T})}{2\rho_d r} \frac{Y_1^* - Y_1}{1 - Y_1^*} Sh_d , \tag{38}$$

where Sh_d is the Sherwood number for mass transfer, $Y_1{}^*$ is the fuel vapor mass fraction at the droplet's surface, $\overline{Y_1} = \rho_1/\rho$, and $(\rho D)_{air}(\hat{T})$ is the fuel vapor diffusivity in air. The Sherwood number is given by

$$Sh_d = (2.0 + 0.6\, Re_d^{\frac{1}{2}} Sc_d^{\frac{1}{3}}) \frac{\ell n\, (1 + B_d)}{B_d} , \tag{39}$$

where $Sc_d = \dfrac{\mu_{air}(\hat{T})}{\rho D_{air}(\hat{T})}$ and $B_d = \dfrac{Y_1^* - Y_1}{1 - Y_1^*}$. The surface mass fraction $Y_1{}^*$ is obtained from

$$Y_1^*(T_d) = \frac{W_1}{W_1 + W_0\left(\dfrac{p}{p_v(T_d)} - 1 \right)} , \tag{40}$$

where W_0 is the local average molecular weight of all species exclusive of fuel vapor and $p_v(T_d)$ is the equilibrium fuel vapor pressure at temperature T_d. To obtain Eq. (40), we have assumed that the droplet temperature is uniform and that the partial pressure of fuel vapor at the droplet's surface equals the equilibrium vapor pressure. For the vapor diffusivity in air we use the empirical correlation

$$(\rho D)_{air}(T) = D_1 T^{D_2} ,$$

where D_1 and D_2 are constants.

The rate of droplet temperature change is determined by the energy balance equation

$$\rho_d \frac{4}{3} \pi r^3 c_\ell \dot{T}_d - \rho_d 4\pi r^2 \dot{R} L(T_d) = 4\pi r^2 Q_d , \tag{41}$$

where c_ℓ is the liquid specific heat, $L(T_d)$ is the latent heat of vaporization, and Q_d is the rate of heat conduction to the droplet surface per unit area. Equation (36) is a statement that the energy conducted to the droplet either heats up the droplet or supplies heat for vaporization. The heat conduction rate Q_d is given by the Ranz-Marshall correlation:[45]

$$Q_d = \frac{K_{air}(\hat{T})(T - T_d)}{2r} Nu_d , \tag{42}$$

where

$$Nu_d = (2.0 + 0.6\, Re_d^{\frac{1}{2}} Pr_d^{\frac{1}{3}}) \frac{\ell n(1 + B_d)}{B_d} ,$$

$$Pr_d = \frac{\mu_{air}(\hat{T}) c_p(\hat{T})}{K_{air}(\hat{T})} ,$$

$$K_{air}(\hat{T}) = \frac{K_1 \hat{T}^{3/2}}{\hat{T} + K_2} ,$$

c_p is the local specific heat at constant pressure and at temperature $\hat{T} = (T + 2T_d)/3$, and K_1 and K_2 are constants.

Consistent with the approximation that the liquid density is constant, we also assume its internal energy I_ℓ is a function of temperature alone. Thus the liquid enthalpy will have a small pressure dependence,

$$h_\ell(T_d,p) = I_\ell(T_d) + p/\rho_d \; . \tag{43}$$

Since the latent heat of vaporization L is the energy required to convert a unit mass of liquid to vapor at constant pressure equal to the equilibrium vapor pressure, the liquid and vapor enthalpies and internal energies and L are related by

$$L(T_d) = h_1(T_d) - h_\ell(T_d,p_v(T_d)) = I_1(T_d) + RT_d/W_1 - I_\ell(T_d) - p_v(T_d)/\rho_d \; . \tag{44}$$

The equation for the acceleration of the droplet distortion parameter is

$$\ddot{y} = \frac{2}{3}\,\frac{\rho}{\rho_d}\,\frac{(\mathbf{u} + \mathbf{u}' - \mathbf{v})^2}{r^2} - \frac{8\,\alpha(T_d)}{\rho_d r^3}\,y - \frac{5\mu_\ell(T_d)}{\rho_d r^2}\,\dot{y} \; , \tag{45}$$

where $\mu_\ell(T_d)$ is the viscosity of the liquid. Equation (45), which is based on the analogy between an oscillating droplet and a spring-mass system,[46] is the equation of a forced, damped harmonic oscillator. The external force is supplied by the gas aerodynamic forces on the droplet. The restoring force is supplied by surface tension forces. Damping is supplied by liquid viscosity. A detailed discussion of Eq. (45) may be found in Ref. 35.

We are now in a position to give the exchange functions $\dot{\rho}^s$, $\mathbf{F}^s$, $\dot{Q}^s$, and $\dot{W}^s$. These are obtained by summing the rates of change of mass, momentum, and energy of all droplets at position $\mathbf{x}$ and time t.[34] Thus one obtains

$$\dot{\rho}^s = -\int f\rho_d 4\pi r^2 R \; d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y} \; ,$$

$$\mathbf{F}^s = -\int f\rho_d (4/3\;\pi r^3 \mathbf{F}' + 4\pi r^2 R\,\mathbf{v})\, d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y} \; , \tag{46}$$

$$\dot{Q}^s = -\int f\rho_d \{4\pi r^2 R[I_\ell(T_d) + \tfrac{1}{2}(\mathbf{v} - \mathbf{u})^2] + 4/3\,\pi r^3 \{c_\ell \dot{T}_d + \mathbf{F}' \cdot (\mathbf{v} - \mathbf{u} - \mathbf{u}')]\}\, d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y} \; ,$$

and

$$\dot{W}^s = -\int f\rho_d 4/3\,\pi r^3 \mathbf{F}' \cdot \mathbf{u}'\, d\mathbf{v}\, dr\, dT_d\, dy\, d\dot{y} \; ,$$

where $\mathbf{F}' = \mathbf{F} - \mathbf{g}$. Physically, $\dot{W}^s$ is the negative of the rate at which the turbulent eddies are doing work in dispersing the spray droplets. Since $\mathbf{u}'$ follows the Gaussian distribution (36) it can be shown that $\dot{W} < 0$, and thus this term always depletes turbulent kinetic energy.

C. Boundary Conditions

In this section we give the physical boundary conditions that are available as standard options in the KIVA-II code. There are also numerical boundary conditions used by the program in conjunction with inflow and outflow boundaries. Numerical boundary conditions are extra conditions that are not required by the equations themselves but that have been found through experience to be necessary in implementing computational boundaries in fluid flow codes.[14] To understand these numerical conditions requires some familiarity with the numerical solution procedure in KIVA-II, and for this reason we defer discussion of inflow and outflow boundaries until Sec. IV.K of this report.

In addition to inflow and outflow boundaries, two types of physical boundaries are available in KIVA-II -- rigid walls and periodic boundaries. There are, in turn, several types of rigid walls depending on velocity and temperature boundary conditions. The velocity boundary conditions on rigid walls can be free slip, no slip, or turbulent law-of-the-wall. Temperature boundary condition options are adiabatic walls and fixed temperature walls. In engine calculations one ordinarily uses turbulent law-of-the-wall velocity conditions with fixed temperature walls. We now give in detail the rigid wall boundary conditions for the gas-phase equations.

Velocity boundary conditions on rigid walls are introduced either by imposing the value of the velocity on walls or the value of the wall stress $\sigma_w = \sigma.\mathbf{n}$, where $\mathbf{n}$ is the unit normal to the wall. On no slip walls, the gas velocity is set equal to the wall velocity:

$$\mathbf{u} = w_{wall}\mathbf{k} \ , \tag{47}$$

where the wall is assumed to be moving with speed w_{wall} in the z-direction. The wall stress is then determined implicitly through Eq. (3). On free-slip and turbulent law-of-the-wall boundaries the normal gas velocity is set equal to the normal wall velocity,

$$\mathbf{u} \cdot \mathbf{n} = w_{wall}\mathbf{k} \cdot \mathbf{n} \ , \tag{48}$$

and the two tangential components of σ_w are explicitly specified. For free-slip walls the tangential components of σ_w are zero. For turbulent law-of-the-wall conditions the tangential components are determined by matching to a logarithmic profile:

$$\frac{v}{u^*} = \begin{cases} 1/\kappa\, \ell n\,(c_{\ell w}\zeta^{7/8}) + B & \zeta > R_c \\ \zeta^{1/2} & \zeta < R_c \end{cases} , \tag{49}$$

where $\zeta = \frac{\rho y v}{\mu_{air}(T)}$ is the Reynolds number based on the gas velocity relative to the wall, $v = |\mathbf{u} - w_{wall}\mathbf{k}|$, which is evaluated a distance y from the wall, and u^* is the shear speed, which is related to the tangential components of the wall stress by

$$\sigma_w - (\sigma_w \cdot \mathbf{n})\mathbf{n} = \rho(u^*)^2 \frac{\mathbf{v}}{v} , \tag{50}$$

where $\mathbf{v} = \mathbf{u} - w_{wall}\mathbf{k}$.

In Eqs. (49) and (50) it is assumed that y is small enough to be in the logarithmic region or the laminar sublayer region of the turbulent boundary layer. The Reynolds number R_c defines the boundary between these two regions. The constants κ, $c_{\ell w}$, R_c, and B in Eq. (49) are related to the $k-\varepsilon$ model constants by

$$\kappa = \sqrt{c_\mu^{1/2}(c_{\varepsilon_2} - c_{\varepsilon_1})Pr_\varepsilon}$$

and

$$B = R_c^{1/2} - 1/\kappa\, \ell n\,(c_{\ell w} R_c^{7/8}) . \tag{51}$$

For commonly accepted values of the $k-\varepsilon$ constants, $B = 5.5$, and $c_{\ell w} = 0.15$, we obtain $\kappa = .4327$ and $R_c = 114$. A derivation of Eqs. (49)-(51) is given in Appendix B.

Temperature boundary conditions on rigid walls are introduced by specifying either the wall temperature or the wall heat flux $J_w = -k\nabla T \cdot \mathbf{n}$. For adiabatic walls, we set J_w equal to zero. For fixed temperature walls that are also either free slip or no slip, the wall temperature is prescribed, and J_w is determined implicity from Eq. (5). For fixed temperature walls using the turbulent law-of-the-wall condition, J_w is determined from the modified Reynolds analogy formula

$$\frac{J_w}{\rho u^* c_p(T - T_w)} = \begin{cases} 1/(Pr_\ell \frac{v}{u^*}) & \zeta \le R_c \\ 1/\left\{Pr\left[\frac{v}{u^*} + \left(\frac{Pr_\ell}{Pr} - 1\right)R_c^{1/2}\right]\right\} & \zeta > R_c \end{cases} , \tag{52}$$

where T_w is the wall temperature and Pr_ℓ is the Prandtl number of the laminar fluid. A derivation of Eq. (52) is given in Appendix B.

In addition to the wall heat loss, there is a source to the internal energy due to frictional heating. Frictional heating occurs whenever turbulent law-of-the-wall velocity conditions are used and has the form

$$f_w = \sigma_w \cdot \mathbf{v}$$

$$= \rho(u^*)^2 v \ ,$$

where f_w is the heating rate per unit area of wall.

In calculations of turbulent flow, boundary conditions are also needed for the turbulent kinetic energy k and its dissipation rate ε. These are taken to be

$$\nabla k \cdot \mathbf{n} = 0$$

and

$$\varepsilon = c_{\mu_\varepsilon} \frac{k^{3/2}}{y} \ , \tag{53}$$

where k and ε are evaluated a distance y from the wall and

$$c_{\mu_\varepsilon} = \left[\frac{c_\mu}{Pr_\varepsilon (c_{\varepsilon_2} - c_{\varepsilon_1})} \right]^{\frac{1}{2}} .$$

Periodic boundaries are only used in KIVA-II when the flow field is assumed to have an N-fold periodicity about the z-axis. When this assumption is used, the computational region is composed of points in the pie-shaped sector $0 \le \theta \le 2\pi/N$, where θ satisfies $cos\,\theta = x/\sqrt{x^2 + y^2}$ and $sin\,\theta = y/\sqrt{x^2 + y^2}$. The periodic boundaries are those for which $\theta = 0$ and $\theta = 2\pi/N$. The conditions imposed on these boundaries can be inferred from the assumed N-fold periodicity. For a scalar quantity q the requirement is that $q(r, \theta, z) = q(r, \theta + 2\pi/N, z)$, where $r = \sqrt{x^2 + y^2}$. For a vector v the requirement is that $\mathrm{v}(r, \theta + 2\pi/N, z) = \mathbf{R}\cdot\mathrm{v}(r, \theta, z)$, where **R** is the rotation matrix corresponding to the angle $2\pi/N$.

Boundary conditions are also needed for the spray equations, and we indicate here what these are for a spray injector and for a rigid wall. When a spray droplet impinges on a rigid wall, we set its velocity equal to the wall velocity, and for purposes of calculating heat and mass transfer with the gas, we set $Re_d = 0$ in Eqs. (39) and (42). There is no heat transfer between the droplet and wall. This is a provisional model for spray/wall interactions that will undoubtedly change as this important problem receives more attention from engine researchers.

Another type of physical boundary for the spray equation is the spray injector. This is a point in space at which we specify a droplet mass flow rate and a distribution of droplet sizes, velocities, temperatures, and oscillation parameters. An arbitrary number of spray injectors may be used in a single KIVA-II calculation. The mass flow rate for each injector is constant between times $T1_{inj}$ and $T2_{inj}$ and is zero otherwise. Two types of injector size distribution are available: monodisperse $(\delta(r - \bar{r}))$ or χ-squared $(1/\bar{r}\, exp(-r/\bar{r}))$. In either case, the number-averaged radius $\bar{r}$ is independent of time. The velocities of injected drops all have magnitude v_{inj}, which can be made an arbitrary function of time through tabular input to the program. The distribution of angles of the droplet velocities are defined relative to a spherical coordinate system in which the positive z'-axis is the spray axis, as depicted in Fig. 1. The direction of the spray axis can be arbitrarily specified and need not coincide with the z-axis of the flow field coordinate system. The polar angle ϕ of the droplet velocity relative to the spray axis is distributed uniformly in the interval $[\phi_{in}, \phi_{out}]$, and the azimuthal angle θ is distributed uniformly in the interval $[0, 2\pi]$. The temperatures of all injected drops have the same value $T_{d_{inj}}$. If the droplet breakup model is in use, then all injected drops have $y = 0$ and $\dot{y} = \dot{y}_{inj}$, where $\dot{y}_{inj}$ is the product of an input dimensionless amplitude and dimensional frequency. A more detailed description of the droplet injection procedure is given in the Computer Program section of this report.

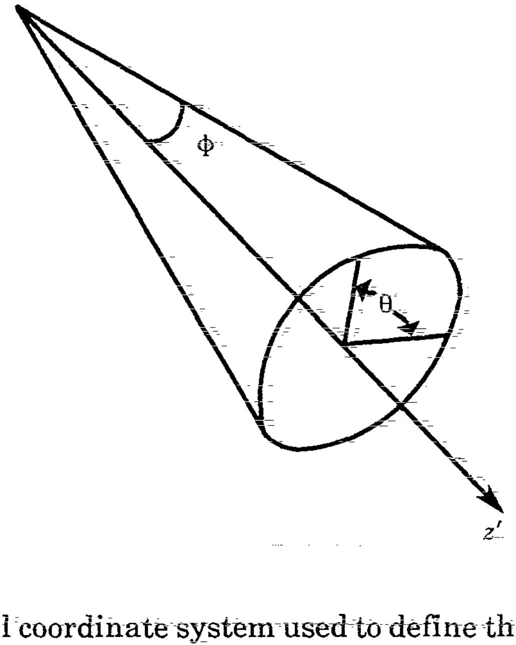


Fig. 1. Spherical coordinate system used to define the distribution of directions of droplet velocities at a spray injector.

III. THE NUMERICAL SCHEME

KIVA-II solves finite-difference approximations to the governing equations of Sec. II. The equations are discretized both in space and time. Before specifying the numerical scheme in detail, we discuss some of its general features and the principal ways in which it differs from the KIVA numerical scheme.

A. Temporal Differencing

The temporal differencing is performed with respect to a sequence of discrete times t^n ($n = 0, 1, 2, \ldots$). The time interval $\Delta t^n = t^{n+1} - t^n$ is the **timestep**, and the integer n is the **cycle number**. The latter is displayed as a superscript, so that Q^n denotes the difference approximation to the quantity Q at time t^n. When Δt appears without a superscript, Δt^n is understood. The difference approximation to the derivative $\partial Q/\partial t$ is the first-order expression $(Q^{n+1} - Q^n)/\Delta t$.

It is in its temporal difference scheme that KIVA-II differs most from KIVA. Just as in KIVA, a cycle is performed in three stages, or phases, but the terms differenced in each phase and their temporal differencing have changed considerably. Phases A and B together constitute a Lagrangian calculation in which computational cells move with the fluid. Phase A is a calculation of spray droplet collision and oscillation/breakup terms and mass and energy source terms due to the chemistry and spray. Phase B calculates in a coupled, implicit fashion the acoustic mode terms (namely the pressure gradient in the momentum equation and velocity dilatation terms in mass and energy equations), the spray momentum source term, and the terms due to diffusion of mass, momentum, and energy. Phase B also calculates the remaining source terms in the turbulence equations. In Phase C, the flow field is frozen and rezoned or remapped onto a new computational mesh. For the detailed description of each phase that is given later, it is convenient to define intermediate quantities that have been partially but not fully updated. Such quantities are identified by superscripts A and B. Thus, for example, Q^A is the computed value of Q at the end of Phase A. (Superscript C is not needed because it is equivalent to superscript $n+1$.)

B. Spatial Differencing

The spatial differencing is based on the ALE method,[17,18] which in three dimensions uses a mesh made up of arbitrary hexahedrons. Spatial difference approximations are constructed by the control-volume or integral-balance approach,[17] which largely preserves the local conservation properties of the differential equations.

The spatial region of interest is subdivided into a number of small **cells** or **zones**, the corners of which are the **vertices**. Together, the cells constitute the **mesh** with respect to which spatial differences are formed. The vertices need not be stationary, but may move in an arbitrarily prescribed manner. This capability includes the Lagrangian and Eulerian descriptions as special cases. In the general case, the cells are asymmetrical; a typical cell is shown in Fig. 2. The vertices are conventionally numbered as shown.

The cells are indexed by integers (i, j, k), which may be regarded as coordinates in logical space. The indices (i, j, k) also label the vertices, with the understanding that vertex (i, j, k) is vertex 4 for cell (i, j, k). The Cartesian coordinates of vertex (i, j, k) are $(x_{ijk}, y_{ijk}, z_{ijk})$, which in general depend on the time t. Thus the position vector to vertex (i, j, k) is

$$\mathbf{x}_{ijk} = x_{ijk}\mathbf{i} + y_{ijk}\mathbf{j} + z_{ijk}\mathbf{k} . \tag{54}$$

The "center" of cell (i, j, k) is defined as the point with coordinates

$$x^c_{ijk} = \frac{1}{8}\sum_{\alpha=1}^{8} x_\alpha ,$$

$$y^c_{ijk} = \frac{1}{8}\sum_{\alpha=1}^{8} y_\alpha ,$$

$$z^c_{ijk} = \frac{1}{8}\sum_{\alpha=1}^{8} z_\alpha , \tag{55}$$

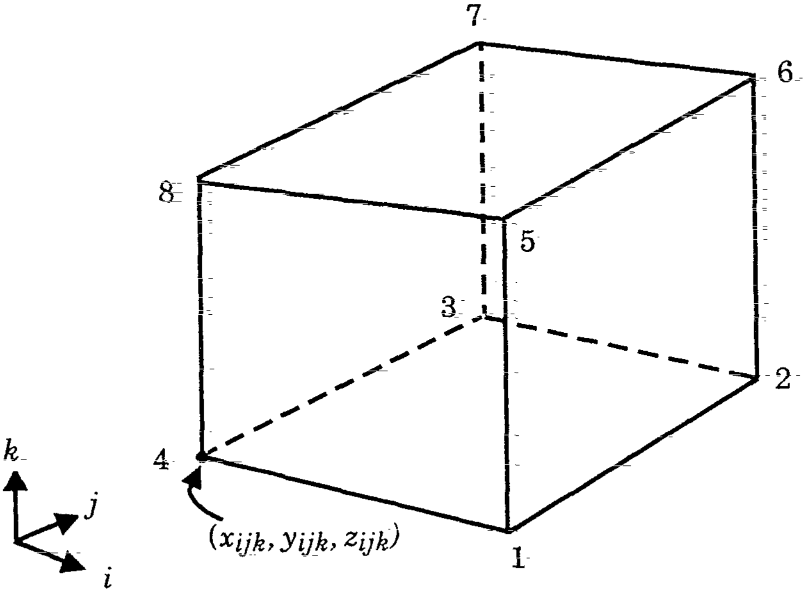


Fig. 2. Typical finite-difference cell.

where (x_a, y_a, z_a) are the coordinates of vertex a of cell (i, j, k). In general, the point $(xc_{ijk}, yc_{ijk}, zc_{ijk})$ is not the center of mass or volume of cell (i, j, k).

It is convenient to define auxiliary cells centered about the vertices. These cells are called **momentum cells** as their main use is in differencing the momentum equations. Momentum cell (i, j, k) is centered about vertex (i, j, k). In contrast to regular cells, which have six faces, momentum cells have twenty-four faces, each of which is comparable in size to one-fourth of a regular cell face. Three of these twenty-four faces lie within each of the eight regular cells which share common volume with the momentum cells. The portion of momentum cell (i, j, k) lying within regular cell (i, j, k) is shown in Fig. 3. The points of intersection of the momentum cell faces with the regular cell edges are defined as the midpoints of the regular cell edges. The points of intersection of the momentum cell edges with the regular cell faces are then defined implicitly by the requirement that the regular cell face be partitioned into four subfaces of equal area by the momentum cell faces. The corners of the momentum cells are then implicitly defined by the requirement that the overlap volume between a regular cell and a momentum cell centered at one of its corners be one-eighth of the regular cell volume. In general, the momentum cell corners do not coincide with the cell centers defined by Eq. (55). The momentum cell corners and the intersection points of momentum cell edges with regular cell faces are not actually solved for as they are not needed.

The location of velocities at cell vertices in the ALE method is convenient because no interpolation is required when determining vertex motion in the Lagrangian phase of the calculation, but it has a major drawback. This is that ALE method solutions are notoriously susceptible to parasitic modes in the velocity field. A major reason for this is that pressure waves tend to propagate along cell diagonals rather than via adjacent cells.[17,18] A "checkerboarding" effect is thereby created in the pressure field, with associated irregularities in the velocity field that are usually suppressed by the introduction of a numerical damping called node coupling.[16,17] In a major improvement to the ALE method, we have alleviated the susceptibility to parasitic modes by the introduction of velocities centered on cell faces.[2] Vertex velocities are retained, and momentum associated with the vertices is conserved, but normal velocity components on cell faces are used to compute cell volume changes in Phase B and fluxing volumes in Phase C. The resulting scheme greatly reduces the need for node coupling, and many problems can be run with no node coupling at all.

Accelerations of the cell-face velocities due to pressure gradients are calculated by constructing momentum control volumes centered about the cell faces. Like the momentum cells the cell-face control volumes have twenty-four faces. Referring to Figs. 2 and 3, the cell-face control volume for the left face of cell (i, j, k) is composed of those portions of

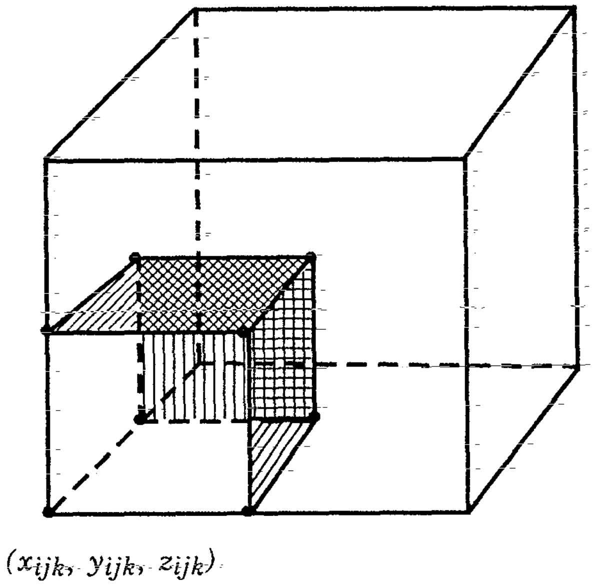


Fig. 3. The portion of momentum cell (i, j, k) lying within regular cell (i, j, k). The three momentum cell faces lying within the regular cell are shaded. Each momentum cell has twenty-four such faces in all.

the momentum cells of vertices 3, 4, 7, and 8 that lie in regular cells (i, j, k) and $(i-1, j, k)$. Control volumes associated with the other cell faces are defined analogously.

The volume of any momentum control volume may be calculated once the volumes of the main computational cells are known. The volume of cell (i, j, k) is denoted by V_{ijk} and is calculated by the following formula:[49]

$$V_{ijk} = \frac{1}{12} \sum_{\alpha=1}^{8} x_\alpha C_\alpha \, , \tag{56}$$

where

$$C_1 = -(y_2 z_3 + y_2 z_4 - y_2 z_5 - y_2 z_6 - y_3 z_2 + y_3 z_4 - y_4 z_2 - y_4 z_3 + y_4 z_5$$
$$+ y_4 z_8 + y_5 z_2 - y_5 z_4 + y_5 z_6 - y_5 z_8 + y_6 z_2 - y_6 z_5 - y_8 z_4 + y_8 z_5) \, ,$$

$$C_2 = (y_1 z_3 + y_1 z_4 - y_1 z_5 - y_1 z_6 - y_3 z_1 - y_3 z_4 + y_3 z_6 + y_3 z_7 - y_4 z_1 + y_4 z_3$$
$$+ y_5 z_1 - y_5 z_6 + y_6 z_1 - y_6 z_3 + y_6 z_5 - y_6 z_7 - y_7 z_3 + y_7 z_6) \, ,$$

$$C_3 = -(y_1 z_2 - y_1 z_4 - y_2 z_1 - y_2 z_4 + y_2 z_6 + y_2 z_7 + y_4 z_1 + y_4 z_2 - y_4 z_7$$
$$- y_4 z_8 - y_6 z_2 + y_6 z_7 - y_7 z_2 + y_7 z_4 - y_7 z_6 + y_7 z_8 + y_8 z_4 - y_8 z_7) \, ,$$

$$C_4 = -(y_1z_2 + y_1z_3 - y_1z_5 - y_1z_8 - y_2z_1 + y_2z_3 - y_3z_1 - y_3z_2 + y_3z_7$$

$$+ y_3z_8 + y_5z_1 - y_5z_8 - y_7z_3 + y_7z_8 + y_8z_1 - y_8z_3 + y_8z_5 - y_8z_7) \; ,$$

$$C_5 = (y_1z_2 - y_1z_4 + y_1z_6 - y_1z_8 - y_2z_1 + y_2z_6 + y_4z_1 - y_4z_8 - y_6z_1$$

$$- y_6z_2 + y_6z_7 + y_6z_6 - y_7z_6 + y_7z_8 + y_8z_1 + y_8z_4 - y_8z_6 - y_8z_7) \; ,$$

$$C_6 = (y_1z_2 - y_1z_5 - y_2z_1 + y_2z_3 - y_2z_5 + y_2z_7 - y_3z_2 + y_3z_7 + y_5z_1$$

$$+ y_5z_2 - y_5z_7 - y_5z_8 - y_7z_2 - y_7z_3 + y_7z_5 + y_7z_8 + y_8z_5 - y_8z_7) \; ,$$

$$C_7 = (y_2z_3 - y_2z_6 - y_3z_2 + y_3z_4 - y_3z_6 + y_3z_8 - y_4z_3 + y_4z_8 + y_5z_6$$

$$- y_5z_8 + y_6z_2 + y_6z_3 - y_6z_5 - y_6z_8 - y_8z_3 - y_8z_4 + y_8z_5 + y_8z_6) \; ,$$

$$C_8 = -(y_1z_4 - y_1z_5 - y_3z_4 + y_3z_7 - y_4z_1 + y_4z_3 - y_4z_5 + y_4z_7 + y_5z_1$$

$$+ y_5z_4 - y_5z_6 - y_5z_7 + y_6z_5 - y_6z_7 - y_7z_3 - y_7z_4 + y_7z_5 + y_7z_6) \; , \tag{57}$$

and the summation extends over the eight corners of cell (i, j, k).

It is also necessary to know the x-, y-, and z-projections of the surface areas of the cell faces. Since each face is common to two cells, there are three independent faces per cell. These are conventionally taken to be the left, front, and bottom faces of the cell as viewed from the perspective of Fig. 2. These faces are shown in Fig. 4. The conventional direction of their vector area elements is outward from the cell, as shown in Fig. 4. The area projections of these faces are calculated as follows:[49]

$$A_{\ell x} = -0.5\,[(y_3 - y_4)(z_8 - z_4) - (y_8 - y_4)(z_3 - z_4) + (y_8 - y_7)(z_3 - z_7) - (y_3 - y_7)(z_8 - z_7)] \; ,$$

$$A_{\ell y} = +0.5\,[(x_3 - x_4)(z_8 - z_4) - (x_8 - x_4)(z_3 - z_4) + (x_8 - x_7)(z_3 - z_7) - (x_3 - x_7)(z_8 - z_7)] \; ,$$

$$A_{\ell z} = -0.5\,[(x_3 - x_4)(y_8 - y_4) - (x_8 - x_4)(y_3 - y_4) + (x_8 - x_7)(y_3 - y_7) - (x_3 - x_7)(y_8 - y_7)] \; ,$$

$$A_{fx} = +0.5\,[(y_1 - y_4)(z_8 - z_4) - (y_8 - y_4)(z_1 - z_4) + (y_8 - y_5)(z_1 - z_5) - (y_1 - y_5)(z_8 - z_5)] \; ,$$

$$A_{fy} = -0.5\,[(x_1 - x_4)(z_8 - z_4) - (x_8 - x_4)(z_1 - z_4) + (x_8 - x_5)(z_1 - z_5) - (x_1 - x_5)(z_8 - z_5)]\ ,$$

$$A_{fz} = +0.5\,[(x_1 - x_4)(y_8 - y_4) - (x_8 - x_4)(y_1 - y_4) + (x_8 - x_5)(y_1 - y_5) - (x_1 - x_5)(y_8 - y_5)]\ ,$$

$$A_{bx} = +0.5\,[(y_3 - y_4)(z_1 - z_4) - (y_1 - y_4)(z_3 - z_4) + (y_1 - y_2)(z_3 - z_2) - (y_3 - y_2)(z_1 - z_2)]\ ,$$

$$A_{by} = +0.5\,[(z_3 - z_4)(x_1 - x_4) - (z_1 - z_4)(x_3 - x_4) + (z_1 - z_2)(x_3 - x_2) - (z_3 - z_2)(x_1 - x_2)]\ ,$$

$$A_{bz} = -0.5\,[(y_3 - y_4)(x_1 - x_4) - (y_1 - y_4)(x_3 - x_4) + (y_1 - y_2)(x_3 - x_2) - (y_3 - y_2)(x_1 - x_2)]\ , \tag{58}$$

where subscripts ℓ, f, and b refer respectively to the left, front, and bottom faces. When needed, the subscripts r, d, and t will be used to refer respectively to the right, back, and top faces of a computational cell. These area projections enable us to define area vectors $\mathbf{A}_\alpha$ associated with the faces α of each regular cell. It is convenient to associate $\mathbf{A}_\alpha$ with both the face and the cell, so that $\mathbf{A}_\alpha$ may be considered to always point outward from the cell under consideration. Thus, for a given face, the sign of $\mathbf{A}_\alpha$ depends on the side of the face from which it is viewed. Consider, for example, the left face of cell (i, j, k), as shown in Fig. 4, and compute $A_{\ell x}$, $A_{\ell y}$, and $A_{\ell z}$ by Eq. (58). From the point of view of cell (i, j, k), the vector $\mathbf{A}_\alpha$ for this face is $A_{\ell x}\mathbf{i} + A_{\ell y}\mathbf{j} + A_{\ell z}\mathbf{k}$. But of cell $(i-1, j, k)$, the vector $\mathbf{A}_\alpha$ for this face is $-A_{\ell x}\mathbf{i} - A_{\ell y}\mathbf{j} - A_{\ell z}\mathbf{k}$. Similar considerations apply to right, front, back, bottom, and top faces.

Similarly, the outward area vector associated with face α of a particular momentum cell is denoted by $\mathbf{A}'_\alpha$, and the outward area vector of face α of a cell-face control volume is

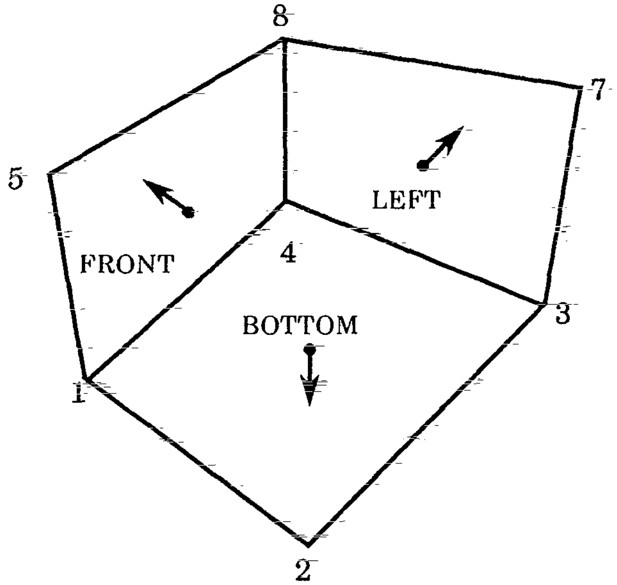


Fig. 4. Cell faces associated with cell (i,j,k).

denoted by $\mathbf{A}''_\alpha$. We shall not write out explicit expressions for the components or projections of these momentum cell-face area vectors, as it will always be possible to eliminate them in favor of the $\mathbf{A}_\alpha$ as discussed below.

In the finite-difference approximations of KIVA, velocities are fundamentally located at the vertices, so that

$$\mathbf{u}_{ijk} = \mathbf{u}\,(x_{ijk}, y_{ijk}, z_{ijk}) \ . \tag{59}$$

Thermodynamic quantities are located at cell centers:

$$Q_{ijk} = Q\,(x^c_{ijk}, y^c_{ijk}, z^c_{ijk}) \ , \tag{60}$$

where $Q = p$, ρ, T, I, or ρ_m, as well as k and ε. Quantities needed at points where they are not fundamentally located are obtained by averaging neighboring values.

Spatial differences are usually performed by integrating the differential term in question over the volume of a typical cell (or momentum cell). Volume integrals of gradient or divergence terms are usually converted into surface area integrals using the divergence theorem. The volume integral of a time derivative may be related to the derivative of the integral by means of Reynolds' transport theorem.[50] Volume and surface area integrals are usually performed under the assumption that the integrands are uniform within cells or on cell faces. Thus area integrals over surfaces of cells become sums over cell faces (or subfaces):

$$\int \mathbf{F} \cdot d\mathbf{A} \rightarrow \sum_\alpha \mathbf{F}_\alpha \cdot \mathbf{A}_\alpha \ . \tag{61}$$

When differencing diffusion terms for cell-centered quantity Q, it is necessary to evaluate $(\nabla Q)_\alpha \cdot \mathbf{A}_\alpha$. Referring to Fig. 5, this quantity is approximated as follows. The points $\mathbf{x}_\ell$ and $\mathbf{x}_r$ are the centers of the cells on either side of face α, and $\mathbf{x}_t$, $\mathbf{x}_b$, $\mathbf{x}_f$, and $\mathbf{x}_d$ are the centers of the four edges bounding face α. We first solve for coefficients $a_{\ell r}$, a_{tb}, and a_{fd} such that

$$a_{\ell r}(\mathbf{x}_\ell - \mathbf{x}_r) + a_{tb}(\mathbf{x}_t - \mathbf{x}_b) + a_{fd}(\mathbf{x}_f - \mathbf{x}_d) = \mathbf{A}_\alpha \ . \tag{62}$$

Note that since the mesh may be nonorthogonal, the vector $\mathbf{x}_\ell - \mathbf{x}_r$ need not be parallel to $\mathbf{A}_\alpha$, and thus a_{tb} and a_{fd} may be nonzero. The finite-difference approximation to $(\nabla Q)_\alpha \cdot \mathbf{A}_\alpha$ is obtained by dotting both sides of Eq. (62) with $(\nabla Q)_\alpha$ and ignoring terms of second and higher order in the cell dimensions:

$$a_{\ell r}(Q_\ell - Q_r) + a_{tb}(Q_t - Q_b) + a_{fd}(Q_f - Q_d) = (\nabla Q)_\alpha \cdot \mathbf{A}_\alpha \, . \tag{63}$$

In Eq. (63) Q_t is the simple average of the values of Q in the four cells surrounding cell edge "t," and Q_b, Q_f, and Q_d are defined analogously.

Area integrals over momentum cell faces are ordinarily converted into area integrals over regular cell faces by the following procedure. Let Q be a quantity that is uniform within regular cells, and consider the volume of overlap between regular cell (i,j,k) and the momentum cell associated with one of its vertices. Three faces of this overlap volume (call them a,b,c) are faces of the momentum call in question, with outward area vectors $\mathbf{A}'_\alpha$, while the other three (call them d,e,f) are surfaces of regular cell (i,j,k), with outward area vectors $\frac{1}{4}\mathbf{A}_\alpha$. But the divergence theorem shows that the integral $\int d\mathbf{A}$ over the entire surface of this overlap volume is zero, so that

$$\mathbf{A}'_a + \mathbf{A}'_b + \mathbf{A}'_c = -\frac{1}{4}(\mathbf{A}_d + \mathbf{A}_e + \mathbf{A}_f) \, . \tag{64}$$

Thus the integral $\int Qd\mathbf{A}$ over the three momentum cell faces in question my be represented by

$$\int Qd\mathbf{A} = Q_{ijk}(\mathbf{A}'_a + \mathbf{A}'_b + \mathbf{A}'_c) = -\frac{1}{4}Q_{ijk}(\mathbf{A}_d + \mathbf{A}_e + \mathbf{A}_f) \, , \tag{65}$$

so that the area vectors $\mathbf{A}'_\alpha$ never need to be explicitly evaluated. A similar procedure is used to express the outward normal areas $\mathbf{A}''_\alpha$ of faces of the cell-face control volumes in terms of the regular cell face areas $\mathbf{A}_\alpha$.

The mass of cell (i,j,k) is denoted by M_{ijk} and is given by

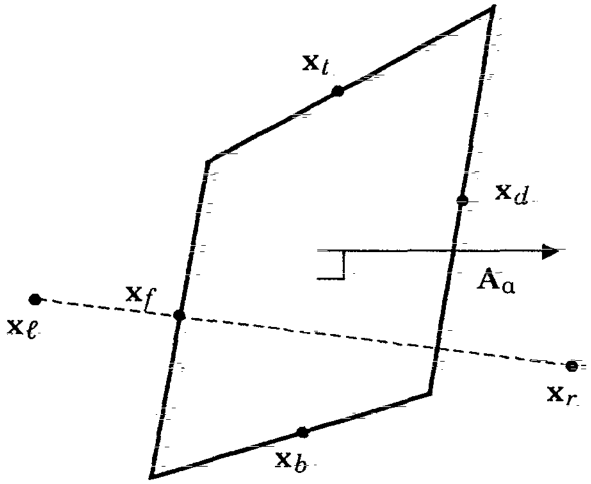


Fig. 5. The six points used to define the gradient of cell-centered quantity Q on cell face α.

$$M_{ijk} = \rho_{ijk} V_{ijk} \; . \tag{66}$$

The mass of momentum cell (i,j,k) is given by

$$\begin{aligned} M'_{ijk} = \frac{1}{8} (M_{ijk} &+ M_{i-1,j,k} + M_{i-1,j-1,k} + M_{i,j-1,k} + M_{i,j,k-1} \\ &+ M_{i-1,j,k-1} + M_{i-1,j-1,k-1} + M_{i,j-1,k-1}) \; . \end{aligned} \tag{67}$$

The mass of left cell-face control volume of cell (i,j,k) is given by

$$M''_{\ell} = \frac{1}{2} (M_{ijk} + M_{i-1,j,k}) \; , \tag{68}$$

and the mass of other cell-face control volumes are defined analogously.

C. Stochastic Particle Technique

A very efficient and accurate method for solving for the spray dynamics is based on the ideas of the Monte Carlo method and of discrete particle methods. In discrete particle methods, the continuous distribution f is approximated by a discrete distribution f':

$$f' = \sum_{p=1}^{NP} N_p \delta(\mathbf{x} - \mathbf{x}_p) \delta(\mathbf{v} - \mathbf{v}_p) \delta(r - r_p) \, \delta(T_d - T_{d_p}) \delta(y - y_p) \delta(\dot{y} - \dot{y}_p) \; . \tag{69}$$

Each particle p is composed of a number of droplets N_p having equal location $\mathbf{x}_p$, velocity $\mathbf{v}_p$, size r_p, temperature T_{d_p}, and oscillation parameters y_p and $\dot{y}_p$. Particle and droplet trajectories coincide (thus, for example, $d\mathbf{x}_p/dt = \mathbf{v}_p$ and $d\mathbf{v}_p/dt = \mathbf{F}_p$), and the particles exchange mass, momentum, and energy with the gas in the computational cells in which they are located. The finite-difference approximations to the ordinary differential equations for the particle trajectories and for the exchange rate functions of Eq. (46) are given in Appendix C.

Our method is a Monte Carlo method in the sense that we sample randomly from assumed probability distributions that govern droplet properties at injection and droplet behavior subsequent to injection. We now show how this random sampling is done. Let us assume that we are given the distribution function $f(x)$ corresponding to the random variable x $(x_1 \leq x \leq x_2)$. The distribution function is defined by $dN = f(x)\, dx$; i.e., this is the number of droplets in the interval dx about the value x. Let us define the random variable

$$y = \int_{x_1}^{x} f(x')\, dx' \ ,$$

and we note that $dN = dy$. Hence,the number of droplets is uniformly distributed with respect to the variable y. Commonly one has available random number generators with a uniform distribution in the range from zero to one. We therefore sample from this distribution, scale by

$$\int_{x_1}^{x_2} f(x)\, dx$$

to obtain y, and then invert to obtain x, which then will be distributed according to $f(x)$. Depending on the form of the distribution function, the integral and its inversion may be performed analytically or, failing that, by a numerical method. This sampling procedure is used in the droplet injection calculation, which is described in Appendix D; in the droplet collision calculation, which is described in Appendix E; in the droplet breakup calculation, described in Appendix F; and when a new value of the gas turbulence velocity $\mathbf{u}'_p$ must be chosen. In the latter case, since each component of $\mathbf{u}'_p$ follows a Gaussian distribution $G(u')$ and since

$$erf(x/\sqrt{4q/3}\,) = 2 \int_{0}^{x} G\,(u')\, du' \ , \qquad (70)$$

it follows that we must invert the error function *erf*. This is done by storing values of the inverse error function $erf^{-1}\,(y)$ evaluated at intervals of 0.05 from $y = 0.0$ to $y = 1.00$. The value of $erf^{-1}\,(1.0)$ is taken to be 2.0. Values of $erf^{-1}\,(y)$ at intermediate values of y are found by linear interpolation. If y is a random number between 0.0 and 1.0, then $2y - 1$ is a random number between -1.0 and 1.0 whose magnitude and sign determine the magnitude and sign of u':

$$u' = \sqrt{4q/3}\ sign\ (2y - 1)\, erf^{-1}\,(|2y - 1|)\ . \qquad (71)$$

Of course, a new $\mathbf{u}'_p$ is only sampled once every correlation time t_{turb} [see Eq. (37)].

When the turbulence correlation time t_{turb} of a spray particle is less than the computational timestep Δt, the droplet equations cannot be integrated directly since the particle "sees" many values of $\mathbf{u}'_p$ in a single timestep. Instead we add random turbulent particle position and velocity changes that are chosen from distributions derived assuming the droplet experiences a linear drag law. When $\Delta t > t_{turb}$, we also set $\mathbf{u}'_p = 0$ when differencing the droplet equations. Appendix G discusses briefly how the distributions of turbulent displacements are obtained, and a detailed derivation may be found in Ref. 33.

D. State Relations

The quantities $I_m(T)$ are obtained from the JANAF tables[38] and are stored in tabular form at intervals of 100 K. A simple linear interpolation is used to determine the $I_m(T)$ at temperatures within the range of the tables. The quantities $c_{vm}(T)$ are simply approximated by differences between adjacent tabular values of $I_m(T)$, divided by 100 K. The temperature T determines the internal energy I via Eq. (11). Conversely, I determines T via the inverse of this relation. This inversion is performed by a simple linear search algorithm which takes advantage of the fact that $(\partial I/\partial T)_{\rho_m}$ is always positive. Let $T_n = 100\, n\ (n = 0, 1, 2, \ldots)$, and choose the initial value of n so that T_n is a reasonable estimate of the correct T. One then evaluates $I_n = I(T_n)$ from Eq. (11). If $I_n > I$, n is reduced, and if $I_{n+1} < I$, n is increased. The search proceeds in this way until I is bracketed by I_n and I_{n+1}; T is then evaluated by linear interpolation between T_n and T_{n+1}.

The liquid fuel internal energy $I_\ell(T)$, the liquid viscosity $\mu_\ell(T)$, and the equilibrium vapor pressure $p_v(T)$ are also stored in tabular form. The liquid latent heat $L(T)$ is first stored at intervals of 100 K, and then the values of $I_\ell(T)$ are calculated at intervals of 100 K from Eq. (44) and known values of $L(T)$, $p_v(T)$, and $I_1(T)$. Because the vapor pressure p_v and liquid viscosity μ_ℓ vary rapidly with temperature, their values are stored at intervals of 10 K up to the fuel critical temperature T_{crit}. Latent heat, vapor pressure, and liquid viscosity tables for many fuels can be found in Refs. 51-53.

E. Lagrangian Phase Difference Equations

With the above background, we are now in a position to specify the KIVA difference equations. It is convenient to give these first for the Lagrangian phase and then for Phase C or the rezone phase. In the equations of this section, we use implicit methods to difference the terms associated with acoustic pressure wave propagation and diffusion of mass, momentum, and energy. In the next section we tell how these coupled implicit equations are solved.

1. Mass Density Equations. The Lagrangian phase difference approximation to Eq. (1) is

$$\frac{(\rho_m)^B_{ijk}\, V^B_{ijk} - (\rho_m)^n_{ijk}\, V^n_{ijk}}{\Delta t} = \sum_a (\rho D)^n_a\, \boldsymbol{\nabla} [\phi_D\, Y^B_m + (1 - \phi_D)\, Y^A_m]_a \cdot \mathbf{A}^n_a + [(\dot{\rho}^c_m)_{ijk} + \dot{\rho}^s_{ijk}\, \delta_{m1}]\, V^n_{ijk} \, . \tag{72}$$

The mass fractions are related to the densities by

$$Y^x_m = \rho^x_m / \rho^x , \tag{73}$$

where $x = n$, A, or B. The Phase A mass densities will be defined shortly. An important feature of Eq. (72) is the use of variable implicitness parameter ϕ_D in differencing the diffusion term. Parameter ϕ_D varies in space and time and is defined at cell centers. Its value, which lies between zero and one, depends on the local diffusion Courant number

$$C_d = \frac{\mu}{\rho} \frac{\Delta t}{\Delta x^2} ,$$

where Δx is a measure of the cell size. When C_d is small compared to unity, ϕ_D is zero and a fully explicit difference approximation is used. When C_d is large compared to unity, ϕ_D is close to unity and an implicit formulation is used. The exact expression for ϕ_D, which is chosen to ensure numerical stability, is given in Appendix H.

The Phase A density of species m includes contributions due to chemistry and spray evaporation:

$$\frac{(\rho_m)^A_{ijk} - (\rho_m)^n_{ijk}}{\Delta t} = (\dot{\rho}^c_m)_{ijk} + \dot{\rho}^s_{ijk} \delta_{m1} . \tag{74}$$

The chemistry source term $(\dot{\rho}_m{}^c)_{ijk}$ is given by Eq. (20) with $\dot{\omega}_r$ replaced by $(\dot{\omega}_r{}^A)_{ijk}$. The integration method for kinetic reaction rates $\dot{\omega}_r$ is described in Appendix I, and that for equilibrium reactions is described in Appendix J. The finite-difference approximation to the spray mass source term $\dot{\rho}_{ijk}{}^s$ is given in Appendix C.

Summing Eq. (72) over all species m gives the following Lagrangian phase difference approximation to the mass density equation (2):

$$\frac{\rho^B_{ijk} V^B_{ijk} - \rho^n_{ijk} V^n_{ijk}}{\Delta t} = \dot{\rho}^s_{ijk} V^n_{ijk} . \tag{75}$$

This shows the total gas mass in a cell changes only due to the spray source. Similarly summing (74) over all species and comparing with (75) shows that Phase B cell masses are known after Phase A:

$$\rho^B_{ijk} V^B_{ijk} = \rho^A_{ijk} V^n_{ijk} = M^A_{ijk} = M^B_{ijk} . \tag{76}$$

Equation (76) can be combined with (72) - (74) to obtain

$$M^B_{ijk} \frac{(Y_m)^B_{ijk} - (Y_m)^A_{ijk}}{\Delta t} = \sum_{\alpha} (\rho D)^n_\alpha \nabla[\phi_D Y^B_m + (1 - \phi_D) Y^A_m]_\alpha \cdot \mathbf{A}^n_\alpha \, . \tag{77}$$

Equation (77) is solved in Phase B.

2. <u>Momentum Equation.</u> The Lagrangian phase difference approximation to the momentum equation (3) is the following:

$$\begin{aligned} \frac{(M')^B_{ijk}\mathbf{u}^B_{ijk} - (M')^n_{ijk}\mathbf{u}^n_{ijk}}{\Delta t} &= -\frac{1}{(\alpha^n)^2} \sum_\beta [\phi_p p^B + (1 - \phi_p) p^n]_\beta (\mathbf{A}')^n_\beta - A_0 \sum_\beta \tfrac{2}{3} \rho^A_\beta k^A_\beta (\mathbf{A}')^n_\beta \\ &+ \sum_\beta [\phi_D \boldsymbol{\sigma}(\mathbf{u}^B) + (1 - \phi_D) \boldsymbol{\sigma}(\mathbf{u}^n)]_\beta \cdot (\mathbf{A}')^n_\beta \qquad (78) \\ &- \frac{1}{\Delta t} (\mathbf{R}'_{ijk} + S'_{ijk}\mathbf{u}^B_{ijk}) + \mathbf{g}(M')^n_{ijk} - (M')^n_{ijk}(\mathbf{ANC})_{ijk}/\Delta t \, . \end{aligned}$$

The index β refers to the faces of momentum control volume (i,j,k), whose normal area vectors at time t^n are $(\mathbf{A}')_\beta{}^n$. The pressure p, turbulent kinetic energy k, and viscous stress tensor σ are regarded as being uniform within the regular cell in which face β lies. Thus the **A**′ can be eliminated in terms of regular cell face areas **A** as described in Sec. B above. This makes it convenient to evaluate momentum changes due to surface stresses by sweeping over cells rather than vertices in the computer program.

In differencing the pressure gradient term, variable implicitness parameter ϕ_p is used. Parameter ϕ_p plays a role for the acoustic mode terms analogous to that of ϕ_D for the diffusion terms. Appendix H gives the exact expression for ϕ_p, which depends on the sound speed Courant number

$$C_s = \frac{c\Delta t}{\Delta x} \, ,$$

where c is the isentropic speed of sound.

The PGS parameter α^n, which is used to artificially raise the Mach number in far subsonic flows,[36] depends on time but not on space. The formula used to determine α^n is given in Appendix A.

The viscous stress tensor σ is a weighted average of the stresses based on time level n velocities and those based on the Phase B velocities. This weighting employs the same variable implicitness parameter ϕ_D that is used for the mass diffusion terms. The value of σ depends on velocity gradients whose difference approximations are given in Appendix K.

The quantities $\mathbf{R}'_{ijk}$ and S'_{ijk} are associated with the implicit coupling of the computed gas and drop velocities. One can interpret S'_{ijk} as an added mass that arises because forces on momentum cell (i,j,k) must accelerate the droplets in addition to the gas in that cell. The appearance of S'_{ijk} in (78) is related to the well-known lowering of the sound speed in two-phase flows.[34] The evaluation of $\mathbf{R}'_{ijk}$ and S_{ijk} is described in Appendix C.

Although we have greatly reduced the susceptibility of KIVA solutions to parasitic velocity modes or alternate node uncoupling, in some problems the velocity field may develop persistent alternative-vertex irregularities of small amplitude. Such irregularities can usually be eliminated by using the alternate node coupler described in Appendix L. The effects of the alternate node coupler are represented by the term $(\mathbf{ANC})_{ijk}$, which is the sum over all regular cells surrounding vertex (i,j,k) of the terms $\delta \mathbf{u}_{ijk}{}^{anc}$ given in Eq. (L-4).

The Phase A vertex velocities include changes due to the spray momentum source, gravitational acceleration, and the alternate node coupler:

$$\frac{[(M')^A_{ijk} + S'_{ijk}]\mathbf{u}^A_{ijk} - (M')^n_{ijk}\mathbf{u}^n_{ijk}}{\Delta t} = -\mathbf{R}'_{ijk}/\Delta t + \mathbf{g}(M')^n_{ijk} - (M')^n_{ijk}(\mathrm{ANC})_{ijk}/\Delta t \, . \qquad (79)$$

3. <u>Cell Face Normal Velocities.</u> In addition to vertex velocities, in Phase B we use face-centered normal velocities. We compute accelerations of these velocities due to the thermodynamic and turbulence pressures p and $\frac{2}{3}\rho k$, and the resulting face-centered normal velocities are then used to calculate the Lagrangian phase cell volume changes. This procedure has been found to reduce dramatically the susceptibility of computed solutions to alternate node uncoupling.[16,17] The reasons for this improvement are discussed in Refs. 47 and 48.

We now describe how the face-centered velocities are computed. Rather than dealing directly with face-centered normal velocities, it is more convenient to introduce a factor of the cell face area. Thus the variables we use are

$$(uA)_\alpha = \mathbf{u}_\alpha \cdot \mathbf{A}_\alpha \, . \qquad (80)$$

An equation for $\mathbf{u} \cdot \mathbf{A}$ may be derived as follows. Consider the momentum balance for control volume V that moves with the fluid:

$$\frac{D}{Dt}\int_V \rho \mathbf{u} \, dv = \mathbf{F} \, , \qquad (81)$$

where $\mathbf{F}$ is the sum of all forces on V. By dotting this equation with area element $\mathbf{A}$ that is moving with the fluid, we obtain

$$\frac{D}{Dt}\int_V \rho \mathbf{u}\cdot\mathbf{A}\, dv = \mathbf{F}\cdot\mathbf{A} + \frac{D\mathbf{A}}{Dt}\cdot\int_V \rho\mathbf{u}\, dv \ . \tag{82}$$

In curved meshes, the last term in this equation gives rise to Coriolis and centrifugal force terms.

Equation (82) is differenced in the following manner. We initialize $(uA)_\alpha$ using vertex velocities $\mathbf{u}^t$ that differ from the Phase B velocities only in that terms in (78) are omitted that are associated with the thermodynamic and turbulence pressures p and $\frac{2}{3}\rho k$:

$$[(M')^B_{ijk} + S'_{ijk}]\,\frac{\mathbf{u}^B_{ijk} - \mathbf{u}^t_{ijk}}{\Delta t} = -\frac{1}{(\alpha^n)^2}\sum_\beta [\phi_p p^B + (1-\phi_p)p^n]_\beta(\mathbf{A}')^n_\beta - A_0\sum_\beta \tfrac{2}{3}\rho^A_\beta k^A_\beta(\mathbf{A}')^n_\beta \ . \tag{83}$$

Alternatively, $\mathbf{u}^t$ differs from $\mathbf{u}^A$ by the addition of the viscous term contributions:

$$[(M')^B_{ijk} + S'_{ijk}]\,\frac{\mathbf{u}^t_{ijk} - \mathbf{u}^A_{ijk}}{\Delta t} = \sum_\beta [\phi_D\sigma(\mathbf{u}^B) + (1-\phi_D)\sigma(\mathbf{u}^n)]_\beta\cdot(\mathbf{A}')^n_\beta \ . \tag{84}$$

The quantities $(uA)_\alpha$ are initialized by

$$(uA)^t_\alpha = \tfrac{1}{4}(\mathbf{u}^t_a + \mathbf{u}^t_b + \mathbf{u}^t_c + \mathbf{u}^t_d)\cdot\mathbf{A}^n_\alpha \ , \tag{85}$$

where a, b, c, and d label the vertices that form the four corners of cell face α. This labeling convention will be understood in what follows.

The finite-difference approximation to (82) is then given by

$$[(M'')^B_\alpha + S''_\alpha]\,\frac{(uA)^B_\alpha - (uA)^t_\alpha}{\Delta t} = -\sum_\gamma \{[\phi_p p^B + (1-\phi_p)p^n]/(\alpha^n)^2 + A_0\tfrac{2}{3}\rho^A k^A\}_\gamma(\mathbf{A}'')^n_\gamma\cdot\mathbf{A}_\alpha$$

$$+\ \frac{\mathbf{A}^t_\alpha - \mathbf{A}^n_\alpha}{\Delta t}\cdot\frac{\mathbf{u}^n_a + \mathbf{u}^n_b + \mathbf{u}^n_c + \mathbf{u}^n_d}{4}[(M'')^B_\alpha + S''_\alpha] \ . \tag{86}$$

The indices γ refer to the faces of cell face control volume α, whose normal area vectors at time t^n are $(\mathbf{A}'')_\gamma{}^n$. In computing $(uA)_\alpha{}^B$ the $\mathbf{A}''$ that do not coincide with regular cell faces are eliminated in favor of the regular cell face areas as is described in Sec. B above. For faces γ that lie entirely within a regular cell, the pressures and values of ϕ_p are taken to be those at the regular-cell center. For faces γ that lie on a regular cell face, the pressure on face γ is obtained by averaging the values of $\phi_p p^B + (1 - \phi_p)p^n$ associated with the regular cells on either side of the cell face.

It is necessary to define an added mass S_α'' associated with the spray droplets in cell-face control volume α. This quantity is given by

$$S_\alpha'' = \frac{S_a' + S_b' + S_c' + S_d'}{(M')_a^B + (M')_b^B + (M')_c^B + (M')_d^B}\,(M'')_\alpha^B \,. \tag{87}$$

To approximate dA/dt in Eq. (82) we have used an area change based on the time level n vertex velocities,

$$\mathbf{A}_\alpha^t = \mathbf{A}_\alpha(\mathbf{x}^n + \mathbf{u}^n\,\Delta t)\;; \tag{88}$$

that is, the areas $\mathbf{A}_\alpha{}^t$ are computed using Eq. (58) with the vertices located at $\mathbf{x}^n + \mathbf{u}^n\,\Delta t$.

4. <u>Internal Energy Equation.</u> The Lagrangian phase finite-difference approximation to the internal energy equation is the following:

$$\begin{aligned}\frac{M_{ijk}^B I_{ijk}^B - M_{ijk}^n I_{ijk}^n}{\Delta t} = &- \frac{p_{ijk}^n + p_{ijk}^B}{2}\,\frac{V_{ijk}^B - V_{ijk}^n}{\Delta t} + (1 - A_0)\,[\phi_D\,\sigma(\mathbf{u}^B){:}\nabla\mathbf{u}^B \\ &+ (1 - \phi_D)\,\sigma(\mathbf{u}^n){:}\nabla\mathbf{u}^n]_{ijk} V_{ijk}^n + \sum_\alpha K_\alpha^n\,\nabla[\phi_D T^B + (1 - \phi_D)\,\tilde{T}]_\alpha \cdot \mathbf{A}_\alpha^n \\ &+ \sum_\alpha (\rho D)_\alpha^n \left\{ \sum_m h_m(T_\alpha^n)\,\nabla[\phi_D Y_m^B + (1 - \phi_D) Y_m^A]_\alpha \right\} \cdot \mathbf{A}_\alpha^n \\ &+ A_0 M_{ijk}^B \varepsilon_{ijk}^A + V_{ijk}^n(\dot{Q}_{ijk}^c + \dot{Q}_{ijk}^s)\,.\end{aligned} \tag{89}$$

The temperature used for the heat conduction calculation is a weighted average, using variable implicitness parameter ϕ_D, of the Phase B temperature T^B and an intermediate temperature $\tilde{T}$ that we define shortly.

The Phase A internal energy I^A contains changes due to chemical heat release and the spray energy source:

$$\frac{M^A_{ijk}I^A_{ijk} - M^n_{ijk}I^n_{ijk}}{\Delta t} = V^n_{ijk}(\dot{Q}^c_{ijk} + \dot{Q}^s_{ijk}) \ . \tag{90}$$

The chemical source $\dot{Q}_{ijk}{}^c$ is given by Eq. (21) with $\dot{\omega}_r$ replaced by $(\dot{\omega}_r{}^A)_{ijk}$. The method for approximating $(\dot{\omega}_r{}^A)_{ijk}$ is described in Appendices I and J, and the difference formula for $\dot{Q}^s$ is given in Appendix C.

The temperature $\tilde{T}$ used in the diffusion term differencing is based on an internal energy that also includes updates due to enthalpy diffusion and turbulence dissipation:

$$M^B_{ijk}\frac{I^t_{ijk} - I^A_{ijk}}{\Delta t} = A_0 M^B_{ijk}\varepsilon^A_{ijk} + \sum_\alpha (\rho D)^n_\alpha \left\{ \sum_m h_m(T^n_\alpha)\nabla[\phi_D Y^B_m + (1-\phi_D)Y^A_m]_\alpha \right\} \cdot \mathbf{A}^n_\alpha \ . \tag{91}$$

To calculate $\tilde{T}$ it is assumed that all heat addition up to this point in the computational cycle has occurred at constant pressure. Thus the temperature change is related to the enthalpy change by

$$h^t_{ijk} = h^n_{ijk} + (c_p)^n_{ijk}(\tilde{T}_{ijk} - T^n_{ijk}) \ , \tag{92}$$

where

$$M^B_{ijk}h^t_{ijk} - M^n_{ijk}h^n_{ijk} = M^B_{ijk}I^t_{ijk} - M^n_{ijk}I^n_{ijk} \ . \tag{93}$$

By substituting for $h_{ijk}{}^t$ in (93) using (92), solving for $\tilde{T}_{ijk}$, and using the fact that $h_{ijk}{}^n = I_{ijk}{}^n + p_{ijk}{}^n/\rho_{ijk}{}^n$, we obtain the following equation for $\tilde{T}_{ijk}$:

$$\tilde{T}_{ijk} = T^n_{ijk} + \frac{1}{(c_p)^n_{ijk}}\left[I^t_{ijk} - I^n_{ijk} + p^n_{ijk}\left(\frac{1}{\rho^A_{ijk}} - \frac{1}{\rho^n_{ijk}} \right) \right] \ . \tag{94}$$

By using (89) - (91) we can derive

$$M^B_{ijk}\frac{I^B_{ijk} - I^t_{ijk}}{\Delta t} = -\frac{p^n_{ijk} + p^B_{ijk}}{2}\,\frac{V^B_{ijk} - V^n_{ijk}}{\Delta t} + (1 - A_0)[\phi_D\sigma(\mathbf{u}^B):\nabla\mathbf{u}^B + (1-\phi_D)\sigma(\mathbf{u}^n):\nabla\mathbf{u}^n]$$

$$+ \sum_\alpha K^n_\alpha \nabla[\phi_D T^B + (1-\phi_D)\tilde{T}]_\alpha \cdot \mathbf{A}^n_\alpha \ , \tag{95}$$

an equation that is solved in Phase B and will be referred to when we describe the solution procedure. For the differencing of the viscous stresses σ and the velocity gradients $\nabla \mathbf{u}$ in (95), the reader is referred to Appendix K.

5. Turbulence Equations. The Lagrangian phase difference approximation to the turbulent kinetic energy equation (7) is

$$\frac{M^B_{ijk} k^B_{ijk} - M^n_{ijk} k^n_{ijk}}{\Delta t} = -\tfrac{2}{3}\rho^B_{ijk} \frac{V^B_{ijk} - V^n_{ijk}}{\Delta t} [(1 - f_{ijk}) k^n_{ijk} + f_{ijk} k^B_{ijk}] + (VD)_{ijk}$$

$$+ \sum_{\alpha} \frac{\mu^n_\alpha}{Pr_k} \nabla [\phi^D k^B + (1 - \phi^D) k^A]_\alpha \cdot \mathbf{A}_\alpha - M^B_{ijk} \frac{\varepsilon^n_{ijk}}{k^n_{ijk}} k^B_{ijk} + (\dot{W}^s)_{ijk} V^n_{ijk} \qquad (96)$$

The difference approximation to the dissipation rate equation (8) is

$$\frac{M^B_{ijk} \varepsilon^B_{ijk} - M^n_{ijk} \varepsilon^n_{ijk}}{\Delta t} = -(\tfrac{2}{3} c_{\varepsilon_1} - c_{\varepsilon_3}) \rho^B_{ijk} \frac{V^B_{ijk} - V^n_{ijk}}{\Delta t} [(1 - f_{ijk}) \varepsilon^n_{ijk} + f_{ijk} \varepsilon^B_{ijk}]$$

$$+ \sum_{\alpha} \frac{\mu^n_\alpha}{Pr_\varepsilon} \nabla [\phi^D \varepsilon^B + (1 - \phi^D) \varepsilon^A]_\alpha \cdot \mathbf{A}_a + c_{\varepsilon_1} \frac{\varepsilon^n_{ijk}}{k^n_{ijk}} (VD)_{ijk}$$

$$- c_{\varepsilon_2} M^B_{ijk} \frac{\varepsilon^n_{ijk}}{k^n_{ijk}} \varepsilon^B_{ijk} + c_s (\dot{W}^s)_{ijk} V^n_{ijk} \frac{\varepsilon^A_{ijk}}{k^n_{ijk}} \; . \qquad (97)$$

The quantity f_{ijk} is zero or unity depending on the sign of the cell volume change $V_{ijk}{}^B - V_{ijk}{}^n$:

$$f_{ijk} = \begin{cases} 1 & \mathit{if}\ V^B_{ijk} - V^n_{ijk} > 0 \\ 0 & \mathit{otherwise}\ . \end{cases} \qquad (98)$$

This prescription for f_{ijk} is chosen to avoid negative computed values of k and ε when there are large volume changes during the Lagrangian phase.

Viscous dissipation of mean flow kinetic energy is represented by the term $(VD)_{ijk}$:

$$(VD)_{ijk} = V^n_{ijk} [\phi_D \sigma(\mathbf{u}^B) : \nabla \mathbf{u}^B + (1 - \phi^D) \sigma(\mathbf{u}^n) : \nabla \mathbf{u}^n]_{ijk} \; . \qquad (99)$$

The difference approximations to the viscous stresses σ and velocity gradients $\nabla \mathbf{u}$ are given in Appendix K.

In differencing the diffusion term, we use a weighted averaged of the Phase A values k^A and ε^A and the Phase B values k^B and ε^B. The amount of implicitness is specified by the parameter ϕ^D, which is chosen in a manner described in Appendix H.

The Phase A values k^A and ε^A differ from the time n values because they include the effects of spray source term $\dot{W}^s$:

$$\frac{M^B_{ijk} k^A_{ijk} - M^n_{ijk} k^n_{ijk}}{\Delta t} = (\dot{W}^s)_{ijk} V^n_{ijk} , \tag{100}$$

and

$$\frac{M^B_{ijk} \varepsilon^A_{ijk} - M^n_{ijk} \varepsilon^n_{ijk}}{\Delta t} = c_s (\dot{W}^s)_{ijk} V^n_{ijk} \frac{\varepsilon^A_{ijk}}{k^n_{ijk}} . \tag{101}$$

6. Volume Change Equation and Equations of State. The volume change of a computational cell in the Lagrangian phase is computed using the Phase B cell face normal velocities:

$$V^B_{ijk} = V^n_{ijk} + \Delta t \sum_{\alpha} (uA)^B_{\alpha} . \tag{102}$$

This approximates the following equation for volume change of volume V moving with the fluid:

$$\frac{DV}{Dt} = \int_S \mathbf{u} \cdot d\mathbf{A} , \tag{103}$$

which can be derived using the Reynolds transport theorem.[50]

The equations of state (10) and (11) are approximated by

$$p^B_{ijk} = \left[\sum_m (\rho_m)^B_{ijk} / W_m \right] R_0 T^B_{ijk} \tag{104}$$

and

$$I^B_{ijk} = I^t_{ijk} + (c_v)^t_{ijk}(T^B_{ijk} - T^t_{ijk}) \; , \qquad (105)$$

where $I_{ijk}{}^t$ is given by Eq. (91) and $T_{ijk}{}^t$ and $(c_v)_{ijk}{}^t$ satisfy

$$I^t_{ijk} = \sum_m (Y_m)^B_{ijk} I_m(T^t_{ijk})$$

and

$$(c_v)^t_{ijk} = \sum_m (Y_m)^B_{ijk} c_{v_m}(T^t_{ijk}) \; . \qquad (106)$$

7. Droplet Equations. The Phase A calculation of droplet positions is defined by

$$\mathbf{x}^A_p = \mathbf{x}^n_p + \Delta t\, \mathbf{v}^n_p + \delta\mathbf{x}'_p \; , \qquad (107)$$

where $\delta\mathbf{x}'_p$ is a random position change that is added when timestep Δt exceeds the particle turbulence correlation time t_{turb} given by Eq. (37) with all quantities evaluated at time level n. The displacement $\delta\mathbf{x}'_p$, and an associated velocity change $\delta\mathbf{v}'_p$, are chosen from distributions given in Appendix G. The particle then exchanges mass, momentum, and energy with the gas in the computational cell in which $\mathbf{x}_p{}^A$ is located.

The Phase A calculation of the remaining particle quantities is accomplished by splitting or differencing sequentially the terms associated with each physical effect, always using the most recently obtained droplet properties when calculating changes due to the next physical effect. The random velocity changes $\delta\mathbf{v}'_p$ are first added to the particle velocities $\mathbf{v}_p{}^n$. Then the droplet oscillation and breakup calculation is performed (cf. Appendix F), followed by the droplet collision calculation (cf. Appendix E). The Phase A calculation is completed with the updates of particle radii and temperatures due to evaporation (cf. Appendix C) and the addition of gravitational acceleration terms to the particle velocities.

The only particle properties that are altered in Phase B are the particle velocities. The calculation of $\mathbf{v}_p{}^B$ is intimately connected with the evaluation of the terms $\mathbf{R}'_{ijk}$ and S'_{ijk}, and hence is described in Appendix C.

None of the particle properties are altered in Phase C.

F. Solution Procedure for Implicit Phase B Equations

The Phase B values of the flow field variables are found by solving the implicit equations of the previous section. The solution procedure is patterned after the SIMPLE method,[19] with individual equations solved using the conjugate residual method.[20] In this section we describe the solution procedure and give the roles of the various subroutines that are involved.

Basically, the SIMPLE method is a two-step iterative procedure. After selecting a predicted value of the Phase B pressure p^B, in step 1 we freeze the **predicted** pressure field and solve for other flow quantities using finite difference equations that difference the diffusion terms implicitly. In the original SIMPLE method, the convection terms are also differenced implicitly, and their effects are included in step 1. In KIVA, convection is calculated in Phase C in a subcycled explicit fashion that offers some significant advantages over implicit methods. In step 2, we freeze the values of the diffusion terms obtained in step 1 and solve for the **corrected** pressure field using equations that difference pressure terms implicitly. Sometimes a Poisson equation for the pressure is derived and solved in step 2. In KIVA, for step 2 we simultaneously solve the cell-face velocity equations, the volume change equations, and a linearized form of the equation of state. By algebraically eliminating the volumes and cell-face velocities from these equations in favor of the pressures, one can show we are also solving a Poisson equation for the pressure in step 2. Following step 2, the **predicted** and **corrected** pressures are compared. If they agree to within a specified convergence tolerance, the equations have been solved, and we proceed to Phase C. If the difference between the pressure fields exceeds the convergence tolerance, the corrected pressure field becomes the new predicted pressure field, and we return to step 1 and repeat the process. Each pass through the two steps will be called an outer iteration.

In Ref. 54 it is argued that one should be able to stop after a small, predetermined number N of outer iterations and have a sufficiently accurate solution of the equations. Such a solution procedure is noniterative and therefore very attractive, but we have found the argument to be flawed. The argument is based on the fact that each outer iteration increases by one order in Δt the accuracy of the computed approximations to the exact solution of the finite difference equations. By stopping after N outer iterations, one introduces temporal truncation errors whose formal orders are equal to or greater than other truncation errors of the difference equations. We have found through computational experience, however, that it is better to iterate to a prescribed convergence tolerance. Although the errors incurred by stopping after N iterations may formally be of high order, they can be unacceptably large in practice.

Because the equations for $(Y_m)^B$, k^B, and ε^B are weakly coupled to the flow field solution, these equations are not included in the outer iteration. The mass fractions $(Y_m)^B$ are used in the calculation of the Phase B pressure p^B in Eq. (104), but the values of the Phase B pressures and velocities do not influence the solution of the $(Y_m)^B$ through

Eq. (77). Thus Eq. (77) is solved for the $(Y_m)^B$ in subroutine YSOLVE before beginning the outer iteration. This results in a considerable computational time savings over schemes, such as those that calculate implicit convection in step 1, that include the mass fraction equations in the outer iteration. We often have ten or more chemical species in our applications, and to solve equations for all these in the outer iteration would greatly increase computational times.

In the cases of k^B and ε^B, the flow field influences their values through the turbulence production terms in Eqs. (96) and (97), but the values of k^B and ε^B do not enter into the flow field equations. Thus Eqs. (96) and (97) are solved after completion of the outer iteration in subroutine KESOLVE. The finite-difference equations have been designed, of course, to give this one-way coupling. Mathematically, the values of k and ε influence the flow through the turbulent diffusivity and the turbulent pressure $\frac{2}{3}\rho k$. This coupling could be accounted for by using Phase B values of k and ε to evaluate the turbulent diffusivity and $\frac{2}{3}\rho k$; but this would greatly increase computational times, is not necessary for stability, and is usually not necessary for accuracy when timesteps are used that satisfy the constraints of Sec. III.H.

Thus the only equations in the outer iteration are the momentum equation, internal energy equation, and the pressure equation. We now describe in more detail the outer iteration. The predicted pressure ${p_{ijk}}^P$ is first initialized by linear extrapolation using the Phase B pressures from the previous two cycles:

$$p^P_{ijk} = (p^B_{ijk})^{n-1} + \frac{\Delta t^n}{\Delta t^{n-1}} [(p^B_{ijk})^{n-1} - (p^B_{ijk})^{n-2}] \; . \qquad (108)$$

This extrapolation has been found to reduce both the number of outer iterations and the number of iterations required to solve the pressure equation.

The first equation solved in step 1 is the momentum equation (78). The predicted pressures ${p_{ijk}}^P$ are used in place of the unknown Phase B pressures, and predicted velocity field $\mathbf{u}_{ijk}{}^P$ is solved for in place of the Phase B velocities. This calculation is performed in subroutine VSOLVE.

The predicted temperature field is next found by using a combination of Eqs. (95), (105), (104), and (76). Equation (105) is used to eliminate I^B in favor of T^B in (95). Equations (104) and (76) are combined to give

$$p^B_{ijk} = \frac{M^B_{ijk}}{V^B_{ijk}} R_o \left[\sum_m \frac{(Y_m)^B_{ijk}}{W_m} \right] T^B_{ijk} \; , \qquad (109)$$

which is then used to eliminate ${V_{ijk}}^B$ from (95). After some rearrangement and replacement of superscripts B with superscripts p to denote we are solving for predicted values, we obtained the following equation for the predicted temperature field:

$$T^p_{ijk} = \left\{ T^t_{ijk} + \frac{p^n_{ijk} + p^p_{ijk}}{2(c_v)^t_{ijk}} \frac{V^n_{ijk}}{M^B_{ijk}} + \frac{\Delta t}{M^B_{ijk}(c_v)^t_{ijk}} \left[\sum_{\alpha} K^n_{\alpha} \nabla [\phi_D T^p + (1 - \phi_D) \tilde{T}]_{\alpha} \cdot \mathbf{A}^n_{\alpha} \right.\right.$$

$$\left.\left. + (1 - A_o)[\phi_D \sigma(\mathbf{u}^p) : \nabla \mathbf{u}^p + (1 - \phi_D) \sigma(\mathbf{u}^n) : \nabla \mathbf{u}^n|_{ijk} \right] \right\}$$

$$/ \left\{ 1.0 + \frac{p^n_{ijk} + p^p_{ijk}}{2(c_v)^t_{ijk} p^p_{ijk}} R_o \left[\sum_m \frac{(Y_m)^B_{ijk}}{W_m} \right] \right\} . \qquad (110)$$

This equation is solved in subroutine TSOLVE. After solving for the predicted temperatures, the predicted cell volumes ${V_{ijk}}^p$ are found from (109) with superscripts p again replacing the superscripts B:

$$V^p_{ijk} = \frac{M^B_{ijk}}{p^p_{ijk}} R_o \left[\sum_m \frac{(Y_m)^B_{ijk}}{W_m} \right] T^p_{ijk} . \qquad (111)$$

Finally, we solve for the corrected pressure field ${p_{ijk}}^c$. This involves simultaneous solution of Eq. (86) for the cell face velocities, Eq. (102) for the corrected cell volumes, and a linearized, isentropic equation of state that relates the corrected pressures and volumes. In Eq. (86) the corrected pressures are used on the right-hand side of the equation in place of p^B. In (102) the corrected volume ${V_{ijk}}^c$ is used in place of ${V_{ijk}}^B$. The equation of state is obtained by combining (109) and (110), neglecting the diffusion and dissipation terms in (110), and linearizing about the predicted cell volumes and pressures. The result is

$$V^c_{ijk} = V^p_{ijk} - \frac{1}{\gamma^t_{ijk}} \frac{V^p_{ijk}}{p^p_{ijk}} (p^c_{ijk} - p^p_{ijk}) , \qquad (112)$$

where

$$\frac{1}{\gamma^t_{ijk}} = \frac{2(c_v)^t_{ijk} + R^o \left[\sum_m \frac{(Y_m)^B_{ijk}}{W_m} \right] \left(1 - \frac{V^n_{ijk}}{V^p_{ijk}} \right)}{2(c_v)^t_{ijk} + \frac{p^n_{ijk} + p^p_{ijk}}{p^p_{ijk}} R^o \left[\sum_m \frac{(Y_m)^B_{ijk}}{W_m} \right]} . \qquad (113)$$

The corrected pressures are found in subroutine PSOLVE.

The predicted and corrected pressure fields are then compared to see if convergence has been attained. The outer iteration is judged to have converged if

$$|p^p_{ijk} - p^c_{ijk}| < \varepsilon_o \left\{ \left[\underset{lmn}{Max} \, (p^c_{\ell mn}) - \underset{lmn}{Min} \, (p^c_{\ell mn}) \right] + 10^{-10} \left[\underset{lmn}{Max} \, (p^c_{\ell mn}) + \underset{lmn}{Min} \, (p^c_{\ell mn}) \right] \right\} \tag{114}$$

for all cells (i,j,k). The convergence tolerance ε_o is typically taken to be 0.10. If (114) is not satisfied in every cell, then the corrected pressure field becomes the new predicted pressure field, and we return to step 1.

If convergence has been attained, then $p_{ijk}{}^B$ is set equal to $p_{ijk}{}^C$, $\mathbf{u}_{ijk}{}^B$ is found from (78) with the most recently predicted velocity field used to evaluate the viscous stresses, and $I_{ijk}{}^B$ is found from (95) with the most recently predicted velocities and temperatures used to evaluate the dissipation and diffusion terms. Finally, the Phase B vertex positions are given by

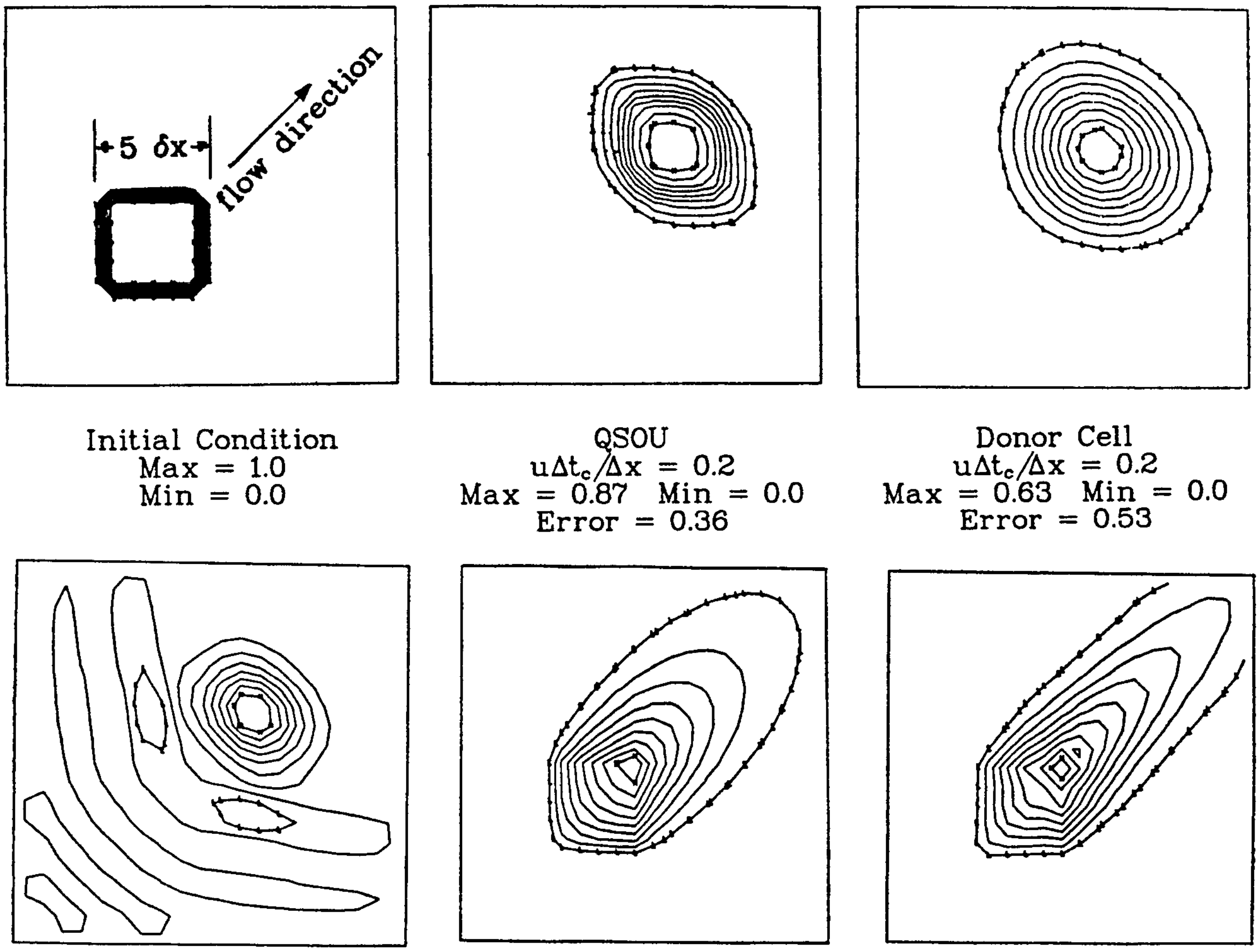
$$\mathbf{x}^B_{ijk} = \mathbf{x}^n_{ijk} + \mathbf{u}^B_{ijk} \Delta t \, . \tag{115}$$

G. Phase C

Phase C is the rezone phase, in which we calculate the convective transport associated with moving the mesh relative to the fluid. This is accomplished in a subcycled, explicit calculation using a timestep Δt_c that is an integral submultiple of the main computational timestep Δt. The timestep Δt_c must satisfy the Courant condition $u_r \Delta t_c / \Delta x < 1$, where u_r is the fluid velocity relative to the grid velocity, but because there is no upper bound on the number of subcycles, the code can run with $u_r \Delta t / \Delta x > 1$. The exact formula for Δt_c is given in the next section. Convective subcycling saves computational time because the rezone calculation takes only about ten percent of the time of the Lagrangian phase calculation.

The user of KIVA-II has the option of using one of two convection schemes: quasi-second-order upwind (QSOU) differencing, described in Appendix M, and partial donor cell (PDC) differencing, described in Appendix N. In addition, when using PDC differencing, the user can vary the amount of "upwinding" through two parameters α_o and β_o. When $\alpha_o = 1$ and $\beta_o = 0$, donor cell or full upwind differencing is used. When $\alpha_o = 0$ and $\beta_o = 1$, interpolated donor cell differencing, or Leith's method,[55] is used. The algorithms are fully described in the appendices, but here we will demonstrate some of their properties through a computational example. Because of poor resolution, the example problem is a severe test of the convection schemes and exaggerates many of their shortcomings.

In the example we convect a scalar field through a two-dimensional mesh of square cells with a uniform velocity directed at a 45 degree angle to the mesh directions. The initial conditions are plotted in Fig. 6. The scalar field is initially unity on a square that has five cells to a side and is zero otherwise. Also shown in Fig. 6 are the results for five different convection schemes after one timestep Δt such that $u\Delta t/\Delta x = 5$, where u is one component of the velocity. The five schemes are subcycled explicit QSOU, donor cell, and interpolated donor cell, each using $u\Delta t_c/\Delta x = 0.2$, and fully implicit donor cell and the QUICK scheme,[56] both using $u\Delta t_c/\Delta x = u\Delta t/\Delta x = 5$. The latter two schemes are not available in KIVA-II, but are included in the examples in order to illustrate the accuracy of commonly used implicit schemes. Given in Fig. 6 for each convection scheme are contour plots of the final computed scalar field, the maximum and minimum computed



Initial Condition
Max = 1.0
Min = 0.0

QSOU
$u\Delta t_c/\Delta x = 0.2$
Max = 0.87 Min = 0.0
Error = 0.36

Donor Cell
$u\Delta t_c/\Delta x = 0.2$
Max = 0.63 Min = 0.0
Error = 0.53

Interpolated Donor Cell
$u\Delta t_c/\Delta x = 0.2$
Max = 1.41 Min = −0.35
Error = 0.41

Implicit Donor Cell
$u\Delta t_c/\Delta x = 5.0$
Max = 0.51 Min = 0.0
Error = 0.83

Implicit QUICK
$u\Delta t_c/\Delta x = 5.0$
Max = 0.66 Min = −0.05
Error = 0.75

Fig. 6. Isopleths from calculations of convection of a square-shaped scalar profile.

values, and the root-mean-square error between the computed solution and the exact solution, which is just a uniform translation five cells in each direction. The error is averaged over the twenty-five cells where the scalar field is unity in the exact solution.

The most accurate method is QSOU, even though it has first-order spatial truncation errors and therefore is formally less accurate than interpolated donor cell, which is second-order accurate in space. The least accurate methods are the implicit methods. This illustrates the fact that although the implicit methods are unconditionally stable, they can have large errors when $u\Delta t/\Delta x > 1$.

A monotone scheme has the property that it introduces no new mimima or maxima in the computed solution. The donor cell and QSOU schemes are monotone, and thus their computed maxima and minima lie between those of the initial conditions. Interpolated donor cell and QUICK are not monotone. Computational oscillations in regions of steep gradients are obtained with nonmonotone schemes and can be especially pronounced in calculations using pure interpolated donor cell.

Although PDC differencing is less accurate than QSOU, we retain it as an option in KIVA-II because it is significantly faster than the QSOU scheme. We recommend use of PDC differencing in calculations in which speed is more important than accuracy or in calculations in which cell Reynolds numbers are less than two. In the latter case physical diffusion is large enough to render negligible the numerical errors associated with PDC differencing. When using PDC differencing, we recommend taking $\beta_o = 1.0$ and $\alpha_o = 0.1$ or 0.2 to suppress computational oscillations. The QSOU scheme should be used to obtain the most accurate calculation for a given mesh resolution. Both schemes can be used in separate calculations of the same problem, as a partial test of convergence. If the computed solutions are the same, then numerical errors associated with convection are small.

We now describe the convection calculation in more detail. The transport of cell-centered quantities is computed by using a volume δV_α that is swept out by regular cell face α each convective subcycle, as it moves from its Lagrangian position (defined by the corner positions $\mathbf{x}_{ijk}{}^{B}$) to its final position (defined by the user-specified $\mathbf{x}_{ijk}{}^{n+1}$). It is convenient to associate δV_α with both the face and the regular cell under consideration, so that δV_α is positive if the volume of the cell in question is increased by moving face α from its Lagrangian to its final position. Thus, for a given face, the sign of δV_α depends on the side of the face from which it is viewed. This convention is entirely analogous to that adopted in subsection B above for the area vectors $\mathbf{A}_\alpha$. The δV_α are evaluated in terms of the cell-face velocities and the old- and new-time grid positions:

$$\delta V_\alpha = \delta V_\alpha^G \, \Delta t_c / \Delta t - (uA)_\alpha^B \, \Delta t_c \, . \tag{116}$$

In (116), $\delta V_\alpha{}^G$ is the volume swept out by cell face α when the four vertices defining the face are moved from their old-time positions $x_{ijk}{}^n$ to their new-time positions $x_{ijk}{}^{n+1}$. The $\delta V_\alpha{}^G$,which are computed using Eqs. (56) and (57), are positive if the volume of cell (i,j,k) is increased by the grid motion. It can be verified using Eq. (102) that the δV_α satisfy

$$V^B_{ijk} + NS \sum_\alpha \delta V_\alpha = V^{n+1}_{ijk} \ , \tag{117}$$

where $NS = \Delta t/\Delta t_c$ is the number of convective subcycles.

The species densities $(\rho_m)_{ijk}{}^v$ after v convective subcycles are given by

$$(\rho_m)^v_{ijk} V^v_{ijk} = (\rho_m)^{v-1}_{ijk} V^{v-1}_{ijk} + \sum_\alpha (\rho_m)^{v-1}_\alpha \delta V_\alpha , \tag{118}$$

where the summation is over the faces of cell (i,j,k). The species densities are initialized at the beginning of Phase C by their Phase B values

$$(\rho_m)^0_{ijk} = (\rho_m)^B_{ijk} \ , \tag{119}$$

and the subcycle volumes $V_{ijk}{}^v$ are given by

$$V^v_{ijk} = [v\, V^{n+1}_{ijk} + (NS - v) V^B_{ijk}] / NS \ , \tag{120}$$

where NS is the number of convective subcycles and $V_{ijk}{}^{n+1}$ is the cell volume based on the final coordinates. The cell face densities $(\rho_m)_\alpha{}^v$ are evaluated either by the quasi-second-order upwind scheme described in Appendix M or by the partial donor cell procedure described in Appendix N. The total density after v convective subcycles is given by

$$\rho^v_{ijk} = \sum_m (\rho_m)^v_{ijk} \ , \tag{121}$$

and the vertex masses after v subcycles are found from Eqs. (66) and (67) using these total densities of (121) and the volumes of Eq. (120).

The specific internal energy $I_{ijk}{}^v$ after v convective subcycles is determined from

$$\rho^v_{ijk} V^v_{ijk} I^v_{ijk} = \rho^{v-1}_{ijk} V^{v-1}_{ijk} I^{v-1}_{ijk} + \sum_\alpha (\rho I)^{v-1}_\alpha \delta V_\alpha \ , \tag{122}$$

where the cell face energy densities $(\rho I)_\alpha{}^v$ are evaluated either by quasi-second-order upwind differencing (Appendix M) or by partial donor cell differencing (Appendix N).

The formula for updating turbulence quantity $q_{ijk}{}^v$ in the subcycle is

$$\rho^{v}_{ijk} V^{v}_{ijk} q^{v}_{ijk} = \rho^{v-1}_{ijk} V^{v-1}_{ijk} q^{v-1}_{ijk} + \sum_{\alpha} (\rho q)^{v-1}_{\alpha} \delta V_{\alpha} , \tag{123}$$

where $q = k$ or $q = k^{3/2}/\varepsilon = L$. We convect turbulence length scale L rather than ε because ε generally has steeper gradients, and therefore larger numerical errors arise when convecting ε. The cell face quantities $(\rho q)_\alpha{}^v$ are evaluated either by quasi-second-order differencing (Appendix M) or by partial donor cell differencing (Appendix N) using the full donor cell limit ($\alpha_o = 1$, $\beta_o = 0$). Since partial donor cell differencing is not monotone except when $\alpha_o = 1$ and $\beta_o = 0$, this limit is used to ensure that negative values of k and L are not obtained in the convection phase.

After completion of all convective subcycles, the final values of cell-centered quantities are set equal to their values after the NS subcycles. The final value of temperature is computed by inverting Eq. (11) using final values of internal energy and mass densities. The final pressure is given by

$$p^{n+1}_{ijk} = R_o T^{n+1}_{ijk} \sum_{m} (\rho_m)^{n+1}_{ijk} / W_m . \tag{124}$$

Convective transport of momentum on subcycle v is calculated in terms of the mass increments across momentum cell faces, which are related to the mass increments across regular cell faces in the following way. The mass increment across cell face α of a particular momentum cell is defined by

$$(\delta M'_{\alpha})^{v-1} = \frac{1}{8} (\rho^{v-1}_{o} \delta V_{o} - \rho^{v-1}_{i} \delta V_{i}) , \tag{125}$$

where o and i are the regular cell faces on either side of the momentum cell face α (between which face α is "sandwiched"), of which i ("inner") is the one that actually cuts into the momentum cell in question, while face o ("outer") does not. As usual, it is convenient to associate $\delta M'_\alpha$ with both the face and the momentum cell in question. When face α is viewed from the other momentum cell to which it is common, o and i are interchanged and the sign of $\delta M'_\alpha$ reverses. One readily verifies that Eq. (125) is consistent with the definition of vertex masses, in the sense that

$$(M')^{\upsilon}_{ijk} = (M')^{\upsilon-1}_{ijk} + \sum_{\alpha} (\delta M'_{\alpha})^{\upsilon-1} , \tag{126}$$

where the summation extends over all faces of momentum cell (i,j,k).

Before fluxing momentum the mass increments $\delta M'_{\alpha}$ corresponding to the twenty-four faces of momentum cell (i,j,k) are added in groups to obtain mass fluxes δM_{β}^{c} through six composite faces β of the momentum cell. Each composite face is formed from the four cell faces α that touch a common regular cell edge that emanates from vertex (i,j,k). The momentum fluxing is then computed by

$$(M')^{\upsilon}_{ijk}\mathbf{u}^{\upsilon}_{ijk} = (M')^{\upsilon-1}_{ijk}\mathbf{u}^{\upsilon-1}_{ijk} + \sum_{\beta} (\delta M^{c}_{\beta})^{\upsilon-1}\mathbf{u}^{\upsilon-1}_{\beta} , \tag{127}$$

where the velocities $\mathbf{u}_{\beta}{}^{\upsilon-1}$ are evaluated by the quasi-second-order upwind scheme of Appendix M or the partial donor cell scheme of Appendix N.

Two special features are provided in conjunction with the convection calculation. The use of partial donor cell differencing sometimes results in the development of unphysical small negative species densities. We therefore provide a reapportionment algorithm which tends to preserve the positivity of these densities, as described in Appendix O. This algorithm is not needed when quasi-second-order differencing is used, since this scheme is monotone.

In an axisymmetric swirling flow with free-slip boundary conditions, the total angular momentum should be conserved. However, the KIVA difference approximations to the momentum equations simply conserve the three Cartesian components of momentum, and this does not imply angular momentum conservation because of truncation errors. In practice, we have found that the only such truncation errors that are significant are those arising from the rezone calculation of Phase C. For this reason, an optional angular momentum correction procedure has been included in Phase C, as described in Appendix P.

H. Accuracy Conditions and Automatic Timestep Control

The timesteps Δt and Δt_c are selected automatically at the beginning of each cycle. Because diffusion terms are differenced implicitly and convective terms are subcycled, there are no stability restrictions on Δt, but there are several accuracy conditions upon which the automatic selection of Δt is based. These will be given in this section. The convection timestep Δt_c must satisfy the Courant condition for stability, and we also describe how this is generalized to an arbitrary mesh.

The accuracy conditions we use to determine Δt cannot give a universally reliable selection because there are many accuracy conditions we have not taken into account. It has been our experience, however, that the criteria we use for determining Δt give temporally accurate solutions in most calculations. The user is cautioned that other accuracy conditions could be important in his application and the timestep should always be varied to test for temporal accuracy.

The first accuracy condition on Δt is that

$$\left| \frac{D\mathbf{u}}{Dt} \right| \Delta t^2 < f_a \Delta x \,, \tag{128}$$

where f_a is some positive real number of order unity and Δx is an average cell dimension. This condition arises because terms of order higher than Δt are ignored in Eq. (115). The accuracy constraint (128) is the only one we use in which the cell size Δx appears. We note that $\Delta t \sim \Delta x^{\frac{1}{2}}$ for condition (128), in contrast to explicit convective stability criteria, which give $\Delta t \sim \Delta x$, and explicit diffusional stability criteria, which give $\Delta t \sim \Delta x^2$. Thus while (128) will reduce the timestep as the mesh is refined, it will not do so as much as the two stability criteria, which had to be observed by the first version of KIVA.[1] Constraint (128) is implemented by calculating a timestep Δt_{acc}:

$$\Delta t_{acc}^{n+1} = \min_{(i,j,k)} \frac{f_a \Delta x_{i,j,k}}{|\mathbf{u}_{ijk}^{B} - \mathbf{u}_{ijk}^{n}|} \,, \tag{129}$$

where

$$\Delta x_{ijk} = [(|\mathbf{x}_1 - \mathbf{x}_4|^2 + |\mathbf{x}_2 - \mathbf{x}_3|^2 + |\mathbf{x}_3 - \mathbf{x}_4|^2 + |\mathbf{x}_2 - \mathbf{x}_1|^2 + |\mathbf{x}_5 - \mathbf{x}_1|^2 + |\mathbf{x}_8 - \mathbf{x}_4|^2)/6]^{\frac{1}{2}} \tag{130}$$

and the subscripts in (130) refer to the vertices of cell (i,j,k) as numbered in Fig. 2. We then constrain Δt^{n+1} to be less than $\Delta t_{acc}{}^{n+1}$, as described below. The default value of f_a is 0.5.

The second accuracy condition on Δt is that

$$|\lambda| \, \Delta t < f_r \,, \tag{131}$$

where f_r is of order unity and λ is an eigenvalue of the rate of strain tensor. This criterion limits the amount of cell distortion that can occur due to mesh movement in the Lagrangian phase. When cells become very distorted, the spatial accuracy of the difference approximations suffers. One example of how (131) works is the following. In a plane parallel shear flow, there is one nonzero eigenvalue $\frac{1}{2}\, \partial u / \partial y$, where u is the streamwise velocity component and y is the cross-stream direction. As depicted in Fig. 7 for a rectangu-

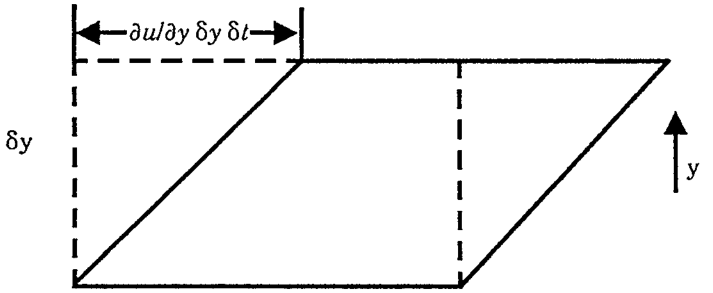


Fig. 7. Cell shapes before (dashed lines) and after (solid lines) the Lagrangian phase in a plane, parallel shear flow.

lar cell aligned with the flow, constraint (131) limits the distance upper and lower cell vertices can move relative to one another, divided by the cell height. Constraint (131) is enforced by calculating a timestep Δt_{rst}:

$$\Delta t^n_{rst} = \min_{(i,j,k)} \frac{f_r}{2\sqrt{a_{ijk}/3}} , \qquad (132)$$

where

$$a_{ijk} = \tfrac{1}{3}(p^2_{ijk} - 3q_{ijk}) \qquad (133)$$

and p_{ijk} and q_{ijk} are given in terms of the rate of strain tensor $s_{\ell m}$ in cell (i,j,k):

$$p_{ijk} = -s_{\ell\ell} \qquad (134)$$

$$q_{ijk} = \varepsilon_{1\ell m} s_{2\ell} s_{3m} + \varepsilon_{\ell 2m} s_{1\ell} s_{3m} + \varepsilon_{\ell m3} s_{1\ell} s_{2m} .$$

In (134) $\varepsilon_{\ell mn}$ is the alternating tensor,[57] and $s_{\ell m}$ is given by

$$s_{\ell m} = \frac{1}{2}\left(\frac{\partial u_\ell}{\partial x_m} + \frac{\partial u_m}{\partial x_\ell}\right)_{ijk} , \qquad (135)$$

where the velocity derivatives are evaluated using time level n velocities and the approximations of Appendix K. It can be verified from the formula for the roots of a cubic polynomial that the denominator of (132) is greater than the magnitudes of all eigenvalues of $s_{\ell m}$, and therefore if we select $\Delta t^n \leq \Delta t_{rst}{}^n$ the constraint (131) will be satisfied. Using $f_r = 1/\sqrt{3}$ has given sufficient accuracy in calculations we have made.

Two other accuracy criteria for Δt are obtained from the need to couple accurately the flow field and source terms due to chemical heat release and mass and energy exchange with the spray. For the chemical heat release, the requirement is that

$$\frac{V_{ijk}\dot{Q}^c_{ijk}}{M_{ijk}I_{ijk}}\Delta t < f_{ch} \ , \tag{136}$$

where f_{ch} is an input constant typically taken to be 0.1. Constraint (136) is the requirement that the total heat release from all chemical reactions in a cell should not exceed a small fraction f_{ch} of the total internal energy in the cell. To enforce (136) we calculate timestep Δt_{ch} by

$$\Delta t^{n+1}_{ch} = \min_{ijk} f_{ch} / \left| \frac{V^n_{ijk}(\dot{Q}^c)^n_{ijk}}{M^n_{ijk}I^n_{ijk}} \right| \tag{137}$$

and choose $\Delta t^{n+1} \leq \Delta t_{ch}{}^{n+1}$. This formula assumes that the fractional rate of energy release varies slowly from cycle to cycle. Spray timestep Δt_{sp} is calculated from a similar formula:

$$\Delta t^{n+1}_{sp} = \min_{ijk} \left\{ f_{ch} / \left| \frac{(\dot{\rho}^s)^n_{ijk}}{\rho^n_{ijk}} \right| , \ f_{ch} / \left| \frac{V^n_{ijk}(\dot{Q}^s)^n_{ijk}}{M^n_{ijk}I^n_{ijk}} \right| \right\} .$$

The main timestep used for cycle $n+1$ is then given by

$$\Delta t^{n+1} = \min\left(\Delta t^{n+1}_{acc}, \Delta t^{n+1}_{rst}, \Delta t^{n+1}_{ch}, \Delta t^{n+1}_{sp}, \Delta t^{n+1}_{gr}, \Delta t_{mx}, \Delta t_{mxca}\right) . \tag{138}$$

The timestep $\Delta t_{gr}{}^{n+1}$ limits the amount by which the timestep can grow:

$$\Delta t^{n+1}_{gr} = 1.02\,\Delta t^n \ . \tag{139}$$

Timesteps Δt_{mx} and Δt_{mxca} are, respectively, an input maximum timestep and a maximum timestep based on an input maximum crank angle in engine calculations.

The initial guess for Δt on cycle 0 is given by input quantity DTI. This is then compared with $\Delta t_{rst}{}^{o}$, Δt_{mx}, and Δt_{mxca} to determine the actual initial timestep Δt^{o}. (If the DTI supplied on a subsequent restart differs from the DTI at cycle 0, the current Δt will be reset to this new DTI.)

The convection timestep Δt_c is based on the Courant stability condition. In a rectangular mesh, this condition is

$$\Delta t_c \leq \min\left(\frac{\Delta x}{|u - b_x|} , \frac{\Delta y}{|v - b_y|} , \frac{\Delta z}{|w - b_z|} \right) , \tag{140}$$

where b_x, b_y, and b_z are the components of the grid velocity **b**. Constraint (140) limits the magnitude of the flux volume in any coordinate direction to a value less than the cell volume. To generalize this to an arbitrary mesh, it is natural to replace (140) by the similar criterion

$$\Delta t_c^n \leq \Delta t_c^{n-1} \min_{\alpha} \frac{V_{ijk}}{|\delta V_\alpha|} , \tag{141}$$

where the δV_α are the flux volumes calculated for cell (i,j,k) using timestep $\Delta t_c{}^{n-1}$. In practice, for accuracy we also reduce the timestep determined from (141) by a factor f_{con}, typically taken to be 0.2.

IV. THE COMPUTER PROGRAM

A. General Structure

The KIVA-II computer program consists of a set of subroutines controlled by a short main program. The general structure is illustrated in Fig. 8, showing a top-to-bottom flow encompassing the entire calculation. Beside each box in the flow diagram appears the name(s) of the primary subroutines(s) responsible for the associated task. In addition to the primary subroutines, Fig. 8 also identifies a number of supporting subroutines that perform tasks for the primaries. Comments at the beginning of each subroutine in the listing describe its purpose, where it is called from, and what subroutines or functions it calls, if any.

KIVA-II is an advanced experimental computer program, not a "black box" production code. Its use requires some knowledge of and experience with numerical fluid dynamics, chemistry, and spray modeling.

KIVA-II was written specifically for use on the CRI Cray family of computers, operating under the Cray Time Sharing System (CTSS) and using the Cray FORTRAN (CFT) and CFT77 compilers. We have several observations concerning our experience with the CFT compilers currently available to us:

- CFT 1.11 is the compiler of choice for short runs and scoping studies using KIVA-II. It will compile the program in about 20 s on a Cray X-MP, and the compiled code will run the baseline sample calculation in about 63 s.
- CFT 1.14 requires about 55 s to compile KIVA-II. The baseline calculation still requires 63 s to run. A principal feature of CFT 1.14 is vectorization of loops containing indirect addressing, utilizing hardware features of the Cray X-MP. While this is quite attractive for some of our codes, it is of minor consequence in KIVA-II, which contains very little indirect addressing.

- CFT77, written in Pascal rather than machine language, requires over 3 min to compile KIVA-II and is best reserved for longer runs in which one can recover the compilation time penalty. The baseline run time is down to about 52 s, 21% faster than with the CFT compilers. Clearly, it is advisable to create a library of relocatable binaries when using CFT 1.14 and CFT77, to minimize the time spent in the compiler.

At present, CFT77 is not a good compiler to use for debugging, as many variables reside in registers. Therefore, symbolic names are frequently inaccessible to the debugger. We understand that the CFT77 developers plan to back off somewhat in register utilization, so that future versions will be more compatible with interactive debuggers. As things now stand, developers of large codes are reluctant to use CFT77 because of this.

Most Cray users outside the Los Alamos-Livermore environment have available the Cray Operating System (COS) and will find KIVA-II generally compatible. The principle incompatibility with outside Cray systems lies in the calls that communicate with the operating system. The functions of all calls to system routines are described in the EPILOG at the end of the FORTRAN listing. WRITE (59,-) statements refer to the user's remote terminal.

Users who do not have a Cray computer face an additional task adapting KIVA-II to run on whatever computer they have. This is because KIVA-II contains statements peculiar to the CFT compilers that permit vectorization of many of the loops in the hydro subroutines. Our efforts at vectorization resulted in making the hydro portion of the code run nearly five times as fast. (Unfortunately, the chemistry and spray subroutines are not amenable to such treatment; this is because each cell in the chemistry and each droplet in the spray follows a unique logic path dependent upon local conditions.)

The unfamiliar statements in KIVA-II are the CFT vector merge functions CVMGT, CVMGP, CVMGM, and CVMGZ, and the vectorization directive CDIR$ IVDEP. The vector merge functions allow many loops to vectorize, in that they can replace IF tests, which do not vectorize. Briefly, the four vector merge functions perform as follows:

- CVMGT (X,Y,L) results in X if L is true, Y if L is not true.
- CVMGP (X,Y,P) results in X if P $\geq$ 0, Y if P $<$ 0.
- CVMGM (X,Y,M) results in X if M $<$ 0, Y if M $\geq$ 0.
- CVMGZ (X,Y,Z) results in X if Z is zero, Y if Z is nonzero.

CDIR$ IVDEP instructs CFT to ignore apparent vector dependencies or recursions. If an imagined recursion causes the loop not to be automatically vectorized by CFT, the IVDEP directive instructs the compiler to create vector code anyway. Note that this directive begins in column 1, which will cause it simply to be treated as a comment by other compilers.

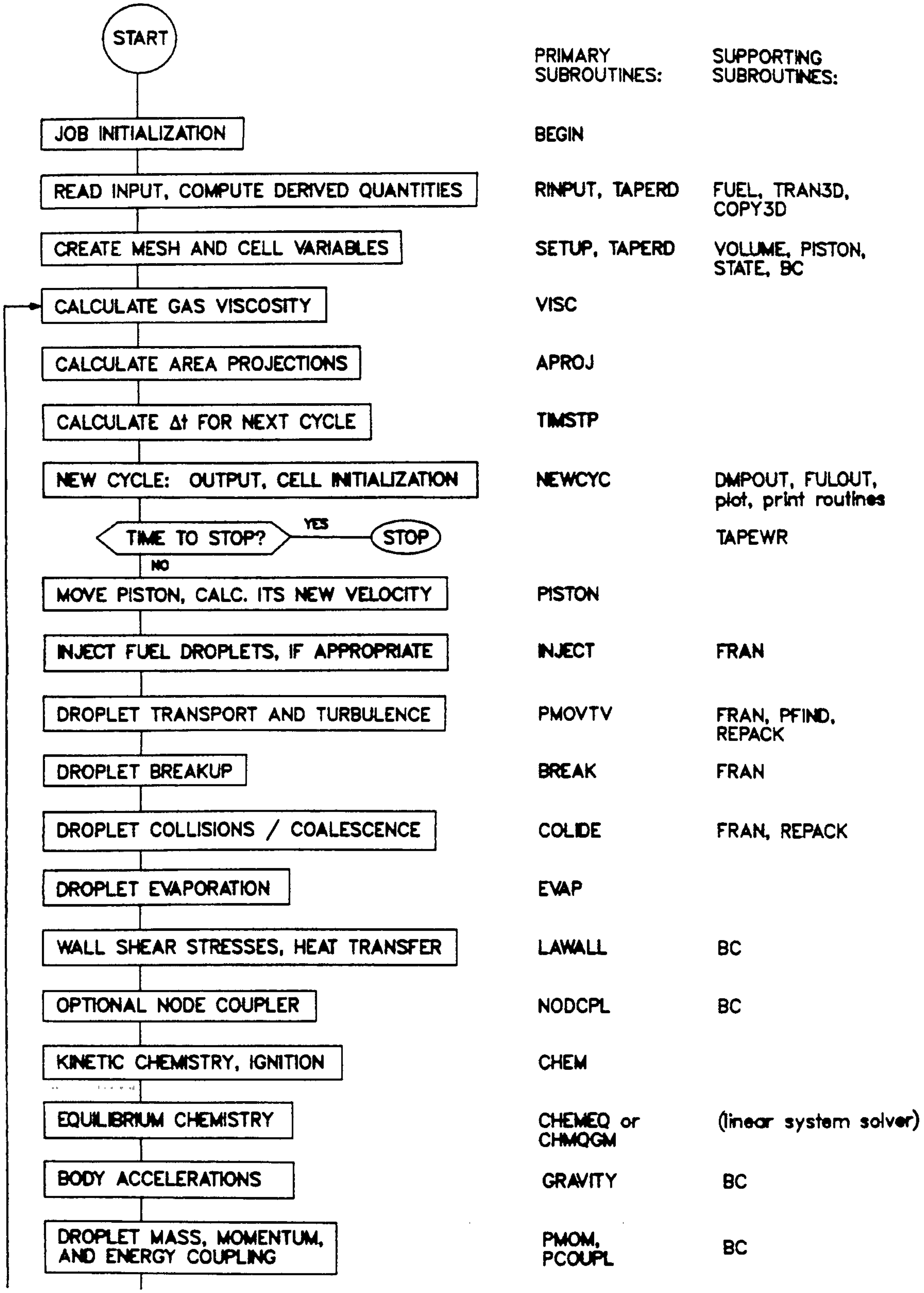

START
PRIMARY SUBROUTINES:
SUPPORTING SUBROUTINES:
JOB INITIALIZATION
BEGIN
READ INPUT, COMPUTE DERIVED QUANTITIES
RINPUT, TAPERD
FUEL, TRAN3D, COPY3D
CREATE MESH AND CELL VARIABLES
SETUP, TAPERD
VOLUME, PISTON, STATE, BC
CALCULATE GAS VISCOSITY
VISC
CALCULATE AREA PROJECTIONS
APROJ
CALCULATE Δt FOR NEXT CYCLE
TIMSTP
NEW CYCLE: OUTPUT, CELL INITIALIZATION
NEWCYC
DMPOUT, FULOUT, plot, print routines
TIME TO STOP?
YES
STOP
TAPEWR
NO
MOVE PISTON, CALC. ITS NEW VELOCITY
PISTON
INJECT FUEL DROPLETS, IF APPROPRIATE
INJECT
FRAN
DROPLET TRANSPORT AND TURBULENCE
PMOVTV
FRAN, PFIND, REPACK
DROPLET BREAKUP
BREAK
FRAN
DROPLET COLLISIONS / COALESCENCE
COLIDE
FRAN, REPACK
DROPLET EVAPORATION
EVAP
WALL SHEAR STRESSES, HEAT TRANSFER
LAWALL
BC
OPTIONAL NODE COUPLER
NODCPL
BC
KINETIC CHEMISTRY, IGNITION
CHEM
EQUILIBRIUM CHEMISTRY
CHEMEQ or CHMQGM
(linear system solver)
BODY ACCELERATIONS
GRAVITY
BC
DROPLET MASS, MOMENTUM, AND ENERGY COUPLING
PMOM, PCOUPL
BC

Fig. 8. General flow diagram for the KIVA-II program.

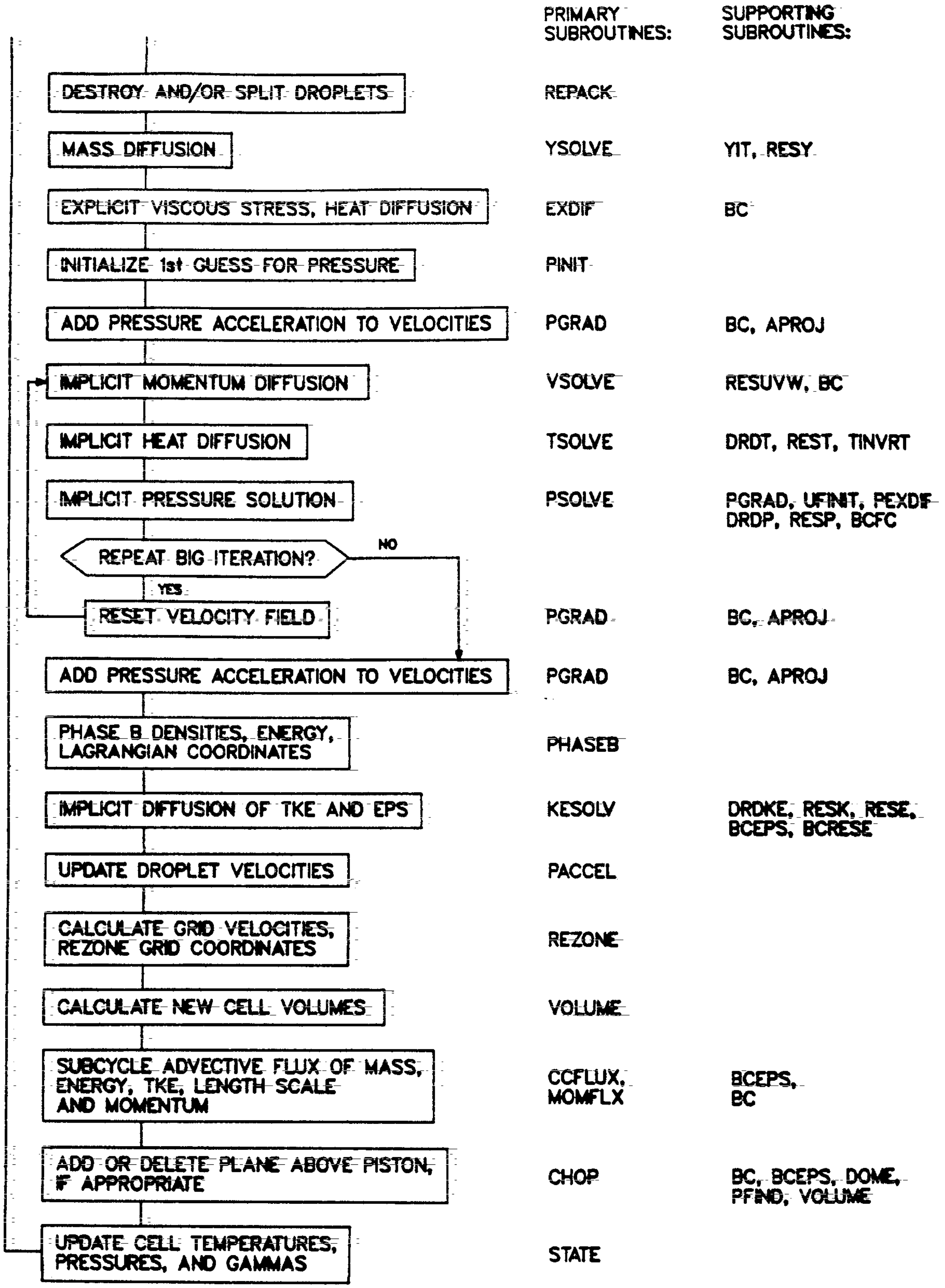


Fig. 8. continued

The non-Cray user can quickly modify the program by installing functions that emulate the vector merges. For example, CVMGT can be emulated by

```
FUNCTION CVMGT (X,Y,L)
LOGICAL L
IF (L) THEN
     CVMGT = X
     RETURN
          ELSE
          CVMGT = Y
          RETURN
ENDIF
END
```

Once the program is running properly, the functions should be replaced by in-line coding for greater efficiency.

Overall subroutine architecture also shows the effects of vectorization. For example, the user may ask why we break some logical task into a set of contiguous separate DO loops, when a single DO loop would appear to suffice. The reason is that excessively long DO loops will not vectorize because of optimization-block size limitations in some CFT compilers predating CFT77. It is not easy to define just how long "too long" is, as it depends not on the number of statements, but rather on the extent of computations involved. These cases were determined empirically; if there was no other reason why a long loop failed to vectorize, we broke it up and achieved vectorization.

The dimensions in the release version allow up to 12 species, 966 vertices, and 2000 computational spray particles. These dimensions may easily be altered via the PARAMETER statement (lines COMD.18 - COMD.19).

The input quantities always required to set up a problem are described in the EPILOG at the end of the listing and are read according to the formats appearing in subroutine RINPUT. The mesh generation is automated for a broad range of engine geometries, and is discussed in Sec. IV.E.

B. The Computing Mesh

The KIVA-II formulation is based on (x,y,z) Cartesian geometry and is applicable to cylindrical (CYL = 1.0) or planar (CYL = 0.0) calculations in either two or three space dimensions. The mesh is composed of a block of cells in logic space, NX cells in the i-direction, by NY cells in the j-direction, by NZ cells in the k-direction.

1. The Five Mesh Types Available.

Figure 9 shows the five mesh types available, determined by the specification of NY, CYL, JSECTR, and THSECT in the input data.

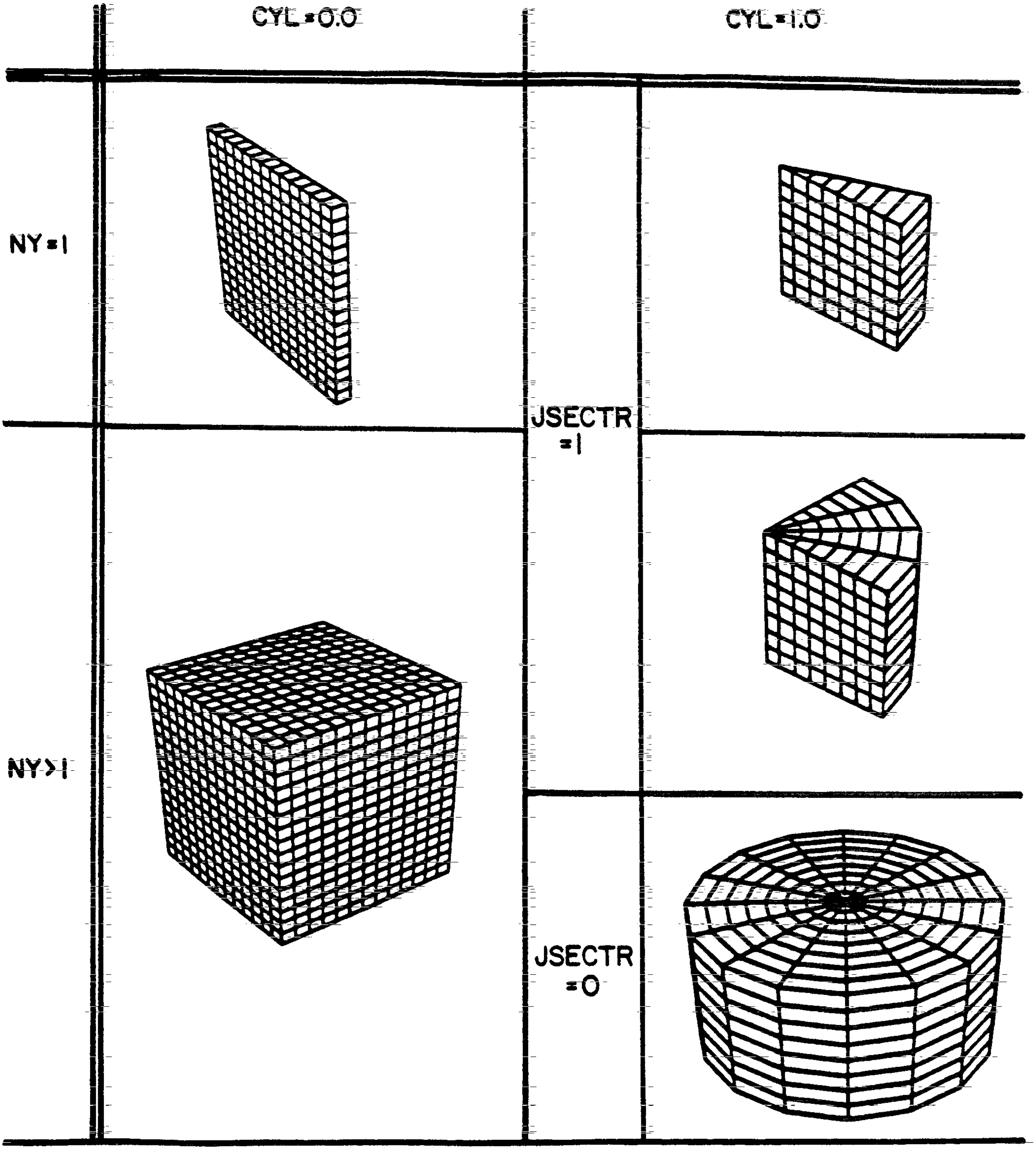


Fig. 9. The five mesh types available in KIVA-II.

The 2-D cylindrical option (NY =1, CYL = 1.0, JSECTR = 1) offers an efficient means of calculating fully symmetric cylindrical geometries. In three dimensions, the cylindrical case results in a pseudo-polar grid. If NY > 1 and JSECTR = 1, the resulting mesh is some sector of a full circle. Figure 10 illustrates a 72° sector, which has been applied to an engine geometry in which the on-axis fuel injector has 5 evenly-spaced nozzles, directed radially. The sector option allows us to model the 5-nozzle feature easily and far more efficiently than zoning a full 360° by taking advantage of the symmetry. The JSECTR =0 option is used when features of the geometry or spray require zoning the full 360°. As shown in Fig. 11, the Cartesian block of cells is curved around and joined to itself.

In all three cylindrical cases, the front and back (derriere) boundaries are periodic. The neighboring-cell relationships between cells facing the front and derriere boundaries are built into KIVA-II, and velocities are mass-averaged across corresponding points. The left boundary is shrunk to zero size to become the central axis, where our prescription at each axial level is to separately mass average each velocity component. For engine applications, the top boundary becomes the cylinder head, which may be flat or domed, and the bottom boundary is the moving piston face, which may be flat or contain a bowl for DISC or diesel designs.

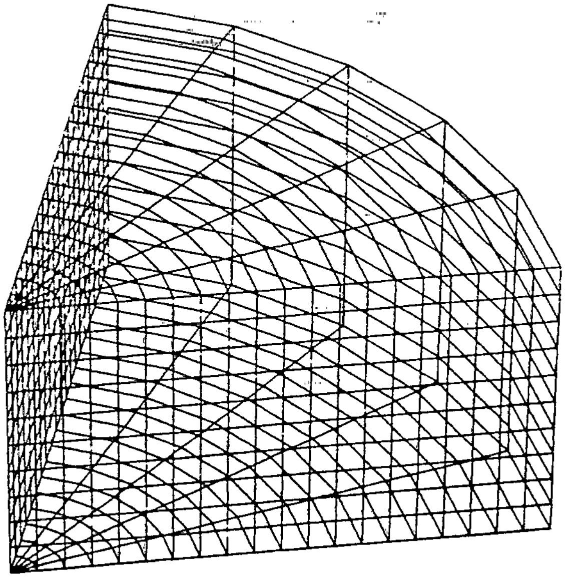


Fig. 10. Perspective view of the outline of a KIVA-II sector mesh. NX = 20, NY = 5, NZ = 10, and THSECT = 72°.

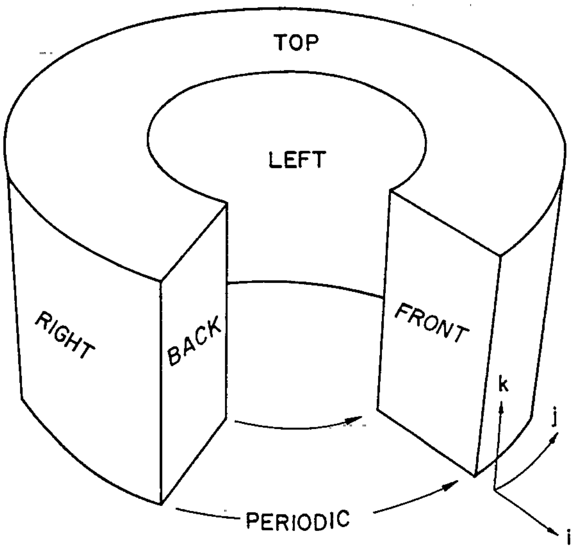


Fig. 11. The KIVA-II 3-D pseudo-polar grid is formed from a Cartesian block of cells through the use of periodic boundary conditions.

The azimuthal dimension of the cylindrical options is given by THSECT, measured in degrees. THSECT is required to be 360° for the full-circle mesh, or an even fraction of 360° for a sector mesh, so that the symmetry condition is satisfied. For example, the mesh in Fig. 10 has THSECT = 72°, as there are 5 spray nozzles being modeled. When NY = 1, subroutine RINPUT ensures that THSECT = 0.50°.

For whichever of the five mesh configurations that the user selects, KIVA-II automatically computes the correct boundary condition treatments for no-slip, free-slip, or free-slip law-of-the-wall, requiring no code modifications by the user, except for special cases such as inflow and outflow treatments, which are discussed in Sec. IV.K.

2. 2-D to 3-D Conversion. Further efficiency in the use of 3-D sector and full circle meshes is made possible through the use of a 2-D to 3-D converter supplied in KIVA-II. In many engine applications, a significant portion of the calculation is the cold flow after IVC and before the spray event, while the flow is truly axisymmetric. This portion can be modeled using the 2-D cylindrical option (NY = 1), then forcing a restart dump just before the spray begins. The user then restarts from this dump, supplying a new value of NY and an appropriate value of THSECT in the input data. Subroutine RINPUT notes that NY has changed and calls subroutine TRAN3D to convert the 2-D mesh with its current solution to a 3-D sector or full circle mesh with the same current solution. The 3-D run then proceeds just as if it had been 3-D all along, and the user can initiate a nonaxisymmetric spray event, having realized a significant reduction in computer time to reach this point in the calculation. The one restriction on this feature is that the parameter NV, which is the dimension of a cell storage array, must be adequate from time t = 0 to accommodate the 3-D mesh.

C. The Indexing Notation

As discussed in Sec. III.B, some variables are located at vertices and some at cell centers. In FORTRAN notation x_{ijk} becomes X(I,J,K), p_{ijk} becomes P(I,J,K), and so on. Thus, the indices (I,J,K) refer to the cell center for cell-centered variables or to vertex (i, j,k) for vertex quantities.

Because the number of vertices in any direction is one greater than the number of cells, it is apparent that the grid in computer storage must be (NX + 1) by (NY + 1) by (NZ + 1) in size. Since our index (I, J,K) refers to both cell centers and vertices, we must allow extra storage planes across the right, back, and top of the logical mesh.

In KIVA-II we replace the triple (I,J,K) subscript by a single subscript, which allows statements to be compactly written. Traditionally single subscripts have also been more efficient, but this is becoming less of an advantage with the increasing sophistication of the newer compilers. When referencing the eight vertices of a cell, we use the 1 through 8 shorthand notion of Fig. 2. In this notation, "I4" refers to vertex 4, the (I,J,K) vertex. I4 is computed as (K − 1)*NXPNYP + (J − 1)*NXP + I, where NXPNYP = (NX + 1)*(NY + 1) = the number of vertices in a plane and NXP = (NX + 1) = the number of radial vertices. When referencing the six neighboring cells to obtain cell-centered variables, we use subscripts with the letter P for + and M for −, when necessary. Thus, we write

IMJK for $(i-1,j,k)$,
I1 for $(i+1,j,k)$,
IJMK for $(i,j-1,k)$,
I3 for $(i,j+1,k)$,
IJKM for $(i,j,k-1)$, and
I8 for $(i,j,k+1)$.

In vector loops that update the 8 vertices of a cell, such as for pressure accelerations or the calculation of vertex masses, note that the sequence is always 6-7-5-8-2-3-1-4. This is dictated by the rule that senior array elements must appear before junior array elements in order to avoid vector dependencies of the results-not-ready or value-destroyed types.

D. Storage of Cell Data

For many applications, KIVA-II can make heavy demands on computer storage. Even with minimal cell resolution, three-dimensional calculations require several thousand cells, and the multiple species and spray model capabilities add to the demand. Because we operate today in a time-sharing environment, reasonably efficient use of computer storage becomes imperative. Accordingly, we have equivalenced as many storage arrays as possible in KIVA-II. The idea is to retain quantities during a calculational cycle only as long as they are needed, and then to reassign the available storage to other quantities. In this version of KIVA-II, the full calculational cycle requires 217 variables of subscript (i,j,k), plus species densities (12 arrays) and species masses (another 12 arrays). As part of the equivalencing, the species densities and species masses share the same storage. The final storage scheme requires 100 arrays rather than 241, a 59% reduction. Figure 12 shows the allocation of these 100 arrays. The ordering from left to right corresponds to the sequence in which subroutines are called during a cycle. Reading down a particular column, the appearance of a variable name signifies reference to it in the associated subroutine.

E. Mesh Generation

Although the features in KIVA-II provide a general capability, applications to internal combustion engine modeling were the principal reason for writing the program. With this in mind, we have included an automated mesh generator in subroutine SETUP that will create a usable 2-D or 3-D cylindrical grid for a wide variety of piston and head shapes for both DISC and diesel engines. The generator requires the use of tabular information as part of the input data.

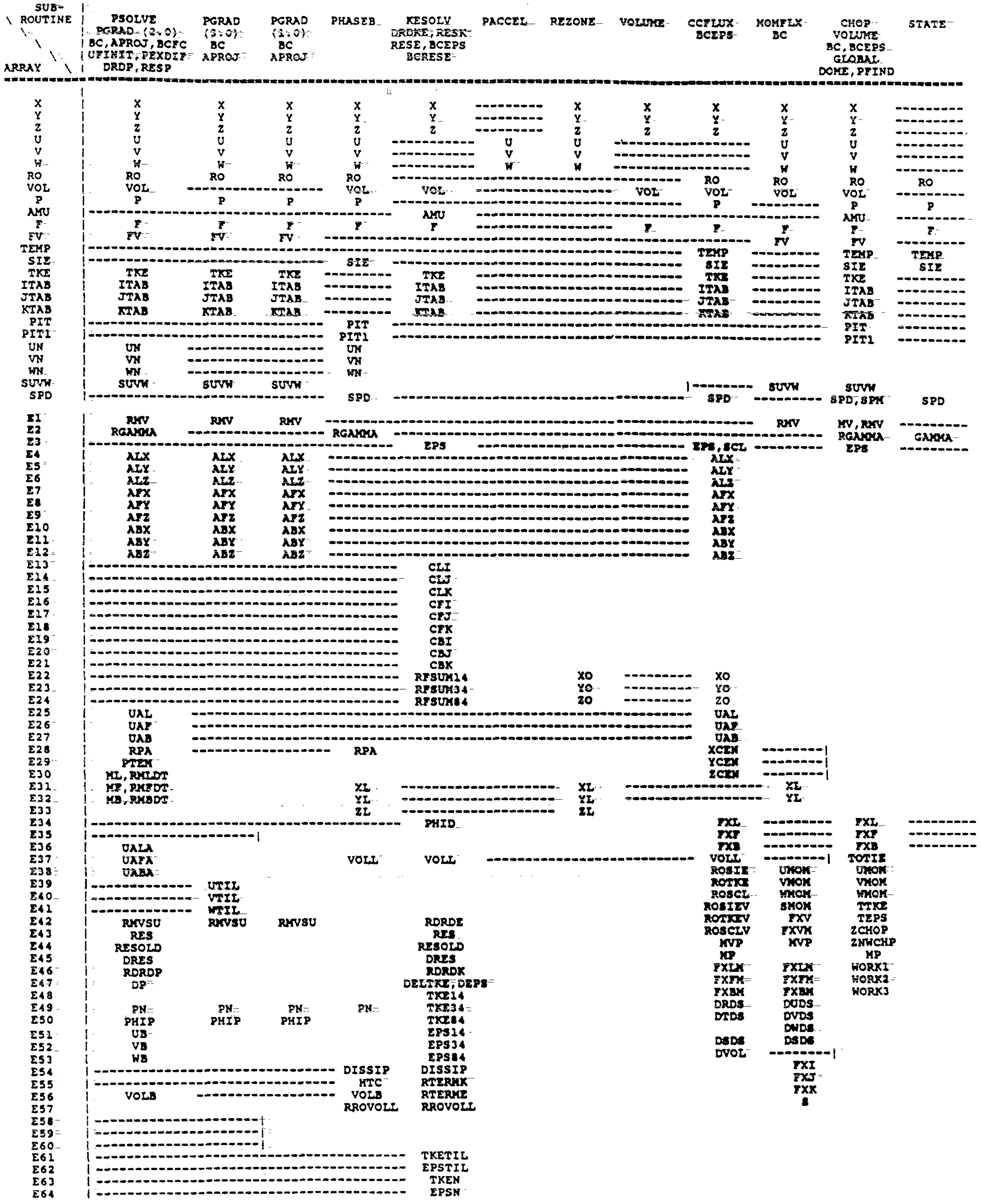
SUB-ROUTINE \ ARRAY	SETUP VOLUME STATE BC	VISC	APROJ	TIMSTP	NEWCYC FULOUT TAPEWR DMPOUT	INJECT	PMOVTV PFIND REPACK	BREAK	COLIDE REPACK	EVAP	LAWALL BC	NODCPL BC
X	X	---------	X	X	X	---------	X	---------	---------	---------	X	X
Y	Y	---------	Y	Y	Y	---------	Y	---------	---------	---------	Y	Y
Z	Z	---------	Z	Z	Z	Z	Z	---------	---------	---------	Z	Z
U	U	---------	---------	U	U	---------	U	U	U	---------	U	U
V	V	---------	---------	V	V	---------	V	V	V	---------	V	V
W	W	---------	---------	W	W	---------	W	W	W	---------	W	W
RO	RO	RO	---------	---------	RO	---------	---------	RO	---------	RO	RO	---------
VOL	VOL	---------	---------	VOL	VOL	---------	---------	---------	VOL	VOL	VOL	VOL
P	P	---------	---------	---------	P	---------	---------	---------	---------	P	P	---------
AMU	AMU	AMU	---------	---------	AMU	---------	---------	---------	---------	---------	---------	---------
F	F	---------	---------	F	F	---------	F	---------	---------	F	F	F
FV	FV	---------	---------	---------	FV	---------	---------	---------	---------	---------	FV	FV
TEMP	TEMP	TEMP	---------	---------	TEMP	---------	TEMP	---------	---------	TEMP	TEMP	---------
SIE	SIE	---------	---------	---------	SIE	---------	---------	---------	---------	SIE	SIE	---------
TKE	TKE	TKE	---------	---------	TKE	---------	TKE	---------	---------	---------	---------	---------
ITAB	ITAB	---------	---------	---------	ITAB	---------	ITAB	---------	---------	---------	---------	---------
JTAB	JTAB	---------	---------	---------	JTAB	---------	JTAB	---------	---------	---------	---------	JTAB
KTAB	KTAB	---------	---------	---------	KTAB	---------	KTAB	---------	---------	---------	---------	---------
PIT		---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------
PIT1		---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------
UN					UN	---------	---------	---------	---------	---------	UN	UN
VN					VN	---------	---------	---------	---------	---------	VN	VN
WN					WN	---------	---------	---------	---------	---------	WN	WN
SUVW	SUVW				SUVW	---------	---------	---------	---------	---------	SUVW	SUVW
SPD	SPD	---------	---------	---------	SPD	---------	---------	---------	---------	SPD	---------	---------
E1	MV,RMV	---------	---------	---------	RMV	---------	---------	---------	---------	---------	RMV	RMV
E2	GAMMA	---------	---------	---------	GAMMA	---------	---------	---------	---------	GAMMA	GAMMA	---------
E3	EPS	EPS	---------	---------	EPS	---------	EPS	---------	---------	---------	---------	---------
E4			ALX	ALX	---------	---------	---------	---------	---------	---------	ALX	ALX
E5			ALY	ALY	---------	---------	---------	---------	---------	---------	ALY	ALY
E6			ALZ	ALZ	---------	---------	---------	---------	---------	---------	ALZ	ALZ
E7			AFX	AFX	---------	---------	---------	---------	---------	---------	AFX	AFX
E8			AFY	AFY	---------	---------	---------	---------	---------	---------	AFY	AFY
E9			AFZ	AFZ	---------	---------	---------	---------	---------	---------	AFZ	AFZ
E10			ABX	ABX	---------	---------	---------	---------	---------	---------	ABX	ABX
E11			ABY	ABY	---------	---------	---------	---------	---------	---------	ABY	ABY
E12			ABZ	ABZ	---------	---------	---------	---------	---------	---------	ABZ	ABZ
E13												
E14												
E15												
E16												
E17												
E18												
E19												
E20												
E21												
E22												
E23												
E24												
E25					SUM3							DUDX
E26					SUM2							DUDY
E27					CONQ							DUDZ
E28										VAPM		DVDX
E29										ENTHO		DVDY
E30												DVDZ
E31	XL											DWDX
E32	YL											DWDY
E33	ZL											DWDZ
E34		---------	---------	FXL								
E35		---------	---------	FXF								
E36		---------	---------	FXB								
E37												AUGMV
E38												
E39												
E40												
E41												
E42												
E43												
E44												
E45												
E46												
E47												
E48												
E49												
E50												
E51												UB
E52												VB
E53												WB
E54										TOTCM		
E55										DHTOT	---------	---------
E56										TOTH		
E57										DSIEP	---------	---------
E58										CPC		
E59					RON	---------	---------	---------	---------	---------	---------	---------
E60					SIEN	---------	---------	---------	---------	---------	---------	---------
E61												
E62												
E63					TKEN	---------	---------	---------	---------	---------	---------	---------
E64					EPSN	---------	---------	---------	---------	---------	---------	---------

Fig. 12. The storage of cell data in KIVA-II. Dashed lines indicate that the quantity to the left must be retained for later use.

```
   SUB-   |
\ ROUTINE |  CHEM      CHEMEQ    PMOM     PCOUPL       YSOLVE      EXDIF     PINIT     PGRAD     VSOLVE      TSOLVE
  \       |             (OR                BC           YIT         BC                 (1.0)     RESUVW      TINVRT
    \     |            CHMQGM)                          RESY                            BC        BC          REST
      \   |                                                                           APROJ                   DRDT
ARRAY   \ |
==========================================================================================================================
          |
    X     |-------------------------------   X            X           X         X         X         X       -------------
    Y     |-------------------------------   Y            Y           Y         Y         Y         Y       -------------
    Z     |-------------------------------   Z            Z           Z         Z         Z         Z       -------------
    U     |-------------------------------   U     ---------------    U     ---------     U         U       -------------
    V     |-------------------------------   V     ---------------    V     ---------     V         V       -------------
    W     |-------------------------------   W     ---------------    W     ---------     W         W       -------------
   RO     |   RO        RO        RO         RO           RO          RO        RO        RO        RO           RO
   VOL    |-------------------------------   VOL          VOL         VOL   -------------------     VOL          VOL
    P     |----------   P     ---------------------------------       P         P         P     ----------       P
   AMU    |------------------------------------------     AMU         AMU   -------------------     AMU          AMU
    F     |    F          F   ---------      F            F           F         F         F         F            F
   FV     |-------------------------------   FV    ---------------    FV    ---------     FV        FV      -------------
  TEMP    |   TEMP      TEMP      TEMP   ---------        TEMP        TEMP  -----------------------------      TEMP
   SIE    |   SIE       SIE   ---------      SIE          SIE         SIE   -----------------------------      SIE
   TKE    |-------------------------------   TKE   ---------------    TKE   ---------     TKE   -----------------------
  ITAB    |------------------------------------------     ITAB        ITAB  ---------     ITAB  ----------     ITAB
  JTAB    |------------------------------------------     JTAB        JTAB  ---------     JTAB  ----------     JTAB
  KTAB    |------------------------------------------     KTAB        KTAB  ---------     KTAB  ----------     KTAB
   PIT    |-----------------------------------------------------------------    PIT   ---------------------------------
  PIT1    |-----------------------------------------------------------------    PIT1  ---------------------------------
   UN     |--------------------     UN     -------------------------    UN    -----------------------------------------
   VN     |--------------------     VN     -------------------------    VN    -----------------------------------------
   WN     |--------------------     WN     -------------------------    WN    -----------------------------------------
  SUVW    |--------------------    SUVW      SUVW  ---------------    SUVW  ---------     SUVW      SUVW    -------------
   SPD    |   SPD       SPD   ---------      SPD          SPD     -------------------------------------------    SPD

   E1     |----------------------------- MV,RMV ---------------       RMV   ---------     RMV       RMV     -------------
   E2     |----------   GAMMA   ---------------------------------     GAMMA     GAMMA   -------------------GAMMA,RGAMMA
   E3     |-------------------------------   EPS   ---------------    EPS   ---------------------------------------------
   E4     |------------------------------------------     ALX         ALX   ---------     ALX       ALX     -------------
   E5     |------------------------------------------     ALY         ALY   ---------     ALY       ALY     -------------
   E6     |------------------------------------------     ALZ         ALZ   ---------     ALZ       ALZ     -------------
   E7     |------------------------------------------     AFX         AFX   ---------     AFX       AFX     -------------
   E8     |------------------------------------------     AFY         AFY   ---------     AFY       AFY     -------------
   E9     |------------------------------------------     AFZ         AFZ   ---------     AFZ       AFZ     -------------
   E10    |------------------------------------------     ABX         ABX   ---------     ABX       ABX     -------------
   E11    |------------------------------------------     ABY         ABY   ---------     ABY       ABY     -------------
   E12    |------------------------------------------     ABZ         ABZ   ---------     ABZ       ABZ     -------------
   E13    |                                               CLI         CLI   -----------------------------      CLI
   E14    |                                               CLJ         CLJ   -----------------------------      CLJ
   E15    |                                               CLK         CLK   -----------------------------      CLK
   E16    |                                               CFI         CFI   -----------------------------      CFI
   E17    |                                               CFJ         CFJ   -----------------------------      CFJ
   E18    |                                               CFK         CFK   -----------------------------      CFK
   E19    |                                               CBI         CBI   -----------------------------      CBI
   E20    |                                               CBJ         CBJ   -----------------------------      CBJ
   E21    |                                               CBK         CBK   -----------------------------      CBK
   E22    |                                         FSUM14,RFSUM14 RFSUM14  -----------------------------    RFSUM14
   E23    |                                         FSUM34,RFSUM34 RFSUM34  -----------------------------    RFSUM34
   E24    |                                         FSUM84,RFSUM84 RFSUM84  -----------------------------    RFSUM84
   E25    |                                               YSPM        DUDX                          DUDX
   E26    |                                               YSPD        DUDY                          DUDY         CV
   E27    |                                                           DUDZ                          DUDZ          R
   E28    |                                               XCEN        DVDX                          DVDX       SIETIL
   E29    |                                               YCEN        DVDY                          DVDY
   E30    |                                               ZCEN        DVDZ                          DVDZ
   E31    |                                                           DWDX                          DWDX       CVTERM
   E32    |                                                           DWDY                          DWDY
   E33    |                                                           DWDZ                          DWDZ
   E34    |                                               PHID        PHID  -------------------     PHID         PHID
   E35    |                                              ENTHDF     DISPTIL ------------------- DISPTIL   -------------
   E36    |                                           SPD14,DYP14     TEM14                         RESU        TEM14
   E37    |                                           SPD34,DYP34     TEM34                         RESV        TEM34
   E38    |                         RU       RU       SPD84,DYP84     TEM84                         RESW        TEM84
   E39    |                         RV       RV           HISP        UTIL  -------------------------------------------
   E40    |                         RW       RW          SPMTIL       VTIL  -------------------------------------------
   E41    |                                              ENTHTIL      WTIL  -------------------------------------------
   E42    |                                               DDY         RMVSU ---------    RMVSU     RMVSU    -------------
   E43    |                                               RES                                   RESUO        RES
   E44    |                                              RESOLD                                 RESVO       RESOLD
   E45    |                                               DRES                                  RESWO        DRES
   E46    |                                              RDRDY                                  DRESU       RDRDT
   E47    |                                              DELTAY                                 DRESV       DTEMP
   E48    |                                                           TKE14                     DRESW
   E49    |                                                           TKE34     PN        PN   ----------      PN
   E50    |                                                           TKE84     PHIP      PHIP ------------------------
   E51    |                                                           EPS14                      UB      -------------
   E52    |                                                           EPS34                      VB      -------------
   E53    |                                                           EPS84                      WB      -------------
   E54    |                                                                                     DISSIP      DISSIP
   L55    |--------------------    DMTOT    DMTOT                                               DUHAT        HTC
   E56    |                                                                                  DVHAT,VOLB      VOLB
   E57    |--------------------    DSIEP    DSIEP                                               DWHAT
   E58    |                        DTKEP    DTKEP                     CPC   -----------------------------      CPC
   E59    |---------------------------------------------------------RON,HTCTIL-------------------------      HTCTIL
   E60    |---------------------------------------------------------SIEN,TTIL             |----------         TTIL
   E61    |                                                          TKETIL ---------------------------------------------
   E62    |                                                          EPSTIL ---------------------------------------------
   E63    |------------------------------------------------------------------------------------------------------------
   E64    |------------------------------------------------------------------------------------------------------------
```

Fig. 12. continued



```
   SUB-   |
\ ROUTINE |     PSOLVE       PGRAD       PGRAD       PHASEB      KESOLV      PACCEL      REZONE      VOLUME      CCFLUX      MOMFLX       CHOP       STATE
  \       |  PGRAD (2,0)     (3,0)       (1,0)                 DRDKE,RESK                                        BCEPS         BC        VOLUME
    \     | BC,APROJ,BCFC      BC          BC                  RESE,BCEPS                                                               BC,BCEPS
      \   | UFINIT,PEXDIF    APROJ       APROJ                   BCRESE                                                                  GLOBAL
ARRAY    \|   DRDP,RESP                                                                                                                 DOME,PFIND
==============================================================================================================================================================
          |
    X     |       X            X           X           X           X      -----------      X           X           X           X           X      -----------
    Y     |       Y            Y           Y           Y           Y      -----------      Y           Y           Y           Y           Y      -----------
    Z     |       Z            Z           Z           Z           Z      -----------      Z           Z           Z           Z           Z      -----------
    U     |       U            U           U           U      -----------      U           U      -----------------------      U           U      -----------
    V     |       V            V           V           V      -----------      V           V      -----------------------      V           V      -----------
    W     |       W            W           W           W      -----------      W           W      -----------------------      W           W      -----------
    RO    |       RO           RO          RO          RO     -----------------------------------------------      RO          RO          RO          RO
    VOL   |      VOL      -----------------------     VOL         VOL     -----------------------     VOL         VOL         VOL         VOL     -----------
    P     |       P            P           P           P      -----------------------------------------------      P      -----------      P           P
    AMU   |--------------------------------------------------     AMU     -----------------------------------------------------------     AMU     -----------
    F     |       F            F           F           F           F      -----------------------      F           F           F           F           F
    FV    |       FV           FV          FV     -----------------------------------------------------------------------      FV          FV     -----------
    TEMP  |--------------------------------------------------------------------------------------------------     TEMP    -----------     TEMP        TEMP
    SIE   |--------------------------------------     SIE     -----------------------------------------------     SIE     -----------     SIE         SIE
    TKE   |      TKE          TKE         TKE     -----------     TKE     -----------------------------------     TKE     -----------     TKE     -----------
    ITAB  |      ITAB         ITAB        ITAB    -----------     ITAB    -----------------------------------     ITAB    -----------     ITAB    -----------
    JTAB  |      JTAB         JTAB        JTAB    -----------     JTAB    -----------------------------------     JTAB    -----------     JTAB    -----------
    KTAB  |      KTAB         KTAB        KTAB    -----------     KTAB    -----------------------------------     KTAB    -----------     KTAB    -----------
    PIT   |--------------------------------------     PIT     -----------------------------------------------------------------------     PIT     -----------
    PIT1  |--------------------------------------     PIT1    -----------------------------------------------------------------------     PIT1    -----------
    UN    |       UN      -----------------------      UN
    VN    |       VN      -----------------------      VN
    WN    |       WN      -----------------------      WN
    SUVW  |      SUVW         SUVW        SUVW                                                                |----------     SUVW        SUVW
    SPD   |--------------------------------------     SPD     -----------------------------------------------     SPD     -----------   SPD,SPM       SPD

    E1    |      RMV          RMV         RMV     -----------------------------------------------------------------------     RMV        MV,RMV   -----------
    E2    |     RGAMMA    -----------------------    RGAMMA   -----------------------------------------------------------------------    RGAMMA      GAMMA
    E3    |--------------------------------------------------     EPS     -----------------------------------   EPS,SCL   -----------     EPS     -----------
    E4    |      ALX          ALX         ALX     -----------------------------------------------------------     ALX
    E5    |      ALY          ALY         ALY     -----------------------------------------------------------     ALY
    E6    |      ALZ          ALZ         ALZ     -----------------------------------------------------------     ALZ
    E7    |      AFX          AFX         AFX     -----------------------------------------------------------     AFX
    E8    |      AFY          AFY         AFY     -----------------------------------------------------------     AFY
    E9    |      AFZ          AFZ         AFZ     -----------------------------------------------------------     AFZ
    E10   |      ABX          ABX         ABX     -----------------------------------------------------------     ABX
    E11   |      ABY          ABY         ABY     -----------------------------------------------------------     ABY
    E12   |      ABZ          ABZ         ABZ     -----------------------------------------------------------     ABZ
    E13   |--------------------------------------------------     CLI
    E14   |--------------------------------------------------     CLJ
    E15   |--------------------------------------------------     CLK
    E16   |--------------------------------------------------     CFI
    E17   |--------------------------------------------------     CFJ
    E18   |--------------------------------------------------     CFK
    E19   |--------------------------------------------------     CBI
    E20   |--------------------------------------------------     CBJ
    E21   |--------------------------------------------------     CBK
    E22   |--------------------------------------------------   RFSUM14                    XO     -----------      XO
    E23   |--------------------------------------------------   RFSUM34                    YO     -----------      YO
    E24   |--------------------------------------------------   RFSUM84                    ZO     -----------      ZO
    E25   |      UAL      -----------------------------------------------------------------------------------     UAL
    E26   |      UAF      -----------------------------------------------------------------------------------     UAF
    E27   |      UAB      -----------------------------------------------------------------------------------     UAB
    E28   |      RPA      -----------------------     RPA                                                         XCEN    ----------|
    E29   |      PTEM                                                                                             YCEN    ----------|
    E30   |    ML,RMLDT                                                                                           ZCEN    ----------|
    E31   |    MF,RMFDT                                XL     -----------------------      XL     -----------------------      XL
    E32   |    MB,RMBDT                                YL     -----------------------      YL     -----------------------      YL
    E33   |                                            ZL     -----------------------      ZL
    E34   |--------------------------------------------------     PHID                                            FXL     -----------     FXL     -----------
    E35   |--------------------|                                                                                  FXF     -----------     FXF     -----------
    E36   |      UALA                                                                                             FXB     -----------     FXB     -----------
    E37   |      UAFA                                 VOLL        VOLL    -----------------------------------     VOLL    ----------|    TOTIE
    E38   |      UABA                                                                                            ROSIE        UMOM        UMOM
    E39   |--------------     UTIL                                                                               ROTKE        VMOM        VMOM
    E40   |--------------     VTIL                                                                               ROSCL        WMOM        WMOM
    E41   |--------------     WTIL                                                                               ROSIEV       SMOM        TTKE
    E42   |     RMVSU        RMVSU       RMVSU                   RDRDE                                           ROTKEV       FXV         TEPS
    E43   |      RES                                              RES                                            ROSCLV       FXVM       ZCHOP
    E44   |     RESOLD                                           RESOLD                                           MVP         MVP        ZNWCHP
    E45   |      DRES                                             DRES                                             MP                      MP
    E46   |     RDRDP                                            RDRDK                                            FXLM        FXLM       WORK1
    E47   |       DP                                          DELTKE,DEPS                                         FXFM        FXFM       WORK2
    E48   |                                                      TKE14                                            FXBM        FXBM       WORK3
    E49   |       PN           PN          PN          PN        TKE34                                            DRDS        DUDS
    E50   |      PHIP         PHIP        PHIP                   TKE84                                            DTDS        DVDS
    E51   |       UB                                             EPS14                                                        DWDS
    E52   |       VB                                             EPS34                                            DSDS        DSDS
    E53   |       WB                                             EPS84                                            DVOL    ----------|
    E54   |--------------------------------------    DISSIP      DISSIP                                                       FXI
    E55   |--------------------------------------     HTC        RTERMK                                                       FXJ
    E56   |      VOLB     -----------------------     VOLB       RTERME                                                       FXK
    E57   |                                         RROVOLL     RROVOLL                                                        S
    E58   |--------------------|
    E59   |--------------------|
    E60   |--------------------|
    E61   |--------------------------------------------------    TKETIL
    E62   |--------------------------------------------------    EPSTIL
    E63   |--------------------------------------------------     TKEN
    E64   |--------------------------------------------------     EPSN
```

Fig. 12. continued

1. The Piston Face. Given the dimensions of the desired piston geometry, the user lays out a half cross section on graph paper, as shown in Fig. 13. At this point, one must decide on the level of resolution available and, based on it, define grid points along the piston silhouette, starting at the bowl axis and ending at the cylinder wall. The outline is required to follow cell edges in the logical mesh and not cut diagonally across a cell. With these points defined, it is a simple matter to create the input table. For each point, the generator requires the logical coordinates (i and k) and the physical coordinates (r and z) in cm, relative to z = 0 being the lowest z-coordinate. In the input data, NPO is the number of points, and the coordinates are in the arrays IPO, KPO, RPO, and ZPO. In addition, NUNIF specifies how many zones out from the centerline are to remain uniform in r from the bottom of the mesh to the top (NUNIF ≥ 0). This feature allows one to maintain the best possible resolution in the spray region. Figure 14 lists the tabular information associated with Fig. 13. The mesh generator first assigns the vertices that have been specified in the table, then places remaining vertices, those within the fluid region, at average positions of their neighbors. The averaging in the code uses equal-weight springs. If this is found to create a less-than-optimum grid, it is a simple matter to modify the algorithm to use unequal-weight springs to shrink or expand cells selectively in the piston bowl.

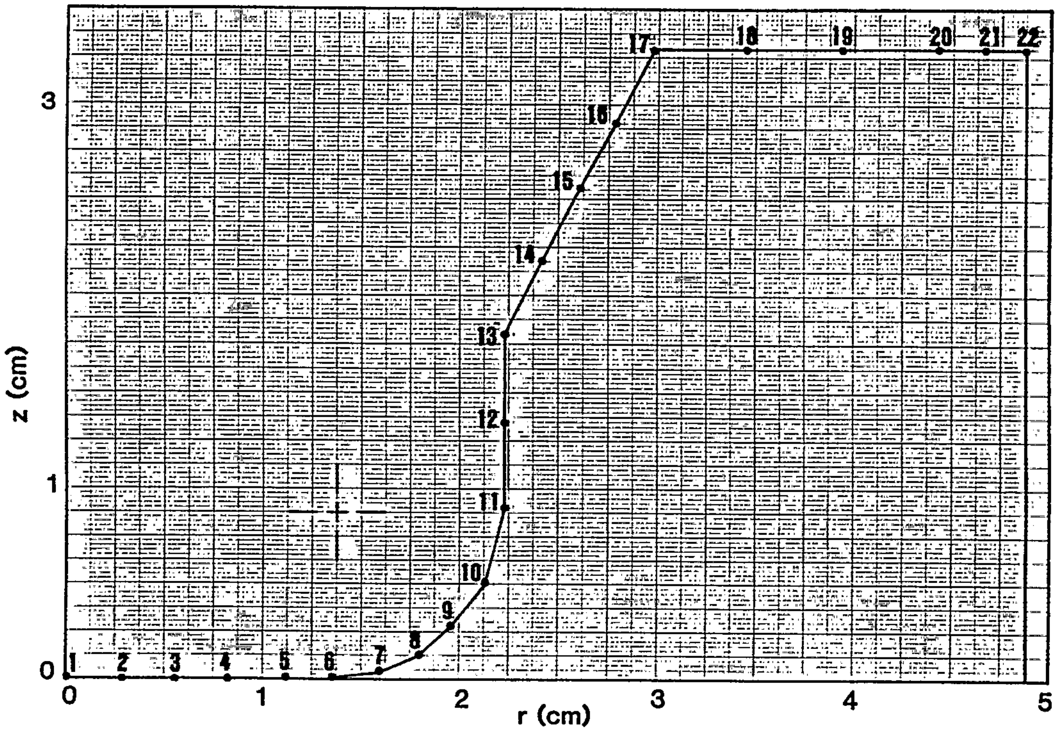


Fig. 13. Chamfered bowl piston silhouette showing grid points to be used by the automatic mesh generator.

```
NPO          22
NUNIF         0.
  1    1    0.0       0.0
  2    1 0.28125      0.0
  3    1 0.5625       0.0
  4    1 0.84375      0.0
  5    1 1.125        0.0
  6    1 1.35         0.0
  7    1 1.60         0.03
  8    1 1.82         0.125
  9    1 1.975        0.275
  9    2 2.15         0.5
  9    3 2.25         0.90
  9    4 2.25         1.35
  9    5 2.25         1.8
  9    6 2.4375       2.175
  9    7 2.625        2.55
  9    8 2.8125       2.925
  9    9 3.0          3.3
 10    9 3.4875       3.3
 11    9 3.975        3.3
 12    9 4.4625       3.3
 13    9 4.70625      3.3
 14    9 4.9215       3.3
```

Fig. 14. Tabular input data corresponding to Fig. 13.

Note that up to this point, we have been discussing the geometry in a purely two-dimensional sense. In fact, KIVA-II initially sets up only the j = 1 azimuthal plane, treating the mesh generation as two-dimensional. Then, for cylindrical meshes, the generator simply rotates this j = 1 configuration about the axis to create the remaining azimuthal planes, j = 2 through j = NYP. Figure 15 shows the axisymmetric mesh that results after rotation through 360°. Deactivated vertices lying entirely within the piston are not drawn. For a 2-D geometry with cylindrical symmetry, only the j = 2 plane is created by the rotation process.

Some piston designs have an offset bowl in the piston face, not concentric with the axis of the cylinder. This is common in diesel engines and represents a truly three-dimensional geometry. Subroutine SETUP automatically allows this option, through the use of the quantity OFFSET in the input data. If OFFSET = 0.0, the result is an axisymmetric geometry, as in the example just discussed. If OFFSET ≠ 0.0, however, the bowl is offset in the x-direction by the distance specified by the value of OFFSET.

An example is shown in Fig. 16a, which shows a bowl offset −0.546 cm from the cylinder midline. To set up this geometry, the user starts by pretending it is an axisymmetric configuration, as in Fig. 16b, and supplies the tabular input data based on these adjusted dimensions. The axisymmetric layout is shown in Fig. 17, and the associated data are listed in Fig. 18. SETUP first creates a grid in the same manner as in the previous example, then checks the value of OFFSET. Because OFFSET = − 0.546, the bowl and all vertices above it are displaced to the left (negative x-direction) by this amount. The radial lines are then straightened from the new center out to the cylinder wall. Next, x- and y-coordinates between the bowl lip and the cylinder wall are uniformly distributed. Figure 19 shows three views of the final mesh SETUP creates.

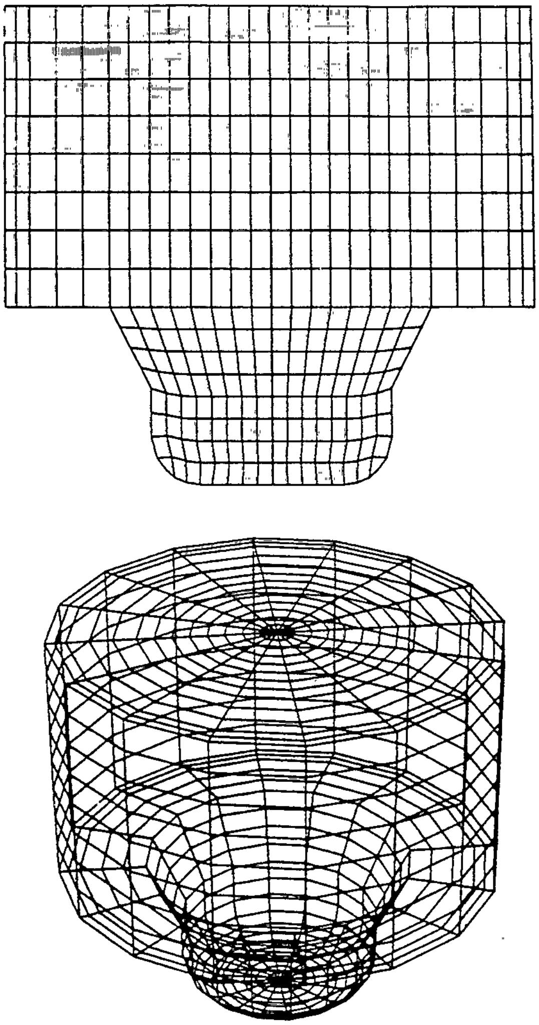


Fig. 15. The KIVA-II computing mesh created using the tabular data of Fig. 14. The top view is a cross section through the mesh; at the bottom is a perspective view of the grid, in which the fluid region is outlined. NX = 13, NY = 16, and NZ = 16.

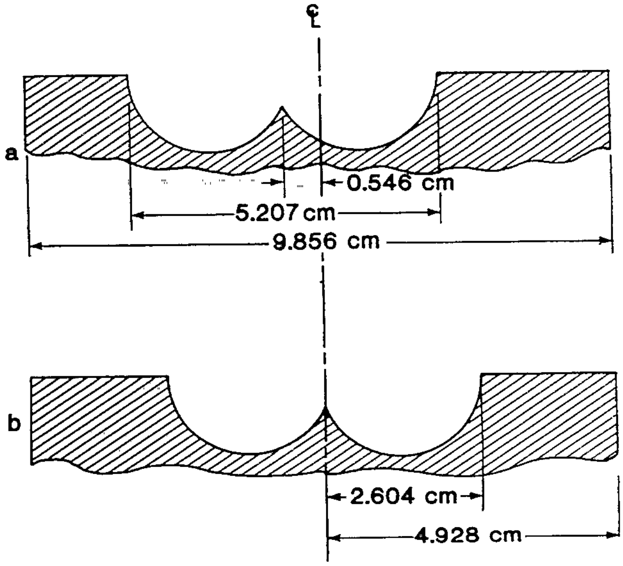


Fig. 16. (a) A piston cup with a bowl offset 0.546 cm to the left.
(b) The user pretends that the desired configuration is axisymmetric and supplies the offset as a separate parameter.

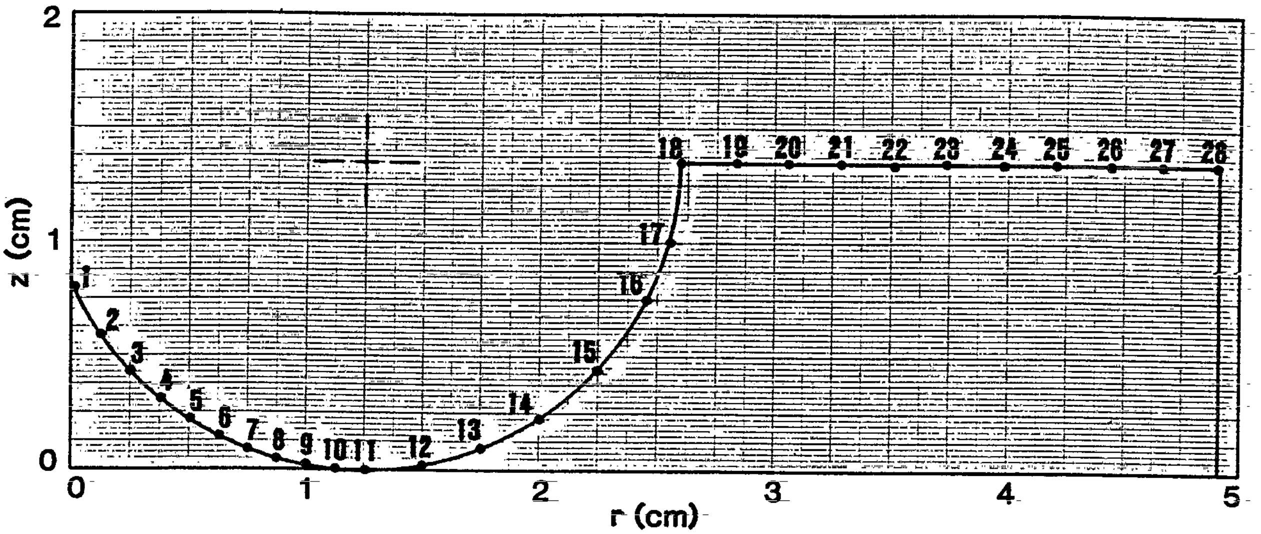


Fig. 17. Piston silhouette based on the symmetric configuration of Fig. 16b.

```
NPD        28
NUNIF       0
 1     1   0.0      0.8
 2     1   0.125    0.575
 3     1   0.25     0.425
 4     1   0.375    0.325
 5     1   0.5      0.22
 6     1   0.625    0.15
 7     1   0.75     0.09
 8     1   0.875    0.05
 9     1   1.0      0.03
10     1   1.125    0.01
11     1   1.25     0.0
11     2   1.5      0.02
11     3   1.75     0.1
11     4   2.0      0.225
11     5   2.25     0.43
11     6   2.475    0.75
11     7   2.563    1.0
11     8   2.604    1.35
12     8   2.836    1.35
13     8   3.068    1.35
14     8   3.301    1.35
15     8   3.533    1.35
16     8   3.766    1.35
17     8   3.998    1.35
18     8   4.230    1.35
19     8   4.463    1.35
20     8   4.695    1.35
21     8   4.928    1.35
```

Fig. 18. Tabular input data corresponding to Fig. 17.

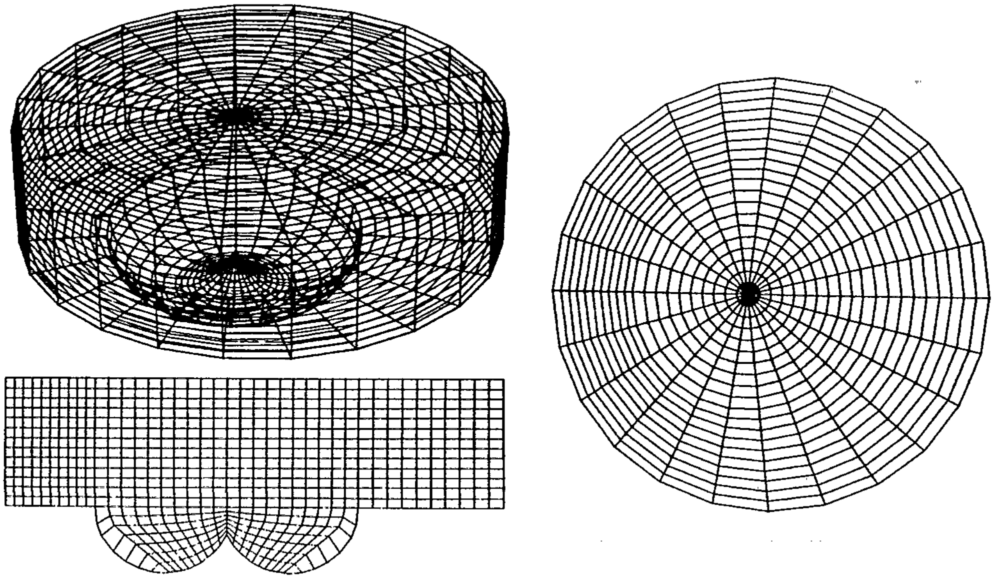


Fig. 19. Perspective, overhead, and cross-sectional views of the KIVA-II computing mesh created from the tabular data of Fig. 15, with the bowl subsequently offset. The mesh dimensions are NX = 20, NY = 24, and NZ = 20.

In the previous examples, the piston cups have been round, as are indeed most cup designs in common use. Some designs, however, employ a square cup, the purpose being to enhance turbulent mixing. A square cup option is included in the KIVA-II mesh generator and requires a 90° sector geometry (JSECTR = 1, CYL = 1.0, THSECT = 90.0, and OFFSET = 0.0). The example shown in Fig. 20 has NX = 13, NY = 12, and NZ = 20. It was created using the input data listed in Fig. 21, which shows three additional variables:

- SQUARE = 1.0 indicates the square cup; SQUARE = 0.0 for the round cups of the previous examples.
- RCORNR is the radius of the corner in cm; at present we require RCORNR > 0.
- NSTRT is the number of zones with straight sides before the corner radius begins. A relationship between NY and NSTRT is implied, and the code checks to ensure that NY − (2*NSTRT) $\geq$ 1.

The mesh generator has been successfully applied to a wide variety of bowl designs. In addition to the chamfered, Mexican-hat, and square bowls of the above examples, we have modeled deep curved and reentrant bowls. The bowl volume is printed by SETUP as a check for the user, along with the total mesh volume.

At the other extreme, a flat-topped piston is obtained by defining KPO = 1 and ZPO = 0.0 for all tabular points. For a fixed Eulerian grid, input ATDC = −180°, SQUISH = 0.0, and STROKE equal to the desired mesh height. (The result is the same with ATDC = 0°, STROKE = 0.0, and SQUISH equal to the mesh height.) Unless

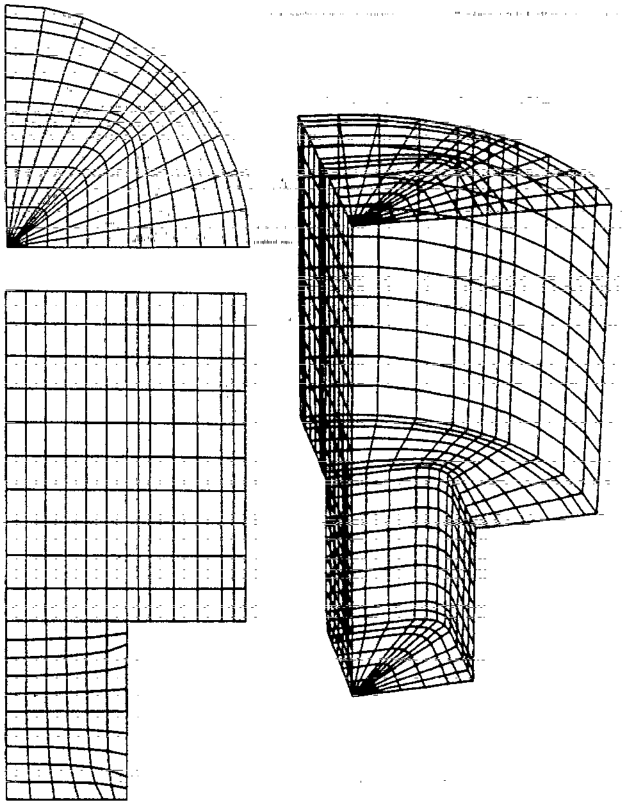


Fig. 20. Overhead, front, and perspective views of a square-cup mesh. The mesh dimensions are NX = 13, NY = 12, and NZ = 20.

```
NPO         24
NUNIF        0
 1    1 0.0000  0.0000
 2    1 0.3442  0.0000
 3    1 0.6883  0.0000
 4    1 1.0325  0.0000
 5    1 1.3767  0.0000
 6    1 1.7208  0.0000
 7    1 2.0650  0.0000
 7    2 2.0650  0.3080
 7    3 2.0650  0.6160
 7    4 2.0650  0.9240
 7    5 2.0650  1.2320
 7    6 2.0650  1.5400
 7    7 2.0650  1.8480
 7    8 2.0650  2.1560
 7    9 2.0650  2.4640
 7   10 2.0650  2.7720
 7   11 2.0650  3.0800
 8   11 2.2715  3.0800
 9   11 2.4780  3.0800
10   11 2.8910  3.0800
11   11 3.3040  3.0800
12   11 3.7170  3.0800
13   11 3.9235  3.0800
14   11 4.1300  3.0800

SQUARE       1.0
RCORNR       0.5
NSTRT        4
```

Fig. 21. Tabular input data for the square-cup example of Fig. 20.

modified by the user, the generator will create uniform δz's. The result is a simple cylinder, which is also useful for other purposes besides engine applications. The piston motion may be turned off to maintain the fixed grid simply by setting RPM = 0.0 in the input data. This will also automatically turn off the swirl, because the swirl ratio is defined as the ratio of air r.p.m. to crankshaft r.p.m. If a swirl is desired in an RPM = 0.0 case, the user will have to patch the ANGVEL statement in subroutine SETUP. A nonzero value of OFFSET can be used with a flat-topped piston if one wishes to place the axis of rotation to one side of the axis of the cylinder. Thus, all planes would have an appearance similar to that illustrated in the overhead view of Fig. 19.

The simple plane coordinates (CYL = 0.0) meshes shown in Fig. 9 are also created automatically. In 2-D, the j = 2 plane is identical to the j = 1 plane, but is at depth δy (DY in the input data) behind the j = 1 (y = 0.0) plane. In 3-D, the j planes behind the j = 1 plane have uniform δy (DY in the input data). This may be easily overridden if nonuniform δy's are desired.

2. The Cylinder Head. Analogous to the definition of a piston silhouette, the generator uses tabular input data when the cylinder head is not perfectly flat. In the input data, NHO is the number of points in the head outline data, again starting at the axis and ending at the cylinder wall. The limiting case is, of course, the flat head, as in the previous examples. Here the use specifies NHO = 0, with no further mesh generation data required after the NHO line.

For a nonflat head, NHO > 0, and the coordinates are in the arrays IHO, KHO, RHO, and ZHO. ZHO is relative to a value of zero at the lowest point. The head shape, typically a dome, may be cylindrically symmetric, ellipsoidal, or semi-ellipsoidal when viewed from above. In addition, the dome may be offset, again using OFFSET in the input data. If NHO > 0, the head is offset rather than the piston bowl, if a bowl exists.

An example that uses all of these features is the mesh shown in Fig. 22. Let us examine the sequence of steps in its creation. First, the piston silhouette is defined, as discussed in the previous section. This is a simple cylindrically symmetric shape with a slightly arched top, as illustrated in the bottom of Fig. 23. The first set of 11 tabular lines in Fig. 24 provide the definition. If we were done, the mesh would appear as shown in Fig. 25a. Second, the basic head shape supplied to the code neglects for the moment offset and ellipsoidal adjustments. This starting profile, drawn at the top of Fig. 23, and entered as the set of 17 lines following NHO in Fig. 24, modifies the mesh to the appearance shown in Fig. 25b. Third, the head is offset 1.20 cm to the left, specified by OFFSET = −1.20 in the input file (Fig. 25c). Finally, the head dome is made semi-ellipsoidal. Whenever NHO > 0, NEO must be specified. If NEO = 0, no tabular data follow, and the head definition would be complete. Otherwise, NEO is the number of ellipsoidal or semi-ellipsoidal k-planes in the dome. The four columns of tabular information that follow NEO are the arrays NCORR, IEMAX, SEMIMJ, and SEMIMN. NCORR is the index of the corresponding line in the NHO table. For a perfect ellipse both left and right, the value of IEMAX is zero, using the semi-major (SEMINJ) and semi-minor (SEMIMN) axes specified at that level. Points along the ellipse are determined by their intersection with each radial grid line in turn, which originate at the ellipse center. If IEMAX = 1, as in this example, the code again creates a perfect ellipse on the left, but on the right chooses the shorter of the ellipse distance and a circular arc whose radius is the RHO at that k-level. This ensures that the ellipsoid on the left will smoothly join the circular arc on the right, as evident in the overhead view of Fig. 22. The final mesh is also shown in the cross-section of Fig. 25d.

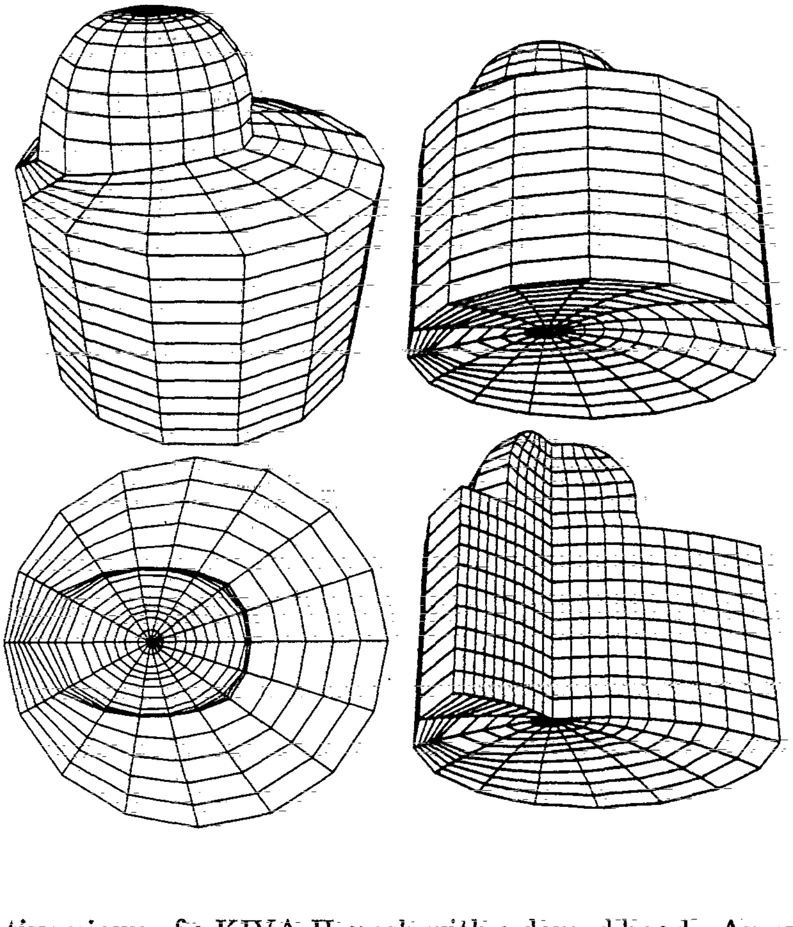


Fig. 22. Perspective views of a KIVA-II mesh with a domed head. An overhead view is shown at the lower left, and a cut with planes j = 11 through j = 16 removed at the lower right. The mesh dimensions are NX = 10, NY = 16, and NZ = 16.

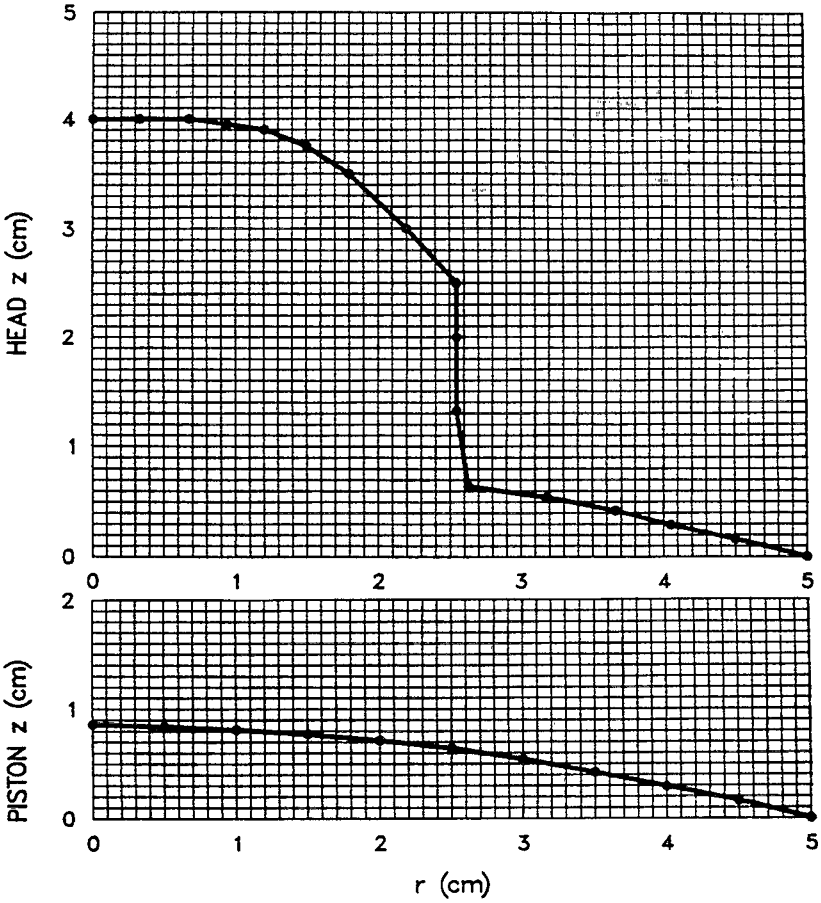


Fig. 23. Grid points used by the mesh generator for the mesh of Fig. 22.

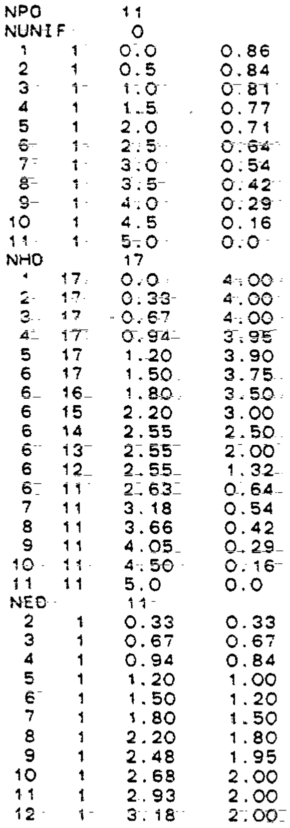
F. Cell and Vertex Flags

In many engine applications, of which the examples in the preceding section are typical, a number of cells are deactivated, as they lie entirely within the piston or the head. In order that calculational DO loops may easily recognize such "obstacle" cells, in addition to the "ghost" cells in the i = NXP, j = NYP, and k = NZP planes, we use a cell flagging scheme. Cells with F = 0.0 are deactivated, whereas cells with F = 1.0 participate as fluid cells. In a vectorized DO loop over all cells (e.g., DO 10 I4 = 1, IJKVEC), F(I4) is used as a coefficient, thus permitting vectorization, as no testing is required.

SETUP also defines a set of vertex flags FV. Analogous to the cell flags, FV = 0.0 describes a vertex lying entirely within an obstacle, and FV > 0.0 describes a fluid vertex. Used in the automatic mesh generator and retained for use thereafter are unique flags for each of the following possible cases:

FV = FLFACE = 1.0 for all vertices on the piston face,

FV = FLBOWL = 2.0 for bowl vertices not on the piston face,



```
NPO          11
NUNIF         0
  1    1    0.0      0.86
  2    1    0.5      0.84
  3    1    1.0      0.81
  4    1    1.5      0.77
  5    1    2.0      0.71
  6    1    2.5      0.64
  7    1    3.0      0.54
  8    1    3.5      0.42
  9    1    4.0      0.29
 10    1    4.5      0.16
 11    1    5.0      0.0
NHO          17
  1   17    0.0      4.00
  2   17    0.33     4.00
  3   17    0.67     4.00
  4   17    0.94     3.95
  5   17    1.20     3.90
  6   17    1.50     3.75
  6   16    1.80     3.50
  6   15    2.20     3.00
  6   14    2.55     2.50
  6   13    2.55     2.00
  6   12    2.55     1.32
  6   11    2.63     0.64
  7   11    3.18     0.54
  8   11    3.66     0.42
  9   11    4.05     0.29
 10   11    4.50     0.16
 11   11    5.0      0.0
NEO          11
  2    1    0.33     0.33
  3    1    0.67     0.67
  4    1    0.94     0.84
  5    1    1.20     1.00
  6    1    1.50     1.20
  7    1    1.80     1.50
  8    1    2.20     1.80
  9    1    2.48     1.95
 10    1    2.68     2.00
 11    1    2.93     2.00
 12    1    3.18     2.00
```

Fig. 24. Tabular input data for the mesh of Fig. 22.

FV = FLSQSH = 3.0 for all of squish region above the piston,
FV = FLDOME = 4.0 for vertices within the head volume but not on the head, and
FV = FLHEAD = 5.0 for vertices on the head surface itself.

In a vectorized DO loop over all vertices (e.g., DO 10 I4 = 1, IJKALL), CVMG- statements are used that embody FV information in such a way as to ensure that deactivated vertices have no effect.

G. Fuel Sprays

The fuel spray injection model in KIVA-II is sufficiently general that a wide variety of engine injectors or continuous sprays may be specified through input data alone. For many users, subroutine INJECT should require no modification. Features of the injection model include a multiple or multihole nozzle capability, with continuous or pulsed hollow cone or solid cone sprays whose origin, profile, and orientation are easily specified. Pulsed sprays may be sinusoidal, square-wave, or be supplied with a tabular velocity profile appropriate for hole-type nozzle injectors. Either a fixed particle radius or a distribution of radii may be injected.

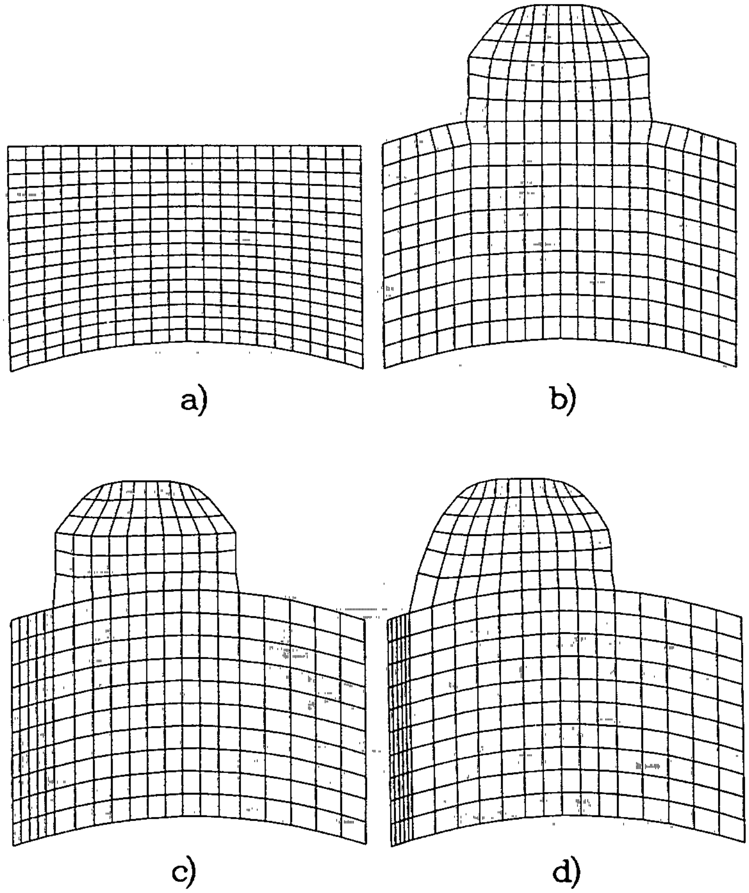


Fig. 25. The four stages in the generation of the mesh in Fig. 22:
(a) The original mesh with an arched piston top,
(b) The addition of a domed head,
(c) The offset of the domed head,
(d) The adjustment of the head shape to a semi-ellipsoidal form, when viewed from above.

These features are now discussed in detail.

1. Spray Origin, Profile, and Orientation. (See Fig. 26.) The input quantity NUMNOZ specifies the number of spray nozzles (1 to 12). For each nozzle, the radius in cm to its location is specified by DRNOZ, measured from (XO,YO), which is the cylinder axis, or in the case of a planar mesh, the front-left-bottom corner of the mesh. The azimuthal rotation in degrees is given by DTHNOZ, measured counterclockwise from the $y = 0$ line. DZNOZ, in cm, locates the nozzle in the axial direction. If DZNOZ < 0, it is interpreted as a distance below the topmost point of the head. This is generally appropriate for engine applications. IF DZNOZ > 0, it is interpreted as a distance above Z(1). This is the appropriate choice for a spray in a fixed mesh, such as a spray combustor or burner.

In order for momentum exchanges with vertices to take place properly, subroutine INJECT ensures that the spray is at least one half cell out radially from the axis in sector

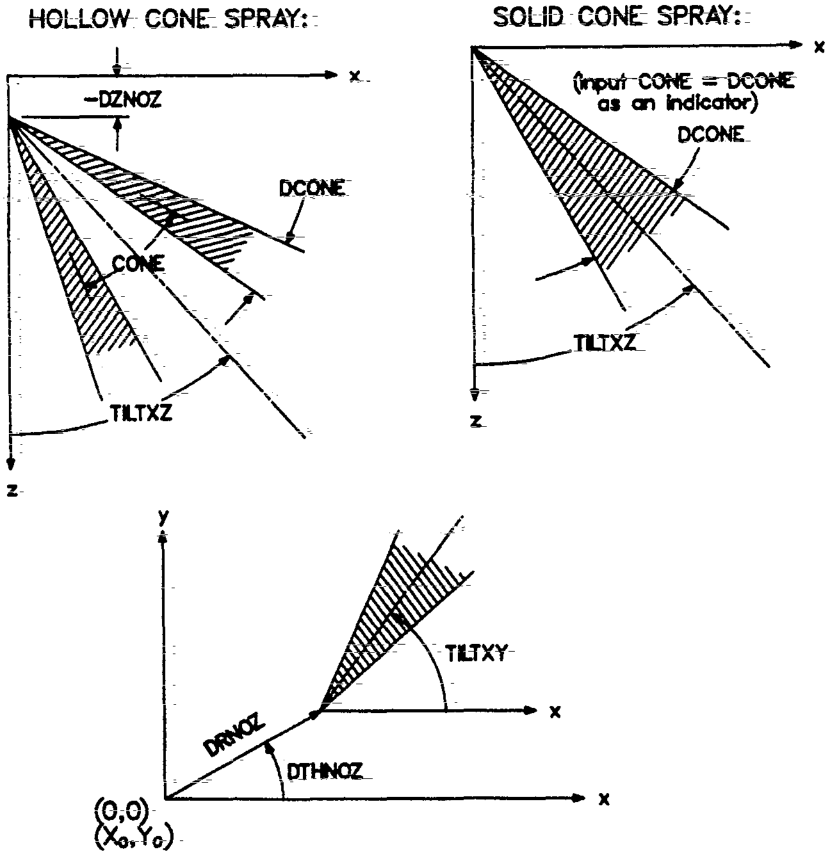


Fig. 26. The input quantities DRNOZ, DTHNOZ, and DZNOZ locate each spray nozzle; TILTXY and TILTXZ define the spray axis in 3-D space; CONE and DCONE allow for hollow cone or solid cone sprays.

meshes and either at least a half cell below the topmost point of the head or a half cell above Z(1).

CONE, DCONE, TILTXZ, and TILTXY, all supplied in degrees, define the profile and orientation of each spray jet. CONE and DCONE provide either a hollow cone spray or a solid cone or pencil spray. CONE is the mean cone angle for hollow cone sprays, and DCONE is the thickness of the spray. If CONE is input identically equal to DCONE, a solid spray results.

The orientation of the spray axis for each nozzle is defined by TILTXZ and TILTXY, where TILTXZ gives the x-direction inclination from vertical in the x-z plane and TILTXY specifies the rotation of the spray axis in the x-y plane. As with DTHNOZ, TILTXY is measured counterclockwise from the positive x-axis. In the case of a sector mesh with NY = 1, the model assumes that the spray jet is azimuthally centered in the half-degree sector, by enforcing TILTXY = THSECT/2.

2. Spray Flow Definition. Several input quantities characterize the flow: The quantity PULSE differentiates between continuous and pulsed spray injection. PULSE =

0.0 is used for a continuous spray, for which TSPMAS is the mass flow rate in g/s, and TNPARC is the number of computational particles injected per second. Alternatively, PULSE > 0 defines a pulsed spray, for which TSPMAS is the total mass in grams to be injected, and TNPARC is the total number of computational particles to be injected.

Clearly, the accuracy of the spray model improves as the number of computational particles is increased, but the code can be significantly slowed down, especially by the particle collision subroutine. The code automatically calculates the mass per computational particle to ensure that the correct total mass or mass flow rate is injected regardless of the choice for TNPARC, which is governed solely by computer time and storage constraints. Some typical values for TNPARC that we have used in engine calculations are 500-1000 (2-D) and 2000-5000 (3-D). For a continuous spray, we typically inject TNPARC = 40000/s. Because of evaporation and an outflow boundary, particles are continuously being destroyed, so that we can get by with a dimension NPAR of only 5000 for the particle storage arrays.

Computational particles are injected with speed VELINJ cm/s, and the angular distribution of particle velocities is uniform within the internal DCONE. The density of the fuel in g/cc is supplied as RHOP, and TPI is the fuel temperature in Kelvin. Particles are moved in subroutine PMOVTV, which will also add the effects of turbulent velocity fluctuations if TURB = 1.0.

Three types of pulsed sprays are available: PULSE = 1.0 defines a spray whose mass is injected in a single half sine-wave pulse. PULSE = 2.0 defines a spray whose mass is injected in a single square-wave pulse. PULSE = 3.0 defines a pulse whose velocity profile is supplied by VELINJ, which here is a table of NUMVEL (up to 100) entries. For this case, subroutine RINPUT calculates the total fuel mass predicted by the velocity table, using the sums of the nozzle areas ANOZ( ), and corrects the velocity profile up or down by the ratio of mass desired to mass predicted. This case is appropriate for hole-type nozzle injectors, rather than pintle nozzles, as the nozzle areas at present are assumed to remain constant with time.

For the cases PULSE = 0.0, 1.0, or 2.0, however, NUMVEL should be 1, the velocity being assumed constant at VELINJ(1). (A comment in subroutine INJECT lists a one-line modification to allow the sinusoidal case to have a sinusoidal velocity profile, in addition to the mass profile.) In addition, ANOZ( ) can be input simply as 1.0 for these three cases.

Injection commences at crank angle CA1INJ and has a duration of CADINJ degrees. CA1INJ is given in degrees ATDC, and hence usually has a negative value. Injection may also be controlled in terms of problem time, which is more convenient for continuous sprays and other nonengine applications. In this case, the appropriate starting time and duration are supplied as T1INJ and TDINJ. For a continuous spray, TDINJ should be set to ∞. A negative value for T1INJ indicates to the code that CA1INJ and CADINJ are to be used instead.

3. Particle Radius. Either a distribution of particle radii (INJDIST = 1) or particles of a fixed radius (INJDIST = 0) may be injected. When a drop size distribution is specified, we sample randomly from a distribution about the Sauter mean radius (SMR), input in cm. The distribution is patterned after experimentally observed data and is described in Appendix D.

In the case of a fixed radius, the value supplied under the name SMR is interpreted as the fixed nozzle radius, in cm. The aerodynamic breakup model should be used (BREAKUP = 1.0) to create a spectrum of sizes, although the breakup model is appropriate in general for both the INJDIST = 1 and INJDIST = 0 options. AMPO is the initial amplitude of droplet oscillation at the injector, based upon a Weber number estimate.

After injection, the particle size is reduced by breakup, if BREAKUP = 1.0, and through evaporation, if EVAPP = 1.0. Liquid fuel particles disappear as they evaporate, as subroutine REPACK destroys any particles whose mass falls below 10^{-3} of the mass of an injected particle. Conversely, particle size increases through coalescence from collisions, if KOLIDE = 1. Particles are split in REPACK into two identical particles, each with half the number of droplets, if their mass grows to twice the mass of an injected particle.

4. Stochastic Injection. When using the large timesteps possible in KIVA-II, the tendency is to inject in bursts, resulting in discrete clumps of computational particles. These clumps may then move more than one cell per cycle, causing an uneven coupling with the mesh cells and vertices along the particle path.

To mitigate this source of computational inaccuracy, KIVA-II injects each particle at some random point along the particle trajectory **behind** the injector. The particles are then immediately moved forward in subroutine PMOVTV to their effective initial locations. This stochastic injection offers a smoother and more uniform particle distribution, resulting in improved coupling with the mesh, and better statistics when spray particle positions and radii are averaged over time.

H. Spark Ignition

Spark ignition is provided by a special energy deposition at the end of the kinetic chemistry subroutine CHEM. The ignition window is specified either by crank angle (CA1IGN to CA1IGN + CADIGN) or problem time (T1IGN to T1IGN + TDIGN), in a manner analogous to the specification of the injection window discussed above. During ignition, the specific internal energy in the specified ignition cell(s) is increased by a factor of (1.0 + XIGNIT*DT) on each timestep. If the temperature in the ignition cell(s) reaches 1600 K before the end of the ignition window, as it ordinarily does, then the special energy deposition is terminated at that point. The ignition cell(s) are specified in the input data as the DO-loop range defined by IIGNL(1) to IIGNR(1), JIGNF(1) to JIGND(1), and KIGNB(1) to KIGNT(1). This allows the choice of a single cell or a block of cells. If JIGNF(1) = 1 and JIGND(1) = NY, ring ignition will result in a 3-D cylindrical run. Dual ignition is an optional feature, as a second ignition region may be specified by setting IIGNL(2), IIGNR(2), JIGNF(2), JIGND(2), KIGNB(2), and KIGNT(2) to nonzero values.

Spark ignition in our UPS-292-SC engine calculations[7] was accomplished in a unique manner. In the UPS engine, a pencil spray of fuel impinges on a spark plug which commences firing when the spray starts and continues firing for 35 crank angle degrees. Our procedure was to not allow fuel and oxygen to coexist in the cells containing the spark plug tip during the 35° period. Each cycle, any available fuel or oxygen was consumed through direct conversion to CO_2 and H_2O in accordance with the oxidation reaction, with appropriate heat release. While quite satisfactory, this procedure is applicable only in such specialized circumstances and would be totally inappropriate in more general applications in which the fuel has experienced significant evaporation and premixing prior to ignition.

I. Initial Bessel Function Swirl Profile

Internal combustion engines are designed to impart a significant amount of swirl in the incoming air, to aid in turbulent mixing and enhance combustion efficiency. The simplest model assumes that the swirl velocity has a wheel-flow profile, but this is not usually a realistic assumption, as the turbulent wall boundary layer forces the swirl velocity to decrease in the wall region. From experimental observation, modelers have determined that a Bessel function profile more accurately represents the flow.

Figure 27 illustrates the Bessel function velocity profile provided in KIVA-II and compares it with wheel flow for the same swirl number. The quantity α (input as SWIPRO) is a dimensionless constant that defines the initial azimuthal velocity profile and lies between 0.0 (the wheel flow limit) and 3.83 (zero velocity at the wall). A value suggested by Wahiduzzaman and Ferguson[58] for typical engine applications is about 3.11. We define our Bessel function profile to give the same angular momentum as wheel flow with the same swirl number. Thus the initial slope of the $\alpha = 3.11$ curve is necessarily higher than the corresponding slope for wheel flow.

A second input quantity, SWIRL, is the initial swirl ratio of air r.p.m. to crankshaft r.p.m. When viewed from above, the swirl is clockwise if SWIRL > 0 and counterclockwise if SWIRL < 0.

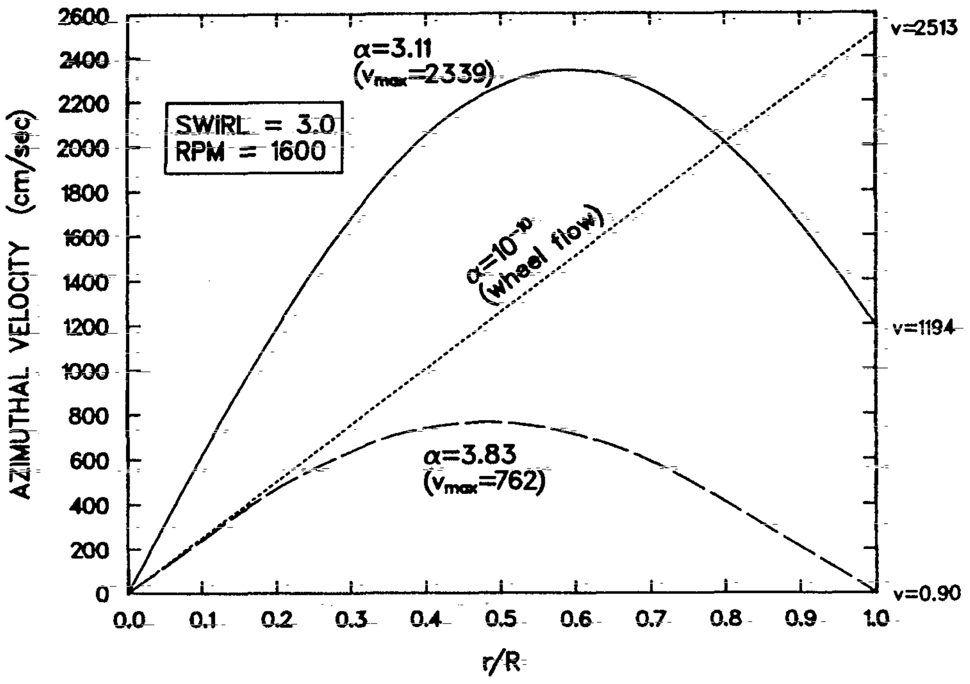


Fig. 27. Bessel function swirl velocity profile provided in KIVA-II setup.

J. Fuel Library

The release version of KIVA-II assumes that 12 chemical species are present, although this number can be increased or decreased as needed. The 12 species, appropriate for a variety of combustion applications, are 1 = fuel, 2 = O_2, 3 = N_2, 4 = CO_2, 5 = H_2O, 6 = H, 7 = H_2, 8 = O, 9 = N, 10 = OH, 11 = CO, and 12 = NO. The sample input data deck contains kinetic and equilibrium chemistry data that correspond to these 12 species. Enthalpy tables are required for all species, and because species 2-12 have been defined, enthalpies for them are provided in DATA statements in subroutine RINPUT.

What remains is to define the fuel, species 1, which requires a number of other properties in addition to its enthalpy. For this purpose, KIVA-II contains a library of the thermophysical properties of 12 common hydrocarbon fuels, embodied in BLOCK DATA FUELIB. At present, the 12 fuel choices are

methane	(CH_4),
propane	(C_3H_8),
n-heptane	(C_7H_{16}),
n-octane	(C_8H_{18}),
n-dodecane	($C_{12}H_{26}$),
~~n-tridecane~~	~~($C_{13}H_{28}$),~~
n-tetradecane	($C_{14}H_{30}$),
n-hexadecane	($C_{16}H_{34}$),

	acetylene	(C_2H_2),
	ethylene	(C_2H_4),
	benzene	(C_6H_6),
and	diesel fuel	(DF2).

To select from this list, the user supplies the fuel formula as a mnemonic in the input data. Subroutine RINPUT then calls FUEL, which performs the following steps:

1) The fuel formula is correlated with an entry in a tabular set of mnemonics, to verify that the requested fuel is one of the 12 available and simultaneously to obtain an index for accessing the data associated with that particular fuel.
2) The fuel enthalpy table is loaded.
3) The molecular weight, critical temperature, and heat of formation are defined for the fuel.
4) If the fuel is normally gaseous (methane, propane, acetylene, or ethylene), the input flags for parcel evaporation, collisions, and breakup (EVAPP, KOLIDE, and BREAKUP) are checked. If they are all off, FUEL has completed its task and control returns to RINPUT. If any of the three flags is on, the run is terminated with a message that the input data are inconsistent.
5) If the fuel is a liquid, several more quantities are set. These are the tables of liquid latent heat of vaporization and liquid vapor pressure, for use in droplet evaporation, along with a table of liquid viscosity, for use in droplet breakup. In addition, the slope and y-intercept of the surface tension vs temperature linear fit are defined. These are based on the surface tension at 350 K, which is a typical fuel temperature, and the critical temperature of the fuel. These are used in the surface tension linear fits in droplet collisions and breakup. Finally, the coefficient for fuel diffusivity in air is set, for use in droplet evaporation.

The tabular information in FUELIB has been drawn from a number of sources.[38,51-53,59] In several cases, these sources do not contain sufficient data to complete the tables at one end or the other of the temperature range. This is true at the low temperature end for some of the latent heat tables, and at the high temperature end for some of the liquid viscosity tables. The comment lines in FUELIB identify the sources of the various tabular data and note where extrapolations from published data were made in order to complete the tables.

The user should thus be aware that development of the library for less-completely documented fuels is an ongoing project. Contributions would be welcomed for inclusion in future code versions, both for improving the data for the existing 12 fuels and for fuels not presently included.

The final consideration concerning fuel is its inclusion in a kinetic chemical reaction. To accomplish this, the user must supply an appropriate input data set, as required for each kinetic reaction. The data set is comprised of forward and backward pre-exponential factors, activation temperatures and temperature exponents, along with stoichiometric species coefficients on the left and right sides of the reaction and exponents of species concentration in both the forward and backward rates of the reaction. Kinetic chemistry is discussed in detail in Appendix I.

K. Inflow and Outflow Boundaries

Some engine and nonengine applications have geometries that require use of inflow and outflow boundaries. Accordingly, we have provided inflow and outflow options of a type that may be encountered. If input flag BOTIN = 1.0, the entire bottom boundary of the mesh is an inflow boundary. If BOTIN = 0.0, then a rigid wall is used for the bottom boundary. At the inflow boundary the normal velocity WIN is specified. Either one, or both, of the right and top boundaries of the mesh may be outflow boundaries. Input flag RTOUT controls the right boundary specification, with RTOUT = 1.0 denoting outflow and RTOUT = 0.0 denoting a rigid wall. The boundary lies between the I = NX and I = NXP planes of cells. Input flag TOPOUT performs an analogous role for the top boundary, which is the boundary between the K = NZ and K = NZP planes. At outflow boundaries, the pressure is specified to be input value PAMB. At both boundaries, we assume the flow is subsonic. We now describe the inflow and outflow options in detail, tell what subroutines are modified to incorporate them, and tell how they may be modified for supersonic flows. Although the inflow and outflow options are rather limited, it is hoped that they can be used as a guide to incorporating boundaries with different locations and different conditions.

In addition to the normal velocity WIN, at inflow boundaries we specify reference species mass densities SPDIN0(M), specific turbulent kinetic energy TKEAMB, and turbulence length scale SCLAMB (~ $k^{3/2}/\varepsilon$). The reference densities are at reference pressure PAMB; the values that are actually imposed at the inflow boundary are obtained from

$$SPDIN(M) = SPDIN0(M)\,(P/PAMB)^{1/\gamma_{amb}} ,$$

where P is the computed pressure in the cell with K index equal to one immediately above the inflow cell and γ_{amb} is the ratio of specific heats of the inflow mixture. Thus at an inflow boundary we are imposing the species mass fractions and the entropy of the incoming fluid and obtaining the pressure by extrapolation and the densities from an isentropic gas equation of state. The inflow internal energies are obtained from pressure P and densities SPDIN(M) using the equations of state (10) and (11). One inflow tangential velocity component is specified and the other is calculated. In cylindrical mesh geometries, the azimuthal or swirl component is specified to be zero. In planar geometries, the y-component of the velocity is specified to be zero.

We note here that the values SPDIN are used for computing mass fluxes but are not used to compute diffusive fluxes at the inflow. In fact, the diffusive fluxes of all cell-centered quantities are taken to be zero at the inflow boundary. This will introduce little error if the Peclet numbers, which give the ratios of the convective to diffusive fluxes, are greater than unity. If this condition is not satisfied, then the appropriate coding changes should be made to calculate diffusive fluxes at the inflow boundary.

The above inflow boundary assumes subsonic flow. If the flow is supersonic, then all thermodynamic conditions and all three components of the velocity should be specified.

At the outflow boundary, the pressure PAMB is specified a distance DISTAMB outside the outflow boundary. This is accomplished in the following manner. For a regular cell face α that lies on the outflow boundary, one of the faces γ of cell-face control volume α also lies on the outflow boundary. In differencing Eq. (86) to find $(uA)_\alpha{}^B$, we take the pressure on cell face γ to be

$$p = \frac{\delta z/2 \cdot PAMB + DISTAMB \cdot [\phi_p p^B + (1 - \phi_p) p^n|_{ijk}}{\delta z/2 + DISTAMB} ,$$

where $\delta z = V_{ijk} / |\mathbf{A}_\alpha|$ and cell (i, j, k) is the interior cell containing face α. When DISTAMB $= 0$, this gives $p =$ PAMB. This is a true specified pressure condition, which unfortunately reflects acoustic waves perfectly and can affect the upstream flow in subsonic calculations. Taking DISTAMB to be a characteristic dimension of the computational region greatly reduces acoustic wave reflection at the outflow boundary. This also allows more rapid convergence when computing steady state flows, by reducing the problem time required to reach steady state. In addition to the above pressure specification, to implement an outflow boundary we set the vertex velocities equal to those one vertex in from the boundary: $\mathbf{u}_{i,j,NZP} = \mathbf{u}_{i,j,NZ}$ at an outflow boundary across the top, and $\mathbf{u}_{NXP,j,k} = \mathbf{u}_{NX,j,k}$ at an outflow boundary across the right. We also use donor cell differencing to compute outflow convective fluxes.

When the flow at the outflow boundary is directed out of the computational region, no further specifications are needed. Occasionally, however, the pressure condition will generate velocities directed into the computational mesh. When this occurs, it is also necessary to prescribe the mass fractions, entropy, turbulent kinetic energy, and turbulent length scale of the incoming fluid. This is accomplished through input reference species densities SPDAMB(M), referenced to pressure PAMB. The inflow densities and internal energy are then found in a manner analogous to that used at the inflow boundary. The inflow turbulence quantities are taken to be TKEAMB and SCLAMB.

No modification of this outflow boundary treatment is required when the flow is supersonic. Although specification of the pressure is incorrect in this case, the errors incurred by this specification will not propagate upstream in the supersonic flow.

A number of subroutines are modified to allow for the inflow and outflow options. Vertex and cell face velocities are prescribed in subroutines BC and BCFC. Since ε is not specified near inflow or outflow boundaries using Eq. (53), this specification must be disabled in subroutines BCEPS and BCRESE. Similarly, law-of-the-wall velocity and temperature conditions should be disabled in subroutine LAWALL. Convective fluxes are modified in subroutine CCFLUX, and the pressure is specified at the outflow boundary in subroutines PEXDIF and RESP. In making any modifications for inflow or outflow boundaries, the user should consider these subroutines carefully and use the existing code as a guide.

L. Output

Monitor prints are produced on the user's remote terminal (our unit 59) at least every 25th cycle. The variables KIVA-II prints are listed in subroutines NEWCYC and TIMSTP. In addition, informational and error messages are always sent to the user's terminal. Except for the above, all KIVA-II output is written to file PLOT, which can be scanned on a CRT-equipped terminal and/or disposed to the operating system to be processed onto microfiche. A four-line monitor print is written on PLOT every cycle, in addition to copies of any messages sent to the remote terminal.

Numerical cell data are optionally available, being produced only if the input flag LPR = 1. The WRITE statements in subroutine LNGPRT indicate the variables listed, which may easily be changed. Because of the vast amount of numbers a multidimensional code produces, we almost always set LPR = 0 and rely on plots, except when debugging. Even then, an interactive debugging utility is generally more useful than a blanket cell print because the utility can print selected numbers at the user's terminal at any desired point in the calculation.

Graphics are the most useful form of output, and information for computer-generated plots is also written to file PLOT. This is followed by a short summary of system totals, computed by subroutine GLOBAL.

Subroutine FULOUT is responsible for calling the various plot and print subroutines. Output is automatically provided for cycles 0 and 1, and thereafter at cyclic intervals (NCFILM), time intervals (TWPLT), or crank angle intervals (CAFILM), as specified in the input data.

KIVA-II can provide three different types of plots -- zone and spray, velocity vector, and contour. The first set of zone plots are simple two-dimensional views through (x-z) plane j = 1 and the opposite (or closest to opposite) j plane if 3-D. The first plot is a mesh cross section that shows all zone edges. The second plot is the mesh outline plus only those spray particles currently in the j = 1 plane and the opposite j plane if 3-D. The third plot is again the mesh outline, but with all spray particles plotted regardless of their j plane. A fourth zone plot is an overhead view of the mesh that also includes all the spray particles. If there are no spray particles present at the time the plots are drawn, the second and third plots are omitted, as is the fourth plot unless the output is for cycle 0. For 2-D applications, only the first two zone plots described above are created, as the third and fourth plots would be meaningless.

The remaining zone plots, in addition to all the velocity vector and contour plots, employ our perspective plot logic. For 2-D applications this is automated, but for 3-D applications the logic requires specification of input data. This is all described in Sec. IV.M.

The plotting routines provided in KIVA-II are adequate for the needs of many users. Because these routines require only graphics primitives (point plotting and vector drawing), they are adaptable to other systems without too much difficulty. Other users, however, will want to use KIVA-II with their own graphics post-processors. Accordingly, we have provided the basic connective linkage for post-processing in KIVA-II, which will simplify the task. It is controlled by the following three input quantities:

- IPOST is the post-processor flag, where 0 means no post-processor file is desired, 1 means to make post-processor dumps onto file TAPE9 starting with this run, and 2 means to continue dumping onto a previous TAPE9, which must exist in local file space.
- CADUMP is the crank angle at which to start dumping onto TAPE9.
- DCADMP is the crank angle interval between dumps, after crank angle CADUMP has been reached.

TAPE9 is initialized or resumed by subroutine DMPINT; writing of file TAPE9 is performed by subroutine DMPOUT. Because each user of a post-processor has their own specific needs, the variables written in the DMPOUT routine provided in KIVA-II are intended only as an example of those quantities that are generally useful.

M. Perspective Plots

The development of three-dimensional computing techniques has brought particular difficulties in the effective presentation of results. We have attempted to display as much useful information as possible by offering perspective pictures, made in the same manner as a photographic record of a three-dimensional scene on a two-dimensional negative.

The perspective plot logic described below is used in producing zone, velocity vector, and contour plots for both 2-D and 3-D applications. However, for 2-D applications, KIVA-II subroutine RINPUT automatically specifies simple head-on 2-D views for each quantity plotted, and the user need not be concerned with supplying perspective view data.

The plot-generating subroutines ZONPLT, VELPLT, PVPLOT, and CONTUR can create a variety of perspective views of the computing grid with spray particles, in addition to fluid velocity vector, spray parcel velocity vector, and contour plots of selected cell variables. For the grid plots, some selected number of the three bounding faces of our pseudo-polar grid are to be drawn. For the vector and contour plots, however, we simply outline the edges of the grid and present a selected plane of vectors or contours within this framework.

Only a few simple concepts are required to describe how a perspective view is generated. The fundamental concept is that of the transparent image plane, analogous to the film in a camera, on which the perspective image is traced (Fig. 28). To minimize distortion, however, we chose an image plane perpendicular to the average line of sight, which extends from some point (x_c,y_c,z_c) near the mesh center out to the eye point (x_e,y_e,z_e), assumed to lie well outside the mesh.

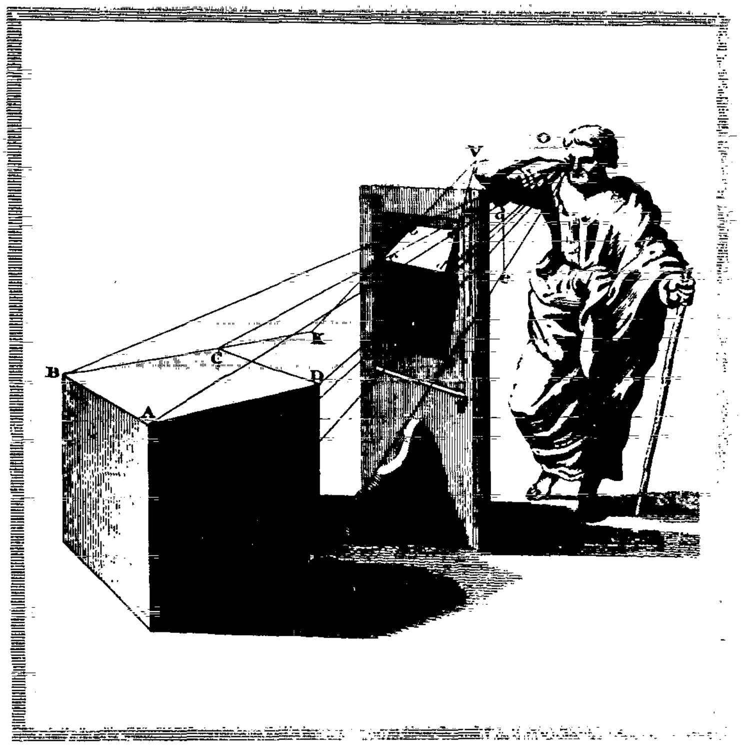


Fig. 28. Construction of perspective picture on an image plane. From *New Principles of Linear Perspective* by Brook Taylor, published in 1715.

A pair of perspective coordinates (ε,η) relates a point on the image plane to a point (x_o,y_o,z_o) in the mesh region and can be obtained by performing suitable transformations. In our case, these are

$$\xi = \bar{x}_e - \bar{y}_e\left(\frac{\bar{x}_o - \bar{x}_e}{\bar{y}_o - \bar{y}_e}\right),$$

$$\eta = \bar{z}_e - \bar{y}_e\left(\frac{\bar{z}_o - \bar{z}_e}{\bar{y}_o - \bar{y}_e}\right),$$

in which

$$\bar{x} = (x - x_c)\cos\theta + (y - y_c)\sin\theta\ ,$$

$$\bar{y} = \cos\phi[(y - y_c)\cos\theta - (x - x_c)\sin\theta] - (z - z_c)\sin\phi\ ,$$

$$\bar{z} = \sin\phi[(y - y_c)\cos\theta - (x - x_c)\sin\theta] + (z - z_c)\cos\phi\ .$$

The two angles θ and ϕ measure the rotation and the tilt, respectively, of the line of sight with respect to (x,y,z) space as indicated in Fig. 29.

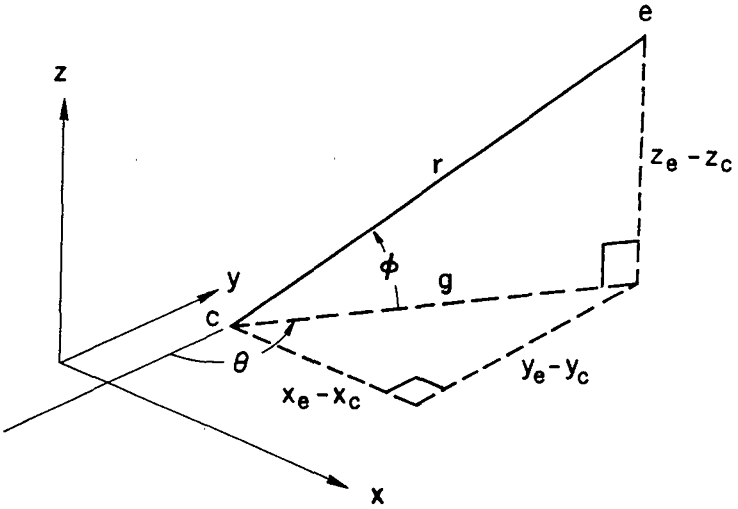


Fig. 29. Relationship of the rotation angles ϕ and θ to the (x,y,z) coordinates.

From the figure, it is evident that

$$g = [(x_e - x_c)^2 + (y_e - y_c)^2]^{\frac{1}{2}} \ ,$$

$$r = [(z_e - z_c)^2 + g^2]^{\frac{1}{2}} \ ,$$

from which

$$\phi = Arcsin\left(\frac{z_e - z_c}{r}\right) ,$$

$$\theta = Arcsin\left(\frac{x_e - x_c}{g}\right) ,$$

if viewed from the front, $(y_e - y_c) < 0$. If viewed from behind, $(y_e - y_c) > 0$, then

$$\theta = -\left[Arcsin\left(\frac{y_e - y_c}{g}\right) + \frac{\pi}{2}\right] \quad for \quad (x_e - x_c) < 0 \ ,$$

$$\theta = Arcsin\left(\frac{y_e - y_c}{g}\right) + \frac{\pi}{2} \quad for \quad (x_e - x_c) > 0 \ .$$

The origin and scale of the (ξ,η) coordinates are not required because we choose a constant shift that centers the mesh outline in the plot frame and a constant scale factor that maximizes the size of the mesh drawn in the available plot frame area. This scaling is done in KIVA-II by first computing the (ξ,η) coordinates of all mesh vertices and then testing for their maximum and minimum values.

Because a straight line segment in three-dimensional space transforms into a straight line segment in the perspective view, only the end points require transformation according to the above equations, and the resulting points are connected by a straight line.

In KIVA-II, we have defined $x_c = y_c = 0.0$, and z_c = (z of the cylinder head minus half the stroke), assuming code applications will be to internal combustion engines. For 3-D plane coordinates, the user should redefine x_c, y_c, and z_c to lie at the center of the mesh. These statements can be easily changed in KIVA-II subroutine RINPUT. Currently, we allow up to five views each for grid plots, velocity vector plots, and contour plots, as per the parameter LV at the beginning of the code. Required input quantities for each individual 3-D plot are as follows.

Grid Plot: XEZ, YEZ, ZEZ, IFACE(6), IEDG, in which the first three refer to (x_e,y_e,z_e) in (x,y,z) space. IFACE is a set of six integer flags that correspond to the six bounding faces of the logical mesh: left, right, front, derriere, bottom, and top, respectively, where a value of 1 means to include and a value of 0 means to exclude the face in the plot. For the 3-D pseudo-polar mesh, appropriate perspective-view values are 0, 1, 0, 0, 1, 1 to draw the right (cylinder wall), bottom (piston face), and top (cylinder head). Finally, IEDG = 0 will draw line segments between all vertices lying on selected faces, and IEDG = 1 will draw only the outlines of the selected faces. The IEDG feature is intended for use in 3-D plane coordinates.

Velocity Vector Plot: XEV, YEV, ZEV, ISLV, JSLV, KSLV. Again, the first three refer to (x_e,y_e,z_e). The last three are integers of which only one can be nonzero. The nonzero choice identifies the I, J, or K index of some plane in the logical mesh where we make a "slice" and draw vectors of the vertex velocities as they would appear normalized in the plane. For an I slice [$0 < \text{ISLV} \leq (\text{NX} + 1)$, JSLV = KSLV = 0], the plane is in (y,z) space. In a 3-D pseudo-polar grid, an I slice plot would be a wraparound at some radius and probably would not be too useful. Of real value for this grid, however, are J and K slice plots. For a J slice [$0 < \text{JSLV} \leq (\text{NY} + 1)$, ISLV = KSLV = 0], the plane is in (x,z) space. In a 3-D pseudo-polar grid, the velocity vectors for the j plane JSLV are drawn on the right side of the frame, and vectors for the opposite (or closest to opposite) j plane are drawn on the left side of the frame. For a K slice [$0 < \text{KSLV} \leq (\text{NZ} + 1)$, ISLV = JSLV = 0], the plane is in (x,y) space. In all cases, the vector length drawn is scaled to the maximum velocity in the plane being plotted.

Contour Plot: XEC, YEC, ZEC, ISLC, JSLC, KSLC. These are analogous to the quantities defined above for the velocity vector plots, except that individual contour plots are drawn for each view specified. Contour plots may be drawn for up to 26 different cell variables, selected according to the set of 26 binary flags in the ICONT input line. The DATA statement in subroutine CONTUR and the comments in subroutine FULOUT describe the sequence. The same remarks concerning slices apply to contour plots, but because contour plots relate to cell-centered variables rather than velocities, the "+ 1" on the range is inapplicable. As provided here, contour plots are composed of vector segments joining points of equal value and are linear in contour increment. Contours are automatically connected across the center (at I = 1) in both J and K slice views in the 3-D pseudo-polar grid, and are drawn to the j = 1 and j = NYP boundaries in K slice views of a 3-D sector mesh.

In both velocity vector and contour plots, it is best to define the eye point such that the line of sight will be as perpendicular as possible to the selected plane. When the eye is not located too far above the mesh, a more pleasing appearance is often produced by setting $z_e = z_c$. As a result, $\phi = 0$, which keeps the image plane perpendicular to the (x,y) plane.

Often we prefer to eliminate perspective entirely. For example, a set of (x,y) velocity vector and contour plots at various k-plane levels viewed from straight overhead generally proves more useful than views from some artistic angle. To produce a straight overhead view, set $x_e = y_e = 0.0$ and $z_e = \infty$ (10^{10} is adequate). Note, however, that $y_e = 0.0$ will cause the 3-D pseudo-polar plot to be rotated 45° such that the periodic boundary (J = 1) will appear at the 4:30 position. To mentally orient ourselves, we prefer to always have the periodic boundary at the 3 o'clock position. This is achieved automatically if we instead specify $y_e = -$RPO(NPO) rather than $y_e = 0.0$.

The coordinate system on our CRT face has its origin (0,0) at the upper left corner, and the values of the two raster indices increase to the right and down to maximum values of 1024. A conversion from image-plane coordinates to corresponding CRT coordinates is thus required for all plotting. In KIVA-II, the view on the image plane extends from ξ_L to ξ_R and from η_B to η_T, with CRT counterparts FIXL to FIXR and FIYB to FIYT. To leave room for labeling, we limit the available CRT face to 1022 raster points wide and 900 points high. Within this rectangular region, we maximize the image-plane view drawn, which requires the ratio

$$XD = \left(\frac{\xi_R - \xi_L}{\eta_T - \eta_B} \right) .$$

If the image-plane view is higher than it is wide [that is, XD < (1022/900)], then the CRT plot range is given by

$$\text{FIXL} = 511 - 450(\text{XD}) ,$$
$$\text{FIXR} = 511 + 450(\text{XD}) ,$$
$$\text{FIYB} = 900 ,$$

and

$$\text{FIYT} = 0 .$$

Conversely, if the image-plane view is wider than it is high [XD > (1022/900)], then the CRT plot range is

$$\text{FIXL} = 0 ,$$
$$\text{FIXR} = 1022 ,$$
$$\text{FIYB} = 900 ,$$

and

$$\text{FIYT} = 900 - (1022/\text{XD}) .$$

In either case, we can now calculate the two conversion factors

$$\text{XCONV} = (\text{FIXR} - \text{FIXL})/(\xi_R - \xi_L)$$

and

$$YCONV = (FIYB - FIYT)/(\eta_T - \eta_B) \ ,$$

required to translate image-plane coordinates for the view being drawn. Thus, if (ξ,η) represents some point on the image plane, the corresponding CRT position (IX,IY) is given by

$$IX = FIXL + (\xi - \xi_L)(XCONV)$$

and

$$IY = FIYB - (\eta - \eta_B)(YCONV) \ .$$

N. Chopper

In engine calculations, the cells can become very thin in the z-direction in the squish region between the piston face and cylinder head, resulting in severe timestep restrictions. To alleviate this condition, subroutine CHOP is used to strip out or add planes of cells across the mesh above the piston, thus providing a control on cell heights in the squish region.

When the chopper is used, grid lines are required to be vertical (z-direction) through the squish region, although radial grid lines are not required to be horizontal. An example of a mesh that meets these criteria is shown in Fig. 22. The volume-of-overlap logic in the chopper accounts only for vertical displacement of vertices. A truly arbitrary volume-of-overlap algorithm for three-dimensional space would be significantly more complex and has not been required for any application so far.

The input quantity NCHOP specifies the minimum number of planes to be left in the squish region and is usually equal to 2 or 3. CHOP automatically removes planes on the compression stroke and restores them on the power stroke. The parameter LNZP prevents the number of cells added on the power stroke from exceeding the available storage.

The quantity DZCHOP, calculated in RINPUT, is a function of NCHOP. The δz's in the squish region are always uniform, and each cycle δz is compared to DZCHOP to determine if a plane is to be added or deleted. The current algorithm for DZCHOP is fairly conservative: in typical engine calculations that begin at ATDC = −90°, chopping will occur between about −45° to −27°, at which time our specified minimum number of planes (NCHOP = 3) is reached.

O. Dump and Restart

Provision is made for running a problem in segments. If the input quantity TLIMD = 1.0, the code compares the job time limit to the time used and writes a restart dump on file TAPE8 when less than 90 seconds of time remain, and the run terminates. In addition, a dump is produced on TAPE8 every NCTAP8 cycles during a run. This feature is provided to minimize time lost due to system or hardware crash. Each time a dump is written TAPE8 is rewound, so that the last dump is the only one saved.

To continue the problem on a subsequent run, the quantity IREST in input data file is set equal to the dump number. The dump file is read as TAPE7. After reading the restart dump, KIVA-II reads the remainder of the input data file. This allows the user to modify data in midcalculation if desired. If this is done, care must be taken not to introduce inconsistencies.

KIVA-II calls a built-in random number generator at several places in the spray subroutines. This generator is intended to be as portable as possible for use on other computers having different word lengths and attempts to give the same number sequence regardless of computer. Occasionally, the user may find it desirable to restart from a dump and calculate identical numbers to those that would result from a single longer run with no restart in the middle. To allow for this, KIVA-II includes the current seeds RANB and RANS for the random number generator in the dump data and resets them as part of the restart procedure.

ACKNOWLEDGMENTS

The authors wish to give special acknowledgment to T. L. McKinley of Cummins Engine Company, who helped in formulating some of the finite difference equations and their numerical solution procedure and who recommended many additional improvements to the computer program. We also thank many users of the preliminary versions of the code who have found "bugs" and suggested improvements. Among these users are John Ramshaw of Idaho National Engineering Laboratory, Rolf Reitz and Ramachandra Diwakar of General Motors Research Laboratories, and Kenneth Marx of Sandia National Laboratories. Finally we thank Adrienne Rosen for her good-natured perseverance in typing the manuscript.

APPENDIX A

DETERMINATION OF THE PGS PARAMETER

The objective of the pressure gradient scaling (PGS) method is to scale up the magnitudes of the pressure fluctuations in far subsonic flows and thereby increase computational efficiency without changing other flow features of interest. The method is implemented by solving equations that are modified only in that the pressure gradient term in the momentum equation is multiplied by a factor $1/\alpha^2$. The quantity α is called the PGS parameter and is constrained to be greater than or equal one. It is shown in Ref. 36 that if the solution to the modified system has pressure fluctuations that are small, in the sense that $\delta p_\alpha/\bar{p}$ is small compared to unity where $\bar{p}$ is an average pressure, then the modified system will have nearly the same solution as the unmodified system, except that the pressure fluctuations will be increased by a factor α^2; that is

$$\delta p_\alpha / \bar{p} \sim \alpha^2 , \qquad \text{(A-1)}$$

where δp_α are the pressure fluctuations of the modified system.

Since in many problems $\delta p/\bar{p}$ is approximately equal to the square of the Mach number,[60] one suspects that the PGS method is in effect increasing the Mach number by a factor of α. That this is true can be shown by examination of the acoustic wave equations with the factor $1/\alpha^2$ multiplying the pressure gradient term. It is easily shown that sound speeds are reduced by a factor $1/\alpha$. Computational efficiency is improved because the efficiency of many numerical methods for solving the pressure equation is improved when the sound speed Courant number $c\Delta t/\Delta x$ is lowered,[36] and the PGS method lowers the Courant number by the factor $1/\alpha$. The PGS method should not be used in problems where it is important to calculate acoustic waves accurately. In many subsonic problems, however, the acoustic mode is not important and the PGS method can be used to enhance computational efficiency.

We now discuss the method for choosing α. The derivation of Ref. 36 shows that α cannot vary in space but can be time-dependent. It follows from the above brief discussion that to optimize computational efficiency one should take α as large as possible while still maintaining $\delta p/\bar{p}$ small compared to unity. In KIVA we choose α to maintain $\delta p_\alpha/\bar{p} \leq 0.04$, unless the value of α so chosen becomes less than unity. In this case, the pressure fluctuations of the unmodified equations, which are those that pertain when $\alpha = 1$, have become larger than 0.04, and we set $\alpha = 1$ and thereby deactivate the method.

In more detail, the algorithm for choosing α is the following. Each cycle we calculate

$$\left(\frac{\delta p_\alpha}{\bar{p}}\right)^n = \max_{ijk} \frac{|p^B_{ijk} - \bar{p}^n|}{\bar{p}^n}, \tag{A-2}$$

where

$$\bar{p}^n = \frac{\sum\limits_{ijk} p^B_{ijk} V^B_{ijk}}{\sum\limits_{ijk} V^B_{ijk}} .$$

According to Eq. (A-1), in order to make the maximum relative pressure fluctuation on the next cycle equal to 0.04, we should take $\alpha^{n+1} = \alpha^*$, where

$$\alpha^*/\alpha^n = [0.04/(\delta p_\alpha / \bar{p})]^{\frac{1}{2}} . \tag{A-3}$$

We have found, however, that taking $\alpha^{n+1} = \alpha^*$ can result in severe oscillations in the computed values of α. The algorithm in KIVA, which works well in practice, is to take

$$\alpha^{n+1} = \begin{cases} \alpha^* & \textit{if} \quad \alpha^* < \alpha^n \\ \alpha^n + \dfrac{\Delta t}{\tau_r}(\alpha^* - \alpha^n) & \textit{if} \quad \alpha^* > \alpha^n \end{cases} . \tag{A-4}$$

Thus if $\alpha^* > \alpha^n$ we allow α to relax to its desired value α^* with relaxation time τ_r. The time τ_r is taken to be the maximum of $20\Delta t^n$ and four times a characteristic acoustic wave transit time across the computational mesh, based on a scaled average sound speed $\bar{c}/\alpha^n$. Hence, we take

$$\tau_r = \max\left\{20\,\Delta t^n\ ,\ \frac{4\,L\alpha^n}{\bar{c}}\right\}, \tag{A-5}$$

where

$$\bar{c} = \sqrt{\frac{1.4\ \bar{p}}{\bar{\rho}}}$$

and $\bar{p}$ and $\bar{\rho}$ are the volume-averaged pressure and mass density in the computational mesh. If the value of α^{n+1} from Eq. (A-4) is less than unity, we set $\alpha^{n+1} = 1$.

APPENDIX B

TURBULENT BOUNDARY LAYER TREATMENT

Wall functions are analytic solutions to simplified turbulence equations and are used to infer wall shear stresses and heat losses in lieu of numerical solution near walls of complete turbulence equations. Numerically, one accomplishes this by matching the computed fluid velocities and temperatures at grid points closest to walls to the wall functions, which then determine the wall shear stresses and heat losses. Numerical solution of complete turbulence equations is usually impractical because one cannot provide sufficient resolution. Although it makes computations affordable, the alternative of using wall functions can introduce large errors because in practice many of the assumptions are violated that one needs to obtain analytic solutions. In the first section of this appendix, we derive the wall functions used in KIVA-II and give the assumptions used in the derivation. In the second section, we tell how the wall function approach is implemented numerically in KIVA-II.

I. DERIVATION OF WALL FUNCTIONS

In this section we first give the assumptions that are needed to derive the wall functions and the simplified equations that result from making these assumptions. We next nondimensionalize the equations and thereby introduce a dimensionless wall heat loss ζ. The quantity ζ is equal to $(J_w Pr)/(\rho u^* c_p T_w)$, where J_w is the wall heat loss per unit area, u^* is the shear speed, and T_w is the wall temperature. It is assumed that ζ is small compared to unity, and we obtain a perturbation solution for the boundary layer profiles with ζ as the perturbation parameter. Finally, we introduce a change of independent variable that makes the wall functions easier to implement numerically.

We use the following assumptions to derive the wall functions:

1. the flow is quasi-steady;
2. the fluid velocity is directed parallel to a flat wall and varies only in the direction normal to the wall;
3. there are no streamwise pressure gradients;
4. there are no chemical reactions in the gas or on the wall surface;

5. there are no spray sources;
6. the dimensionless wall heat loss ζ is small compared to unity;
7. Reynolds numbers are large (i.e., $\mu >> \mu_\ell$ where μ_ℓ is the laminar viscosity); and
8. Mach numbers are small, so that dissipation of turbulent kinetic energy is a negligible source to the internal energy.

The above list leads to provisional wall functions that closely resemble those commonly used in conjunction with the $k-\varepsilon$ turbulence model.[26] Assumptions 1-6 are frequently violated at grid points near walls in internal combustion engine calculations. Some of the assumptions can be made more valid by providing more resolution near walls. For example, the measure of the flatness of a wall is the ratio of y, the normal distance from the grid point to the wall, to r, the wall radius of curvature. By diminishing y, one lessens this ratio, and the wall looks flatter to the flow. The validity of other assumptions, for example assumption 3, will not be improved with increased resolution. To obtain more universal wall functions, it would be desirable to relax those assumptions whose validity does not depend on mesh resolution, and the analysis that follows can serve as a basis for future extensions of the theory.

With assumptions 1-8 above, the $k-\varepsilon$ equations near a wall become

$$\mu \frac{\partial u}{\partial y} = \tau_w = constant \ , \tag{B-1}$$

$$K \frac{\partial T}{\partial y} = J_w = constant \ , \tag{B-2}$$

$$\frac{\partial}{\partial y}\left(\frac{\mu}{Pr_k}\frac{\partial k}{\partial y}\right) + \mu\left(\frac{\partial u}{\partial y}\right)^2 - \rho\varepsilon = 0 \ , \tag{B-3}$$

and

$$\frac{\partial}{\partial y}\left(\frac{\mu}{Pr_\varepsilon}\frac{\partial \varepsilon}{\partial y}\right) + c_{\varepsilon_1}\frac{\varepsilon}{k}\mu\left(\frac{\partial u}{\partial y}\right)^2 - c_{\varepsilon_2}\rho\frac{\varepsilon^2}{k} = 0 \ , \tag{B-4}$$

where

$$\mu = c_\mu \rho \frac{k^2}{\varepsilon} \ , \tag{B-5}$$

$$K = \frac{\mu c_p}{Pr} \ , \tag{B-6}$$

$$p = \rho \frac{R}{\bar{W}} T = constant \ . \tag{B-7}$$

In these equations y is the normal coordinate to the wall and u is the velocity component tangent to the wall. In the absence of chemical reactions, the species mass fractions are constant, and hence the mean molecular weight $\bar{W}$ is constant. Although it is not necessary for the analysis that follows and does not alter the results in any fundamental way, for simplicity we also assume that c_p is constant.

We now nondimensionalize the equations. The dimensional quantities characterizing our problem are the wall shear stress τ_w, the wall heat loss J_w, the wall temperature T_w, the wall density ρ_w, and the specific heat c_p. From these a characteristic velocity, the shear speed, is defined by

$$u^* = \sqrt{\frac{\tau_w}{\rho_w}} \ . \tag{B-8}$$

The only quantity with dimensions of length is the distance y from the wall, and accordingly when a length scale is needed, we use y. We nondimensionalize u by u^*, T by T_w, k by $(u^*)^2$, and ε by $(u^*)^3/y$. Written in terms of dimensionless dependent variables (for which we use the same symbols as their dimensional counterparts), but retaining dimensional independent variable y, the equations become

$$c_\mu \rho \frac{k^2}{\varepsilon} y \frac{\partial u}{\partial y} = 1 \ , \tag{B-1$'$}$$

$$c_\mu \rho \frac{k^2}{\varepsilon} y \frac{\partial T}{\partial y} = \frac{J_w Pr\, u^*}{c_p \tau_w T_w} = \zeta \ , \tag{B-2$'$}$$

$$\frac{\partial}{\partial y}\left[\frac{c_\mu}{Pr_k} \rho \frac{k^2}{\varepsilon} y \frac{\partial k}{\partial y} \right] + c_\mu \rho \frac{k^2}{\varepsilon} y \left(\frac{\partial u}{\partial y} \right)^2 - \rho \frac{\varepsilon}{y} = 0 \ , \tag{B-3$'$}$$

$$\frac{\partial}{\partial y}\left[\frac{c_\mu}{Pr_\varepsilon} \rho \frac{k^2}{\varepsilon} y \frac{\partial}{\partial y}\left(\frac{\varepsilon}{y} \right) \right] + c_{\varepsilon_1} c_\mu \rho k \left(\frac{\partial u}{\partial y} \right)^2 - c_{\varepsilon_2} \rho \frac{\varepsilon^2}{y^2 k} = 0 \ , \tag{B-4$'$}$$

and

$$\rho T = 1 \; . \tag{B-7'}$$

We now assume the dimensionless wall heat loss ζ is small. Each of the dependent variables is expanded in a power series in ζ:

$$\begin{aligned}
u &= u_0 + u_1\zeta + \ldots \, , \\
T &= T_0 + T_1\zeta + \ldots \, , \\
k &= k_0 + k_1\zeta + \ldots \, , \\
\varepsilon &= \varepsilon_0 + \varepsilon_1\zeta + \ldots \, , \\
\rho &= \rho_0 + \rho_1\zeta + \ldots \, ,
\end{aligned} \tag{B-9}$$

where u_i, T_i, k_i, ε_i, and ρ_i are functions of y alone. Now note that dividing (B-2′) by (B-1′) yields

$$\frac{\partial T}{\partial y} / \frac{\partial u}{\partial y} = \zeta \, , \tag{B-10}$$

from which we immediately obtain

$$\frac{dT_0}{dy} = 0 \tag{B-11}$$

and

$$\frac{dT_{i+1}}{dy} = \frac{du_i}{dy} \quad \mathit{for} \quad i \geq 0 \; .$$

Consequently $T_0 = 1$ and

$$T_{i+1} = u_i + c_i \; , \tag{B-12}$$

where c_i is a constant. Our strategy will be to solve the equations for an isothermal boundary layer for u_0, k_0, and ε_0. These zero-order terms in ζ will be the wall functions for u, k, and ε. Then the temperature wall function will be obtained from the expansion to first order in ζ,

$$T = 1 + (u_0 + c_0)\zeta \,, \tag{B-13}$$

where we have used (B-9) and (B-12).

Solutions for the zero-order terms are

$$k_0 = c_\mu^{-\frac{1}{2}} \,,$$

$$\varepsilon_0 = [c_\mu^{\frac{1}{2}}(c_{\varepsilon_2} - c_{\varepsilon_1})Pr_\varepsilon]^{-\frac{1}{2}} = 1/\kappa \,,$$

and

$$u_0 = 1/\kappa \, \ell n \, y + const \,. \tag{B-14}$$

Thus a logarithmic velocity profile is obtained, and the analysis shows that the Karmann constant κ is related to the other $k-\varepsilon$ model constants and cannot be independently specified. For the standard values of the model constants given in Table II, we have $\kappa = 0.4327$, which differs slightly from the commonly accepted value of 0.40.[61] Determination of the constants in the velocity formula of (B-14) requires consideration of the laminar sublayer, wherein the laminar kinematic viscosity ν_ℓ becomes important. A dimensional argument gives

$$u_0 = \frac{1}{\kappa} \, \ell n \, \frac{yu^*}{\nu_\ell} + B \,, \tag{B-15}$$

where the constant B has the experimentally determined value of 5.5 for smooth walls.[61]

In dimensional terms, the wall functions for k, ε, and u become

$$k = c_\mu^{-\frac{1}{2}}(u^*)^2 \,, \tag{B-16}$$

$$\varepsilon = \frac{1}{\kappa} \, \frac{(u^*)^3}{y} = \frac{c_\mu^{3/4}}{\kappa} \, \frac{k^{3/2}}{y} \,, \tag{B-17}$$

and

$$\frac{u}{u^*} = \frac{1}{\kappa}\,\ell n\,\frac{yu^*}{\nu_\ell} + B\ . \tag{B-18}$$

From (B-16), we obtain

$$\frac{\partial k}{\partial y} = 0\ , \tag{B-19}$$

which is the boundary condition used for the k-equation in KIVA-II. Equation (B-17) is used directly to determine the value of ε at the centers of computational cells next to walls. Because of (B-17), whenever a length scale L is needed in the KIVA-II input or equations, it is related to k and ε by

$$L = \frac{c_\mu^{3/4}}{\kappa}\,\frac{k^{3/2}}{\varepsilon}\ .$$

We do not use (B-18) directly for the velocity wall function because this would require iterative solution for the unknown shear speed u^*. Instead we change the independent variable by replacing yu^*/ν_ℓ by its 1/7-power law value[63]

$$\frac{yu^*}{\nu_\ell} = c_{\ell w}\left(\frac{yu}{\nu_\ell}\right)^{7/8}\ , \tag{B-20}$$

where $c_{\ell w} = 0.15$. We obtain

$$\frac{u}{u^*} = \frac{1}{\kappa}\,\ell n\left[c_{\ell w}\left(\frac{yu}{\nu_\ell}\right)^{7/8}\right] + B\ , \tag{B-21}$$

which can be easily solved for u^*, once y and u are known.

Equation (B-21) is only valid in the logarithmic region, where $(yu^*)/\nu_\ell >> 1$. If $yu^*/\nu_\ell < 1$, then we are in the laminar sublayer and another formula must be used. Although the flow in the laminar sublayer is not truly laminar, we use the laminar formula

$$\frac{u}{u^*} = \left(\frac{yu}{\nu_\ell}\right)^{\frac{1}{2}}\ . \tag{B-22}$$

The transition between (B-21) and (B-22) is made at the point $R_c = yu/\nu_\ell$ where they predict the same u^*. Solving

$$R_c^{1/2} = \frac{1}{\kappa}\, \ell n\, [c_{\ell w} R_c^{7/8}] + B \qquad \text{(B-23)}$$

for R_c gives $R_c = 114$. Strictly speaking, we should also not apply (B-17) in the laminar sublayer, but we continue to use it for lack of a better alternative.

Although we obtain the wall shear stress from (B-21), note that if assumptions 1-8 are valid we could also obtain this stress from Eq. (B-16) and the computed value of k at a grid point in the logarithmic region. Some authors[26] use (B-16) to eliminate u^* in the argument of the logarithm in (B-18) and thereby obtain an equation that can easily be solved for u^* once u and k are known a distance y from the wall. To our knowledge, no one has tested the relative accuracies of these different equations for u^*.

We now turn our attention to the temperature wall function. From (B-13), in the logarithmic velocity region, the dimensional temperature equation is

$$\frac{T}{T_w} = 1 + \frac{J_w Pr\, u^*}{c_p \tau_w T_w} \left(\frac{u}{u^*} + c_0 \right), \qquad \text{(B-24)}$$

where c_0 is a constant whose value must be determined from experiment. Unfortunately, good experimental evidence for c_0 is lacking, and we determine c_0 by matching to a laminar temperature profile in the laminar sublayer region. More precisely we assume that

$$\frac{T}{T_w} = 1 + \frac{J_w}{K_\ell T_w} y = 1 + \frac{J_w Pr_\ell u^*}{c_p \tau_w T_w} \frac{u}{u^*} \qquad \text{(B-25)}$$

for $yu/\nu_\ell < R_c$, where K_ℓ and Pr_ℓ are the laminar heat conduction coefficient and Prandtl number. By equating (B-24) and (B-25) at $yu/\nu_\ell = R_c$, where $u/u^* = R_c^{\frac{1}{2}}$, we obtain

$$c_0 = R_c^{\frac{1}{2}} \left(\frac{Pr_\ell}{Pr} - 1 \right). \qquad \text{(B-26)}$$

This is only a provisional value for c_0 that must be tested in experimental comparisons.

II. NUMERICAL IMPLEMENTATION

We now describe the numerical implementation of the turbulent boundary layer equations, first for the momentum equation and then for the internal energy, k-, and ε-equations. Consider a typical cell adjacent to the wall, as shown in Fig. B-1. Vertices e, f, g, and h lie on the wall, and vertices a, b, c, and d are in the fluid. To evaluate the shear stress we need to know the velocity u tangent to the wall, evaluated a distance y from the wall, and the laminar kinematic viscosity ν_ℓ. The tangential velocity u is evaluated by

$$u = |\tfrac{1}{4}(\mathbf{u}_a + \mathbf{u}_b + \mathbf{u}_c + \mathbf{u}_d)| . \tag{B-27}$$

Equation (B-27) assumes the normal velocity at points a, b, c, and d is negligible, so that the tangential component may be replaced by the magnitude of the velocity. The distance y from the wall is calculated by

$$y = \left| \tfrac{1}{4}[(\mathbf{x}_a - \mathbf{x}_e) + (\mathbf{x}_b - \mathbf{x}_f) + (\mathbf{x}_c - \mathbf{x}_g) + (\mathbf{x}_d - \mathbf{x}_h)] \cdot \frac{\mathbf{A}_\alpha}{|\mathbf{A}_\alpha|} \right| , \tag{B-28}$$

where $\mathbf{A}_\alpha$ is the area vector of the face of the cell that lies on the wall. The kinematic viscosity is evaluated by

$$\nu_\ell = \frac{\mu_{air}(T)}{\rho} , \tag{B-29}$$

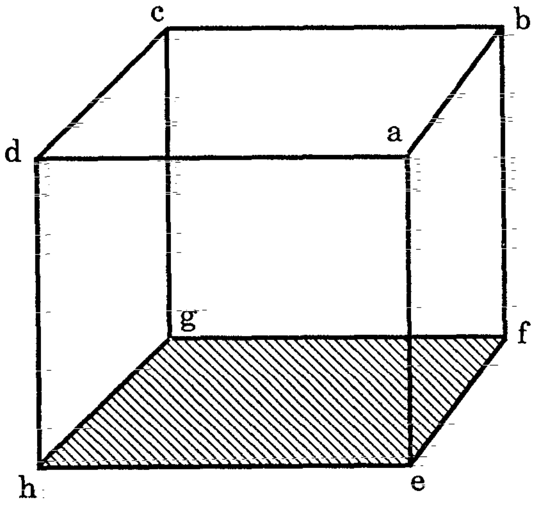


Fig. B-1. Typical cell adjacent to a wall.

where μ_{air} is given by Eq. (24) and ρ and T are the density and temperature of the cell. The shear stress τ_w is evaluated using Eq. (B-8) and either Eq. (B-21), if $yu/\nu_\ell \geq R_c$, or Eq. (B-22), if $yu/\nu_\ell \leq R_c$.

With τ_w thus determined, the product $\tau_w A \Delta t$ gives the total change in fluid momentum occurring on a timestep due to wall friction associated with the cell in question. One-fourth of this change is apportioned to each of the vertices e, f, g, and h. These changes are effected by taking the change in momentum of vertex i (i = e, f, g, h) to be parallel to the velocity at the vertex next to it and in the fluid. For example the vector momentum change for vertex e is then

$$M'_e \delta \mathbf{u}_e = -\tfrac{1}{4}\, \tau_w A \Delta t\, \mathbf{u}^n_a / |\mathbf{u}^n_a| \ , \qquad \text{(B-30)}$$

where M_i' are the vertex masses. These momentum changes are added in Phase A. It should be noted that the momentum changes at vertices i = e, f, g, and h are only those changes due to the particular wall cell in question. Similar changes to the momentum of each of these vertices will result from the other wall cells to which it is common.

The formulation given above assumes the wall is stationary. If the wall is in motion (e.g., a moving piston), it is necessary to transform the velocities into a coordinate frame moving with the wall before applying the equations. The new velocities must then be transformed back into the laboratory frame.

The wall heat flux J_w is computed from Eqs. (B-24) and (B-26), if $yu/\nu_\ell > R_c$, and from Eq. (B-25), if $yu/\nu_\ell \leq R_c$. The specific heat c_p is given by its value in the wall cell in question, and the temperature T is given by the average of the temperatures in the wall cell and the fluid cell above it. The product $J_w A \Delta t$ then gives the energy lost to the wall, which is therefore subtracted from the internal energy of the cell.

It is also necessary to allow for the kinetic energy dissipated by the wall friction. This frictional dissipation is approximated by $\tau_w u A \Delta t$, which is added to the internal energy of the wall cell in question.

Implementation of the boundary conditions for the turbulence equations is straightforward. The k-equation condition, Eq. (B-19), is enforced by allowing no diffusive flux of k through the face of the wall cell that lies on the wall. The value of ε in the wall cell is determined by Eq. (B-17) using the computed value of k in the wall cell and y equal to one-half the value of Eq. (B-28).

APPENDIX C

NUMERICAL SOLUTION OF THE EQUATIONS GOVERNING SPRAY DYNAMICS

In this appendix, we give the finite-difference approximations to the ordinary differential equations governing droplet trajectories and to the integrals that give the rates of mass, momentum, and energy exchange between the gas and spray. We also describe the solution procedure and the roles of the subroutines in which the spray calculations are performed.

The calculated spray source terms are added to the gas mass, momentum, and energy equations in subroutine PCOUPL. The source term $\dot{\rho}_{ijk}{}^s$, which gives the rate of mass addition to the gas per unit volume due to spray evaporation, is differenced as follows:

$$\dot{\rho}^s_{ijk} = -\frac{1}{V^n_{ijk}\Delta t}\sum_{p\epsilon(i,j,k)} N'_p \rho_d \frac{4}{3}\pi[(r^A_p)^3 - (r'_p)^3] . \qquad \text{(C-1)}$$

The summation is over all particles located in computational cell (i,j,k). The quantities N'_p and r'_p are provisional values of the number and radius of droplets in particle p. These may differ from $N_p{}^n$ and $r_p{}^n$ because of the droplet collision and breakup calculations. The liquid density ρ_d is assumed to be constant and equal for all droplets. The finite-difference approximation for $r_p{}^A$ is given below.

The source term $\dot{Q}_{ijk}{}^s$ in the internal energy equation is differenced as follows:

$$\dot{Q}^s_{ijk} = -\frac{1}{V^n_{ijk}\Delta t}\sum_{p\epsilon(i,j,k)} N'_p \rho_d \frac{4}{3}\pi\{(r^A_p)^3 I_\ell(T^A_{d_p}) - (r'_p)^3 I_\ell(T'_{d_p})$$

$$+ (r^A_p)^3(\mathbf{v}^t_p - \mathbf{v}'_p)\cdot(\mathbf{v}^t_p - \mathbf{u}^n_{\ell mn} - \mathbf{u}'_p) + \tfrac{1}{2}[(r^A_p)^3 - (r'_p)^3](\mathbf{v}^t_p - \mathbf{u}^n_{\ell mn})^2\} . \qquad \text{(C-2)}$$

In Eq. (C-2) r'_p, T'_{d_p}, and $\mathbf{v}'_p$ are droplet radii, temperatures, and velocities that have been partially updated due to droplet collisions and breakups. The velocity $\mathbf{v}'_p$ also contains the gravitational acceleration update. The calculation of the Phase A droplet temperature $T_{d_p}{}^A$ is described below, and $\mathbf{u}'_p$ is calculated as in Sec. III.C. The subscript (ℓ,m,n) denotes the indices of the momentum cell in which particle p is located. The velocity $\mathbf{v}_p{}^t$ is a partially updated particle velocity that is obtained by solving the following finite-difference approximation to the particle acceleration equation:

$$\frac{\mathbf{v}_p^t - \mathbf{v}_p'}{\Delta t} = D_p(\mathbf{u}_{\ell mn}^n + \mathbf{u}_p' - \mathbf{v}_p^t) . \tag{C-3}$$

In this equation D_p is the particle drag function, which is defined below.

The source term $\dot{W}_{ijk}{}^s$ to the turbulent kinetic energy equation is differenced in the following manner:

$$\dot{W}_{ijk}^s = - \frac{1}{V_{ijk}^n \Delta t} \sum_{p\epsilon(ijk)} N_p' \frac{4}{3} \pi\rho_d (r_p^A)^3 (\mathbf{v}_p^t - \mathbf{v}_p') \cdot \mathbf{u}_p' , \tag{C-4}$$

where $\mathbf{v}_p{}^t$ is obtained from Eq. (C-3).

The gas and droplet momentum equations are differenced in an implicit fashion that circumvents timestep limitations due to the strong coupling of gas and droplet velocities. The finite-difference approximation to the gas momentum equation can be written

$$(M')_{ijk}^B \mathbf{u}_{ijk}^B - (M')_{ijk}^n \mathbf{u}_{ijk}^n = \mathbf{E}_{ijk} - \sum_{p\epsilon(ijk)} N_p' \frac{4}{3} \pi\rho_d [(r_p^B)^3 \mathbf{v}_p^B - (r_p')^3 \mathbf{v}_p'] , \tag{C-5}$$

where $\mathbf{E}_{ijk}$ represents all contributions to the Lagrangian phase momentum change of vertex (i,j,k) except those due to spray momentum exchange. The finite-difference approximation to the droplet acceleration equation is implicit in the gas velocity and linearly implicit in the droplet velocity:

$$\frac{\mathbf{v}_p^B - \mathbf{v}_p'}{\Delta t} = D_p(\mathbf{u}_{ijk}^B + \mathbf{u}_p' - \mathbf{v}_p^B) , \tag{C-6}$$

where the drag function D_p is given by

$$D_p = \frac{3}{8} \frac{\rho_{ijk}^n}{\rho_d} \frac{|\mathbf{u}_{ijk}^n + \mathbf{u}_p' - \mathbf{v}_p'|}{r_p^A} C_D(Re_p) . \tag{C-7}$$

The particle velocity $\mathbf{v}'_p$ has already been updated due to gravitational acceleration. The drag coefficient C_D and droplet Reynolds number Re_p, which are defined in Sec.II.B, are evaluated using time level n values of the fluid variables.

When Eq. (C-6) is solved for $\mathbf{v}_p{}^B$ and the result is substituted into Eq. (C-5), one obtains, after some manipulation,

$$[(M')^B_{ijk} + S'_{ijk}]\mathbf{u}^B_{ijk} - (M')^n_{ijk}\mathbf{u}^n_{ijk} = \mathbf{E}_{ijk} - \mathbf{R}'_{ijk} \, , \qquad \text{(C-8)}$$

where

$$S'_{ijk} = \sum_{p\epsilon(ijk)} N'_p \frac{4}{3} \pi \rho_d (r^B_p)^3 \frac{\Delta t D_p}{1 + \Delta t D_p} \qquad \text{(C-9)}$$

and

$$\mathbf{R}'_{ijk} = \sum_{p\epsilon(ijk)} N'_p \frac{4}{3} \pi \rho_d \left[(r^B_p)^3 \frac{\mathbf{v}'_p + \Delta t D_p \mathbf{u}'_p}{1 + \Delta t D_p} - (r'_p)^3 \mathbf{v}'_p \right] \qquad \text{(C-10)}$$

depend on explicitly known values of the gas and droplet variables. These arrays are calculated in subroutine PMOM. When the Phase B gas velocity is known, the Phase B droplet velocities are then computed from Eq. (C-6) in subroutine PACCEL.

The particle radius and temperature changes are obtained by evaporating the particles sequentially at constant pressure. By coupling the particle evaporation rates more closely, the sequential evaporation calculation allows the use of larger timesteps than a simultaneous evaporation calculation. The difference approximations of the evaporation calculation are implicit in the droplet temperature but explicit in the gas temperature and vapor mass fraction. Even though the particles are evaporated sequentially, this explicitness can produce unphysical changes in the computed gas temperature and vapor mass fraction when heat and mass transfer rates to a single particle are large. To prevent this, for each particle we compute an evaporation timestep δt_{ev} based on heat and mass transfer rates, and subcycle the evaporation calculation a number of times equal to $\Delta t / \delta t_{ev} = N_{ev}$.

The evaporation calculation is performed in subroutine EVAP. We first solve implicitly for the updated droplet temperature. The equation we approximate is obtained from Eq. (41) by eliminating R using Eq. (38) and eliminating Q_d using Eq. (42):

$$\rho_d \tfrac{2}{3} r^2 c_\ell \dot{T}_d = K_{air}(T - T_d) Nu_d - L(T_d)(\rho D)_{air} B_d(T_d) Sh_d \, , \qquad \text{(C-11)}$$

where

$$B_d(T_d) = \frac{Y^*_1(T_d) - Y_1}{1 - Y^*_1(T_d)} \qquad \text{(C-12)}$$

and $Y^*{}_1(T_d)$ is given by Eq. (40). The finite-difference approximation to Eq. (C-11) is

$$\rho_d \tfrac{2}{3}(r_p^v)^2 c_\ell(T_{d_p}^v) \frac{T_{d_p}^{v+1} - T_{d_p}^v}{\delta t_{ev}} = K_{air}^v(\hat{T}_{ijk} - T_{d_p}^{v+1}) V_{Nu}^v \frac{\ell n[1 + B_d(T_{d_p}^{v+1})]}{B_d(T_{d_p}^{v+1})}$$

$$- L(T_{d_p}^v)(\rho D)_{air}^v V_{sh}^v \ell n[1 + B_d(T_{d_p}^{v+1})] , \qquad \text{(C-13)}$$

where superscript v denotes the value of a quantity after v evaporation subcycles. We initialize $T_{d_p}{}^o = T'{}_{d_p}$ and $r_p{}^o = r'{}_p$. The quantity B_d is calculated from the formula

$$B_d(T_{d_p}^{v+1}) = \frac{Y_1^*(T_{d_p}^{v+1}) - (\hat{Y}_1)_{ijk}}{1 - Y_1^*(T_{d_p}^{v+1})} . \qquad \text{(C-14)}$$

Cell (i,j,k) is the cell in which particle p is located, and $(\hat{Y}_1)_{ijk}$ in (C-14) and $\hat{T}_{ijk}$ in (C-13) are intermediate values of the vapor mass fraction and gas temperature that have been modified due to evaporation of particles with subscripts less than p and evaporation of the current particle p on subcycles less than v. Formulas for these quantities are given below. The heat conductivity $K_{air}{}^v$ and mass diffusivity $(\rho D)_{air}{}^v$ are calculated using $\hat{T}_{ijk}$ and $T_{d_p}{}^v$. The quantities $V_{sh}{}^v$ and $V_{Nu}{}^v$ are given by

$$V_{sh}^v = 2.0 + 0.6\, Re_d^{\frac{1}{2}} Sc_d^{\frac{1}{3}}$$

and

$$V_{Nu}^v = 2.0 + 0.6\, Re_d^{\frac{1}{2}} Pr_d^{\frac{1}{3}} , \qquad \text{(C-15)}$$

where the drop Reynolds number Re_d, Schmidt number Sc_d, and Prandtl number Pr_d are calculated using $r_p{}^v$, $T_{d_p}{}^v$ and the intermediate gas temperature $\hat{T}_{ijk}$.

Following implicit solution of (C-13) for $T_{d_p}{}^{v+1}$, the drop radius $r_p{}^{v+1}$ is obtained from

$$(r_p^{v+1})^2 = max \left\{ 0.0 \, , (r_p^v)^2 - \delta t_{ev} \frac{(\rho D)_{air}^v}{\rho_d} V_{sh}^v \frac{\ell n[1 + B_d(T_{d_p}^v)] + \ell n[1 + B_d(T_{d_p}^{v+1})]}{2} \right\} \qquad \text{(C-16)}$$

which approximates Eq. (38) for the rate of drop radius change.

The intermediate temperatures and species densities are obtained as follows. Before the particle evaporation calculation we initialize

$$M_{ijk} = \rho^n_{ijk} V^n_{ijk} ,$$

$$(M_1)_{ijk} = (\rho^n_1)_{ijk} V^n_{ijk} ,$$

$$(\overline{MH})_{ijk} = (\rho^n_{ijk} I^n_{ijk} + p^n_{ijk}) V^n_{ijk} ,$$

and

$$(H_0)_{ijk} = I^n_{ijk} + \frac{p^n_{ijk}}{\rho^n_{ijk}} . \tag{C-17}$$

Then as each particle is evaporated we modify the above arrays by

$$\hat{M}_{ijk} \leftarrow \hat{M}_{ijk} - \frac{4}{3} \pi \rho_d [(r_p^{v+1})^3 - (r_p^v)^3] N'_p ,$$

$$(\hat{M}_1)_{ijk} \leftarrow (\hat{M}_1)_{ijk} - \frac{4}{3} \pi \rho_d [(r_p^{v+1})^3 - (r_p^v)^3] N'_p ,$$

and

$$(\overline{MH})_{ijk} \leftarrow (\overline{MH})_{ijk} - \frac{4}{3} \pi \rho_d [(r_p^{v+1})^3 I_\ell (T^{v+1}_{d_p}) - (r_p^v)^3 I_\ell (T^v_{d_p})] N'_p , \tag{C-18}$$

if particle p lies in cell (i, j, k). Finally, we calculate

$$(\hat{Y}_1)_{ijk} = (\hat{M}_1)_{ijk} / \hat{M}_{ijk} \tag{C-19}$$

and

$$\hat{T}_{ijk} = T^n_{ijk} + \frac{(\overline{MH})_{ijk} / \hat{M}_{ijk} - (H_0)_{ijk}}{(c_p)^n_{ijk}} . \tag{C-20}$$

These new intermediate temperatures and species densities are then used when calculating the radius and temperature changes for particle p on the next subcycle or for particle $p+1$ if we have completed the evaporation calculation for particle p.

The choice for δt_{ev} is based on the idea that the heat transfer to a computational particle in one timestep should not exceed some fraction of the energy available for transfer. The heat transfer rate to a computational particle on the first evaporation subcycle is approximately

$$Q_p = Nu_p K_{air} (\widehat{T}_{ijk} - T'_{d_p}) 2\pi r'_p N'_p \ . \tag{C-21}$$

The gas energy available for transfer is approximately

$$E_g = (c_p)^n_{ijk} (\widehat{T}_{ijk} - T'_{d_p}) \rho^n_{ijk} V^n_{ijk} \ . \tag{C-22}$$

The criterion for δt_{ev} is thus

$$Q_p \delta t_{ev} \leq f E_g \ , \tag{C-23}$$

where $f < 1.0$. Using $f = \frac{1}{2}$ and substituting from (C-21) and (C-22) gives

$$\delta t_{ev} \leq \frac{\rho^n_{ijk} V^n_{ijk} (c_p)^n_{ijk}}{Nu_p K_{air} 4\pi r'_p N'_p} \ . \tag{C-24}$$

A similar criterion based on mass transfer considerations gives

$$\delta t_{ev} \leq \frac{\rho^n_{ijk} V^n_{ijk}}{Sh_p (\rho D)_{air} 4\pi r'_p N'_p} \ . \tag{C-25}$$

Since $Sh_p \approx Nu_p < V_{sh}$, where V_{sh} is given by (C-15), and $(\rho D)_{air} \approx K_{air}/(c_p)_{ijk}{}^n \approx \mu_{air}$, where μ_{air} is the viscosity of air, we replace (C-24) and (C-25) with the single criterion

$$\delta t_{ev} \leq \frac{\rho^n_{ijk} V^n_{ijk}}{V_{sh} \mu_{air} 4\pi r'_p N'_p} \ . \tag{C-26}$$

More precisely, we set $\delta t_{ev} = \Delta t/N_{ev}$, where the number of subcycles N_{ev} is the smallest positive integer such that δt_{ev} satisfies (C-26).

APPENDIX D

PARTICLE RADIUS SELECTION AT INJECTION

Through input switch INJDIST, the user specifies one of two size distributions associated with droplets injected into the computational mesh. If INJDIST = 0, a monodisperse distribution is used [$\delta(r - r_o)$], with the injected drop size r_o given by input parameter SMR. This option can be used in conjunction with the breakup model (Appendix F) to calculate atomization according to the method of Reitz.[4] One injects droplets with r_o equal to the nozzle radius for hole nozzles or equal to half the nozzle opening size for nozzles with pintles. The "atomization" of these large injected drops into smaller droplets is then calculated by the breakup model.

If INJDIST = 1, a χ-squared distribution is used for the sizes of injected droplets:

$$f(r) = \frac{1}{\bar{r}} e^{-r/\bar{r}} , \tag{D-1}$$

where $\bar{r}$ is the number-averaged drop radius, which for the distribution of (D-1) is related to the input Sauter mean radius r_{32} by

$$\bar{r} = \tfrac{1}{3} r_{32} . \tag{D-2}$$

The quantity r_{32} is given by input parameter SMR when INJDIST = 1. There are many ways to obtain a specified size distribution when injecting particles because one has the freedom to choose the number of drops per particle. The method we use, which is also used in the droplet breakup calculation, samples most frequently those portions of the size distribution where the most mass occurs. These drops will usually exchange the most mass, momentum, and energy with the gas. In the remainder of this appendix, we detail how the radii of injected particles are chosen to obtain the specified drop size distribution (D-1).

In addition to the drop size distribution (D-1), we can define another distribution $g(r)$ in such a way that $g(r)dr$ is the probability that a particle has drops with radii in the range $(r, r + dr)$. The number of drops per particle is then proportional to the ratio $f(r)/g(r)$. Best resolution of the drop-size distribution is obtained where the values of $g(r)$ are largest, and it follows that to obtain the best resolution of the size distribution where the most drop mass is located, $g(r)$ should be proportional to the mass distribution $r^3 f(r)$ and the number of drops per particle should be proportional to $1/r^3$. From this it follows that the total droplet mass associated with each particle should be constant. This constant is determined by dividing the input total spray mass to be injected TSPMAS by the input total number of parcels to be injected TNPARC.

The distribution $g(r)$ is normalized to unit total weight by taking

$$g(r) = \frac{r^3}{6\,\bar{r}^4}\, e^{-r/\bar{r}} \; . \tag{D-3}$$

According to our procedure for selecting radius values randomly with the distribution $g(r)$, we must first find the cumulative distribution $h(r)$ associated with $g(r)$, and then apply the inverse of $h(r)$ to random numbers uniformly distributed in the interval (0,1). The distribution $h(r)$ can be seen to be

$$h(r) = 1 - e^{-r/\bar{r}} \left[1 + r/\bar{r} + \frac{1}{2}\,(r/\bar{r})^2 + \frac{1}{6}\,(r/\bar{r})^3 \right] \; . \tag{D-4}$$

The inversion of $h(r)$ is performed numerically. We store values of $h(r)$ in increments of $0.12\,\bar{r}$ between $r = 0$ and $r = 12\,\bar{r} = 4\,r_{32}$. The value of $h(12\,\bar{r})$ is taken to be unity. (This involves only a slight inaccuracy since $h(12\,\bar{r})$ is in reality greater than 0.997.) If XX is a random number in the interval (0,1), we find that value of n for which

$$h[0.12\,\bar{r}\,(n-1)] \le XX < h[0.12\,\bar{r}\,n] \; . \tag{D-5}$$

Then the corresponding drop radius is

$$r = 0.12\,\bar{r}\,n = 0.04\,r_{32} n \; . \tag{D-6}$$

APPENDIX E

THE DROPLET COLLISION CALCULATION

Consistent with the viewpoint of the stochastic particle method, drop collisions are calculated by a sampling procedure. The alternative is to try to represent the complete distribution of drop properties that arise due to drop collisions. For example, having calculated the collision frequency between a drop associated with particle A and all drops associated with another particle, we could proceed in two ways. In the first way, we could use the collision frequency to calculate the probable number of drops in particle A that undergo collisions with drops in the other particle. To represent the distribution of collision behavior, this number of drops would be subtracted from particle A, and one or more new particles would be created having the properties of the drops resulting from the collisions. We tried such a procedure with the result that we quickly had more particles than could be accommodated by computer storage. In the second way, which is the way we use, the collision frequency is used to calculate the probability P that a drop in particle A will undergo a collision with a drop in the other particle. Then all the drops in particle A behave in the same manner; they either do or do not collide, and the probability of the former event is P. Since all the drops in particle A behave in the same way, no new particles have to be created. Then the probability distribution of outcomes is recovered by ensemble averaging over many computations or, in a steady-state calculation, by time averaging over a long time. The above brief description gives the basic idea of our collision calculation, which we will now describe in more detail.

For each pair of particles, the collision calculation proceeds as follows. The collision calculation is performed for the pair of particles if, and only if, they are in the same computational cell. To facilitate the description of the collision calculation, we will call the drops of larger radius "collectors" and those of smaller radius "droplets." For purposes of the collision calculation, the drops associated with each particle are considered to be uniformly distributed throughout the computational cell in which they are located. Thus, we calculate the collision frequency ν of a collector drop with all droplets according to

$$\nu = \frac{N_2^n}{V_{ijk}^n} \pi (r_1^n + r_2^n)^2 |\mathrm{v}_1 - \mathrm{v}_2| . \qquad \text{(E-1)}$$

The subscripts 1 and 2 refer to the properties of the collectors and droplets, $N_2{}^n$ is the number of droplets in particle 2, and $V_{ijk}{}^n$ is the volume of the cell in which both particles are located. The probability P_n that a collector undergoes n collisions with droplets follows a Poisson distribution,

$$P_n = e^{-\bar{n}} \frac{\bar{n}^n}{n!} \ , \tag{E-2}$$

with mean value $\bar{n} = \nu \Delta t$ where Δt is the computational timestep. Thus, the probability of no collisions is $P_0 = e^{-\bar{n}}$. A random number XX is chosen in the interval (0,1). If $XX < P_0$, then no collisions are calculated between the drops in particles 1 and 2.

If $XX \geq P_0$, we chose a second random number YY, $0 < YY < 1$, that determines the outcome of the collision. $\sqrt{YY}\,(r_1 + r_2)$ is the collision impact parameter b. If $b < b_{cr}$, where b_{cr} is the critical impact parameter below which coalescence occurs, then the result of every collision is coalescence. If $b \geq b_{cr}$, then each collision is a grazing collision. The value of b_{cr} depends on the drop radii, the relative velocity between the drops, and the liquid surface tension coefficient. The expression we use for b_{cr} can be found in Sec. II.B.

Suppose the outcome of the collision is coalescence. Then the number of coalescences n for each collector is determined by finding the value for n for which

$$\sum_{k=0}^{n-1} P_k \leq XX < \sum_{k=0}^{n} P_k \ . \tag{E-3}$$

For each collector drop, n droplets are subtracted from their associated parcel, and the size, velocity, and temperatures of the collector drops are appropriately modified. If there is an insufficient number of droplets to have n coalesce with each collector, then n is recomputed so that all $N_2{}^n$ droplets coalesce, and the particle associated with the droplets is removed from the calculation.

There is a timestep limitation associated with the above calculation of drop coalescence, and this is that the computational timestep Δt be small compared to the collision time Δt_d for the droplets. The latter is given by

$$\frac{1}{\Delta t_d} = \nu_d = \frac{N_1^n}{V_{ijk}^n} \pi (r_1^n + r_2^n)^2 |\mathrm{v}_1 - \mathrm{v}_2| \ , \tag{E-4}$$

where $N_1{}^n$ is the number of collector drops. When $\Delta t << \Delta t_d$, the probable number of droplets that coalesce in a timestep, which is $\nu_d \, \Delta t \, N_2{}^n$, will be less than $N_2{}^n$. Hence, most of the time the number of droplets in particle 2 will not be depleted in one timestep due to collisions.

This timestep limitation is much less severe than that we would have required if we had allowed only one coalescence per timestep. With one coalescence per timestep, we would have had to compute $\nu\delta t << 1$ where ν is given by Eq. (C-1). The equations for ν and ν_d differ in that the number of droplets $N_2{}^n$ is used in computing ν. In Appendices D and F it is shown that we usually have $N_2{}^n >> N_1{}^n$ and hence $\nu >> \nu_d$.

Suppose now that the outcome of each collision is a grazing collision. In this case, only one collision is calculated for each drop. This introduces an additional timestep constraint that Δt be small compared to the collision times between drops of nearly equal size, since grazing collisions usually occur between drops of nearly equal size. Grazing collisions are calculated between N pairs of drops, where N is the minimum of $N_1{}^n$ and $N_2{}^n$. The N collectors and droplets are then returned to their particles in such a way that mass, momentum, and energy are conserved.

APPENDIX F

THE DROPLET OSCILLATION AND BREAKUP CALCULATION

In this appendix we describe the numerical solution procedure for the equations governing spray droplet oscillation and breakup. A detailed description of the breakup model can be found in Ref. 35. To calculate droplet oscillation and breakup, we require two additional particle arrays y_p and $\dot{y}_p$. The quantity y_p is proportional to the displacement of the droplet's surface from its equilibrium position, divided by the droplet radius. Droplet breakup occurs if and only if y_p exceeds unity. The time rate of change of y_p is $\dot{y}_p$, and the time rate of change of $\dot{y}_p$ is given by Eq. (45).

To update the values of y_p and $\dot{y}_p$ each computational cycle, we make use of the exact solution of Eq. (45) assuming constant coefficients:

$$y(t) = \frac{We}{12} + exp\,(-t/t_d)\left[\left(y(0) - \frac{We}{12}\right) cos\,wt + \frac{1}{\omega}\left(\dot{y}(0) + \frac{y_0 - \dfrac{We}{12}}{t_d}\right) sin\,\omega t\right] , \qquad \text{(F-1)}$$

where

$$We = \frac{\rho\, u^2 r}{\alpha} \qquad \text{(F-2)}$$

is the Weber number, u is the relative velocity between the gas and droplet, α is the surface tension coefficient,

$$t_d = \frac{2}{5}\,\frac{\rho_d r^2}{\mu_\ell} \tag{F-3}$$

is the viscous damping time, μ_ℓ is the liquid viscosity, and

$$\omega^2 = 8\,\frac{\alpha}{\rho_d r^3} - \frac{1}{t_d^2} \tag{F-4}$$

is the square of the oscillation frequency. For each particle we first calculate We, t_d, and $\omega 2$. A value of $\omega 2 \leq 0$ occurs only for very small drops for which distortions and oscillations are negligible. Thus if $\omega 2 \leq 0$, we set $y_p{}^{n+1} = \dot{y}_p{}^{n+1} = 0$.

If $\omega 2 > 0$, we next calculate the amplitude A of the undamped oscillation:

$$A^2 = \left(y_p^n - \frac{We}{12}\right)^2 + \left(\frac{\dot{y}_p^n}{\omega}\right)^2 . \tag{F-5}$$

If $We/12 + A \leq 1.0$, then according to Eq. (F-1), the value of y will never exceed unity and breakup will not occur. Most particles will pass the test $We/12 + A \leq 1.0$, and for these we simply update y_p and $\dot{y}_p$ using Eq. (F-1):

$$y_p^{n+1} = \frac{We}{12} + exp\,(-\Delta t/t_d)\left[\left(y_p^n - \frac{We}{12}\right)\cos\omega\Delta t + \frac{1}{\omega}\left(\dot{y}_p^n + \frac{y_p^n - \frac{We}{12}}{t_d}\right)\sin\omega\Delta t\right] \tag{F-6}$$

and

$$\dot{y}_p^{n+1} = \left(\frac{We}{12} - y_p^{n+1}\right)/t_d + exp\,(-\Delta t/t_d)\left[\left(\dot{y}_p^n + \frac{y_p^n - \frac{We}{12}}{t_d}\right)\cos\omega\Delta t - \omega\left(y_p^n - \frac{We}{12}\right)\sin\omega\Delta t\right] . \tag{F-7}$$

If $We/12 + A > 1.0$, then breakup is possible on the current timestep. We calculate the breakup time t_{bu} assuming that the drop oscillation is undamped for its first period. This will be true for all except very small drops. The breakup time t_{bu} is the smallest root greater than t^n of the equation

$$\frac{We}{12} + A\,cos[\omega(t - t^n) + \phi] = 1 \ , \tag{F-8}$$

where

$$cos\,\phi = \frac{y_p^n - \frac{We}{12}}{A}$$

and

$$sin\,\phi = -\frac{\dot{y}_p^n}{A\omega} \ .$$

If time t^{n+1} is less than t_{bu}, then no breakup occurs on the current timestep, and we use Eqs. (F-6) and (F-7) to update y_p and $\dot{y}_p$.

Breakup is calculated only if $t_{bu} \le t^{n+1}$. When this is true, the Sauter mean radius r_{32} of the product drops is calculated from Eq. (31), and Eq. (32) is used to calculate the velocity w of the product drops normal to the relative velocity between the parent drop and gas. When evaluating r_{32} and w, $\dot{y}_p$ is evaluated at t_{bu} using Eq. (F-1). The radius r_{bu} of the product drops is then chosen randomly from a χ-square distribution with Sauter mean radius r_{32}. In choosing r_{bu} we sample most frequently from those portions of the χ-squared distribution where the most mass resides, as is described in Appendix D. To conserve mass, the number of drops N associated with the computational particle is adjusted according to

$$N_p^{n+1} = N_p^n \left(\frac{r_p^n}{r_{bu}} \right)^3 \ . \tag{F-9}$$

We also add to the particle velocity a component with magnitude w and direction randomly chosen in a plane normal to the relative velocity vector between the parent drop and gas. This procedure does not conserve momentum in detail, but it does so on the average. Following breakup, we assume the product drops are not distorted or oscillating, and accordingly we set $y_p{}^{n+1} = \dot{y}_p{}^{n+1} = 0$.

APPENDIX G

CALCULATION OF DROPLET TURBULENT DISPERSION

Turbulence effects on the spray particles are modeled by adding to the gas velocity $\mathbf{u}$ a fluctuating velocity $\mathbf{u}'_p$, where each component of $\mathbf{u}'_p$ is randomly chosen from a Gaussian distribution with standard deviation $\sqrt{2/3\,k}$ and k is the specific turbulent kinetic energy of the gas in the computational cell in which particle p is located. The fluctuating velocity $\mathbf{u}'_p$ is a piecewise constant function of time, changing discontinuously after passage of turbulence correlation time t_{turb}, which is determined by Eq. (37). The sum $\mathbf{u} + \mathbf{u}'_p$ is then the gas velocity that the particle "sees" when calculating its drag, heat and mass exchange with the gas and its oscillation and breakup. We also subtract from the turbulent kinetic energy the work done by the fluctuating components in dispersing the spray droplets.

For each computational particle, we chose one of two numerical procedures for solving the equations of this model. The choice depends on the relative magnitudes of Δt and t_{turb}. When $\Delta t < t_{turb}$, which is most often the case, it is possible to solve for particle positions $\mathbf{x}_p{}^B$ and velocities $\mathrm{v}_p{}^B$ by straightforward difference approximations:

$$\frac{\mathbf{x}_p^B - \mathbf{x}_p^n}{\Delta t} = \mathbf{v}_p^n \tag{G-1}$$

and

$$\frac{\mathbf{v}_p^B - \mathbf{v}_p^n}{\Delta t} = D_p(\mathbf{u}_{ijk}^B + \mathbf{u}_p' - \mathbf{v}_p^B) + \mathbf{g}\ , \tag{G-2}$$

where D_p is the particle drag function (see Appendix C) and the particle is located in momentum cell (i,j,k). The gas turbulent velocity $\mathbf{u}'_p$ is held fixed for a number of computational cycles k such that

$$t^{n+k-1} - t^n < t_{turb} \leq t^{n+k} - t^n\ , \tag{G-3}$$

where $\mathbf{u}'_p$ was last chosen on cycle n and t_{turb} is the turbulence time evaluated at the position of the particle on cycle $n+k$. Section III.C gives the method used for randomly sampling for $\mathbf{u}'_p$. The velocity $\mathbf{u}'_p$ also enters in a straightforward fashion in the difference approximations to the equations for mass and energy exchange given in Appendix C and the difference equations for droplet oscillation and breakup of Appendix F.

When $\Delta t > t_{turb}$ it is no longer possible to use these difference approximations because the particle "sees" more than one turbulent velocity $\mathbf{u}'_p$ on the current cycle. Possible approaches to this problem are to restrict Δt to be some fraction of t_{turb} or to subcycle the droplet equations using subcycle timestep δt such that $\delta t < t_{turb}$. These methods are computationally inefficient, however, when t_{turb} is smaller than Δt, and are unusable when t_{turb} becomes orders of magnitude smaller than Δt.

Our approach to this problem forsakes some accuracy to obtain computational efficiency. When $\Delta t > t_{turb}$ we choose random velocity and position changes from probability distributions that we have derived[33] for the droplet turbulent velocity and position changes. Thus, independent of how large Δt is relative to t_{turb}, our method requires the choice of only two random numbers each timestep, one to determine its turbulent velocity change and one to determine its turbulent position change.

This approach is inaccurate in several respects. First, when $\Delta t > t_{turb}$, we ignore the effects of the fluctuating velocity $\mathbf{u}'_p$ on heat and mass exchange and on droplet breakup. Second, in deriving the probability distributions for turbulent velocity and position change, we have assumed that the drag function D_p, k, and t_{turb} are constant for a given particle on the current timestep. In particular, the effects of a nonlinear drag law are not included in the probability distributions. There appears to be no alternative to the assumption of a linear drag law because use of a nonlinear law renders intractable the problem of deriving probability distributions for turbulent velocity and position changes.

The derivation of the distributions is given in Ref. 33, and here we only summarize the results. The assumption of a linear drag law allows us to treat each component of the velocity and position changes independently. For each component the distributions are Gaussian; the velocity distribution has variance

$$\sigma^2_{u'} = \frac{1 - exp\,(-D_p t_{turb})}{1 + exp\,(-D_p t_{turb})}\,[1 - exp\,(-2D_p \Delta t)]\,\sigma^2 \,, \tag{G-4}$$

and the position distribution has variance

$$\sigma^2_{x'} = \left\{ t_{turb}\Delta t - \frac{2t_{turb}}{D_p}\,[1 - exp\,(-D_p \Delta t)] + \frac{\sigma^2_{u'}}{D_p^2 \sigma^2} \right\} \sigma^2 \,, \tag{G-5}$$

where $\sigma^2 = \frac{2}{3}k$. An additional quantity, a turbulence persistence time t_{per}, is used in choosing the turbulent position change and enters because the distribution of velocity and position changes are not independent. This quantity is given by

$$t_{per} = \frac{t_{turb}[1 - exp\,(-D_p \Delta t)] - \dfrac{\sigma_{u'}^2}{D_p \sigma^2}}{\sigma_{u'}^2/\sigma^2} \; . \tag{G-6}$$

When $\Delta t > t_{turb}$ the particle velocity and position are updated using

$$\frac{\mathbf{x}_p^B - \mathbf{x}_p^n}{\Delta t} = \mathbf{v}_p^n + \frac{\delta \mathbf{x}'}{\Delta t} \tag{G-7}$$

and

$$\frac{\mathbf{v}_p^B - \mathbf{v}_p^n}{\Delta t} = D_p(\mathbf{u}_{ijk}^B - \mathbf{v}_p^B) + \mathbf{g} + \frac{\delta \mathbf{u}'}{\Delta t} \; , \tag{G-8}$$

where $\delta\mathbf{x}'$ and $\delta\mathbf{u}'$ are the turbulent position and velocity changes. First, each component of $\delta\mathbf{u}'$ is chosen from a Gaussian distribution with variance $\sigma_{u'}{}^2$. Then $\delta\mathbf{x}'$ is calculated from

$$\delta\mathbf{x}' = t_{per}\,\delta\mathbf{u}' + \delta\mathbf{x}_b' \; , \tag{G-9}$$

where each component of $\delta\mathbf{x}'_b$ is chosen from a Gaussian distribution with variance $\sigma_{x'}{}^2 - t_{per}{}^2\sigma_{u'}{}^2$.

APPENDIX H

THE VARIABLE IMPLICITNESS PARAMETERS

Variable implicitness is used in KIVA-II when differencing the diffusion terms and terms associated with pressure wave propagation. The amount of implicitness is chosen, in part, to ensure numerical stability of the difference approximations to the individual terms in question. If stability were our only concern, fully-implicit schemes would be used. Computational efficiency can be gained, however, by minimizing the amount of implicitness. When the timesteps are small enough, KIVA-II will automatically use stable explicit schemes for which no costly iterative solution is required. When implicitness is required, KIVA-II uses a partially implicit difference scheme in which there is some weighting of both the old- and new-time values of the solution variable. It has been our experience that most iterative procedures for implicit equations, including the conjugate residual method used in KIVA-II, converge more slowly for fully implicit than for partially implicit schemes. A fully implicit scheme is only used by KIVA-II in the limit of an infinitely large timestep. In this section we give the analyses upon which the forms of the implicitness parameters are based. In addition to motivating the forms we use for the implicitness parameters, the analyses are interesting in that they reveal the nature of some of the numerical errors inherent in KIVA-II solutions.

We first give the analysis for determining the implicitness parameter ϕ_p for the pressure gradient terms. The form of ϕ_p is obtained by considering the KIVA-II finite difference equations applied to the problem of one-dimensional inviscid acoustic wave motion in a gas with nearly uniform density ρ_o and pressure p_o. We use a computational mesh with uniform cell size Δx and cross-sectional area A. In one dimension the approximations to the vertex momentum equation (78) and cell face velocity equation (86) both reduce to the same form:

$$\rho_o \frac{u_{j+\frac{1}{2}}^{n+1} - u_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{1}{\alpha^2}\left[\phi_p \frac{p_{j+1}^{n+1} - p_j^{n+1}}{\Delta x} + (1-\phi_p)\frac{p_{j+1}^{n} - p_j^{n}}{\Delta x}\right] = 0 \, . \qquad \text{(H-1)}$$

In (H-1) $u_{j+\frac{1}{2}}$ is the velocity at a vertex or a cell face between cells j and $j+1$. All Phase B quantities equal their advanced-time values (e.g., $p_j^B = p_j^{n+1}$) since convection terms are negligible for acoustic waves. An equation for the pressure is obtained by combining Eq. (102) for the cell volume change and the linearized form (112) of the equation of state:

$$V_j^B = V_o + \Delta t A\,[u_{j+\frac{1}{2}}^{n+1} - u_{j-\frac{1}{2}}^{n+1}] \qquad \text{(H-2a)}$$

and

$$p_j^{n+1} - p_j^n = -\frac{\gamma p_o}{V_o}(V_j^B - V_o) \, . \qquad \text{(H-2b)}$$

In (H-2) V_j^B is the Phase B cell volume and V_o is the initial uniform cell volume, $V_o = A\Delta x$. Combining (H-2a) and (H-2b) gives

$$\frac{p_j^{n+1} - p_j^n}{\Delta t} + \gamma p_o \frac{u_{j+\frac{1}{2}}^{n+1} - u_{j-\frac{1}{2}}^{n+1}}{\Delta x} = 0 \, . \qquad \text{(H-3)}$$

By using (H-1) to eliminate the velocities in (H-3), we could derive and analyze the difference approximation to the pressure equation. Our approach is to keep (H-1) and (H-3) in their present forms and solve for both $u_{j+\frac{1}{2}}$ and p_j.

Numerical solutions of (H-1) and (H-3) can be found in terms of the Fourier components of p_j and $u_{j+\frac{1}{2}}$. Substituting the values

$$p_j^n = A^n \gamma p_o \, exp\,(ikj\,\Delta x)$$

and

$$u_{j+\frac{1}{2}}^n = B^n c_o \, exp\,[ik\,(j+\tfrac{1}{2})\,\Delta x] \tag{H-4}$$

into (H-1) and (H-3) and solving for A^{n+1} and B^{n+1} yields

$$\begin{pmatrix} A^{n+1} \\ \\ B^{n+1} \end{pmatrix} = \begin{pmatrix} \dfrac{1 + a^2(1-\phi_p)}{1 - a^2\phi_p} & \dfrac{a}{1 - a^2\phi_p} \\ \dfrac{a}{1 - a^2\phi_p} & \dfrac{1}{1 - a^2\phi_p} \end{pmatrix} \begin{pmatrix} A^n \\ \\ B^n \end{pmatrix} , \tag{H-5}$$

where

$$a = -2\,i\,C_s \sin \psi/2 \ ,$$

$$C_s = \frac{c_o \Delta t}{\Delta x} \ ,$$

$$c_o = \frac{1}{\alpha}\sqrt{\frac{\gamma p_o}{\rho_o}} \ ,$$

and

$$\psi = k\,\Delta x \ .$$

Numerical solutions will be stable if and only if both eigenvalues of the above matrix are less than or equal unity.

The eigenvalues λ of the matrix in (H-5) can be seen to be

$$\lambda = \frac{1 - 2(1-\phi_p)\,C_s^2 \sin^2 \psi/2 \pm 2C_s \sin \psi/2 \sqrt{(1-\phi_p)^2 C_s^2 \sin^2 \psi/2 - 1}}{1 + 4\,\phi_p C_s^2 \sin^2 \psi/2} \ . \tag{H-6}$$

If $(1 - \phi_p)\, C_s \leq 1$, the two λ are complex conjugates and have common magnitude:

$$|\lambda| = [1 + 4\phi_p C_s^2 \sin^2 \psi/2]^{-\frac{1}{2}} . \tag{H-7}$$

Thus the scheme is stable if $(1 - \phi_p)C_s \leq 1$ or, equivalently, if $\phi_p \geq 1 - 1/C_s$.

Motivated by this result, we calculate the value of ϕ_p based on the local Courant number C_s:

$$(\phi_p)^n_{ijk} = \begin{cases} 0 & if \quad (C_s)^n_{ijk} \leq 1/f \\ 1 - 1/[f(C_s)^n_{ijk}] & if \quad (C_s)^n_{ijk} > 1/f \end{cases} , \tag{H-8}$$

where

$$(C_s)^n_{ijk} = \sqrt{\frac{\gamma^n_{ijk} p^n_{ijk}}{(a^n)^2 \rho^n_{ijk}}}\ \frac{\Delta t}{\Delta x} , \tag{H-9}$$

$$\Delta x = min\,(\Delta x_i \ , \ \Delta x_j \ , \ \Delta x_k) \ ,$$

$$\Delta x_i^2 = \tfrac{1}{2}[|\mathbf{x}_1^n - \mathbf{x}_4^n|^2 + |\mathbf{x}_2^n - \mathbf{x}_3^n|^2] ,$$

$$\Delta x_j^2 = \tfrac{1}{2}[|\mathbf{x}_3^n - \mathbf{x}_4^n|^2 + |\mathbf{x}_2^n - \mathbf{x}_1^n|^2] ,$$

$$\Delta x_k^2 = \tfrac{1}{2}[|\mathbf{x}_8^n - \mathbf{x}_4^n|^2 + |\mathbf{x}_5^n - \mathbf{x}_1^n|^2] , \tag{H-10}$$

and the $\mathbf{x}_l^n$'s refer to the vertex locations of cell (i,j,k) as numbered in Fig. 2. The quantity f is an empirically determined safety factor. The above analysis gives $f \geq 1.0$ as a sufficient condition for stability in one-dimensional problems with uniform cell sizes and material properties. Using $f = 2.5$ has been found to give stable results in all our test calculations.

When $C_s < 0.4$, ϕ_p will be zero and an explicit difference approximation will be used by KIVA-II. The cell face velocities are first found from Eq. (86), and then the Phase B cell volumes and Phase B pressures are found from Eqs. (102) and (112). Since the magnitudes of the eigenvalues give the amount of numerical damping, Eq. (H-7) shows acoustic waves are not numerically damped by this explicit scheme, although there is some numerical dispersion of acoustic waves. This is the difference scheme used in the original KIVA code[1] to calculate pressure wave propagation.

We now motivate the choice for ϕ_D, the variable implicitness parameter for the diffusion equations. The form of ϕ_D is indicated by an analysis of the KIVA-II difference approximation to the one-dimensional diffusion equation with constant diffusion coefficient ν:

$$\frac{Y_j^{n+1} - Y_j^n}{\Delta t} = \nu\left[(1-\phi_D)\frac{Y_{j+1}^n - 2Y_j^n + Y_{j-1}^n}{\Delta x^2} + \phi_D\frac{Y_{j+1}^{n+1} - 2\,Y_j^{n+1} + Y_{j-1}^{n+1}}{\Delta x^2}\right] . \tag{H-11}$$

Substituting

$$Y_j^n = A^n\, exp\,(ikj\,\Delta x)$$

into (H-11) gives the following equation for A^{n+1}:

$$A^{n+1}/A^n = \frac{1 + 2(1-\phi_D)\,C_d\,(cos\,\psi - 1)}{1 - 2\phi_D\,C_d\,(cos\,\psi - 1)} , \tag{H-12}$$

where

$$C_d = \frac{\nu\Delta t}{\Delta x^2}$$

and

$$\psi = k\Delta x \ .$$

Now we require that the approximation not only be stable, but that the amplitudes not change sign:

$$0 \le \frac{A^{n+1}}{A^n} \le 1 \ . \tag{H-13}$$

It is easily seen that the right-hand inequality in (H-3) is always satisfied. To satisfy the left-hand inequality, since the denominator of (H-12) is always positive, we need the numerator to be positive. The numerator is minimized for $\psi = \pi$ and has the value $1 - 4(1 - \phi_D)C_d$. This leads to the criterion

$$(1-\phi_D)\,C_d \le \tfrac{1}{4}$$

or

$$\phi_D \geq 1 - \frac{1}{4C_d} \; . \tag{H-14}$$

Motivated by this result, we compute ϕ_D from the equation

$$\phi_D = \begin{cases} 0 & if \quad (C_d)^n_{ijk} \leq 1/4 \\ 1 - 1/[f(C_d)^n_{ijk}] & if \quad (C_d)^n_{ijk} > 1/4 \end{cases} \; , \tag{H-15}$$

where

$$(C_d)^n_{ijk} = \frac{\mu^n_{ijk}}{\rho^n_{ijk}} M \Delta t \frac{\Delta x_i^2 \Delta x_j^2 + \Delta x_i^2 \Delta x_k^2 + \Delta x_j^2 \Delta x_k^2}{\Delta x_i^2 \Delta x_j^2 \Delta x_k^2} \; , \tag{H-16}$$

$$M = max\left(2 + A_3 , \frac{1}{Pr} , \frac{1}{Sc} , \frac{1}{Pr_k} , \frac{1}{Pr_\varepsilon}\right) , \tag{H-17}$$

and Δx_i, Δx_j, and Δx_k are defined in (H-10). In (H-17), A_3 is the ratio of the second to first coefficients of viscosity; Sc is the Schmidt number; and Pr, Pr_k, and Pr_ε are Prandtl numbers for diffusion of heat, turbulence kinetic energy, and turbulence dissipation rate. For real gases, the arguments in (H-17) are normally near unity, and one value of ϕ_D is used to calculate diffusion of all quantities. If one has an application for which there is a large variation in the values of these arguments, then it is advisable to use separate values of ϕ_D for diffusion of each quantity.

Note that if we had required just stability of the difference approximation, that is $A^{n+1}/A^n \geq -1$, from (H-12) it would have been sufficient to take

$$\phi_D \geq \tfrac{1}{2}(1 - 1/(2C_d)) \; , \tag{H-18}$$

or even $\phi_D = \frac{1}{2}$. This latter scheme is the Crank-Nicolson method[21] and is second-order accurate in time. Although more accurate in time, the Crank-Nicolson method and other methods that violate (H-14) have "overshoots" and are inaccurate for large values of C_d.

APPENDIX I

KINETIC CHEMICAL REACTIONS

Here we specify the procedure for evaluating the progress rates $\dot{\omega}_r{}^A$ for the kinetic chemical reactions. Since there is no direct chemical coupling between different cells, we may focus attention on a representative cell (i,j,k) and suppress the subscript ijk.

The progress rates $\dot{\omega}_r$ for the kinetic reactions are computed under the assumption that, for each reaction r, every participating species is either inert $(a_{mr} = b_{mr})$ or appears on only one side of the reaction $(a_{mr}b_{mr} = 0)$. We first calculate the quantities

$$\Omega_{fr} = k^n_{fr} \prod_m (\tilde{\rho}_m / W_m)^{a'_{mr}}$$

and

$$\Omega_{br} = k^n_{br} \prod_m (\tilde{\rho}_m / W_m)^{b'_{mr}} , \qquad \text{(I-1)}$$

where $k_{fr}{}^n$ and $k_{br}{}^n$ are evaluated from Eq. (17) with T replaced by T^n and $\tilde{\rho}_m$ denotes an intermediate value of ρ_m that has been updated due to kinetic reactions $< r$ but not reactions $\geq r$. We next identify the species, call it species K, for which

$$W_m (b_{mr} - a_{mr})(\Omega_{fr} - \Omega_{br})/\rho_m$$

is a minimum. This species is called the **reference species** for reaction r; it is the species whose density is in greatest danger of being driven negative. Once K is identified, we define

$$\Lambda_T = \tilde{\rho}_K + \Delta t W_K [b_{Kr}\Omega_{fr} + a_{Kr}\Omega_{br}] ,$$

$$\Lambda_B = \tilde{\rho}_K + \Delta t W_K [a_{Kr}\Omega_{fr} + b_{Kr}\Omega_{br}] . \qquad \text{(I-2)}$$

Then $\dot{\omega}_r{}^A$ is given by

$$\dot{\omega}^A_r = \frac{\tilde{\rho}_K(\Lambda_T/\Lambda_B - 1)}{\Delta t W_K (b_{Kr} - a_{Kr})} .$$

This prescription makes the part of $\partial\rho_k/\partial t$ that is due to reaction r linearly implicit in ρ_K, which prevents ρ_K from being driven negative no matter how large Δt is.

APPENDIX J

EQUILIBRIUM CHEMICAL REACTIONS

In this appendix we describe the two procedures for evaluating the progress rates $\dot{\omega}_r{}^A$ for the equilibrium reactions. For hydrocarbon combustion, a fast, algebraic solver is provided in subroutine CHMQGM; for more general circumstances, an iterative solver is provided in subroutine CHEMEQ. The choice of subroutines is determined by input flag KWIKEQ. If KWIKEQ = 0, CHEMEQ is used; if KWIKEQ = 1, CHMQGM is used. In the latter case one does not need to input the stoichiometric coefficients and constants used to compute equilibrium constants in Eq. (19); these are stored in data statements for the specific set of reactions used in CHMQGM. We first describe the more general, iterative solver.

I. SUBROUTINE CHEMEQ

Here we describe the procedure in subroutine CHEMEQ for evaluating the progress rates $\dot{\omega}_r{}^A$ for the equilibrium reactions during Phase A. The $\dot{\omega}_r{}^A$ are implicitly determined by the requirement that the Phase A species densities $\rho_m{}^A$ must satisfy an approximation to the equilibrium constraint conditions of Eq. (18):

$$\prod_m (\rho_m^A/W_m)^{b_{mr}-a_{mr}} = K_c^r(\tilde{T})\, exp\, \{D_r(T^A - \tilde{T})\}\ , \tag{J-1}$$

where

$$D_r = \left.\frac{\partial\, \ell n\, K_c^r}{\partial T}\right|_{T=\tilde{T}}$$

and $\tilde{T}$ and T^A are partially updated temperatures that will be defined shortly. In (J-1) we have linearized $\ell n\, K_c{}^r$ about its starting value $\tilde{T}$. (The spatial indice *ijk* will be suppressed throughout this appendix, as there is no direct chemical coupling between different cells.) These constraint conditions constitute a coupled nonlinear equation system which is solved by an iterative procedure.[25] The iteration scheme used is an improvement over the earlier scheme used in CONCHAS-SPRAY[16] and includes the effects on the equilibrium constraints $K_c{}^r$ of heat release from the equilibrium reactions.[62] For simplicity, we assume that the fuel species (species 1), of which the spray particles are composed, does not participate in any of the equilibrium reactions.

The iteration scheme consists, in essence, of the following ingredients: (a) preconditioning of the equilibrium constraint conditions to make them more nearly linear in the progress variables, (b) application of a one-step SOR-Newton iteration[63] to the preconditioned system, followed by (c) switching to a full Newton-Raphson iteration if the simpler SOR-Newton iteration fails to converge in a specified number of steps. If the equilibrium reactions are weakly coupled, convergence usually occurs before the full Newton-Raphson iteration is called into play. However, if the reactions are not weakly coupled, the interaction between them is properly accounted for by the matrix inversion in the Newton-Raphson procedure.

Within this appendix, we denote by $\tilde{\rho}_m$ and $\tilde{T}$ partially updated species densities and fluid temperatures that contain the contributions due to kinetic chemical reactions on the current timestep. $\tilde{T}$ is given by

$$\tilde{T} = T^n + \left[\sum_r \omega_r^A Q_r \right] / (\rho^n c_p^n) \ ,$$

where the sum is over kinetic reactions.. These values serve as initial values for the iteration procedure. The final converged values of the species densities are the $\rho_m{}^A$. The species densities are related to their initial values by

$$\rho_m = \tilde{\rho}_m + W_m \sum_s (b_{ms} - a_{ms}) \omega_s \ , \qquad \text{(J-2)}$$

and the partially updated temperatures are related to there initial values by

$$T = \tilde{T} + \left[\sum_s Q_s \omega_s \right] / (\rho^n c_p^n) \ , \qquad \text{(J-3)}$$

where $\omega_s = \Delta t \, \dot{\omega}_s$ and the summations are over all equilibrium reactions. We denote by $\omega_s{}^A$ the values of ω_s for which $\rho_m = \rho_m{}^A$ and $T = T^A$. The values of $\dot{\omega}_s{}^A$ are then simply $\dot{\omega}_s{}^A = \omega_s{}^A / \Delta t$. Because of Eqs. (J-2) and (J-3), we may regard Eqs. (J-1) as a coupled nonlinear system of equations for the unknown quantities $\omega_s{}^A$. Since these equations will be solved iteratively, we introduce an iteration index ν, which will be displayed as a superscript. Thus the approximation to $\omega_s{}^A$ after iteration ν is denoted by $\omega_s{}^\nu$, and the corresponding approximations to $\rho_m{}^A$ and T^A are

$$\rho_m^\nu = \tilde{\rho}_m + W_m \sum_s (b_{ms} - a_{ms}) \omega_s^\nu \qquad \text{(J-4)}$$

and

$$T^{\nu} = \tilde{T} + \left[\sum_{s} Q_s \omega_s^{\nu} \right] / (\rho^n c_p^n) \; . \tag{J-5}$$

It is understood that $\omega_s{}^0 = 0$, so that $\rho_k{}^0 = \tilde{\rho}_k$ and $T^0 = \tilde{T}$. It will also be necessary to refer to intermediate species densities defined by

$$\rho_m(\nu,s) = \tilde{\rho}_m + W_m \sum_{z=1}^{s-1} (b_{mz} - a_{mz})\,\omega_z^{\nu} + W_m \sum_{z=s}^{N} (b_{mz} - a_{mz})\,\omega_z^{\nu-1} \tag{J-6}$$

and intermediate temperatures $T(\nu,s)$ defined by

$$T(\nu,s) = T + \left[\sum_{z=1}^{s-1} Q_z \omega_z^{\nu} + \sum_{z=s}^{N} Q_z \omega_z^{\nu-1} \right] / (\rho^n c_p^n) \; , \tag{J-7}$$

where N is the number of equilibrium reactions.

One further notational convention will prove useful. We introduce a vector $\rho = (\rho_1, \rho_2, ..., \rho_{NS}, T)$ whose components are the species densities ρ_m and temperature T. Functions of the ρ_m and T can then be compactly written simply as functions of the vector variable ρ. The notation ρ^{ν} refers, of course, to the vector whose components are the $\rho_m{}^{\nu}$ and T^{ν}, and $\rho(\nu,s)$ refers to the vector with components $\rho_m(\nu,s)$ and $T(\nu,s)$.

We now proceed to consider how the equilibrium constraints of Eq. (J-1) might be preconditioned to make them more nearly linear in the ω_s. The first step is to identify the principal or dominant dependence of the left member of Eq. (J-1) upon the $\omega_s{}^A$. The form of this quantity suggests that we determine, for each reaction s, the species m for which the factor $(\rho_m/W_m)^{b_{ms}-a_{ms}}$ depends most sensitively, in some appropriate sense, on the ω_s. Let this be the species with index $m = \mu(s)$ and denote $b_{\mu(s),s} - a_{\mu(s),s}$ by q_s. The species $m = \mu(s)$ will be referred to as the **reference species** for reaction s. The dominant dependence of the left member of Eq. (J-1) on the $\omega_s{}^A$ is now regarded as being contained in the factor $(\rho^A_{\mu(s)}/W_{\mu(s)})^{q_s}$. Since $\rho_{\mu(s)}$ itself is linear in the ω_s, this dominant dependence can be made to manifest itself linearly by raising both sides of Eq. (J-1) to the power $p_s \equiv 1/q_s$. We therefore replace the constraint conditions of Eq. (J-1) by the preconditioned constraint conditions

$$\prod_{m} (\rho_m^A/W_m)^{(b_{ms} - a_{ms})\,p_s} = [K_c^s(\tilde{T})]^{p_s} \, exp\,[\,p_s D_s (T^A - \tilde{T})] \; . \tag{J-8}$$

It is convenient to introduce quantities $F_s(\rho)$ and $G_s(\rho)$ defined by

$$G_s = K_c^s(\tilde{T}) \prod_m (\rho_m/W_m)^{a_{ms} - b_{ms}} \tag{J-9}$$

and

$$F_s = G_s^{-p_s} - exp\,[p_s D_s(T^A - \tilde{T})] \,, \tag{J-10}$$

in terms of which Eq. (J-8) becomes simply

$$F_s(\rho^A) = 0 \,. \tag{J-11}$$

We have yet to specify how the reference species are to be determined. For simplicity, we define the reference species for reaction s as the species for which the factor $(\rho_m/W_m)^{b_{ms}-a_{ms}}$ depends most sensitively on ω_s alone, without regard for the other progress variables. That is, $\mu(s)$ is the value of m for which the quantity

$$R_{ms} \equiv \left(\frac{\rho_m}{W_m}\right)^{a_{ms} - b_{ms}} \frac{\partial}{\partial\omega_s} \left(\frac{\rho_m}{W_m}\right)^{b_{ms} - a_{ms}} \tag{J-12}$$

is largest in magnitude. This quantity is easily evaluated from Eq. (J-2), with the result

$$R_{ms} = (W_m/\rho_m)\,(b_{ms} - a_{ms})^2 \,. \tag{J-13}$$

This depends on ρ_m, so it is necessary to specify which species densities are to be used in the evaluation of the reference species. This will be done below.

In the subsequent development, we shall require the partial derivatives $\partial F_s/\partial\omega_t$, which are also easily evaluated from Eqs. (J-2) and (J-3). The result is

$$\frac{\partial F_s}{\partial\omega_t} = p_s G_s^{-p_s} A_{st} - (p_s D_s Q_t/\rho c_p)\, exp\,\{p_s D_s(T^k - \tilde{T})\} \,, \tag{J-14}$$

where the matrix $A_{st}(\rho)$ is defined by

$$A_{st} = \sum_m \frac{W_m}{\rho_m}\,(b_{ms} - a_{ms})\,(b_{mt} - a_{mt}) \,. \tag{J-15}$$

By virtue of Eqs. (J-2) and (J-3), the quantities $F_s(\rho)$, $G_s(\rho)$, and $A_{st}(\rho)$ may alternatively be regarded as functions of the progress variables ω_s, and this will be done without special comment when it is appropriate or convenient to do so.

The iteration scheme as a whole is structured as follows. The first N_0 iterations are performed with a one-step SOR-Newton algorithm. If convergence has not already occurred, all subsequent iterations are performed with a Newton-Raphson algorithm (except as noted in the description below). We currently take $N_0 = 7$. The iteration scheme is considered to have converged when $|G_s^{p_s} F_s| < \varepsilon$ for all s. Currently ε is taken to be 0.02. We now proceed to a detailed description of the SOR-Newton and Newton-Raphson algorithms that are used.

A one-step SOR-Newton iteration procedure, applied to the system of Eq. (J-11), takes the form[60]

$$\omega_s^{\nu+1} = \omega_s^{\nu} - \frac{\Omega\, F_s(\omega_1^{\nu+1}, \ldots, \omega_{s-1}^{\nu+1}, \omega_s^{\nu}, \ldots, \omega_N^{\nu})}{\partial F_s(\omega_1^{\nu+1}, \ldots, \omega_{s-1}^{\nu+1}, \omega_s^{\nu}, \ldots, \omega_N^{\nu})/\partial\omega_s^{\nu}} , \qquad \text{(J-16)}$$

where Ω is the overrelaxation parameter. Equations (J-14), (J-10), (J-6), and (J-7) allow us to rewrite Eq. (J-16) in the more useful form

$$\omega_s^{\nu+1} = \omega_s^{\nu} + \frac{\Omega\, q_s[G_s^{p_s}\, exp\,\{p_s D_s (T - \tilde{T})\} - 1]}{A_{ss} - (D_s Q_s/\rho c_p)\, G_s^{p_s}\, exp\,\{p_s D_s (T - \tilde{T})\}} , \qquad \text{(J-17)}$$

where G_s and A_{ss} are evaluated at $\rho(\nu+1,s)$ and T is evaluated at $T(\nu+1,s)$. It is not necessary to actually evaluate $\rho(\nu+1, s)$ by means of Eq. (J-6), because if the ρ_m are continually updated as running sums then $\rho(\nu+1,s)$ is simply the "current" value of ρ just prior to the evaluation of $\omega_s^{\nu+1}$.

Strictly speaking, since p_s is considered constant in evaluating $\partial F_s/\partial\omega_t$, p_s (and therefore q_s) should be held constant and not allowed to vary with ν. This might seem all the more advisable in view of the fact that p_s varies discontinuously with the ρ_m or ω_s. In practice, however, we have found that convergence is slightly accelerated if p_s and q_s in Eq. (J-17) are allowed to vary by reevaluating $\mu(s)$ in terms of the $\rho_m(\nu+1,s)$ on every iteration. No problems have yet been experienced in doing so, but if such problems were to occur it would merely be necessary to hold p_s and q_s fixed with the values determined by the initial species densities $\tilde{\rho}_m$.

The value of $\omega_s{}^{\nu+1} - \omega_s{}^{\nu}$ given by Eq. (J-17) is subjected to the restriction

$$0.9\,\delta\omega_s^{min} \le \omega_s^{\nu+1} - \omega_s^{\nu} \le 0.9\,\partial\omega_s^{max} , \qquad \text{(J-18)}$$

where

$$\delta\omega_s^{min} = \left[\begin{matrix} min \\ m \end{matrix} \quad \frac{W_m(a_{ms} - b_{ms})}{\rho_m(\nu+1,s)} \right]^{-1} , \tag{J-19}$$

$$\delta\omega_s^{max} = \left[\begin{matrix} max \\ m \end{matrix} \quad \frac{W_m(a_{ms} - b_{ms})}{\rho_m(\nu+1,s)} \right]^{-1} \tag{J-20}$$

are the minimum and maximum values of $\omega_s^{\nu+1} - \omega_s^\nu$ that preserve the nonnegativity of the ρ_m.

A standard Newton-Raphson iteration applied to the system of Eq. (J-11) yields

$$\sum_t [\partial F_s(\omega_1^\nu, \ldots, \omega_N^\nu)/\partial\omega_t^\nu](\omega_t^{\nu+1} - \omega_t^\nu) = -F_s(\omega_1^\nu, \ldots, \omega_N^\nu) . \tag{J-21}$$

Using Eqs. (J-14), (J-10), (J-4), and (J-5), we may rewrite this as

$$\sum_t [A_{st} - (D_s Q_t/\rho c_p) G_s^{p_s} exp\{p_s D_s (T^\nu - \tilde{T})\}](\omega_t^{\nu+1} - \omega_t^\nu) = q_s [G_s^{p_s} exp\{p_s D_s (T^\nu - \tilde{T})\} - 1] , \tag{J-22}$$

where G_s and A_{st} are evaluated using ρ^ν. To obtain the $\omega_s^{\nu+1}$, it is necessary to invert an $N \times N$ matrix. This may be done using any of the standard methods, one or more of which are usually available as modular library subroutines in large computer centers.

Again, p_s and q_s should strictly be held constant in Eq. (J-22). In practice, however, we find it slightly advantageous to allow them to vary with ν by reevaluating the reference species in terms of the ρ_m^ν on every iteration.

In spite of the preconditioning, Eq. (J-22) occasionally yields values of $\omega_s^{\nu+1}$ that drive one or more of the $\rho_m^{\nu+1}$ negative. (In particular, this may happen when a single trace species of very small concentration is involved in two or more reactions, as the matrix then becomes ill-conditioned.) When this occurs, the values of $\omega_s^{\nu+1} - \omega_s^\nu$ given by Eq. (J-22) are all reduced in magnitude by a factor $\alpha (0 < \alpha < 1)$, and the $\rho_m^{\nu+1}$ are recomputed accordingly. If any of them are still negative, the reduced values of $\omega_s^{\nu+1} - \omega_s^\nu$ are further reduced by another factor of α, and the $\rho_m^{\nu+1}$ are recomputed again. If necessary, this procedure is repeated up to N_α times, whereupon if negative values of the $\rho_m^{\nu+1}$ still persist, the $\omega_s^{\nu+1}$ given by Eq. (J-22) are simply discarded. The iteration is then repeated as an SOR-Newton iteration, with the $\omega_s^{\nu+1}$ obtained from Eq. (J-17). We currently use the values $\alpha = 0.3$ and $N_\alpha = 6$.

II. SUBROUTINE CHMQGM

Subroutine CHMQGM utilizes an algorithm devised by Meintjes and Morgan[24] for the solution for the simultaneous equilibria of six reactions important in hydrocarbon oxidation:

$$\begin{aligned}
&N_2 \rightleftharpoons 2N \\
&O_2 \rightleftharpoons 2O \\
&2CO_2 \rightleftharpoons 2CO + O_2 \\
&H_2 \rightleftharpoons 2H \\
&2OH \rightleftharpoons O_2 + H_2 \\
&2H_2O \rightleftharpoons 2H_2 + O_2 \, .
\end{aligned} \qquad \text{(J-23)}$$

In contrast to CHEMEQ, which solves for reaction progress increments $\omega_r{}^A$, CHMQGM solves for the equilibrium molar concentrations of the chemical species. Four additional equations are needed to solve for the ten concentrations, and these are just the element-conservation relations for carbon, hydrogen, oxygen, and nitrogen. The temperature $\tilde{T}$ used to evaluate the equilibrium constants $K_c{}^r$ for reactions (J-23) is given by

$$\tilde{T} = T^n + \left[\sum_r Q_r \, \omega_r \right] / (\rho^n c_p^n) \, , \qquad \text{(J-24)}$$

where the sum is over all kinetic chemical reactions. Because the formulation does not include the effects on the equilibrium constants of heat release from the equilibrium reactions, small cycle to cycle oscillations in temperature and species concentrations can occur in some applications.[62] These oscillations are small in most hydrocarbon combustion problems, however, and it is better to use subroutine CHMQGM instead of CHEMEQ because it is much faster.

The increase in speed of subroutine CHMQGM is achieved because the ten equations for the concentrations are algebraically reduced to a much simpler system, which can be quickly solved. The details of this reduction can be found in Ref. 24, and here we only summarize the simple system that is solved. First the equilibrium and element-conservation equations involving nitrogen, which are uncoupled from the remainder of the system, are easily solved for the concentrations of molecular and atomic nitrogen. The remaining eight equations are then combined algebraically to obtain two cubic equations in two unknowns. These are scaled concentrations of atomic hydrogen and carbon monoxide. The simultaneous cubic equations are solved by Newton-Raphson iteration, using the scaled concentrations from the previous cycle as a first guess. Because of the manner in which the equations are scaled, it is always necessary to have at least trace amounts of carbon present.

The values of reaction rates $\dot{\omega}_r{}^A$ are not needed when subroutine CHMQGM is used: The mass and energy source terms are differenced by

$$\dot{\rho}^c_m = (\rho^A_m - \tilde{\rho}_m)/\Delta t \tag{J-25}$$

and

$$\dot{Q}^c = -\sum_m \dot{\rho}^c_m (\Delta h^o_f)_m \ , \tag{J-26}$$

where $\tilde{\rho}_m$ is the initial density of species m before equilibrium reactions and Eq. (J-26) follows from combining Eqs. (20), (21), and (22).

APPENDIX K

CALCULATION OF VELOCITY GRADIENTS AND VISCOUS STRESSES

According to Eq. (4), evaluation of the viscous stress tensor requires evaluation of the velocity gradients. These gradients and stresses are taken to be cell-centered quantities and are considered uniform within regular cells.

Velocity gradients are obtained in the following manner. We begin with the identity

$$\frac{\partial u_\ell}{\partial x_m} = \nabla \cdot (u_\ell \mathbf{e}_m) \ , \tag{K-1}$$

where u_ℓ is the velocity component in the ℓth Cartesian coordinate direction and $\mathbf{e}_m$ is the unit vector in the mth Cartesian coordinate direction. After integrating Eq. (K-1) over computational cell (i,j,k) and using Gauss' Divergence Theorem, one obtains

$$\int_{V_{ijk}} \frac{\partial u_\ell}{\partial x_m}\, dV = \int_{S_{ijk}} u_\ell \mathbf{e}_m \cdot d\mathbf{A} \; , \tag{K-2}$$

where S_{ijk} is the surface of cell (i,j,k). The left-hand side of this equation is differenced as

$$\left(\frac{\partial u_\ell}{\partial x_m}\right)_{ijk} V_{ijk} \; , \tag{K-3}$$

and the right-hand side is approximated by

$$\sum_\alpha (u_\ell)_\alpha \mathbf{e}_m \cdot \mathbf{A}_\alpha \; , \tag{K-4}$$

where $(u_\ell)_\alpha$ is the equal-weighted average of the four values of u_ℓ associated with the four vertices bordering face α, and A_α is the outward area projection vector of face α.

After calculating these velocity gradients, the stress components in cell (i,j,k) are obtained from

$$[\sigma_{\ell m}(\mathbf{u}^\nu)]_{ijk} = \mu^n_{ijk}\left[\left(\frac{\partial u_\ell}{\partial x_m}\right)^\nu_{ijk} + \left(\frac{\partial u_m}{\partial x_\ell}\right)^\nu_{ijk}\right] + \lambda^n_{ijk}\left(\frac{\partial u_n}{\partial x_n}\right)^\nu_{ijk} \delta_{\ell m} \; , \tag{K-5}$$

where the superscript ν denotes to time level of the velocity field.

APPENDIX L

THE ALTERNATE NODE COUPLER

In this appendix we describe the alternate node coupler used in KIVA. This is a procedure for damping the hourglass velocity modes that occur in numerical fluid dynamics calculations that have velocities located at computational cell vertices. The basic idea of our procedure is to detect and subtract, in each computational cell, velocity modes for which the finite-difference approximations to the mean velocity and velocity gradients are zero. This is done in such a way that linear momentum is conserved. Since these modes have no calculated mean velocity or velocity gradients, no physical forces are introduced by the numerical procedure. The procedure does introduce a small numerical damping whose nature is discussed in Ref. 16.

In typical hourglass velocity modes, the average velocity and velocity gradients at each cell center are zero when these are calculated from centered differences involving just the vertex velocities of the cell. Thus in the absence of an alternate node coupler, no force will be calculated to damp the hourglass mode.

In order to detect the hourglass modes in KIVA, for each computational cell, we construct a set of vertex velocities with approximately the same computed mean velocity and velocity gradients as the velocity field in the KIVA calculation, but which does not have the hourglass modes. This velocity for vertex ℓ of computational cell (i,j,k) is given by

$$\mathbf{u}_\ell^g = \frac{\sum_m M'_m \mathbf{u}_m^n}{\sum_m M'_m} + \left[\frac{\partial \mathbf{u}^n}{\partial x}\right]_{ijk} (x_\ell - \bar{x}_{ijk}) + \left[\frac{\partial \mathbf{u}^n}{\partial y}\right]_{ijk} (y_\ell - \bar{y}_{ijk}) + \left[\frac{\partial \mathbf{u}^n}{\partial z}\right]_{ijk} (z_\ell - \bar{z}_{ijk}) \, , \qquad \text{(L-1)}$$

where the summation is over the vertices m of cell (ijk); M'_m is the mass associated with vertex m;

$$\left[\frac{\partial \mathbf{u}^n}{\partial x}\right]_{ijk}, \; \left[\frac{\partial \mathbf{u}^n}{\partial y}\right]_{ijk}, \text{ and } \left[\frac{\partial \mathbf{u}^n}{\partial z}\right]_{ijk}$$

are the velocity gradients calculated as in Appendix K; $\bar{x}_{ijk}$ is the center-of-mass x-coordinate

$$\bar{x}_{ijk} = \frac{\sum_m M'_m x_m}{\sum_m M'_m} \; ; \qquad \text{(L-2)}$$

and $\bar{y}_{ijk}$ and $\bar{z}_{ijk}$ are center-of-mass y- and z-coordinates defined in an analogous fashion. It is seen that $\mathbf{u}_\ell{}^g$ is the sum of four terms. The first is independent of ℓ and is the mass-averaged vertex velocity. The remaining three sets of velocities vary linearly in physical space, have mass-averaged velocity equal to zero, and, as we will show below, each has a derivative approximately equal to that of $\mathbf{u}^n$ in one Cartesian coordinate direction and derivatives equal to zero in the other two directions. It is natural to assume that velocity fields that vary linearly in space do not possess hourglass modes.

The hourglass modes for computational cell (i,j,k) are then given by

$$\mathbf{u}_\ell^{anc} = \mathbf{u}_\ell^n - \mathbf{u}_\ell^g \, . \qquad \text{(L-3)}$$

Since the computed mean value and derivatives of $\mathbf{u}_\ell{}^g$ are equal to those of $\mathbf{u}_\ell{}^n$ in cell (i,j,k), the mean value and derivatives of $\mathbf{u}_\ell{}^{anc}$ are zero in cell (i,j,k). To damp the hourglass modes we subtract a fraction $\delta\mathbf{u}_\ell{}^{anc}$ of $\mathbf{u}_\ell{}^{anc}$ from the velocity of each vertex ℓ:

$$\delta\mathbf{u}_\ell^{anc} = \frac{ANC}{2}\,\mathbf{u}_\ell^{anc} . \tag{L-4}$$

The quantity ANC is a user-input parameter whose value is typically taken to be 0.05. A factor of $\frac{1}{2}$ is inserted into Eq. (L-4) so that the effect of our alternate node coupler conforms in some simple cases to that of previous node couplers.[49]

It is easy to verify that our node coupler conserves momentum. Indeed, by summing $\mathbf{M}'_\ell\delta\mathbf{u}_\ell{}^{anc}$ over all vertices ℓ and substituting from Eqs. (L-4), (L-3), and (L-1), one obtains

$$\sum_\ell M'_\ell \delta\mathbf{u}_\ell^{anc} = \frac{ANC}{2}\sum_\ell M'_\ell(\mathbf{u}_\ell^n - \mathbf{u}_\ell^g) = \frac{ANC}{2}\left(\sum_\ell M'_\ell\mathbf{u}_\ell^n - \sum_\ell M'_\ell\mathbf{u}_\ell^n\right) = 0 . \tag{L-5}$$

We now show that the computed velocity derivatives of $\delta\mathbf{u}_\ell{}^{anc}$ in cell (i,j,k) are approximately zero. The finite-difference approximations to these velocity derivatives are given in Appendix K. It can be seen that

$$\beta_1\left[\frac{\partial\mathbf{u}^1}{\partial x}\right]_{ijk} + \beta_2\left[\frac{\partial\mathbf{u}^2}{\partial x}\right]_{ijk} = \left[\frac{\partial}{\partial x}(\beta_1\mathbf{u}^1 + \beta_2\mathbf{u}^2)\right]_{ijk} , \tag{L-6}$$

where $[\partial/\partial x]_{ijk}$ denotes the finite-difference approximation to the derivative in the x-direction in cell (ijk), β_1 and β_2 are constant, and $\mathbf{u}_1$ and $\mathbf{u}_2$ are two sets of velocity values associated with the vertices of cell (i,j,k). Using the defining equations of $\delta\mathbf{u}_\ell{}^{anc}$ and Eq. (L-6), it is seen that in order to show

$$\left[\frac{\partial(\delta\mathbf{u}^{anc})}{\partial x}\right]_{ijk} = 0 , \tag{L-7}$$

it is sufficient to show

$$\left[\frac{\partial\mathbf{u}^g}{\partial x}\right]_{ijk} = \left[\frac{\partial\mathbf{u}^n}{\partial x}\right]_{ijk} . \tag{L-8}$$

Using Eqs. (L-1) and (K-2)-(K-4), we see that

$$\left[\frac{\partial \mathbf{u}^g}{\partial x}\right]_{ijk} = \frac{1}{V^n_{ijk}}\left\{\bar{\mathbf{u}}_{ijk}\sum_\alpha \mathbf{A}_\alpha\cdot\mathbf{i} + \left[\frac{\partial \mathbf{u}^n}{\partial x}\right]_{ijk}\sum_\alpha(\bar{x}_\alpha - \bar{x}_{ijk})\mathbf{A}_\alpha\cdot\mathbf{i}\right.$$

$$\left. + \left[\frac{\partial \mathbf{u}^n}{\partial y}\right]_{ijk}\sum_\alpha(\bar{y}_\alpha - \bar{y}_{ijk})\mathbf{A}_\alpha\cdot\mathbf{i} + \left[\frac{\partial \mathbf{u}^n}{\partial z}\right]_{ijk}\sum_\alpha(\bar{z}_\alpha - \bar{z}_{ijk})\mathbf{A}_\alpha\cdot\mathbf{i}\right\}, \qquad \text{(L-9)}$$

where

$$\bar{u}_{ijk} = \frac{\sum_\ell M'_\ell u^n_\ell}{\sum_\ell M'_\ell},$$

the sums are over the faces α of cell (i,j,k), and $\bar{x}_\alpha$, $\bar{y}_\alpha$, and $\bar{z}_\alpha$ are the arithmetic averages of the x-, y-, and z-coordinates of the four vertices bordering face α. By using the identity

$$\sum_\alpha \mathbf{A}_\alpha = 0, \qquad \text{(L-10)}$$

Eq. (L-9) can be reduced to

$$\left[\frac{\partial \mathbf{u}^g}{\partial x}\right]_{ijk} = \frac{1}{V^n_{ijk}}\left\{\left[\frac{\partial \mathbf{u}^n}{\partial x}\right]_{ijk}\sum_\alpha \bar{x}_\alpha\mathbf{A}_\alpha\cdot\mathbf{i} + \left[\frac{\partial \mathbf{u}^n}{\partial y}\right]_{ijk}\sum_\alpha \bar{y}_\alpha\mathbf{A}_\alpha\cdot\mathbf{i} + \left[\frac{\partial \mathbf{u}^n}{\partial z}\right]_{ijk}\sum_\alpha \bar{z}_\alpha\mathbf{A}_\alpha\cdot\mathbf{i}\right\}. \qquad \text{(L-11)}$$

Now $\sum_\alpha \bar{x}_\alpha \mathbf{A}_\alpha \cdot \mathbf{i}$ is a finite-difference approximation to the surface integral $\int_{S_{ijk}} x\mathbf{i}\cdot d\mathbf{A}$, where S_{ijk} is the surface of cell (i,j,k). Applying Gauss' Divergence Theorem to this surface integral gives

$$\int_{S_{ijk}} x\mathbf{i}\cdot d\mathbf{A} = \int_{V_{ijk}} \nabla\cdot(x\mathbf{i})\,dV = V_{ijk}.$$

Hence we obtain

$$\sum_\alpha \bar{x}_\alpha\mathbf{i}\cdot\mathbf{A}_\alpha \approx V_{ijk}. \qquad \text{(L-12)}$$

Similarly one can show

$$\sum_{\alpha} \bar{y}_{\alpha}\mathbf{i}\cdot\mathbf{A}_{\alpha} \approx 0 \tag{L-13}$$

and

$$\sum_{\alpha} \bar{z}_{\alpha}\mathbf{i}\cdot\mathbf{A}_{\alpha} \approx 0 \; . \tag{L-14}$$

Using Eqs. (L-12), (L-13), and (L-14) in Eq. (L-11) gives

$$\left[\frac{\partial \mathbf{u}^{g}}{\partial x}\right]_{ijk} \approx \left[\frac{\partial \mathbf{u}^{n}}{\partial x}\right]_{ijk} \; , \tag{L-15}$$

which is the desired result. The analogous results for the derivatives in the y- and z-directions are obtained similarly.

Thus the computed derivatives of $\delta\mathbf{u}_{\ell}^{anc}$ in cell (i,j,k) are nearly zero. Not all previous alternate node couplers have had this desirable property. For example, consider the node coupler of the SALE-3D computer program.[49] In a two-dimensional calculation in which the solution is invariant in the k-direction, SALE-3D superimposes on the vertex velocities of cell (i,j,k) a velocity field of the form

$$\delta\mathbf{u}_{\ell}^{anc} \sim \frac{(-1)^{\ell}}{M_{\ell}'}\,\Delta^{anc} \; , \tag{L-16}$$

where $\Delta^{anc} = \mathbf{u}_1 + \mathbf{u}_3 - \mathbf{u}_2 - \mathbf{u}_4$. If the x- and i-coordinate directions coincide, if the computation cells are rectangular, and if centered differencing is used to calculate $\partial(\delta\mathbf{u}^{anc})/\partial x$, the result is

$$\left[\frac{\partial(\delta\mathbf{u}^{anc})}{\partial x}\right]_{ijk} \sim -\frac{1}{M_1} + \frac{1}{M_2} + \frac{1}{M_3} - \frac{1}{M_4} \; .$$

Thus $[\partial(\delta\mathbf{u}^{anc})/\partial x]_{ijk}$ is nonzero if this sum is nonzero.

Most previous alternate node couplers have also had the undesirable property that they sometimes subtract hourglass modes even when these are not present. Consider a computed velocity field in which $\mathbf{u} = \alpha\mathbf{x}$. By our assumption this velocity field has no hourglass modes. The alternate node coupler in CONCHAS-SPRAY[16] subtracts a velocity field

$$\delta\mathbf{u}_{\ell}^{anc} \sim (-1)^{\ell}\,\Delta^{anc} \; ,$$

where again $\Delta^{anc} = u_1 + u_3 - u_2 - u_4$. Hence $\Delta^{anc} \sim x_1 + x_3 - x_2 - x_4$, and the algorithm superimposes a nonzero hourglass mode in cells where $x_1 + x_3 - x_2 - x_4$ is not zero.

A node coupler similar in spirit to ours has been proposed by Margolin.[64] His node coupler has in common with ours the property that the superimposed vertex velocities have no associated mean velocity or velocity gradients. His method, however, introduces undesirable hourglass modes when the cell vertices do not satisfy $x_1 + x_3 - x_2 - x_4 = 0$.

APPENDIX M

QUASI-SECOND-ORDER DIFFERENCING

In this appendix we describe the quasi-second-order upwind (QSOU) differencing scheme that can be used in Phase C to calculate convection. Many methods[65, 66, 67] have been proposed for obtaining what are called monotone finite difference schemes. The QSOU method is a modification of a scheme proposed by van Leer,[22] and the basic idea is perhaps best understood by considering the family of upwind differencing schemes for one-dimensional convection that are represented in Fig. M-1. In each scheme we assume the density profile within a cell is linear and has a value at the cell center equal to the computed old-time value of density for that cell. The old-time densities in cells $i-1$, i, $i+1$, and $i+2$ are plotted using dots in Fig. M-1, and the density profiles in cells i and $i+1$ are the solid lines. The four schemes of Fig. M-1 differ only in the slope used for the density profile within each cell. For each scheme, the mass convected across a boundary moving from point A to point B is the area under the density profile between points A and B. Because of this method for calculating convection, the new-time total mass in a cell is just the area under all the old-time density profiles between the new boundaries of the cell. Assuming the left boundary of cell $i+1$ does not move, the new-time mass of cell $i+1$ is the area crosshatched in Fig. M-1.

Before discussing the four schemes of Fig. M-1 we need to define some terms. A function ρ is **monotone increasing (decreasing)** if $x_1 < x_2$ implies $\rho(x_1) \leq \rho(x_2)$ $(\rho(x_1) \geq \rho(x_2))$. A **monotone** function is one that is either monotone increasing or monotone decreasing. Now let x_i denote the location of computed density ρ_i. A difference scheme for convection is **weakly monotone** if it has the following property: if $\rho_{i+1}{}^n$ lies between $\rho_i{}^n$ and $\rho_{i+2}{}^n$ and $x_{i+1}{}^{n+1}$ lies between $x_i{}^n$ and $x_{i+2}{}^n$, then $\rho_{i+1}{}^{n+1}$ lies between $\rho_i{}^n$ and $\rho_{i+2}{}^n$. A difference scheme is **strongly monotone** if it has the following property: if

$$\rho_i^n \leq \rho_{i+1}^n \leq \rho_{i+2}^n \leq \rho_{i+3}^n \quad (\rho_i^n \geq \rho_{i+1}^n \geq \rho_{i+2}^n \geq \rho_{i+3}^n)$$

and

$$x_i^n \le x_{i+1}^{n+1} \le x_{i+2}^{n+1} \le x_{i+3}^n ,$$

then

$$\rho_i^n \le \rho_{i+1}^{n+1} \le \rho_{i+2}^{n+1} \le \rho_{i+3}^n \quad (\rho_i^n \ge \rho_{i+1}^{n+1} \ge \rho_{i+2}^{n+1} \ge \rho_{i+3}^n) .$$

The strongly monotone property is very desirable because strongly monotone schemes do not have the undershoots and overshoots of many higher order methods.[21] It is easy to show that for the family of schemes of Fig. M-1, a sufficient condition for strong-monotonicity is the following: if $\rho_{i+1}{}^n$ lies between $\rho_i{}^n$ and $\rho_{i+2}{}^n$, then the density profiles in cells i, $i+1$, and $i+2$, taken together, form a monotone function. We call this condition the **monotone-profiles** condition.

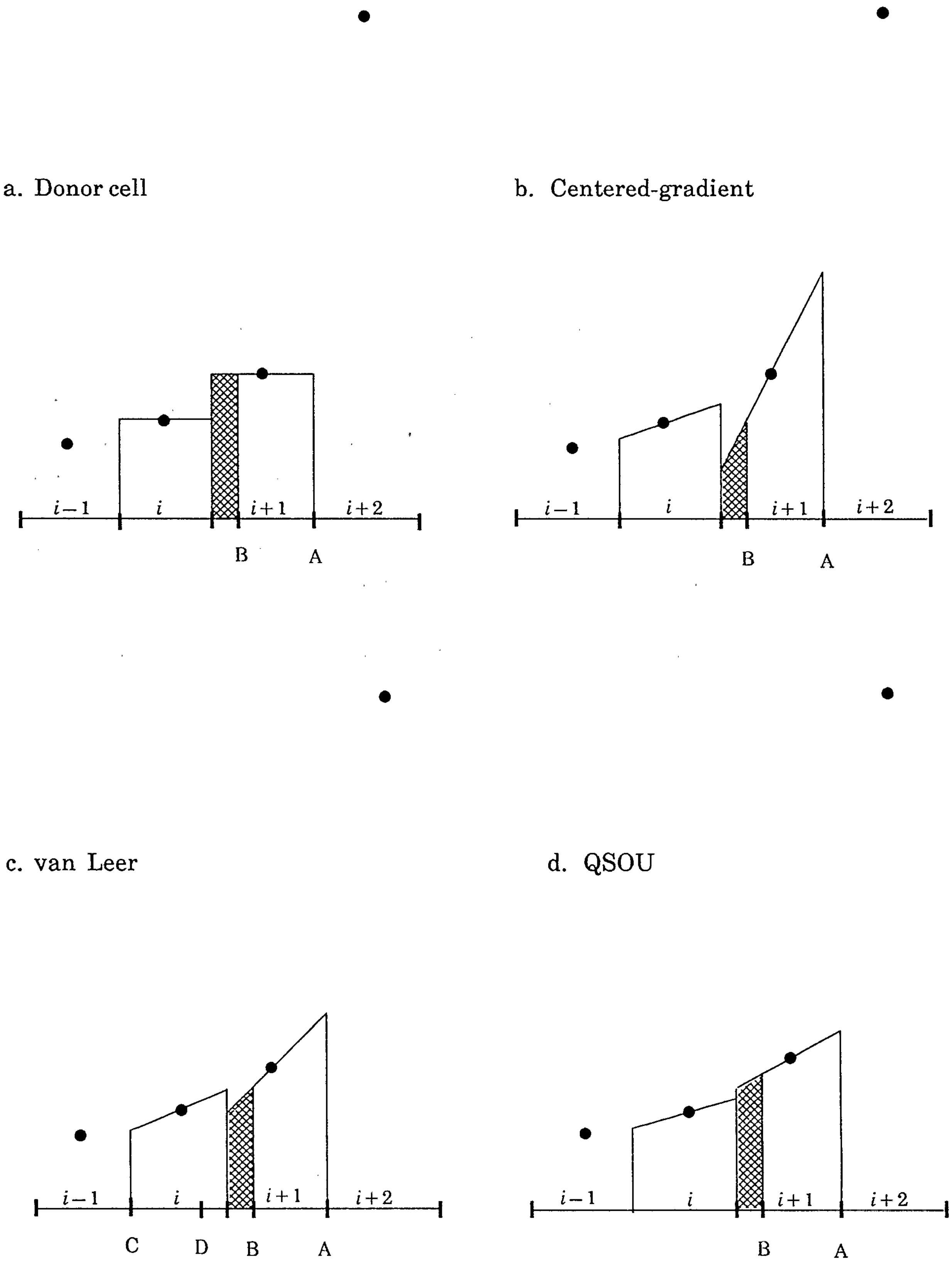
Let us now consider each of the schemes of Fig. M-1. In donor cell differencing, the slope within each cell is taken to be zero. Donor cell differencing satisfies the monotone-profiles condition and hence is strongly monotone. Donor cell differencing is first-order accurate in space, however, and has too much numerical diffusion for many applications.

To remedy this accuracy problem, one might consider the centered-gradient scheme of Fig. M-1b. Here the slope $\partial\rho/\partial x$ within each cell is given by

$$\left.\frac{\partial\rho}{\partial x}\right|_i = \frac{\rho_{i+1} - \rho_{i-1}}{2\Delta x} . \tag{M-1}$$

This scheme is stable and second-order accurate, but not weakly monotone. The reason is shown in Fig. M-1b. The density profiles in regions of large change in $\partial\rho/\partial x$ can have large undershoots or overshoots. In Fig. M-1b the density profile in cell $i+1$ has values less than $\rho_i{}^n$, and hence $\rho_{i+1}{}^{n+1}$, which is the average value of ρ under the crosshatched area of the profile, is less than $\rho_i{}^n$.

Van Leer's scheme corrects this problem by limiting the magnitude of the slope.[22] If $\rho_i{}^n$ lies between $\rho_{i-1}{}^n$ and $\rho_{i+1}{}^n$, then the magnitude of the slope $\partial\rho/\partial x|_i$ is required to be small enough that the density profile in cell i assumes values between $\rho_{i-1}{}^n$ and $\rho_{i+1}{}^n$. If $\rho_i{}^n$ does not lie between $\rho_{i-1}{}^n$ and $\rho_{i+1}{}^n$, then the slope $\partial\rho/\partial x|_i$ is taken to be zero. The resulting change in the slope in cell $i+1$ is shown in Fig. M-1c. The slope in cell i is not limited because the density profile in this cell already assumes values between $\rho_{i-1}{}^n$ and $\rho_{i+1}{}^n$. It can be shown that Van Leer's scheme is weakly monotone and is second-order accurate for computed densities for which the slope-limiting procedure is not used. The



a. Donor cell

b. Centered-gradient

c. van Leer

d. QSOU

Fig. M-1. Density profiles for a family of upwind convection schemes.

scheme is not strongly monotone. For example, if the left boundary of cell i is moved from point C to point D in Fig. M-1c, then we would have $\rho_i^{n+1} > \rho_{i+1}^{n+1}$ even though the values ρ_{i-1}^n, ρ_i^n, ρ_{i+1}^n, and ρ_{i+2}^n are monotone increasing.

The QSOU scheme was devised to satisfy the monotone-profiles condition and hence to be strongly monotone. The QSOU scheme might also be called the minimum gradient scheme because if ρ_i^n lies between ρ_{i-1}^n and ρ_{i+1}^n then the slope is taken to be

$$\left.\frac{\partial\rho}{\partial x}\right|_i = \frac{sign\,(\rho_i^n - \rho_{i-1}^n)}{\Delta x}\,min\,(|\rho_i^n - \rho_{i-1}^n|\,,\,|\rho_{i+1}^n - \rho_i^n|\;. \tag{M-2}$$

As with van Leer's scheme, if ρ_i^n does not lie between ρ_{i-1}^n and ρ_{i+1}^n then $\partial\rho/\partial x|_i$ is taken to be zero. In the example of Fig. M-1d, the slopes in cells i and $i+1$ are both limited by this prescription.

We now show the QSOU prescription for $\partial\rho/\partial x|_i$ satisfies the monotone-profiles condition. Suppose $\rho_{i-1}^n \le \rho_i^n \le \rho_{i+1}^n$; the monotone-decreasing case is handled similarly. Then, according to the QSOU prescription for the slopes,

$$\left.\frac{\partial\rho}{\partial x}\right|_i = min\left(\frac{\rho_{i+1}^n - \rho_i^n}{\Delta x}\;,\;\frac{\rho_i^n - \rho_{i-1}^n}{\Delta x}\right),$$

$$0 \le \left.\frac{\partial\rho}{\partial x}\right|_{i-1} \le \frac{\rho_i^n - \rho_{i-1}^n}{\Delta x}\;,$$

and

$$0 \le \left.\frac{\partial\rho}{\partial x}\right|_{i+1} \le \frac{\rho_{i+1}^n - \rho_i^n}{\Delta x}\;. \tag{M-3}$$

From (M-3) we see that the density profiles are monotone increasing within cells $i-1$, i, and $i+1$. It remains to show that

$$\rho_{i-1} + \left.\frac{\partial\rho}{\partial x}\right|_{i-1}\frac{\Delta x}{2} \le \rho_i - \left.\frac{\partial\rho}{\partial x}\right|_i\frac{\Delta x}{2}$$

and

$$\rho_i + \left.\frac{\partial\rho}{\partial x}\right|_i \frac{\Delta x}{2} \le \rho_{i+1} - \left.\frac{\partial\rho}{\partial x}\right|_{i+1} \frac{\Delta x}{2} \;, \tag{M-4}$$

and these inequalities are easily proven to be true by using the inequalities (M-3).

The QSOU scheme is second-order accurate in space only when $(\rho_{i+1} - \rho_i)/\Delta x$ is a constant independent of i, but the scheme will be nearly second-order accurate when $(\rho_{i+1} - \rho_i)/\Delta x$ varies slowly on the scale of the mesh spacing. When $(\rho_{i+1} - \rho_i)/\Delta x$ is constant, the scheme reduces to interpolated donor cell differencing (see Appendix N). When $\partial\rho/\partial x|_i = 0$, the scheme reduces to donor cell differencing. The QSOU scheme is method for selecting an amount of upwind differencing that maximizes accuracy while maintaining strong-monotonicity.

For future reference, we summarize here the QSOU scheme for a one-dimensional mesh with variable cell sizes. Let $\Delta x_i = x_{i+1} - x_i$ and $\Delta\rho_i = \rho_{i+1} - \rho_i$. One first calculates the slopes in each cell according to

$$\left.\frac{\partial\rho}{\partial x}\right|_i = \begin{cases} sign\,(\Delta\rho_i)\, min\left(\dfrac{|\Delta\rho_i|}{\Delta x_i} \;,\; \dfrac{|\Delta\rho_{i-1}|}{\Delta x_{i-1}}\right) & if \quad \Delta\rho_i \Delta\rho_{i-1} > 0 \\ 0 & if \quad \Delta\rho_i \Delta\rho_{i-1} < 0 \;. \end{cases} \tag{M-5}$$

Then the density ρ_α, used for fluxing through cell face α between cells i and $i+1$, is given by

$$\rho_\alpha = \begin{cases} \rho_i + \left.\dfrac{\partial\rho}{\partial x}\right|_i (x_\alpha - x_i)\left(1 - \dfrac{\delta V_\alpha}{V_i}\right) & if \quad \delta V_\alpha > 0 \\ \rho_{i+1} - \left.\dfrac{\partial\rho}{\partial x}\right|_{i+1} (x_{i+1} - x_\alpha)\left(1 + \dfrac{\delta V_\alpha}{V_{i+1}}\right) & if \quad \delta V_\alpha < 0 \;. \end{cases} \tag{M-6}$$

In Eq. (M-6) x_α is the location of cell face α, δV_α is the flux volume, taken positive if cell i is the upwind cell, and V_i is the volume of cell i.

There are many possible ways to extend this method to a three-dimensional mesh of arbitrary hexahedrons, and we have chosen a simple extension in which the fluxes in each coordinate direction depend only on gradients in that coordinate direction. Consider the determination of the quantities $\rho_\alpha{}^v$, where ρ is one of the cell-centered quantities to be convected ($\rho = \rho_m$, ρI, ρk, or $\rho k^{3/2}/\varepsilon$), v is the convective subcycle number, and α is the common face between cells (i,j,k) and $(i+1,j,k)$. Fluxes in the j- and k-coordinate directions are treated analogously. Using a straightforward extension of (M-5), we first calculate the cell-centered derivatives of ρ with respect to distance s in the i-coordinate direction:

$$\left.\frac{\partial\rho}{\partial s}\right|^{\nu}_{ijk} = \begin{cases} sign\,(\Delta\rho^{\nu}_{i})\, min\left(\dfrac{|\Delta\rho^{\nu}_{i}|}{|\Delta \mathbf{x}_{i}|}\,,\, \dfrac{|\Delta\rho^{\nu}_{i-1}|}{|\Delta \mathbf{x}_{i-1}|}\right) & if\;\; \Delta\rho^{\nu}_{i}\Delta\rho^{\nu}_{i-1} > 0 \\ 0 & if\;\; \Delta\rho^{\nu}_{i}\Delta\rho^{\nu}_{i-1} < 0\;. \end{cases} \qquad \text{(M-7)}$$

In (M-7),

$$\Delta\rho^{\nu}_{i} = \rho^{\nu}_{i+1,j,k} - \rho^{\nu}_{i,j,k}$$

and

$$\Delta \mathbf{x}_{i} = \mathbf{x}^{c}_{i+1,j,k} - \mathbf{x}^{c}_{i,j,k}\;,$$

where the cell-centered locations $\mathbf{x}_{ijk}{}^{c}$ are calculated using Eq. (55) and the new-time vertex locations. Then $\rho_{\alpha}{}^{\nu}$ is obtained from an extension of (M-6):

$$\rho^{\nu}_{\alpha} = \begin{cases} \rho^{\nu}_{i,j,k} + \left.\dfrac{\partial\rho}{\partial s}\right|^{\nu}_{i,j,k} |\mathbf{x}_{\alpha} - \mathbf{x}^{c}_{i,j,k}|\left(1 - \dfrac{\delta V_{\alpha}}{V^{\nu}_{i,j,k}}\right) & if\;\; \delta V_{\alpha} > 0 \\ \rho^{\nu}_{i+1,j,k} - \left.\dfrac{\partial\rho}{\partial s}\right|^{\nu}_{i+1,j,k} |\mathbf{x}_{\alpha} - \mathbf{x}^{c}_{i+1,j,k}|\left(1 + \dfrac{\delta V_{\alpha}}{V^{\nu}_{i+1,j,k}}\right) & if\;\; \delta V_{\alpha} < 0\;. \end{cases} \qquad \text{(M-8)}$$

In (M-8) $\mathbf{x}_{\alpha}$ is the simple average of the new-time locations of the four vertices of cell face α, $V_{i,j,k}{}^{\nu}$ is the cell volume after ν convective subcycles and is given by Eq. (120), and δV_{α} is the flux volume associated with face α. The quantity δV_{α} is positive if volume is being added to cell $(i+1, j, k)$ by the movement of face α, in which case cell (i,j,k) is the upwind cell, and δV_{α} is negative if cell $(i+1,j,k)$ is losing volume, in which case cell $(i+1,j,k)$ is the upwind cell.

Special prescriptions are needed when cells are located next to computational boundaries. If face α lies on a wall or an outflow boundary and cell (i,j,k) is the fluid cell, one of whose faces is α, then

$$\left.\frac{\partial\rho}{\partial s}\right|^{\nu}_{i,j,k} = 0\;,$$

where the derivative is with respect to distance in the coordinate direction going into the wall. The derivatives in the other coordinate directions are unaffected, unless other faces of cell (i,j,k) lie on computational boundaries. If face α lies on an inflow boundary, then (M-7) is used with modification to calculate the derivative with respect to the coordinate direction going into the inflow boundary. The modification is that the density and location of the cell center on the other side of the inflow boundary are taken to be the prescribed inflow density and center $\mathbf{x}_\alpha$ of cell face α, respectively. If face α lies on an inflow or outflow boundary, we also replace prescription (M-8) with pure donor cell differencing.

Prescriptions are also required for the velocities $\mathbf{u}_\beta{}^v$ used for fluxing momentum across the composite faces of the momentum cells. We describe fluxing of u-momentum in the i-coordinate direction; fluxing of other velocity components in the other coordinate directions is treated similarly. First, using a straightforward extension of (M-5), we calculate for each vertex the derivatives with respect to distance in the i-coordinate direction:

$$\frac{\partial u}{\partial s}\bigg|_{ijk}^{v} = \begin{cases} sign\,(\Delta u_i^v)\, min\left(\dfrac{|\Delta u_i^v|}{|\Delta \mathbf{x}_i|}\,,\ \dfrac{|\Delta u_{i-1}^v|}{|\Delta \mathbf{x}_{i-1}|}\right) & if \quad \Delta u_i^v\, \Delta u_{i-1}^v > 0 \\[2ex] 0 & if \quad \Delta u_i^v\, \Delta u_{i-1}^v < 0\ . \end{cases} \tag{M-9}$$

In (M-9)

$$\Delta u_i^v = u_{i+1,j,k}^v - u_{i,j,k}^v$$

and

$$\Delta \mathbf{x}_i = \mathbf{x}_{i+1,j,k}^{n+1} - \mathbf{x}_{i,j,k}^{n+1}\ .$$

When vertex (i,j,k) lies on a computational boundary, then the derivative, with respect to distance in the coordinate direction going into the boundary, is taken to be zero.

In order to calculate $u_\beta{}^v$ by extending (M-6), we also need a quantity analogous to $\delta V_\alpha / V_i$. In a one-dimensional calculation with constant cross-sectional area, this quantity is the Courant number based on the fluid velocity relative to the grid velocity. It is convenient to base its counterpart for momentum fluxing on the ratio of the mass flux $(\delta M_\beta{}^c)^v$, defined in Sec. III.G, to an open-flow-area mass $(M_\beta{}^o)^v$ that is defined by

$$(M_\beta^o)^v = \frac{1}{4} \sum_{(i,j,k)} F_{i,j,k}\, M_{i,j,k}^v\ , \tag{M-10}$$

where $F_{i,j,k}$ is the cell flag for cell (i,j,k), defined in Sec. IV.F, $M_{i,j,k}{}^{\nu}$ is the mass of cell (i,j,k) after ν convective subcycles, and the sum is over the four regular cells that have in common the regular cell edge passing through face β and joining vertices (i,j,k) and $(i+1,j,k)$. The formula for calculating $u_\beta{}^{\nu}$ is then

$$u^{\nu}_{\beta} = \begin{cases} u^{\nu}_{i,j,k} + \left.\dfrac{\partial u}{\partial s}\right|^{\nu}_{i,j,k} \dfrac{\left|\mathbf{x}^{n+1}_{i+1,j,k} - \mathbf{x}^{n+1}_{i,j,k}\right|}{2} \left[1 - \dfrac{(\delta M^{c}_{\beta})^{\nu}}{(M^{o}_{\beta})^{\nu}}\right] & if \quad (\delta M^{c}_{\beta})^{\nu} > 0 \\ u^{\nu}_{i+1,j,k} - \left.\dfrac{\partial u}{\partial s}\right|^{\nu}_{i+1,j,k} \dfrac{\left|\mathbf{x}^{n+1}_{i+1,j,k} - \mathbf{x}^{n+1}_{i,j,k}\right|}{2} \left[1 + \dfrac{(\delta M^{c}_{\beta})^{\nu}}{(M^{o}_{\beta})^{\nu}}\right] & if \quad (\delta M^{c}_{\beta})^{\nu} < 0\,. \end{cases} \tag{M-11}$$

The mass flux $(\delta M_\beta{}^c)^\nu$ is taken to be positive if mass is being added to vertex $(i+1,j,k)$ by the movement of face β, in which case momentum cell (i,j,k) is the upwind cell. If $(\delta M_\beta{}^c)^\nu$ is negative, momentum cell $(i+1,j,k)$ is the upwind cell.

APPENDIX N

PARTIAL DONOR CELL DIFFERENCING

Here we describe the partial donor cell differencing procedure that can be used to evaluate the cell-face quantities $Q_\alpha{}^\nu$, where Q stands for any of the variables ρ_m, ρI, ρk, ρL, or $\mathbf{u}$. We first describe the procedure for the cell-centered quantities ρ_m, ρI, ρk, and ρL. Let the regular cell in question be called cell 1, and let the neighboring cell that is common to the face α be called cell 2. The quantity $Q_\alpha{}^\nu$ is evaluated as an upstream-weighted average of $Q_1{}^\nu$ and $Q_2{}^\nu$. The first step is therefore to determine which cell is the upstream or donor cell. This is clearly determined by the sign of δV_α: if $\delta V_\alpha > 0$, then cell 2 is the upstream or donor cell while cell 1 is the downstream or acceptor cell, and vice versa. Thus we define

$$Q^{\nu}_{d} = \begin{cases} Q^{\nu}_{2} & if \quad \delta V_{\alpha} \;>\; 0 \\ Q^{\nu}_{1} & if \quad \delta V_{\alpha} \;<\; 0 \end{cases} \tag{N-1}$$

and

$$Q^{\nu}_{a} = \begin{cases} Q^{\nu}_{1} & if \quad \delta V_{\alpha} \;>\; 0 \\ Q^{\nu}_{2} & if \quad \delta V_{\alpha} \;<\; 0\,, \end{cases}$$

where the subscripts "d" and "a" refer to "donor" and "acceptor" respectively. The partial donor cell prescription for $Q_\alpha{}^v$ is then given by

$$Q^v_\alpha = \tfrac{1}{2}\, Q^v_d(1 + \alpha_0 + \beta_0 C) + \tfrac{1}{2}\, Q^v_a(1 - \alpha_0 - \beta_0 C)\ , \tag{N-2}$$

where α_0 and β_0 are adjustable coefficients ($0 \leq \alpha_0 + \beta_0 C \leq 1$), and

$$C = \frac{2|\delta V_\alpha|}{V_1 + V_2} \tag{N-3}$$

is an effective Courant number based on fluid speed relative to the mesh.

If $\alpha_0 = \beta_0 = 0$, the above prescription reduces to centered differencing of the convective terms, which is unconditionally unstable unless compensated by a sufficient amount of diffusion. If $\alpha_0 = 1$ and $\beta_0 = 0$, pure donor cell or upwind differencing results. This scheme is stable, but is too diffusive for most applications. Its effective numerical diffusivity is $\frac{1}{2}|u|\Delta x$ in the x-direction, $\frac{1}{2}|v|\Delta y$ in the y-direction, and $\frac{1}{2}|w|\Delta z$ in the z-direction. If $\alpha_0 = 0$ and $\beta_0 = 1$, the so-called interpolated donor cell scheme results. This is effectively a weighted average of centered and donor cell differencing, with the weighting factor set at the value for marginal stability. This scheme is less diffusive than pure donor cell differencing but is not monotone (see Sec.III.G). The optimum values of α_0 and β_0 in any particular calculation must be determined empirically, usually with reference to the nominal values $\alpha_0 = 0.1$, $\beta_0 = 1$ as a starting point.

To prevent the possible development of negative turbulent quantities, the values $\alpha_0 = 1$, $\beta_0 = 0$ are used instead of the input values when $Q = \rho k$ or ρL. That is, the convective transport of turbulent energy and length scale is always done by pure donor cell differencing.

The procedure for evaluating $\mathbf{u}_\beta$ on the momentum cell composite faces is entirely similar but is based on $\delta M_\beta{}^c$ instead of δV_α. Let the vertex in question be called vertex 1, and let the neighboring vertex which shares face β be vertex 2. Define

$$\mathbf{u}^v_d = \begin{cases} \mathbf{u}^v_2 & \textit{if} \quad (\delta M^c_\beta)^v \ > \ 0 \\ \mathbf{u}^v_1 & \textit{if} \quad (\delta M^c_\beta)^v \ < \ 0 \end{cases} \tag{N-4}$$

and

$$\mathbf{u}^v_a = \begin{cases} \mathbf{u}^v_1 & \textit{if} \quad (\delta M^c_\beta)^v \ > \ 0 \\ \mathbf{u}^v_2 & \textit{if} \quad (\delta M^c_\beta)^v \ < \ 0\ . \end{cases}$$

Then $\mathbf{u}_\alpha{}^\nu$ is given by

$$u_\alpha^\nu = \tfrac{1}{2}\,\mathbf{u}_d^\nu(1 + \alpha_0 + \beta_0 C') + \tfrac{1}{2}\,\mathbf{u}_a^\nu(1 - \alpha_0 - \beta_0 C')\,, \qquad \text{(N-5)}$$

where

$$C' = \frac{4\left|(\delta M_\beta^c)^\nu\right|}{M_a^{n+1} + M_b^{n+1} + M_c^{n+1} + M_d^{n+1}}\,, \qquad \text{(N-6)}$$

where a, b, c, and d refer to the four regular cells sharing the regular cell edge joining vertices 1 and 2.

APPENDIX O

ROBIN HOOD ALGORITHM

The Phase C rezoning algorithm mimics the advection terms of the Navier-Stokes equations and consequently suffers from the usual numerical difficulties associated with those terms. The worst two problems are numerical diffusion and dispersion truncation errors. The first of these is well known, and the KIVA program allows the user the option of using interpolated donor cell differencing or quasi-second-order upwind (QSOU) differencing to reduce this artificial smoothing to more or less acceptable levels while retaining stability. Dispersion errors can be significant when using interpolated donor cell differencing, causing artificial ripples in the solution. The diagnostic characteristic is spatial oscillations with a period of several cells, which tend to be most severe in regions of steep gradients such as shock waves or flames. These oscillations can be severe enough to drive species masses negative in some cells, and it is this behavior that the Robin Hood (RH) algorithm was designed to prevent. If the relevant quantities are positive in a cell, it is unaffected by RH, and RH is not needed, and therefore not used, if QSOU differencing is used.

RH is a primitive form of the flux limiter that derives its name from the fact that it steals from rich cells and gives to the poor. Suppose that a given cell has a negative species mass. RH searches the six facing neighboring cells and picks the one with the largest mass of the particular species. Enough species mass is removed from that cell to bring its poor neighbor up to zero mass for that species. The specific internal energies and turbulent kinetic energies and length scales of the two cells must be adjusted for this mass exchange. If the rich cell does not have enough species mass to bring its neighbor up to zero, all of the mass of that species is transferred to its neighbor. In this way, negative values are reduced without introducing new ones.

While the RH procedure may seem ad hoc, it probably does more good than harm. The diffusion that it represents is typically localized in space and time, and it does reduce the magnitude of dispersive errors slightly. Most importantly, it represents a simple attempt to limit convective fluxes to physically realistic values, maintaining positivity in quantities that must be positive to allow calculation of other quantities such as chemical reaction rates.

APPENDIX P

ANGULAR MOMENTUM CONSERVATION LOGIC

In an axisymmetric swirling flow with free-slip boundaries, the total angular momentum should be conserved. However, in their basic form the KIVA difference approximations to the momentum equations simply conserve the three Cartesian components of momentum, and this does not imply angular momentum conservation because of truncation errors. This lack of conservation is a serious problem for calculations of swirling flow in internal combustion engine cylinders because the truncation error effects are typically larger than the legitimate physical swirl decay due to boundary layer drag. In practice, only the truncation errors in the rezone calculation are found to be significant. We have therefore devised an optional angular momentum correction procedure which is incorporated into the rezone calculation of Phase C.

The essence of the procedure is most easily explained in the context of a model problem, namely the pure Eulerian convection of momentum represented by the differential equation

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = 0 \ . \tag{P-1}$$

Consider the augmented differential system

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \beta[(y - y_0)\mathbf{i} - (x - x_0)\mathbf{j}] \ , \tag{P-2}$$

$$\frac{\partial(\rho s)}{\partial t} + \nabla \cdot (\rho s \mathbf{u}) = 0 \ , \tag{P-3}$$

$$s = (x - x_0)v - (y - y_0)u \ , \tag{P-4}$$

which is to be regarded as a system of five equations for the five dependent variables u, v, w, β, and s. Equation (P-4), which shows that s is the angular momentum per unit mass about the axis $(x,y) = (x_0,y_0)$, plays the role of a constraint which implicitly determines the dependent variable $\beta = \beta(\mathbf{x},t)$. An equation for β can be obtained by combining Eqs. (P-2)-(P-4), and when this is done one finds that β is identically zero. Thus the system of Eqs. (P-2) and (P-4) is precisely equivalent to the original Eq. (P-1).

The idea now is to write difference approximations for Eqs. (P-2)-(P-4) instead of the equivalent Eq. (P-1). These difference equations will of course not be equivalent to those obtained by naively differencing Eq. (P-1) because of truncation errors. Differencing Eqs. (P-2)-(P-4) is preferable because this system explicitly contains a conservation equation for the angular momentum density s, and so a conservative differencing of this equation will automatically conserve angular momentum.

A suitable difference scheme for Eqs. (P-1)-(P-4) is

$$\frac{(\rho \mathbf{u})^{n+1} - (\rho \mathbf{u})^n}{\Delta t} + <\nabla \cdot> (\rho \mathbf{u}\mathbf{u})^n = \beta^{n+1}[(y - y_0)\mathbf{i} - (x - x_0)\mathbf{j}] \ , \tag{P-5}$$

$$\frac{(\rho s)^{n+1} - (\rho s)^n}{\Delta t} + <\nabla \cdot> (\rho s \mathbf{u})^n = 0 \ , \tag{P-6}$$

$$s^{n+1} = (x - x_0)v^{n+1} - (y - y_0)u^{n+1} \ , \tag{P-7}$$

where $<\nabla \cdot>$ is the spatial difference operator used to approximate the differential divergence operator. This scheme may be rewritten as

$$(\rho \mathbf{u})^{n+1} = \rho^{n+1}\tilde{\mathbf{u}} + \Delta t \ \beta^{n+1}[(y - y_0)\mathbf{i} - (x - x_0)\mathbf{j}] \ , \tag{P-8}$$

$$(\rho s)^{n+1} = (\rho s)^n - \Delta t <\nabla \cdot> (\rho s \mathbf{u}) \ , \tag{P-9}$$

$$s^{n+1} = (x - x_0)v^{n+1} - (y - y_0)u^{n+1} \ , \tag{P-10}$$

where $\tilde{\mathbf{u}}$ is the value of $\mathbf{u}^{n+1}$ that would result from Eq. (P-5) if β^{n+1} were zero. This of course is the value of $\mathbf{u}^{n+1}$ that would be obtained by simply differencing Eq. (P-1). The vertical component of Eq. (P-8) is just

$$w^{n+1} = \tilde{w} \ , \tag{P-11}$$

which shows that the procedure does not affect w, as one would expect. The component of Eq. (P-8) in the direction of $(x - x_0)\mathbf{i} + (y - y_0)\mathbf{j}$ is

$$(x - x_0)u^{n+1} + (y - y_0)v^{n+1} = (x - x_0)\tilde{u} + (y - y_0)\tilde{v} \ . \tag{P-12}$$

We now observe that Eqs. (P-10) and (P-12) are two equations for the two unknown quantities u^{n+1} and v^{n+1} in terms of s^{n+1}, which is obtained from Eq. (P-9). Notice that it is not necessary to explicitly solve for β^{n+1}. The solution of these equations is easily found to be

$$u^{n+1} = d^{-1}[(x - x_0)^2\tilde{u} + (x - x_0)(y - y_0)\tilde{v} - (y - y_0)s^{n+1}] \ , \tag{P-13}$$

$$v^{n+1} = d^{-1}[(x - x_0)(y - y_0)\tilde{u} + (y - y_0)^2\tilde{v} + (x - x_0)s^{n+1}] \ , \tag{P-14}$$

where $d = (x - x_0)^2 + (y - y_0)^2$. Once u^{n+1} and v^{n+1} have been determined, s^{n+1} is of no further interest and need not be retained.

The net result of the procedure may be described in the following way. One first calculates a provisional value of $\mathbf{u}^{n+1}$ using the basic difference scheme that one would adopt on other grounds, without regard to angular momentum conservation, and one further calculates s^{n+1} using the same scheme. One retains the vertical and radial components of this $\mathbf{u}^{n+1}$ but discards the tangential or azimuthal component and replaces it by the value that agrees with s^{n+1}. Described in this way, the procedure sounds ad hoc and unjustified, but the preceding development shows that it is in fact a well-defined and consistent difference approximation to the differential problem.

Application of this procedure to the rezone phase of KIVA is simple and straightforward. At the start of the rezone phase, the vertex-centered quantity s_{ijk} is initialized in terms of the Phase B velocities using Eq. (P-4) above:

$$s^B_{ijk} = (x^B_{ijk} - x_0)v^B_{ijk} - (y^B_{ijk} - y_0)u^B_{ijk} \ . \tag{P-15}$$

The angular momentum density $s_{ijk}{}^{\nu+1}$ is computed by replacing $\mathbf{u}_{ijk}$ with s_{ijk} in Eq. (127) of the main text, so that angular momentum is fluxed in the same way as linear momentum. The corresponding values of $u_{ijk}{}^{\nu+1}$ and $v_{ijk}{}^{\nu+1}$ then serve as the provisional values $\tilde{u}_{ijk}$ and $\tilde{v}_{ijk}$, and the final values of $u_{ijk}{}^{\nu+1}$ and $v_{ijk}{}^{\nu+1}$ are obtained from Eqs. (P-13) and (P-14) with x and y replaced by $x_{ijk}{}^{\nu+1}$ and $y_{ijk}{}^{\nu+1}$, u and v replaced by $\tilde{u}_{ijk}$ and $\tilde{v}_{ijk}$, and

u^{n+1} and v^{n+1} replaced by $u_{ijk}{}^{\nu+1}$ and $v_{ijk}{}^{\nu+1}$. The intermediate vertex positions are given by

$$\mathbf{x}^{\nu}_{ijk} = [(NS - \nu)\mathbf{x}^{\beta}_{ijk} + \nu\,\mathbf{x}^{n+1}_{ijk}]/NS \ ,$$

where NS is the number of convective subcycles. The correction procedure is optional and is activated by an input flag. It is always deactivated if input flag CYL is zero.

REFERENCES

1. A. A. Amsden, J. D. Ramshaw, P. J. O'Rourke, and J. K. Dukowicz, "KIVA: A Computer Program for Two- and Three-Dimensional Fluid Flows with Chemical Reactions and Fuel Sprays," Los Alamos National Laboratory report LA-10245-MS (February 1985).

2. A. A. Amsden, J. D. Ramshaw, L. D. Cloutman, and P. J. O'Rourke, "Improvements and Extensions to the KIVA Computer Program," Los Alamos National Laboratory report LA-10534-MS (October 1985).

3. A. A. Amsden, T. D. Butler, P. J. O'Rourke, and J. D. Ramshaw, "KIVA: A Comprehensive Model for 2D and 3D Engine Simulations," SAE Technical Paper 850554 (1985).

4. R. D. Reitz and R. Diwakar, "Structure of High Pressure Fuel Sprays," SAE Technical Paper 870598 (1987).

5. R. Diwakar, "Three-Dimensional Modeling of the In-Cylinder Gas Exchange Processes in a Uniflow-Scavenged Two-Stroke Engine," SAE Technical Paper 870596 (1987).

6. F. Grasso, M. -J. Wey, F. V. Bracco, and J. Abraham, "Three Dimensional Computations of Flows in a Stratified Charge Rotary Engine," SAE Technical Paper 870409 (1987).

7. P. J. O'Rourke and A. A. Amsden, "Three-Dimensional Numerical Simulations of the UPS-292 Stratified Charge Engine," SAE Technical Paper 870597 (1987).

8. M. A. Theobald, "A Numerical Simulation of Diesel Autoignition," Ph.D. thesis, Massachusetts Institute of Technology (1986).

9. A. J. Brown, "A Stochastic Mixing Model for Predicting Emissions in a Direct Injection Diesel Engine," Ph.D. thesis, Massachusetts Institute of Technology (1986).

10. W. K. Cheng and R. A. Gentry, "Effects of Charge Non-Uniformity on Diesel Heat Release Analysis," SAE Technical Paper 861568 (1986).

11. R. A. Gentry, B. J. Daly, and A. A. Amsden, "KIVA-COAL: A Modified Version of the KIVA Program for Calculating the Combustion Dynamics of a Coal-Water Slurry in a Diesel Engine Cylinder," Los Alamos National Laboratory report LA-11045-MS (August 1987).

12. W. C. Rivard, O. A. Farmer, and T. D. Butler, "RICE: A Computer Program for Multicomponent Chemically Reactive Flows at All Speeds," Los Alamos Scientific Laboratory report LA-5812 (March 1975).

13. H. C. Gupta and S. A. Syed, "REC-P3 (Reciprocating Engine Combustion, Planar Geometry, Third Version): A Computer Program for Combustion in Reciprocating Engines," MAE Report No. 1431, Mechanical and Aerospace Engineering Department, Princeton University (1979).

14. J. D. Ramshaw and J. K. Dukowicz, "APACHE: A Generalized-Mesh Eulerian Computer Code for Multicomponent Chemically Reactive Fluid Flow," Los Alamos Scientific Laboratory report LA-7427 (January 1979).

15. T. D. Butler, L. D. Cloutman, J. K. Dukowicz, and J. D. Ramshaw, "CONCHAS: An Arbitrary Lagrangian-Eulerian Computer Code for Multicomponent Chemically Reactive Fluid Flow at All Speeds," Los Alamos Scientific Laboratory report LA-8129-MS (November 1979).

16. L. D. Cloutman, J. K. Dukowicz, J. D. Ramshaw, and A. A. Amsden, "CONCHAS-SPRAY: A Computer Code for Reactive Flows with Fuel Sprays," Los Alamos National Laboratory report LA-9294-MS (May 1982).

17. C. W. Hirt, A. A. Amsden, and J. L. Cook, *J. Comput. Phys.* **14**, 227 (1974).

18. W. E. Pracht, *J. Comput. Phys.* **17**, 132 (1975).

19. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow* (Hemisphere Publishing Corporation, Washington D.C., 1980).

20. P. J. O'Rourke and A. A. Amsden, "Implementation of a Conjugate Residual Iteration in the KIVA Computer Program," Los Alamos National Laboratory report LA-10849-MS (October 1986).

21. P. J. Roache, *Computational Fluid Dynamics* (Hermosa Publishers, Albuquerque, New Mexico, 1982).

22. B. van Leer, *J. Comput. Phys.* **32**, 101 (1979).

23. J. D. Ramshaw, *Phys. Fluids* **23**, 675 (1980).

24. K. Meintjes and A. P. Morgan, "Element Variables and the Solution of Complex Chemical Equilibrium Problems," General Motors Research Publication GMR-5827 (April 1987).

25. J. D. Ramshaw and A. A. Amsden, *J. Comput. Phys.* **59**, 3, 484 (1985).

26. B. E. Launder and D. B. Spalding, *Mathematical Models of Turbulence* (Academic Press, New York, 1972).

27. W. C. Reynolds, "Modeling of Fluid Motions in Engines - An Introductory Overview," in *Combustion Modeling in Reciprocating Engines*, Eds. J. N. Mattavi and C. A. Amann (Plenum Press, New York 1981).

28. D. B. Spalding, *Chemical Engineering Science* **26**, 95 (1971).

29. B. F. Magnussen and B. H. Hjertager, Sixteenth Symposium (International) on Combustion (The Combustion Institute, Pittsburgh, Pennsylvania, 1977) pp. 719-729.

30. B. H. Hjertager, *Comb. Sci. Tech.* 27, 159 (1982).

31. J. Abraham, F. V. Bracco, and R. D. Reitz, *Combustion and Flame* **60**, 309 (1985).

32. J. K. Dukowicz, *J. Comput. Phys.* **35**, 229 (1980).

33. P. J. O'Rourke, "Statistical Properties and Numerical Implementation of a Model for Droplet Dispersion in a Turbulent Gas," Los Alamos National Laboratory report LA-UR-87-2376, accepted by *J. Comput. Phys.*

34. P. J. O'Rourke, "Collective Drop Effects in Vaporizing Liquid Sprays," Ph.D.thesis 1532-T, Princeton University (August 1981).

35. P. J. O'Rourke and A. A. Amsden, "The TAB Method for Numerical Calculation of Spray Droplet Breakup," SAE Technical Paper 872089 (1987).

36. J. D. Ramshaw, P. J. O'Rourke, and L. R. Stein, *J. Comput. Phys.* **58**, 361 (1985).

37. P. J. O'Rourke, oral presentation to the Twenty-third Direct Injected Stratified Charge Working Group Meeting, Los Alamos National Laboratory, March 1986.

38. D. R. Stull and H. Prophet, "JANAF Thermochemical Tables," 2nd ed. (U. S. Department of Commerce/National Bureau of Standards, NSRDS-NBS 37, June 1971). N. W. Chase et al., *J. Phys. Chem. Ref. Data* 3, 311 (1974).

39. J. O. Hinze, *Appl. Sci. Research* 1, 263, 273 (1948).

40. L. Martinelli, R. D. Reitz, and F. V. Bracco, "Comparison of Computed and Measured Dense Spray Jets," Ninth International Colloquim on Dynamics of Explosions and Reactive Systems, Poitiers, France, 1983.

41. A. U. Chatwani and F. V. Bracco, "Computation of Dense Spray Jets," ICLASS-85, London, 1985.

42. F. V. Bracco, "Modeling of Engine Sprays," SAE Paper 850394 (1985).

43. F. A. Williams, *Phys. Fluids* 1, 541 (1958).

44. H. Lamb, *Hydrodynamics* (Dover Publications, New York, 1932).

45. G. M. Faeth, *Prog. Energy Combust. Sci.* 3, 191 (1977).

46. G. I. Taylor, "The Shape and Acceleration of a Drop in a High Speed Air Stream" in *The Scientific Papers of G. I. Taylor*, Ed. G. K. Batchelor Vol III (University Press, Cambridge, England, 1963).

47. P. J. O'Rourke, "The KIVA Computer Program for Multidimensional Chemically Reactive Fluid Flows with Fuel Sprays," in *Numerical Simulation of Combustion Phenomena*, Eds. R. Glowinski, B. Larrouturou, and R. Temam (Springer-Verlag, New York, 1985).

48. A. Brandt, J. E. Dendy, and H. Ruppel, *J. Comput. Phys.* **34**, 348 (1980).

49. A. A. Amsden and H. M. Ruppel, "SALE-3D: A Simplified ALE Computer Program for Calculating Three-Demensional Fluid Flow," Los Alamos National Laboratory report LA-8905 (November 1981).

50. P. A. Thompson, *Compressible-fluid Dynamics*, (McGraw-Hill, New York, 1972).

51. N. B. Vargaftik, *Tables on the Thermophysical Properties of Liquids and Gases* (John Wiley & Sons, Hemisphere Publishing Corporation, Washington, D.C., 1975).

52. F. Rossini, et al., *Selected Values of Physical and Thermodynamic Properties of Hydrocarbons and Related Compounds* (Carnegie Press, Pittsburgh, Pennsylvania, 1953).

53. R. C. Reid and T. K. Sherwood, *The Properties of Liquids and Gases* (McGraw-Hill, New York, 1966).

54. R. I. Issa, *J. Comput. Phys.* **62**, 1, 40 (1986).

55. C. E. Leith, *Methods in Comput. Phys.* 4, 1 (1965).

56. B. P. Leonard, *Comput. Meths. Appl. Mech. Eng.* **19**, 59 (1979).

57. H. Jeffreys, *Cartesian Tensors* (Cambridge University Press, Cambridge, England, 1965).

58. S. Wahiduzzaman and C. R. Ferguson, "Convective Heat Transfer from a Decaying Swirling Flow within a Cylinder," 8th International Heat Transfer Conference, paper #86-IHTC-253 (August 1986).

59. J. W. Rose and J. R. Cooper, Eds., *Technical Data on Fuel* (John Wiley & Sons, New York, 1977).

60. F. A. Williams, *Combustion Theory* (Benjamin/Cummings Publishing Company, Menlo Park, California, 1985).

61. H. Schlichting, *Boundary-Layer Theory*, 6th ed. (McGraw-Hill, New York, 1968).

62. J. D. Ramshaw and A. A. Amsden, *J. Comput. Phys.* **71**, 1, 224 (1987).

63. J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, New York, 1970).

64. L. G. Margolin, "A Method for Treating Hourglass Patterns," in Los Alamos National Laboratory report LA-9196-PR (April 1982).

65. S. T. Zalesak, *J. Comp. Phys.* **31**, 335 (1979).

66. M. Chapman, *J. Comp. Phys.* 44, 84 (1981).

67. L. D. Cloutman, "Numerical Experiments with CONCHAS-SPRAY: Installation of the FLOE Algorithm," Los Alamos National Laboratory report LA-10626-MS (February 1986).