

NIST-GCR-92-605

Numerical Analysis Support for Compartment
Fire Modeling and Incorporation of Heat
Conduction into a Zone Fire Model

William E Moss



United States Department of Commerce
Technology Administration
National Institute of Standards and Technology

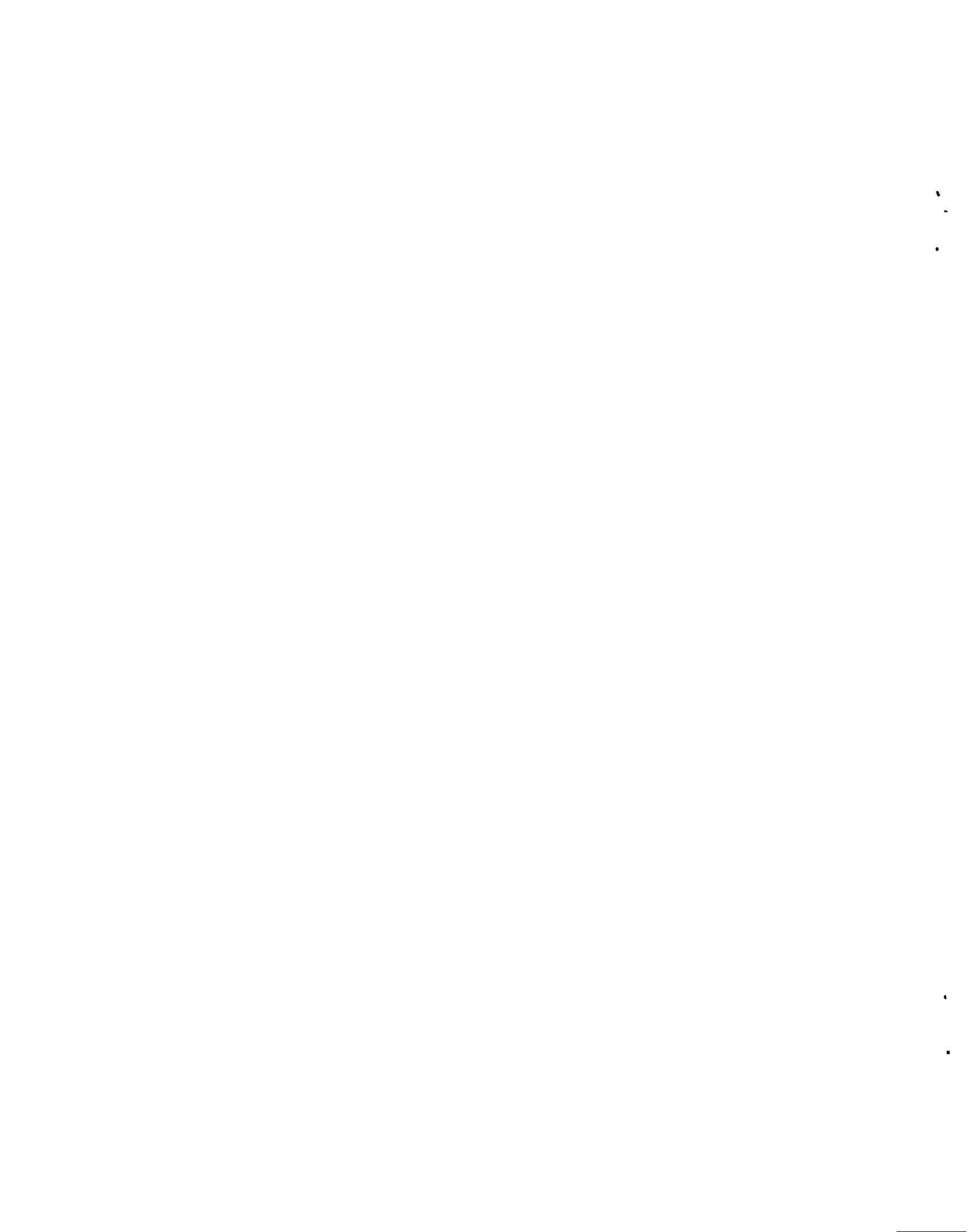
NIST-GCR-92-605

**Numerical Analysis Support for Compartment
Fire Modeling and Incorporation of Heat
Conduction into a Zone Fire Model**

William E **Moss**



United States Department of Commerce
Technology Administration
National Institute of Standards and Technology



NIST-GCR-92-605

Numerical Analysis Support for Compartment
Fire Modeling and Incorporation of Heat
Conduction into a Zone Fire Model

William E Moss

Clemson University
Department of Mathematical Sciences
Clemson, S. C. 29634-1907

Issued March 1992



Sponsored by:
U.S. Department of Commerce
Barbara Hackman Franklin, *Secretary*
Technology **Administration**
Robert M. White, *Under Secretary for Technology*
National Institute of Standards and Technology
John W. Lyons, *Director*

Notice

This report was prepared for the Building and Fire Research Laboratory of the National Institute of Standards and Technology under grant number 60NANB8D0857. The statements and conclusions contained in this report are those of the authors and do not necessarily reflect the views of the National Institute of Standards and Technology or the Building and Fire Research Laboratory.

Contents

1	Introduction	1
1.1	Background	1
1.2	Overview	2
2	MCCFM: A Zone Fire Model	3
2.1	Basic Equations.	3
2.2	Submodels, Sources of Mass and Energy	6
2.3	Code Structure	9
3	CONRAD1: Adding Conduction and Radiation to MCCFM	13
3.1	MOL, the Piecewise Cubic Hermite Collocation Approach	16
3.2	A Graded Spatial Mesh	19
3.3	Code Structure	21
4	CONRAD2: Functional Equations for Conduction	28
4.1	Solving the Heat Equation	30
4.2	Code Structure	32
5	Numerical Experiments	37
6	Future Work	40
A	Notation	42
	References	43

List of Figures

1	Two Layer Zone Model Configuration	4
2	Two Room Example	5
3	Case2: $h_b \leq Y_1 + y_{1,floor} \leq h_t$ and $Y_2 > H_t$	8
4	MCCFM Structure	13
5	Ceiling and Upper Layer Heat Transfer	15
6	Ceiling Temperature Profiles	20
7	Subroutine F Details	33

Numerical Analysis Support for Compartment Fire Modeling and Incorporation of Heat Conduction into a Zone Fire Model

William F. Moss*

Abstract

This report summarizes numerical fire modeling research conducted for NIST Grant Number 60NANB8D0857 from August 15, 1988 to March 31, 1991. The research goal for the first year of the grant was to determine the best available numerical technology for use in zone fire modeling. The goal for the second year was to incorporate heat conduction into a zone fire model in a numerically robust and efficient manner. Three prototype zone fire models named MCCFM, CONRAD1 and CONRAD2 were constructed to test the numerical technology used to realize these goals. These zone fire models and their implementations as Fortran codes are presented. The code MCCFM, developed during the first year of the grant, demonstrates the advantages of using mass as a solution variable instead of density. CONRAD1 and CONRAD2 examine two strategies for coupling the heat conduction equation (a one dimensional partial differential equation) with the zone fire modeling ordinary differential equations. CONRAD1 performs this coupling via the method of lines by using standard piecewise cubic Hermite polynomial basis functions to represent the unknown temperature profiles in the ceiling, wall, and floor heat conduction nodes. CONRAD2 reduces the heat conduction problem to a set of implicitly defined functional equations, a strategy never before used in zone fire modeling. Both CONRAD1 and CONRAD2 use a differential-algebraic equation solver. Supporting numerical results are presented with timings for a Sun Sparcstation 2.

1 Introduction

1.1 Background

This report summarizes numerical fire modeling research conducted for NIST Grant Number 60NANB8D0857 from August 15, 1988 to March 31, 1991. The research

*Department of Mathematical Sciences, Clemson University, Clemson, SC 29634-1907, U.S.A.
(bmoss@math.clemson.edu).

goal for the first year of the grant was to determine the best available numerical technology for use in zone fire modeling. The goal for the second year was to incorporate heat conduction into a zone fire model in a numerically robust and efficient manner. **Three** prototype zone fire models named MCCFM, CONRAD1 and CONRAD2 were constructed to test the numerical technology used to realize these goals. These zone fire models and their implementations **as** Fortran codes are presented.

The code MCCFM, developed during the first year of the grant, demonstrates the advantages of using **mass** **as** a solution variable instead of density. Some of the technology used in MCCFM was later incorporated into the NIST code CCFM.VENTS; in particular, the choice of solution variables. Both MCCFM and CCFM.VENTS use the stiff ordinary differential equation solver DEBDF from DEPAC.

CONRAD1 and CONRAD2 examine two strategies for coupling the heat conduction equation (a one dimensional partial differential equation) with the zone fire modeling ordinary differential equations. CONRAD1 performs this coupling via the method of lines by using standard piecewise, cubic Hermite polynomial basis functions to represent the unknown temperature profile in the ceiling, wall, and **floor** heat conduction nodes. CONRAD2 reduces the heat conduction problem to a set of implicitly defined functional equations, a strategy never before used in zone fire modeling. CONRAD1 and CONRAD2 both use the stiff differential-algebraic equation (DAE) solver DASSL. CONRAD1 is viewed **as** a benchmark code and has been used to verify the correctness of the approach in CONRAD2. Some of the technology in CONRAD2 is currently being incorporated in the NIST code CFAST; in particular, the differential-algebraic equation solver DASSL **and** the strategy for incorporating heat conduction. Supporting numerical results are presented. All timing are for a Sun Sparcstation 2 using Sun Fortran 1.4.

The basic premise of a zone fire model is that **an** enclosure can be divided into a number of regions **or** zones each with approximately uniform conditions. The zones interact by exchanging **mass** and energy. Mass and energy conservation along with expressions relating mass, energy, density, volume, temperature, and pressure can be used to show that many formulations exist for tracking conditions in zones. These formulations are equivalent in the sense that one formulation may be converted to another using physical laws such **as** the ideal gas law **or** definitions of such quantities **as** density or internal energy. Computationally, zone fire modeling is challenging due to the numerical characteristics of the basic conservation equations used to simulate **mass and energy exchange between various zones** (see [1]).

1.2 Overview

Section 2, presents the zone fire model code MCCFM. The basic ordinary differential equations solved, the submodels used, and the structure of the code are discussed.

Section 3 outlines the design of the code CONRAD1. First, heat conduction is modeled in the standard way by an initial-boundary value problem **for** the heat

equation, a partial differential equation. Second, the piecewise cubic Hermite polynomial expansion approach to the methods of lines is used to convert an initial-boundary value for a partial differential equation into a differential-algebraic equation system. The spatial mesh used in this process is graded (not uniform) and is dependent on the length of time the code will be run. Finally, the structure of the code is discussed.

Section 4 outlines the design of the code CONRAD2. It provides the details of how the heat conduction problem is converted into a system of functional equations. Again, graded spatial meshes are used in heat conduction nodes.

Section 5 is a condensation of the lab notes for the second year of the grant. What was tried, what worked, and what did not work is pointed out.

Finally, Section 6 outlines future projects to further improve the modeling of heat transfer in zone fire models.

2 MCCFM: A Zone Fire Model

In this section we examine the basic equations of the zone fire model MCCFM, briefly discuss submodels for fire and for natural and forced transfer of mass and energy through vents, and describe the code MCCFM. MCCFM was developed as a part of the first year of the grant. The project title for this portion of the grant was “Numerical Analysis Support for Compartment Fire Modeling Code Development.” This grant ran from August 15, 1988 to August 15, 1989. The goal of this project was to find the best available numerical methods for zone fire modeling and use them in MCCFM. Some of the technology developed for MCCFM was later used to improve the NIST code CCFM.VENTS (see [2, 3, 4]). In particular, the choice of solution variables used in MCCFM was a notable contribution. MCCFM also demonstrated how to compute vent flow efficiently. The vent flow equations used by MCCFM and CCFM.VENTS are essentially the same. The strategy used to compute these equations is different, however. Though MCCFM’s vent algorithm was never installed in the final version of CCFM.VENTS, the improvements made to the CCFM.VENTS vent algorithm resulted in a 2 to 3 times speed-up.

2.1 Basic Equations

The code MCCFM models the transfer of mass and energy as a fire evolves in a multiple room building. As illustrated in Figure 1 each room in a zone fire model is usually divided into two control volumes, an upper layer of hot gases and smoke and a lower layer of air. The gas in each layer has attributes of mass, energy (enthalpy), density, temperature, and volume denoted respectively by m_i , q_i , ρ_i , T_i , and V_i where $i = L$ for the lower layer and $i = U$ for the upper layer. Enthalpy is a composite term which is the sum of the internal energy of the gas and any work done by expansion of the gas. The upper and lower layer gases are considered ideal

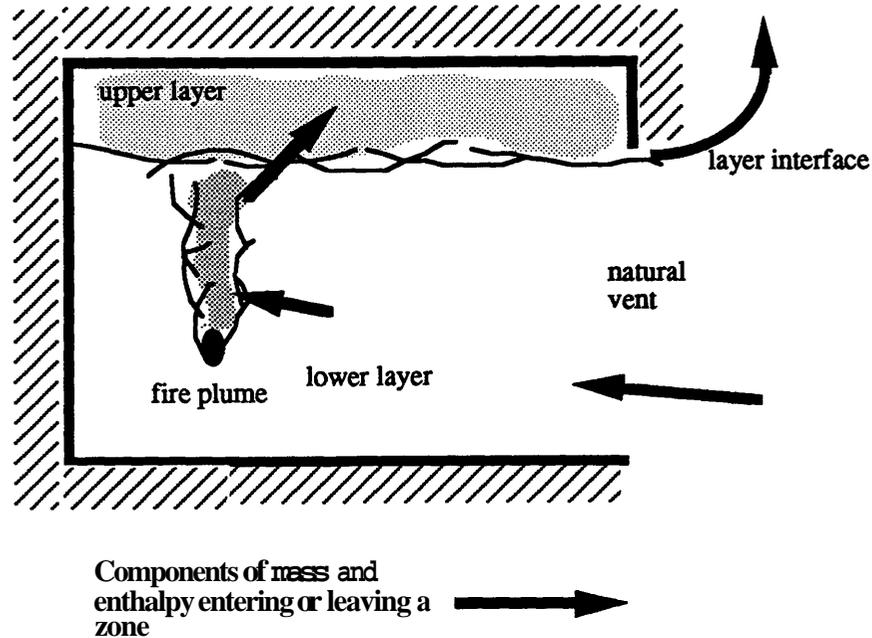


Figure 1: Two Layer Zone Model Configuration

so that their pressures, densities, and temperatures are related by the ideal gas law. Strictly, speaking both the upper and lower layers should have different pressures. It is a simplifying assumption of most zone fire models that these pressures are treated as the same. This common pressure is denoted by P , no layer distinguishing subscript being required. Figure 1 illustrates the sources of energy for a two layer zone model.

Although vertical pressure variation is ignored when using the ideal gas law, it is not ignored when modeling the flow of mass and energy through a vent. The term vent is used here for any opening such as a door, window, slit, hole, or crack that will allow gas to pass from one room to another. Unless a room is completely sealed off from the outside, the room pressure will be close to 1 atmosphere which is about 10^5 Pascals (Pa). A pressure drop across a vent as low as .1 Pa can result in considerable mass and energy transfer. For the purposes of vent flow calculations, a vertically dependent hydrostatic pressure is used. The common layer pressure P is treated as the pressure at the floor, and the pressure at height h above the floor is computed as $P - \rho_L g h$ if $h \leq Y$, where Y is the height above the floor of the interface between the upper and lower layers. If $h > Y$, then the pressure at height h is computed as $P - \rho_L g Y - \rho_U g (h - Y)$. Since room floors may have different elevations, P is further decomposed as $P = P_{\text{datum}} - \rho_{\text{ambient}} g y_{\text{floor}} + \Delta P$ where P_{datum} is the ambient pressure at the datum level, ρ_{ambient} is the ambient density,

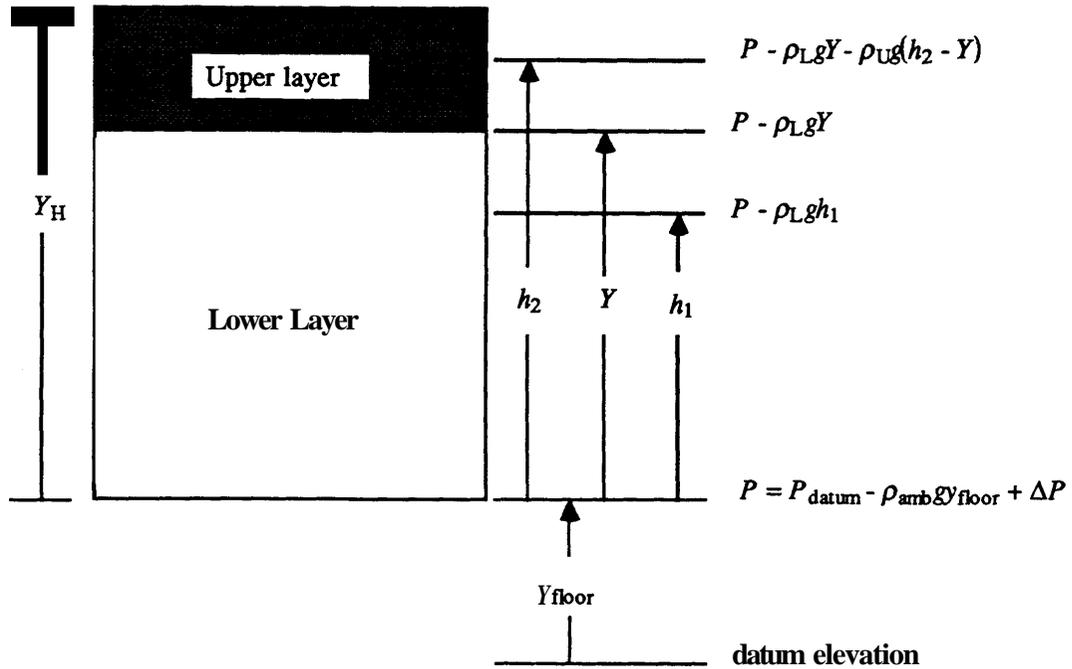


Figure 2: Two Room Example

y_{floor} is the height of the floor above the datum level, and ΔP is called the pressure offset. In MCCFM the term ambient refers to the conditions outside the building. These conditions are taken to be uniform around the building. Unlike the NIST code CCFM.VENTS, MCCFM does not attempt to model outside pressure differentials caused by winds. Consequently, before the fire starts $\Delta P = 0$. Figure 2 illustrates the variation of pressure with elevation.

The code MCCFM solves an initial value problem for a set of ordinary differential equations (ODE's). There are four ODE's for each room which track the pressure offset, ΔP , the upper layer height above the floor, Y , the lower layer mass, m_L , and the upper layer mass, m_U . At any time t after the fire starts, all physical variables can be derived from these four "solution" variables. The four ODE's are derived from conservation of energy and mass in each of the two layers. The choice of solution variables and the derivation of the ODE's was carefully discussed in [1, 2]. In these reports, it was pointed out that this particular choice of solution variables has certain advantages over the other possible choices. An early version of CCFM.VENTS (CCFM.HOLE [5]) used densities instead of masses for two of the solution variables. This code had a difficult time starting because at the time the fire starts, the upper layer density is indeterminate; it is the ratio of the upper layer mass to upper layer volume and both are zero. On the other hand, at the time the fire starts, upper layer mass is well defined and is zero. Early in the development

of MCCFM, the layer masses were chosen as solution variables and somewhat later this choice was adopted for use in CCFM.VENTS. This change of solution variables made a marked improvement in the performance of CCFM.VENTS.

The evolution of the solution variables is driven by all sources of mass and energy (enthalpy) such as fires, natural vents, forced vents, radiation, convection, and conduction. In what follows, let the rate at which mass is added to the upper and lower layers be denoted by \dot{m}_U and \dot{m}_L , and let the rate at which energy is added to the upper and lower layers be denoted by \dot{q}_U and \dot{q}_L . The four basic ODE's solved for each room are

$$\frac{d\Delta P}{dt} = \frac{\gamma - 1}{V}(\dot{q}_U + \dot{q}_L) \quad (1)$$

$$\frac{dY}{dt} = \frac{\gamma - 1}{\gamma PV}((Y_H - Y)\dot{q}_L - Y\dot{q}_U) \quad (2)$$

$$\frac{dm_U}{dt} = \dot{m}_U \quad (3)$$

$$\frac{dm_L}{dt} = \dot{m}_L, \quad (4)$$

where V denotes room volume, Y_H denotes the distance from the floor to the ceiling, and γ denotes the ratio of specific heat at constant pressure to the specific heat at constant volume. MCCFM uses a value of $\gamma = 1.4$ which corresponds to air. When not indeterminate, layer densities can be computed from the definition of density as

$$\rho_U = \frac{m_U}{(Y_H - Y)A}$$

$$\rho_L = \frac{m_L}{YA},$$

where A denotes the area of the room floor, and in turn, layer temperatures can be computed from the ideal gas law as

$$T_U = \frac{P}{\rho_U R}$$

$$T_L = \frac{P}{\rho_L R},$$

where R denotes the universal gas constant.

2.2 Submodels, Sources of Mass and Energy

The \dot{m} and \dot{q} source terms in equations (1)-(4) are derived from submodels for fires, natural vents, and forced vents. These submodels are essentially the same as those used in CCFM.VENTS (see [2]).

The submodel for a fire is based on the observation that rising hot gases from a fire carry with them some of the cooler surrounding gas. This process is referred

to as entrainment. A fraction of the energy released by the fire is considered to be lost due to radiation and conduction of heat through ceilings, walls, and floors. In MCCFM this energy is removed from the system. This is a serious defect in the model because radiative heating of room ceilings, walls, and floors was found to be significant during the development of the codes **CONRAD1** and **CONRAD2** (see Sections 3 and 4).

The submodel for a natural vent is based on Bernoulli's law for flow through an orifice. Suppose the vent connects rooms 1 and 2, and that room 1 is at the higher pressure (source room). Then

$$\dot{m}_{\text{vent}} = c_{\text{vent}} A_{\text{vent}} \sqrt{2\rho_1 \Delta P_{\text{vent}}}, \quad (5)$$

where \dot{m}_{vent} is the rate at which mass is being lost from room 1 and added to room 2, c_{vent} denotes the vent coefficient, A_{vent} denotes the area of the vent, ρ_1 denotes the density of the gas in room 1, and $\Delta P_{\text{vent}} = P_1 - P_2$ denotes the pressure drop across the vent. Equation (5) is only valid if the density, ρ_1 , and the pressure drop, ΔP_{vent} , are constant over the vertical extent of the vent. As mentioned above, pressure is treated as a piecewise linear function of elevation above the floor in the vent calculation. If a layer interface falls between the top and bottom of the vent, a further complication arises from the fact that a jump or discontinuity occurs in the density as the layer interface is crossed. The vent submodel divides the vent into a number of horizontal "slabs" of gas and then treats each slab using equation (5). In each slab ρ_1 is constant and pressure is a linear function of elevation above the floor. The value of $\sqrt{\Delta P_{\text{vent}}}$ for the slab flow calculation is given by

$$\sqrt{\Delta P_{\text{vent}}} = \frac{2}{3} \frac{|\Delta P_{\text{top}}| + \sqrt{|\Delta P_{\text{top}} \Delta P_{\text{bot}}|} + |\Delta P_{\text{bot}}|}{\sqrt{|\Delta P_{\text{top}}|} + \sqrt{|\Delta P_{\text{bot}}|}} \quad (6)$$

which takes into account the vertical variation in pressure (see [2]).

The layer in each room can be above the vent, above or below the layer in the adjoining room or below the vent. Figure 3 illustrates one of the ten cases to be considered when computing vent flow. The number of slabs for each case vary from one to three. Each slab can have one or two flows depending on whether the slab contains a neutral plane.¹ The mass flow through the vent is then computed using equations (5) and (6) using values of ρ_1 , ΔP_{top} and ΔP_{bot} according to which case the layer heights satisfy. The speed of this method for computing vent flow is then achieved by noting that during a fire simulation the vent algorithm may be invoked many times but usually subsequent invocations use the same case. Unneeded computations can be avoided by recording the case that was used last and then trying it first on the next invocation of the vent algorithm.

The vent submodel has the following input-output description. The vent is assumed to connect rooms 1 and 2.

¹elevation where the pressure drop across the vent is zero

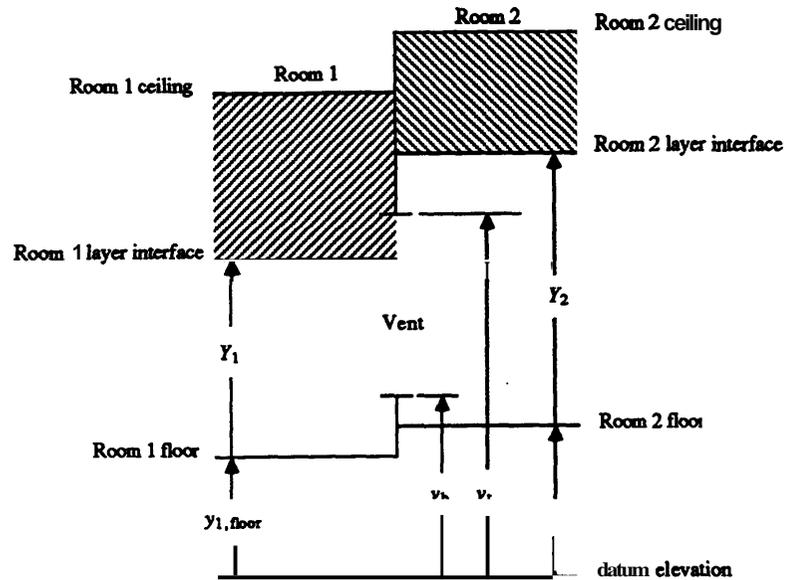


Figure 3: Case2: $h_b \leq Y_1 + y_{1, floor} \leq h_t$ and $Y_2 > H_t$

Inputs:

- o pressure offsets for **rooms** 1 and 2,
- distance from the datum to the layer interface in **rooms** 1 and 2,
- distance from the datum to the **ceiling** in **rooms** 1 and 2,
- o distance from the datum to the **floor** in **rooms** 1 and 2,
- o density **of** the layers in rooms 1 and 2,
- o temperatures of the layers in rooms 1 and 2,
- o distance from the datum to the top of the vent,
- o distance from the datum to the bottom of the vent,
- o area **of** the vent.

outputs:

- o **mass flow** rates into the layers in **rooms** 1 and 2,
- o energy **flow** rates into the layers in **rooms** 1 and 2.

The submodel for a forced vent in MCCFM is a simplified version of that found in CCFM.VENTS. It is assumed that a fan is being used to move mass through the vent and that the supply (or exhaust) rate of the fan is not influenced by the pressure drop across the fan. The input-output description of the forced vent submodel is similar to that of the vent submodel except that the volume flow rate generated by the fan is added to this description, while the offset pressures and distances from datum to floor are removed.

2.3 Code Structure

MCCFM assumes that a fire starts in one or more rooms at time $t = 0$. The initial values for the solution variables are $\Delta P = 0$, $Y = Y_H$, $m_U = 0$, and $m_L = \rho_{\text{ambient}} V$. These initial values, together with the system of ODE's described above, form an initial value problem. The number of ODE's in this system is four times the number of rooms. MCCFM solves this initial value problem and returns approximate values for the solution variables and related quantities at the final time, t_{final} , and every t_{print} seconds.

As was pointed out in [1], this system of ODE's is stiff. In general, a system is stiff if the phenomenon being modeled possesses multiple time scales that vary by several orders of magnitude. Physically, the system described here is stiff because the offset pressure ΔP adjusts to changing conditions much faster than does layer interface height or layer masses. Before about 1970, software for solving stiff systems was not generally available. Beginning with the work of Gear [6], the numerical solution of stiff systems has been an active area of research by numerical analysts. In [1] the package DEPAC was discussed. DEPAC contains three ODE solvers, one of which, called DEBDF, is suitable for solving stiff systems to moderate accuracy. DEBDF is a variable order, variable stepsize, backward differentiation formula code. One feature that distinguishes this code from its competitors is that it allows the specification of an absolute and relative error tolerance for each solution variable. Consequently, all variables do not have to be computed to the same accuracy. Early in the development of MCCFM, the decision was made to use DEBDF, since it had been determined that a stiff solver was required.

The simulation time interval $[0, t_{\text{final}}]$ for MCCFM can be broken into two types of subintervals, stiff transient and stiff. In the stiff transient subintervals, the pressure offset rapidly rises to a quasi-steady state value; that is, in a fraction of a second, typically, the pressure offset rises to a certain value (quasi-steady state) and from that point on the pressure changes slowly with time until another stiff transient occurs. The simulation begins with a stiff transient. Each time a layer interface passes a vent top (or bottom) or the fire output takes a jump, a new stiff transient begins. Nonstiff solvers can generally integrate over the stiff transient subintervals. Outside of these very short time intervals, a stiff solver is required. Since there is a large overhead associated with switching solvers, it is more efficient to use the stiff solver throughout the computation. DEBDF must work especially hard to integrate

over the initial stiff transient. To decrease the time required to integrate the initial stiff transient, **MCCFM** changes the initial value of A P from 0 to **an** approximate quasi-steady state value, **AP_∞**, in those cases when the initial stiff transient is over quickly. Here ΔP_{∞} is computed from a lumped one room model. **This** lumped model has a single fire which is the sum **of** all the **fires** in the full model. It has a single vent to the outside which has an area equal to the total area of vents to the outside in the full model. The density and temperature of the vented **gas** is taken to be ambient. Bernoulli's law is used to model the vent. The solution for A P in this lumped model is derived in [1]. From this solution the duration of the initial stiff transient and ΔP_{∞} can be found.

MCCFM **allows** the user to name his output file interactively, but otherwise it runs in run in batch mode. The user specifies his problem in a data file. The following is a sample data file for a two room case. The units used here are time in seconds, pressure in Pascals, distance in meters, **mass** in kilograms, energy rates in watts, and temperatures in degrees Kelvin.

Input:

```
'RELATIVE ERROR TOLERANCES FOR P, Y, MU, WL'
1.D-6 1.D-6 1.D-6 1.D-6
'ABSOLUTE ERROR TOLERANCES FOR P, Y, MU, WL'
1.D-6 1.D-6 1.D-6 1.D-6
'NUMBER OF ROOMS'
2
'DISTANCE:FIRE TO FLOOR,CEILING TO FLOOR,DATUM TO FLOOR;FLOOR AREA,FIRE WATTAGE'
0. 3. 0. 6. 0.1D+7
0. 3. 0. 6. 0.D0
'NUMBER OF VENTS'
2
'AREA,DISTANCE:DATUM TO TOP,DATUM TO BOTTOM,ROOM NUMBERS ON EACH SIDE'
1.0 1.0 0.0 1 -1
1.0 1.0 0.0 1 2
'NUMBER OF FORCED VENTS'
0
'AREA,DISTANCE:DATUM TO TOP,DATUM TO BOTTOM,FLOW RATE,ROOM NUMBERS ON EACH SIDE'
'TFINAL, TPRINT'
300. 30.
'IF .TRUE. THE JACOBIAN AID ITS EIGENVALUES UILL BE PRINTED'
.FALSE.
'IF LOGPRT IS .TRUE. MASS AND ENERGY RATES THROUGH THE VENTS UILL BE PRINTED'
.FALSE.
'1 FOR DEBDF INTERMEDIATE HODE, ELSE 0'
1
```

MCCFM has the option to print the **mass** and energy rates through each vent and the eigenvalues **of** the Jacobian matrix computed by the **ODE** solver **DEBDF**. These features were helpful in an early stage **of** development for debugging the vent algorithm and for understanding the nature of the stiffness exhibited by the system

of ODE's.

Physical constants such as specific heats and the universal gas constant are stored in a block data subprogram. Before calling the **ODE** solver, the program unit **MAIN** in **MCCFM** reads the data file, and then sets solution variable tolerances, computes ΔP_∞ , computes solution variable normalizations, sets initial values for the solution variables, and writes out a summary of the configuration. In **MCCFM** the solution variables are nondimensionalized by dividing by the normalizations. The pressure offset, \mathbf{AP} , is normalized by ΔP_∞ , the upper layer height, \mathbf{Y} , is normalized by the height of the ceiling above the **floor**, Y_H , and the upper and lower layer masses, m_U and m_L , are normalized by the initial mass of air in the room. Next, **MAIN** calls the ODE solver once for each time at which output is requested.

The output corresponding to the sample data file above follows.

output:

```

SOLVER = DEBDF

G = 9.8 CP = 1000.0 R = 285.7 GAMMA = 1.4
PDATUM = 101325.0
AMBIENT DENSITY = 1.20000 AMBIENT TEMP = 295.53
LAMDAT = .35 LAMDAT = .60
TFINAL = 300.00 TPRINT = 30.00
PRESSURE TRANS TIME = 1.0399791766053D-03
IGNORING INITIAL PRESSURE TRANSIENT

ROOM      DELTA      YH      FH      AREA      QDTFIR
  1      0.00000      3.00000      0.00000      6.00000      0.10D+07
  2      0.00000      3.00000      0.00000      6.00000      0.00D+00

VENT      AVENT      HVTOP      HVBOT      FROM      TO
  1      1.00000      1.00000      0.00000      1      -1
  2      1.00000      1.00000      0.00000      1      2

FVENT      VDOT      HFVTOP      HFVBOT      FROM      TO

RELERR =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05
ABSERR =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05

TIME TO FIRST CALL OF DEBDF = 1.9999999552965D-02

ROOM  TIME      PRESSURE  LAYER  HASSU  HASSL  RHOU  RHOL  TEMPU  TEMPL
  1   30.00      0.2112E+00  0.64   3.73   4.64   0.2641  1.2000  1342.70  295.54
  2   30.00      0.9763E+00  1.01   4.61   7.28   0.3860  1.2000  918.84  295.54

FLAG IDID = -1
ROOM  TIME      PRESSURE  LAYER  HASSU  HASSL  RHOU  RHOL  TEMPU  TEMPL
  1   60.00      -0.7432E+00  0.59   2.75   3.66   0.1903  1.0261  1863.72  345.62

```

2	60.00	-0.4392E+00	0.51	4.96	3.49	0.3316	1.1452	1069.57	309.67
ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEHPU	TEHPL
1	90.00	-0.8216E+00	0.60	2.62	3.65	0.1819	1.0094	1949.20	351.34
2	90.00	-0.6581E+00	0.54	4.42	3.50	0.2998	1.0704	1182.98	331.30
ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEHPL
1	120.00	-0.7693E+00	0.60	2.61	3.68	0.1810	1.0255	1959.82	345.82
2	120.00	-0.7080E+00	0.56	4.00	3.54	0.2739	1.0479	1294.89	338.43
ROOH	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEMPU	TEHPL
1	150.00	-0.6903E+00	0.59	2.62	3.72	0.1810	1.0479	1959.29	338.41
2	150.00	-0.6991E+00	0.57	3.70	3.60	0.2542	1.0449	1395.31	339.39
ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEHPL
1	180.00	-0.6084E+00	0.59	2.63	3.76	0.1814	1.0706	1954.49	331.24
2	180.00	-0.6696E+00	0.58	3.48	3.64	0.2397	1.0492	1479.39	338.01
ROOH	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEHPU	TEHPL
1	210.00	-0.5367E+00	0.58	2.64	3.80	0.1819	1.0901	1949.22	325.33
2	210.00	-0.6214E+00	0.58	3.33	3.68	0.2291	1.0603	1547.69	334.46
ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEHPL
1	240.00	-0.4733E+00	0.58	2.65	3.83	0.1824	1.1069	1944.02	320.38
2	240.00	-0.5648E+00	0.58	3.22	3.72	0.2212	1.0747	1603.05	329.99
ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEHPL
1	270.00	-0.4173E+00	0.57	2.66	3.86	0.1829	1.1216	1939.00	316.20
2	270.00	-0.5078E+00	0.58	3.13	3.76	0.2152	1.0895	1648.00	325.49
ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEHPU	TEHPL
1	300.00	-0.3689E+00	0.57	2.67	3.88	0.1833	1.1343	1934.42	312.66
2	300.00	-0.4549E+00	0.57	3.07	3.80	0.2105	1.1036	1684.63	321.33

CALL	ITERS	TIME	ACCUM TIME
1	420	0.580	0.580
2	193	0.350	0.930
3	19	0.030	0.960
4	20	0.030	0.990
5	35	0.100	1.090
6	14	0.020	1.110
7	8	0.010	1.120
8	18	0.050	1.170
9	6	0.000	1.170
10	12	0.020	1.190

TIME TO FINISH = 1.2799999620765

The system of ODE's can be written in the standard form

$$\frac{dy}{dt} = F(t, y) \quad (7)$$

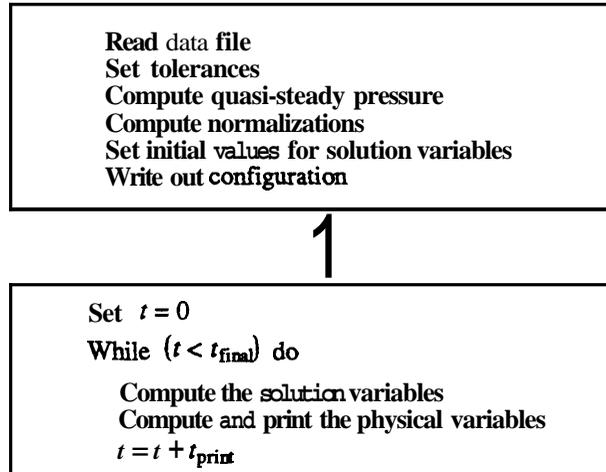


Figure 4: MCCFM Structure

where y and F are N -vector functions with N equal to four times the number of rooms. The components of y are the normalized solution variables for each room. The function F in equation (7) is defined in the subroutine F. Subroutine F begins with a loop which, for each room, computes values of all the variables using the current values of the normalized solution variables. Next, subroutine F executes a loop which, for each vent, computes the mass and energy transfer rates for the adjoining rooms. This is followed by a similar loop over the forced vents. Subroutine F ends with a loop which, for each room, computes the right hand sides of the four ODE's associated with the room. Inside the vent loop is a call to a subroutine VENT and inside the forced vent loop is a call to a subroutine FVENT. Coded in subroutines VENT and FVENT are the natural and forced vent submodels (see [2]). Figure 4 illustrates the structure of MCCFM.

3 CONRAD1: Adding Conduction and Radiation to MCCFM

In this section we examine the zone fire modeling code CONRAD1 which is MCCFM with the addition of two new submodels, one for energy transfer by heat conduction in ceilings, walls, and floors, and one for energy transfer by radiation. The codes CONRAD1 and CONRAD2 were developed as part of the second year of the grant. CONRAD2 is described in the next section. The goal for the second year of the grant was to incorporate into MCCFM the best available numerical methods for modeling heat conduction. Some of the technology used in CONRAD1 and CONRAD2 will likely be included in future updates of the NIST zone fire modeling code CFAST (see [7]).

CONRAD1 is a benchmark code which attempts to model heat conduction as accurately as possible. CONRAD2 trades off accuracy for speed; that is, it produces output that is nearly as accurate as that of CONRAD1, but its execution time is much faster. The goal of this project was to put the best available numerical methods into CONRAD1 and then to invent new methods for CONRAD2. CONRAD1 was constructed for comparison purposes; that is, it has been used to establish the accuracy of CONRAD2. CONRAD1 and CONRAD2 use the same radiation submodel.

CONRAD1 divides the interior surfaces of a room into four heat conduction nodes a ceiling node, an upper wall node, a lower wall node, and a floor node. The wall nodes join at the interface between the upper and lower layers. Consequently, the areas of the wall nodes are functions of time. Heat conduction through these nodes is taken to be one dimensional in the direction perpendicular to the node surfaces. Each node has an interior surface and an exterior surface. It is a simplifying assumption of CONRAD1 and CONRAD2 that the exterior node surfaces transfer heat to ambient. The temperature profile through a node is modeled using the one dimensional heat equation. Initially, the temperature profile in each node is taken to be constant at the ambient temperature. The boundary conditions specify the interior and exterior heat fluxes. These boundary conditions are of the form heat flux in equals heat flux out and are completely general. Any available model for heat flux at the interior and exterior node surfaces can be used. The approach used here has been to consider these fluxes as the sum of a convective term and a radiative term. In the early stages of the development of CONRAD1, radiative heat flux was modeled using a simple temperature to the fourth power model. This radiation model was later replaced by a four wall model due to Dr. Glenn Forney documented in [8]. The convection model used here is a simple flux proportion to temperature difference model. The proportionality constant is taken to be independent of temperature. Again, it should be emphasized that any available models for convection and radiation can be substituted here.

For each conduction node there is an initial-boundary value problem for the heat equation which is coupled with the the ODE's of MCCFM. For the ceiling conduction node, the equations are

$$\frac{\partial u}{\partial t} = -\frac{H}{C_p \rho} \frac{\partial^2 u}{\partial x^2}(x, t), \quad 0 < x < W, t > 0 \quad (8)$$

$$u(x, 0) = T_{\text{amb}}, \quad 0 < x < W \quad (9)$$

$$-H \frac{\partial u}{\partial x}(0, t) = H_i [T_U(t) - u(0, t)] + S_{\text{rad}} \quad (10)$$

$$-H \frac{\partial u}{\partial x}(W, t) = H_e [u(W, t) - T_{\text{amb}}] + \sigma \epsilon_e [u(W, t)^4 - T_{\text{amb}}^4], \quad (11)$$

where $u(x, t)$ denotes the temperature at a distance x into the ceiling at time t and T_{amb} denotes the ambient temperature. The term S_{rad} denotes the radiation flux

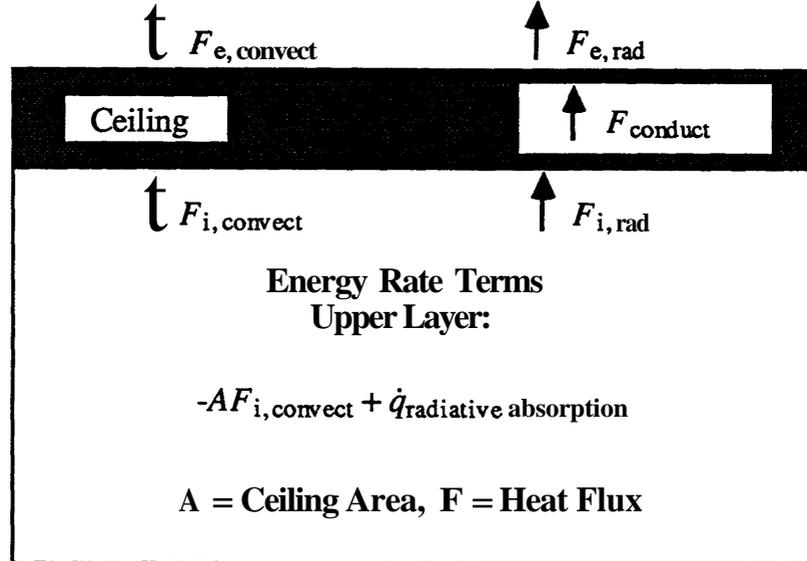


Figure 5: Ceiling and Upper Layer Heat Transfer

into the interior ceiling surface and T_U denotes the temperature of the upper layer. The remaining symbols in these formulas are physical constants: H , C , and ρ denote the thermal conductivity, the specific heat, and the density of the ceiling material; H_i and H_e denote the convective heat transfer coefficients for the interior and exterior ceiling surfaces; ϵ_e denotes the effective emittance of the exterior ceiling surface; and σ denotes the Stefan-Boltzmann constant. The four basic ODE's of MCCFM, equations (1)–(4), are coupled to the above initial-boundary value problem by the presence of the variable T_U in the convective term of the boundary condition (10) and through the radiation term S_{rad} in equation (10). In addition, the energy transfer rate \dot{q}_U now must have a term $-AH_i(T_U(t) - u(0, t))$ to account for the transfer of energy to the interior ceiling surface via convection and a term to account for the transfer of energy to the upper layer gas due to absorption of radiant energy. These heat transfer terms are illustrated in Figure 5.

For each room the four basic equations of MCCFM together with the equations for the four heat conduction nodes yields a system of four ODE's, each with an initial condition, plus four partial differential equations (PDE's), each with an initial condition and two boundary conditions. The method of lines (MOL) was chosen to convert the PDE's into ODE's. The result is an initial value problem for a differential-algebraic equation (DAE) system. If each heat conduction node generates K ODE's via the method of lines, then the total number of DAE's to be solved is $4 + 4(K + 2)$ times the number of rooms since each conduction node contributes K ODE's and 2 boundary conditions (algebraic equations).

For a parabolic equation such as the heat equation, the standard MOL approach

is to discretize the spatial variable either by replacing the spatial derivatives by finite difference approximations, or by expanding the unknown function as a linear combination of spatial basis functions with time dependent coefficients and deriving the ODE's via collocation. CONRAD1 uses the second approach and CONRAD2, the first. The MOL generally produces a stiff system. Although this method was proposed many years before, it was not until the advent of stiff ODE solvers in the 1970's that implementation of this method was practical. We quote the authors of [9] on the advantages of the MOL.

“There are two important advantages to the MOL approach. First, it is computationally efficient. The ODE software takes on the burden of time discretization and of choosing the time steps in a way that maintains accuracy and stability in the evolving solution. Most production ODE software is written to be robust and computationally efficient. Also, the person using a MOL approach has only to be concerned with discretizing spatial derivatives, thus reducing the work required to write a computer program.”

3.1 MOL, the Piecewise Cubic Hermite Collocation Approach

It is a well-known result in approximation theory that a function possessing four continuous derivatives on a closed, finite interval can be approximated to fourth order accuracy using piecewise cubic Hermite interpolation. This method of interpolation matches the function and its first derivative at a set of breakpoints. Between the breakpoints, the function is approximated by a cubic polynomial. The resulting approximation has a continuous first derivative (see [10]). CONRAD1 expands the temperature profile through a conduction node in the form (illustrated again with the ceiling node)

$$u(x, t) = \sum_{i=1}^{n_x} [a_i(t)\phi_i(x) + b_i(t)\psi_i(x)]$$

where ϕ_i and ψ_i denote the standard basis functions for piecewise cubic Hermite interpolation with breakpoints $0 = x_1 < \dots < x_{n_x} = W$. Next, $2n_x - 2$ ODE's for the $2n_x$ unknown coefficients, $a_i(t)$, $b_i(t)$, $i = 1, \dots, n_x$, can be derived by requiring that the heat equation be satisfied at the following two Gaussian points in each subinterval:

$$p_{2j-1} = x_j + \frac{3 - \sqrt{3}}{6}(x_{j+1} - x_j), \quad p_{2j} = x_{j+1} - \frac{3 - \sqrt{3}}{6}(x_{j+1} - x_j), \quad j = 1, \dots, n_x - 1.$$

The boundary conditions (10) and (11) provide two additional algebraic equations. When the boundary conditions can be easily differentiated with respect to time, they can be converted into ODE's. This discretization method leads to $2n_x$ differential

equations where n_b is the number of breakpoints. These equations have the form

$$A \frac{dy}{dt} = \frac{H}{C_p \rho} B y + F,$$

where A and B are $2n_b \times 2n_b$ matrices and A is nonsingular. For $n_b = 4$, the vectors y and F have the form

$$y = \begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \\ a_3 \\ b_3 \\ a_4 \\ b_4 \end{pmatrix}, \quad F = \begin{pmatrix} H_i T'_U + S'_{\text{rad}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

and the matrices A and B have the form

$$A = \begin{pmatrix} H_i & -H & 0 & 0 & 0 & 0 & 0 & 0 \\ \phi_1(p_1) & \psi_1(p_1) & \phi_2(p_1) & \psi_2(p_1) & 0 & 0 & 0 & 0 \\ \phi_1(p_2) & \psi_1(p_2) & \phi_2(p_2) & \psi_2(p_2) & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_2(p_3) & \psi_2(p_3) & \phi_3(p_3) & \psi_3(p_3) & 0 & 0 \\ 0 & 0 & \phi_2(p_4) & \psi_2(p_4) & \phi_3(p_4) & \psi_3(p_4) & 0 & 0 \\ 0 & 0 & 0 & 0 & \phi_3(p_5) & \psi_3(p_5) & \phi_4(p_5) & \psi_4(p_5) \\ 0 & 0 & 0 & 0 & \phi_3(p_6) & \psi_3(p_6) & \phi_4(p_6) & \psi_4(p_6) \\ 0 & 0 & 0 & 0 & 0 & 0 & H_e & H \end{pmatrix} \quad (12)$$

and

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \phi_1''(p_1) & \psi_1''(p_1) & \phi_2''(p_1) & \psi_2''(p_1) & 0 & 0 & 0 & 0 \\ \phi_1''(p_2) & \psi_1''(p_2) & \phi_2''(p_2) & \psi_2''(p_2) & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_2''(p_3) & \psi_2''(p_3) & \phi_3''(p_3) & \psi_3''(p_3) & 0 & 0 \\ 0 & 0 & \phi_2''(p_4) & \psi_2''(p_4) & \phi_3''(p_4) & \psi_3''(p_4) & 0 & 0 \\ 0 & 0 & 0 & 0 & \phi_3''(p_5) & \psi_3''(p_5) & \phi_4''(p_5) & \psi_4''(p_5) \\ 0 & 0 & 0 & 0 & \phi_3''(p_6) & \psi_3''(p_6) & \phi_4''(p_6) & \psi_4''(p_6) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (13)$$

In the earliest stage of CONRAD1 development, the radiation terms in equations (10) and (11) were either missing or the radiation term in (10) was similar to that in (11). Consequently, these equations could be differentiated with respect to time analytically. The required expression for T_U' was obtained by differentiating the ideal gas law and substituting equations (1)–(3). The ODE solver DDRIV2, due to Kahanner [10], was used to solve the full set of ODE's. The number of ODE's in the full system was $4 + 8n_x$ times the number of rooms in case each conduction node used n_x breakpoints. The solver DEBDF used in MCCFM was no longer applicable because DEBDF only handles systems of the form $y' = F(t, y)$, while this system has the form $A(t, y(t))y' = F(t, y)$. DDRIV2 handled this stiff system without difficulty.

When the radiation submodel of Dr. Glenn Forney [8] was incorporated, the analytic differentiation of the boundary conditions was no longer possible because the radiation term, S_{rad} , in boundary condition (10) is an implicitly defined, complex, nonlinear function of the variables \mathbf{P} , \mathbf{Y} , m_U , and m_L and the interior surface temperatures of the four conduction nodes. Now there was no choice but to leave the boundary conditions as algebraic equations. The resulting DAE system consists of $2n_x - 2$ ODE's and two algebraic equations for each conduction node. The full set of equations consists of $8n_x - 4$ ODE's plus eight algebraic equations per room in case all four conduction nodes use n_x breakpoints.

The DAE solver DASSL by Dr. Linda Petzold [9, 11] was chosen for use in CONRAD1 and CONRAD2. It is the most widely used production code for DAE's at this time. Like DEBDF, DASSL is based on backward differentiation methods. It also has a user interface similar to that of DEBDF which includes specification of a relative error tolerance for each solution component.

The following brief description of DASSL is taken from [9]. DASSL is a code for

solving systems of DAE's of the form

$$F(t, \mathbf{y}, \mathbf{y}') = 0 \quad (14)$$

$$\mathbf{y}(t_0) = \mathbf{y}_0 \quad (15)$$

$$\mathbf{y}'(t_0) = \mathbf{y}'_0, \quad (16)$$

where \mathbf{F} , \mathbf{y} , and \mathbf{y}' are N -dimensional vectors. The basic idea for solving DAE systems using numerical ODE methods, originating with Gear [12], is to replace the derivative in (14) by a difference approximation, and then to solve the resulting system for the solution at the current time t_{n+1} using Newton's method. For example, replacing the derivative in (14) by the first order backward difference, we obtain the implicit Euler formula

$$F\left(t_{n+1}, \mathbf{y}_{n+1}, \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{h_{n+1}}\right) = 0, \quad (17)$$

where $h_{n+1} = t_{n+1} - t_n$. This nonlinear system is then usually solved using some variant of Newton's method. The algorithms used in **DASSL** are an extension of this basic idea. Instead of always using the first order formula (17), **DASSL** approximates the derivative using the k th order backward differentiation formula, where k ranges from one to five. At every step it chooses the order k and the stepsize h_{n+1} , based on the behavior of the solution. **DASSL** can solve index zero and one systems. The index of the **DAE** system (14) is the minimum number of times that all or part of this system must be differentiated with respect to t in order to determine \mathbf{y}' as a continuous function of \mathbf{y} and t .

3.2 A Graded Spatial Mesh

The **MOL** chooses the time discretization to maintain accuracy and stability, but the user must choose the spatial discretization. For a uniform mesh with spacing h , piecewise cubic Hermite approximation of the temperature profile is possible with accuracy of order h^4 . During the first seconds of a fire simulation, conduction node temperature profiles typically have steep gradients near the interior node surfaces. Consequently, uniform meshes are not efficient. Again, quoting from [11]

“In many applications, such as combustion modeling, the use of a fixed spatial mesh leads either to poor accuracy or to a fine mesh and large computational effort. One way to circumvent this difficulty is to let the mesh points change with the time t .”

There was not sufficient time during the development of **CONRAD1** and **CONRAD2** to research in depth the topic of moving meshes. It is clear that this is a currently active topic of research in several quarters (see [13, 14, 15]). What was developed for **CONRAD1** is a graded mesh scheme with the grading dependent on the final simulation time t_{final} . This graded mesh is roughly optimized for the case when the

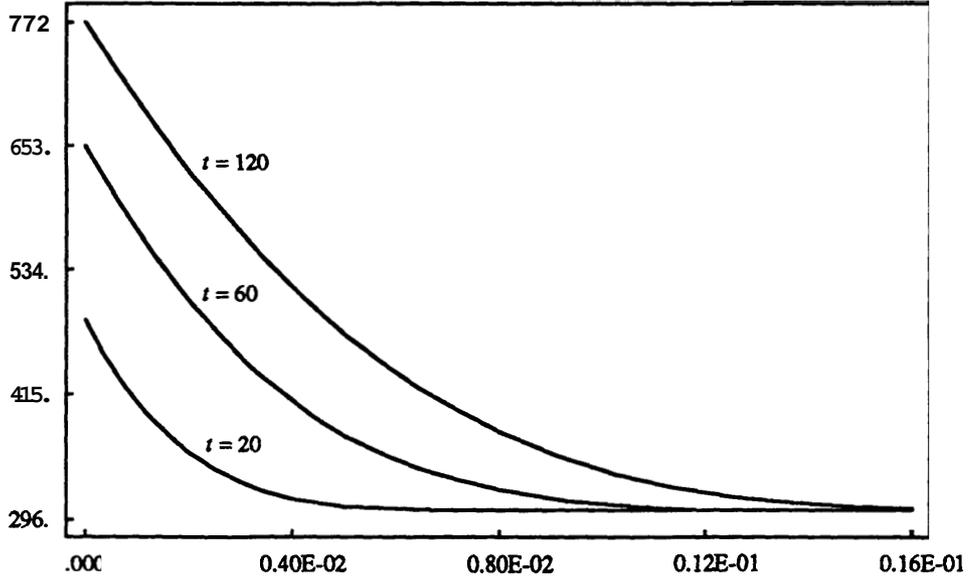


Figure 6: Ceiling Temperature Profiles

fire energy release rate takes a step jump at $t = 0$ and then is constant thereafter. In this case, the steepest temperature gradients occur near the interior surfaces of the conduction nodes. As the simulation evolves, these temperatures profiles tend to flatten out. Figure 6 shows the ceiling temperature profiles at various times during a one room simulation. The room is 3 m long, 2 m wide, 3 m high and contains a 1 Mw fire on the floor. It has a single 1 m² vent to the outside. All conduction nodes are made from gypsum and the radiation submodel of Dr. Glenn Forney is used. The simulation runs for 2 minutes.

The general qualitative features of these profiles are exhibited by the semi-infinite ($0 < x < \infty$) rod solution to the heat equation:

$$u(x, t) = T_{\text{amb}} + (u_1 - T_{\text{amb}})\text{erfc}\left(\frac{x}{2\alpha\sqrt{t}}\right), \quad (18)$$

in case the $u(0, t) = u_1$, a constant, and the initial temperature is T_{amb} . Here $\alpha = \sqrt{\frac{H}{c_v \rho}}$ and erfc denotes the complementary error function. Using this solution as a guide, the following graded mesh algorithm was developed to accommodate both short times and times near t_{final} using $n_x \geq 3$ breakpoints. It is impossible for a single, fixed mesh to be optimum for all times $0 < t < t_{\text{final}}$. If the warning message at the beginning of the following algorithm is activated, the user should adjust the value of t_{print} or t_{final} as indicated to get the best results.

Graded mesh algorithm:

```

If  $t_{\text{print}} < t_{\text{final}}/30$ , then
    write 'For best results  $t_{\text{print}} > t_{\text{final}}/30$ '
endif

 $x_1 = 0$ 
 $x_{n_x} = W$ 
 $x_b = \min[2\alpha\sqrt{t_{\text{final}}} \operatorname{erfc}(.05), \frac{W}{2}]$ 
 $x_p = \min[2\alpha\sqrt{t_{\text{print}}} \operatorname{erfc}(.05), \frac{x_b}{4}]$ 
If  $n_x = 3$ , then
     $x_2 = x_p$ 
else
     $x_2 = x_p$ 
     $x_{n_x-1} = x_b$ 
     $h = \frac{x_b - x_p}{n_x - 3}$ 
    For  $i = 3, \dots, n_x - 2$ 
         $x_i = x_p + (i - 2)h$ 
    endif

```

For a fixed time $t > 0$, the heat equation solution in (18) flattens out for $x > X(t) = 2\alpha\sqrt{t} \operatorname{erfc}(.05)$. To resolve the temperature profile at this time, a breakpoint should be placed near $X(t)$. Two of the breakpoints are generated this way. The breakpoint x_2 is the minimum of $X(t_{\text{print}})$ and $\frac{x_b}{4}$ with the x_b term required to provide short time accuracy when t_{print} is not sufficiently small. The breakpoint x_{n_x-1} is the minimum of $X(t_{\text{final}})$ and $\frac{W}{2}$. By the time $X(t)$ has reached the ceiling midpoint, $\frac{W}{2}$, the temperature profile for $x > \frac{W}{2}$ is linear enough so that breakpoints at $\frac{W}{2}$ and W suffice.

3.3 Code Structure

The program unit **MAIN** of **CONRAD1** begins by reading in a data file with the following structure.

Input:

```

'RELATIVE ERROR TOLERANCES FOR P, Y, MU, ML, NODETEMP'
1.D-6 1.D-6 1.D-6 1.D-6 1.D-2 1.D-2
'ABSOLUTE ERROR TOLERANCES FOR P, Y, MU, ML, NODETEMP'
1.D-6 1.D-6 1.D-6 1.D-6 0.D0 0.D0
'NUMBER OF ROOMS'
2
'FIRE WATTS, DIM: DATUM TO FLOOR, XROOM, YROOM, ZROOM, GXFIRE, GYFIRE, GZFIRE'
'FOLLOWED BY RADIATIO P, NODES, CODES(0=NULL, 1=CONCRETE, 2=GYP SUM, 3=KAOWOOL)'
0.1D+7 0. 3. 2. 3. 1. 1. 0.
.TRUE. 4 2 2 2 2
0.D0 0. 3. 2. 3. 1. 1. 0.
.TRUE. 4 2 2 2 2
'NUMBER OF VENTS'
2
'AREA, DIS: DATUM TO TOP, DATUM TO BOTTOM, ROOM NUMBERS ON EACH SIDE'
1.0 1.0 0.0 1 -1
1.0 1.0 0.0 1 2
'NUMBER OF FORCED VENTS'
0
'AREA, DIS: DATUM TO TOP, DATUM TO VENT, FLOW RATE, ROOM NUMBERS ON EACH SIDE'
'TFINAL, TPRINT'
300. 30.
'LOGICAL, TS, TF: DIAGHOSTICS AT INTERMEDIATE STEPS IN INTERVAL [TS, TF]'
.FALSE. 0.0 0.5
'NUMBER OF BREAKPOINTS NX'
5

```

As was the case with MCCFM, the user's primary means of communication with CONRAD1 is through this data file. The code does, however, interactively request that the user supply the name for the output file. The user must also supply information for each room in the data file. A logical is used to indicate whether the radiation submodel should be used in a room. An integer is used to indicate, the number of conduction nodes to be used in a room. Currently, only two values are accepted as input, 0 and 4. The value 0 indicates that the conduction submodel should not be used in the room. For each room the supported cases are four node conduction plus radiation; four node conduction without radiation; and no conduction, no radiation (defaults to MCCFM). This flexibility is provided so that a user can activate the conduction and radiation submodels in the room of fire origin and possibly in nearby rooms, yet turn them off in rooms that are not close to the room of fire origin.

The user must specify a material code for each conduction node. Currently, four values are accepted by the code. The value 0 indicates a null material with zero convective heat transfer coefficient and zero emissivity, while values of 1, 2, and 3 indicate the materials concrete, gypsum, and kaowool, respectively. Although CONRAD1 includes DAE equations for a node consisting of null material, no heat is transferred to or from such a node. By using null conduction materials, conduction nodes can be inactivated selectively.

A logical followed by the limits of a time interval are provided to turn on di-

agnostic output during the indicated time interval. **This** is provided primarily for debugging purposes and was used in the early stages of development. The same number of breakpoints is used for **all** heat conduction nodes because early numerical experiments indicated no special advantage to having greater flexibility.

After reading the data file, **MAIN** processes the description of the conduction nodes and sets up arrays containing the appropriate physical constants for each material. Next, for each different type of material, the breakpoints and collocation points are generated and the matrices **A** and **B** in (12) and (13) are computed. These matrices **will** later be used to construct the full set of differential equations for each room. **MAIN** then continues by computing tolerances, normalizations, and initial values for the solution variables in each room. After printing out the configuration, **MAIN** produces output every t_{print} seconds by placing calls to the **DAE** solver **DASSL** in a loop. The output corresponding to the above data file follows.

output:

```

NRMNODE = 0 -- NULL
NRMNODE = 1 -- CONCRETE
NRMNODE = 2 -- GYPSUM
NRMNODE = 3 -- KAOUOOL

NODE = 1 -- CEILING
NODE = 2 -- UWALL
NODE = 3 -- LUALL
NODE = 4 -- FLOOR

MATERIALS USED
MT BK      H   CCP  RHOU THUALL  EPO  EPI      CHT  HCI  HCO
  2  5  0.16  900.  800.  0.0160  0.90  0.90  0.222D-06  30.0  5.0

ABSORPTION COEFFICIENT FOR UPPER LAYER =      1.0000000000000D-01
ABSORPTION COEFFICIENT FOR LOUER LAYER =      1.0000000000000D-02

NEQN =      88

RTOL =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-01
ATOL =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05  0.0000D+00

SOLVER = DASSL

G = 9.8 CP = 1000.0 R = 285.7 GAMMA = 1.4
PDATUM = 101325.0
AMBIENT DENSITY = 1.20000 AMBIENT TEMP = 295.53
LAMDAR = .35 LAMDAT = .60
TFINAL =      300.00 TPRINT =      30.00

```

PRESSURE TRAMS TIME = 1.6899661619836D-03

IGNORIBG INITIAL PRESSURE TRANSIENT

ROOM	FH	XROOH	YROOM	ZROOM	GXFIRE	GYFIRE	GZFIRE	QDTFIR
1	0.000	3.000	2.000	3.000	1.000	1.000	0.000	0.10D+07
2	0.000	3.000	2.000	3.000	1.000	1.000	0.000	0.00D+00

ROOH	RADIATION	BODES	HATERIALS
1	T	4	2 2 2 2
2	T	4	2 2 2 2

VENT	AVENT	HVTOP	HVBOT	FROM	TO
1	1.00000	1.00000	0.00000	1	-1
2	1.00000	1.00000	0.00000	1	2

FICTIOUS LAYER THICKNESS, NORHALIZED = 1.0000000000000D-05

TIME TO FIRST CALL OF DASSL = 5.9999998658895D-02

ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEMPU	TEHPL
1	30.00	-0.3566E+00	0.72	5.69	4.80	0.4155	1.1119	853.55	318.93
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	525.77	531.03	344.74	366.81					
	295.64	295.64	295.55	295.56					

ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEMPU	TEHPL
2	30.00	0.1465E+00	1.85	5.02	13.29	0.7283	1.1961	486.97	296.50
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	364.84	366.27	297.45	297.50					
	295.57	295.57	295.53	295.53					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEHPU	TEHPL
1	60.00	-0.6820E+00	0.73	5.28	4.55	0.3876	1.0391	915.06	341.28
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	626.55	632.81	388.97	422.72					
	296.09	296.10	295.65	295.71					

ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEMPU	TEHPL
2	60.00	-0.3580E-01	0.74	10.47	5.21	0.7723	1.1719	459.17	302.61
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	377.06	377.89	301.39	302.58					
	295.69	295.69	295.54	295.54					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEMPU	TEHPL
1	90.00	-0.1146E+01	0.72	5.03	4.05	0.3669	0.9407	966.47	377.00
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	699.15	705.45	440.18	483.63					
	298.54	298.60	296.18	296.50					

ROOH	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOH	RHOL	TEHPU	TEHPL
2	90.00	-0.7623E+00	0.31	13.53	2.08	0.8376	1.1277	423.39	314.46
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	373.10	373.50	307.11	309.50					
	296.41	296.43	295.56	295.56					

ROOM	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEMPU	TEMPL
1	120.00	-0.1396E+01	0.74	4.79	3.90	0.3535	0.8761	1003.18	404.77
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	753.67	759.81	489.90	538.72					
	305.01	305.21	297.79	298.77					
ROOM	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEMPU	TEMPL
2	120.00	-0.1151E+01	0.12	14.63	0.82	0.8468	1.1262	418.77	314.88
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	372.63	372.85	310.29	313.44					
	298.18	298.23	295.64	295.66					
ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEWPU	TEMPL
1	150.00	-0.1589E+01	0.76	4.61	3.78	0.3434	0.8282	1032.69	428.19
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	797.11	803.04	536.37	587.53					
	315.40	315.79	300.90	302.95					
ROOM	TIME	PRESSURE	LAYER	MASSU	MASSL	RHOU	RHOL	TEWPU	TEMPL
2	150.00	-0.1426E+01	0.03	14.92	0.23	0.8383	1.1227	423.02	315.86
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	375.68	375.81	312.37	316.21					
	300.67	300.75	295.85	295.90					
ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
1	180.00	-0.1736E+01	0.77	4.48	3.67	0.3354	0.7920	1057.34	447.77
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	833.27	838.97	579.57	631.19					
	328.71	329.30	305.71	309.17					
ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
2	180.00	-0.1527E+01	0.01	14.88	0.04	0.8282	1.1097	428.18	319.58
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	379.85	379.93	315.31	319.70					
	303.42	303.52	296.20	296.33					
ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEWPU	TEMPL
1	210.00	-0.1851E+01	0.78	4.38	3.58	0.3287	0.7631	1078.79	464.70
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	864.38	869.86	619.39	670.27					
	343.82	344.61	312.20	317.27					
ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
2	210.00	-0.1574E+01	0.00	14.71	0.00	0.8175	1.0980	433.81	322.98
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	384.47	384.52	318.32	323.16					
	306.17	306.28	296.69	296.92					
ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
1	240.00	-0.1946E+01	0.79	4.29	3.50	0.3230	0.7389	1098.00	479.94
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	891.90	897.17	656.16	705.61					
	359.76	360.72	320.20	326.89					
ROOM	TIME	PRESSURE	LAYER	MASSU	MASSL	RHOU	RHOL	TEMPU	TEMPL
2	240.00	-0.1610E+01	0.00	14.52	0.00	0.8069	1.1095	439.51	319.64
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									

389.29	389.31	321.37	326.63
308.86	308.97	297.28	297.64

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEWPL
1	270.00	-0.2028E+01	0.79	4.21	3.42	0.3179	0.7179	1115.57	493.97
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	916.74	921.83	690.22	737.92					
	375.80	376.89	329.43	337.66					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEWPL
2	270.00	-0.1644E+01	0.00	14.34	0.00	0.7967	1.0741	445.12	330.17
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	394.17	394.18	324.53	330.19					
	311.47	311.58	297.95	298.46					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEWPL
1	300.00	-0.2099E+01	0.80	4.14	3.35	0.3133	0.6994	1131.90	507.05
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	939.57	944.49	721.95	767.75					
	391.43	392.62	339.59	349.18					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEWPL
2	300.00	-0.1677E+01	0.00	14.17	0.00	0.7870	1.0626	450.61	333.74
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	399.05	399.04	327.70	333.77					
	314.02	314.13	298.71	299.38					

CALL	ITERS	TIME	ACCUM TIME
1	413	41.970	41.970
2	27	3.890	45.860
3	49	4.340	50.200
4	39	1.240	51.440
5	27	2.510	53.950
6	24	1.090	55.040
7	19	1.030	56.070
8	17	1.570	57.640
9	35	8.540	66.180
10	7	0.400	66.580

TIME TO FINISH = 66.740000914782

Subroutine F in CONRAD1 defines the DAE system in the **form** of equation (14). Each equation in the system is in residual form; that is, its right hand side is zero. The organization of this subroutine is the same as that for **MCCFM**: a loop over the rooms to compute current values for **all** variables in terms of the current values for the solution variables, a loop over the vents to accumulate vent **mass** and energy flow rates for each **room**, a loop over the forced vents to accumulate forced vent **mass** and energy **flow** rates for each **room**, and a loop over the **rooms** to define the **DAE's** for each **room**. Inside this last loop is a **call** to subroutine RAD4 which contains the radiation submodel. If conduction is turned on in a **room**, then there are $4 + 8n_x$ unknowns for this room which are the normalized offset pressure, the

normalized layer interface height, the normalized upper layer mass, the normalized lower layer mass, and for each of the four conduction nodes the values of temperature and temperature gradient at the n_x breakpoints. The corresponding DAE equation set consists of the following $4 + 8n_x$ equations:

- an ODE for normalized offset pressure (MCCFM),
- an ODE for normalized layer interface height (MCCFM),
- an ODE for normalized upper layer **mass** (MCCFM),
- an ODE for normalized lower layer mass (MCCFM),
- an algebraic boundary condition at the interior ceiling node surface,
- $2n_x - 2$ ODE's generated by collocation,
- an algebraic boundary condition at the exterior ceiling node surface,
- an algebraic boundary condition at the interior upper wall node surface,
- $2n_x - 2$ ODE's generated by collocation,
- an algebraic boundary condition at the exterior upper wall node surface,
- an algebraic boundary condition at the interior lower wall node surface,
- $2n_x - 2$ ODE's generated by collocation,
- an algebraic boundary condition at the exterior lower **wall** node surface,
- an algebraic boundary condition at the interior floor node surface,
- $2n_x - 2$ ODE's generated by collocation,
- an algebraic boundary condition at the exterior floor node surface.

A similar set of equations is generated for each room in which conduction is turned on. For rooms in which conduction is turned **off**, only the first four ODE's in this list apply. If conduction is turned on in **all** rooms, the total number of equations, n_{eqn} , in the DAE system is $4 + 8n_x$ times the number of rooms. The implicit method employed by the DAE solver DASSL will solve at each time step a nonlinear system of n_{eqn} equations in n_{eqn} unknowns using a variant of Newton's methods. The dominant cost **for** DASSL is the cost of forming a Jacobian matrix and solving a linear system having this Jacobian as its coefficient matrix. The cost of solving the linear system is proportional to n_{eqn}^3 . Consequently, the execution time of CONRAD1 is heavily influence by the number of rooms that have conduction turned on. Here is where an efficient use of breakpoints pays **off**. Using a large number of equispaced breakpoints per node, for example, would make the execution time of CONRAD1 extremely long.

4 CONRAD2: Functional Equations for Conduction

As mentioned in the previous section, if conduction is turned on in all rooms the total number of equations, n_{eqn} , in the DAE system of CONRAD1 is $4 + 8n_x$ times the number of rooms. CONRAD2 reduces this number to 12 times the number of rooms without significant loss of accuracy. This statement is based on experiments with three materials: concrete, gypsum, and kaowool. The reduction in execution time achieved by CONRAD2 is significant and, as explained in the previous section, is due to the fact that the dominant cost in the computation is proportional to n_{eqn}^3 .

The development of a code like CONRAD2 was the goal of the second year of this NIST grant; however, it was not clear at the outset that a code with the features of CONRAD2 was theoretically possible. Success in creating CONRAD2 can be attributed to several factors: methodical numerical experimentation; intuition based on the author's theoretical PDE training; timely discussions with Dr. Glenn Forney, the scientific officer for this grant; and luck. There is nothing in the original proposal that suggests the final form of CONRAD2; it is totally unlike anything that was proposed.

Roughly, the idea is to take the initial-boundary value problem for the heat equation and reduce it to pair of functional equations. To explain further, consider the following standard problem. Find the temperature profile $u(x, t)$ that satisfies

$$\frac{\partial u}{\partial t} = \frac{A}{C_p \rho} \frac{\partial^2 u}{\partial x^2}(x, t), \quad 0 < x < W, \quad t > 0 \quad (19)$$

$$u(x, t_0) = g(x), \quad 0 \leq x \leq W \quad (20)$$

$$u(0, t) = f_0(t), \quad t \geq 0 \quad (21)$$

$$u(W, t) = f_1(t), \quad t \geq 0, \quad (22)$$

where g , f_0 , and f_1 are given continuous functions. The existence and uniqueness theory (see [16]) for this problem shows that $u(x, t)$ is uniquely determined by initial temperature profile, g , and the time histories $f_0(\tau)$ and $f_1(\tau)$ for $0 \leq \tau \leq t$; that is, there is a functional relation of the form

$$u(x, t) = G[g, f_0(0 \leq \tau \leq t), f_1(0 \leq \tau \leq t)](x, t). \quad (23)$$

Also, if $0 < t_0 < t$, then

$$u(x, t) = G[u(\cdot, t_0), f_0(t_0 \leq \tau \leq t), f_1(t_0 \leq \tau \leq t)](x, t). \quad (24)$$

In [16] the Laplace transform is used to construct the functional G. In the special case when the initial temperature is constant, say $u(x, 0) = T_{\text{amb}}$, the functional G can be written in terms of convolution integrals; with the spatial variable transformed so that $W = 1$, it follows that

$$u(x, t) = T_{\text{amb}} - \int_0^t f_0(t - \tau) \frac{\partial}{\partial x} \theta_3\left(\frac{x}{2}, \tau\right) d\tau - \int_0^t f_1(t - \tau) \frac{\partial}{\partial x} \theta_3\left(\frac{1-x}{2}, \tau\right) d\tau,$$

where $\theta_3(x, t)$ is a theta function which can be expressed as

$$\theta_3(x, t) = 1 + 2 \sum_{k=1}^{\infty} e^{-\pi^2 k^2 t} \cos 2\pi k x .$$

From equation (23) it follows that

$$\begin{aligned} \frac{\partial u}{\partial x}(0, t) &= \frac{\partial}{\partial x} G[g, f_0(0 \leq \tau \leq t), f_1(0 \leq \tau \leq t)](0, t) \\ \frac{\partial u}{\partial x}(W, t) &= \frac{\partial}{\partial x} G[g, f_0(0 \leq \tau \leq t), f_1(0 \leq \tau \leq t)](W, t) . \end{aligned}$$

For each heat conduction node in a room, two functional equations are added to the system of **CONRAD2**. For the ceiling, these equations have the form

$$\begin{aligned} H \frac{\partial}{\partial x} G[T_{\text{amb}}, u(0, 0 \leq \tau \leq t), u(W, 0 \leq \tau \leq t)](0, t) \\ + H_i(T_U(t) - u(0, t)) + S_{\text{rad}} = 0 \end{aligned} \quad (25)$$

$$\begin{aligned} H \frac{\partial}{\partial x} G[T_{\text{amb}}, u(0, 0 \leq \tau \leq t), u(W, 0 \leq \tau \leq t)](W, t) \\ + \sigma \epsilon_e((u(W, t))^4 - T_{\text{amb}}^4) = 0 . \end{aligned} \quad (26)$$

The corresponding unknowns are $u(0, t)$ and $u(W, t)$.

The set of unknowns for **CONRAD2** is then constructed as follows. For each room in which conduction is turned on there are 12 unknowns: the normalized offset pressure, the normalized layer interface height, the normalized upper layer mass, the normalized lower layer mass, and for each of the four conduction nodes the temperatures at the interior and exterior surfaces. The corresponding differential-functional equation (DFE) system for each room consists of the following 12 equations.

- an ODE for normalized offset pressure,
- an ODE for normalized layer interface height,
- an ODE for normalized upper layer mass,
- an ODE for normalized lower layer mass,
- a functional equation for the interior ceiling node surface,
- a functional equation for the exterior ceiling node surface,
- a functional equation for the interior upper wall node surface,
- a functional equation for the exterior upper wall node surface,
- a functional equation for the interior lower wall node surface,

- a functional equation for the exterior lower **wall** node surface,
- a functional equation for the interior **floor** node surface,
- a functional equation for the exterior **floor** node surface.

4.1 Solving the Heat Equation

We now present the methods by which the **DFE** system of **CONRAD2** is solved. We have again used the DAE solver **DASSL** to find an approximate solution to this **DFE** system. Use of a **DAE** solver in this way appears to be new.

The main detail of the implementation that needs to be explained is how the first terms in equations (25) and (26) are computed. The computation is performed in subroutine **CNDUCT** which is discussed below. Suppose that the **DFE** system has been integrated to time t and that the ceiling temperature profile, $u(x, t)$, $0 \leq x \leq W$, has been saved. Suppose that for $\Delta t > 0$ values for $u(0, t + \Delta t)$ and $u(W, t + \Delta t)$ are available. To compute the first terms in equations (25) and (26) at time $t + \Delta t$, the temperature profile must be advanced to time $t + \Delta t$ and its endpoint gradients approximated. The theoretical basis for this approach is the property of the heat equation given in equation (24).

The simplest possible methods were used to implement this part of **CONRAD2** because development time was at a premium at this point in the project, and it was felt that most of the available time should be spent in testing the **DFE** solution method. As in **CONRAD1**, we chose to use the **MOL** to solve the heat equation, but for **CONRAD2** the second spatial derivative was approximated at each of the interior meshpoints by a centered difference formula. Again, a graded mesh with n_x breakpoints was used. This implementation of the **MOL** produces a system of $n_x - 2$ **ODE's** for the $n_x - 2$ unknown temperatures at the interior breakpoints. This system of **ODE's** was solved by the backward Euler method. Generally, this procedure leads to an nonlinear equation implicitly relating the solution at time t to the solution at time $t + \Delta t$, but because the heat equation is linear, the equation relating the solution at time $t + \Delta t$ and the solution at time t is linear. In fact, the solution at time $t + \Delta t$ can be found by solving a tridiagonal system. The temperature gradient at $x = 0$ and time $t + \Delta t$ can be approximated by interpolating the temperature values at $x_1 = 0$, x_2 , and x_3 by a quadratic and then evaluating the derivative of the quadratic at $x = 0$. A similar procedure can be used to approximate the temperature gradient at $x = W$ and time $t + \Delta t$. The only parameter left to choose here is the time stepsize Δt .

Once the **DAE** solver **DASSL** has integrated forward in time to the time t and has chosen the stepsize k for its next time step, it predicts values for the solution variables at time $t + k$ and then homes in on the true values of these solution variables by generating successive Newton iterates; that is, **DASSL** advances the solution to time $t + k$ by solving a nonlinear system of equations. After some experimentation,

where for $i = 1, \dots, n_x - 1$

$$\begin{aligned} a_i &= 1 + \frac{s}{h_{i-1}h_i} \\ b_i &= \frac{-s}{h_{i-1}(h_{i-1} + h_i)} \\ c_i &= \frac{-s}{h_i(h_{i-1} + h_i)}. \end{aligned}$$

The gradients at time $t + k$ at the points $x = 0$ and $x = W$ are estimated from quadratics passing through the first three and last three temperature values. Using standard divided difference notation, first terms in equations (25) and (26) at time $t + k$ are computed respectively by

$$\begin{aligned} H \frac{u[x_1, x_2] - u[x_1, x_2, x_3]h_1}{W} \\ H \frac{u[x_{n_x-1}, x_{n_x}] - u[x_{n_x-2}, x_{n_x-1}, x_{n_x}]h_{n_x-1}}{W} \end{aligned}$$

4.2 Code Structure

The organization of CONRAD2 is only a slight variation of that of CONRAD1. In this section we will simply point out the differences.

In program unit MAIN the breakpoints for each type of material are computed and stored for later use. The DFE system in residual form is again defined in subroutine F. The final loop of subroutine F computes the DFE's for each room. Just before this final loop, the tridiagonal heat conduction matrices for each material are computed and factored. These matrices are dependent on the current time stepsize k mentioned in the previous section and thus cannot be computed outside of subroutine F. It should be noted that it is only necessary to compute the breakpoints and heat conduction matrices for each different material and not for each heat conduction node. Inside the final loop of subroutine F, the gradients discussed in the previous section, are computed for each of the conduction nodes in a room. This computation requires the solution of a tridiagonal system of linear equations for each conduction node. Because factorizations were computed before the final loop, all that is required at this point is forward substitution, backward substitution, and gradient estimation using divided differences. Figure 7 shows the details of subroutine F in CONRAD2 in a block format.

In their default configurations each conduction node in CONRAD1 contributes 10 equations to its DAE system, while in CONRAD2 each conduction node only contributes 2 functional equations. Consequently, the execution time for CONRAD2 is much less than that for CONRAD1 especially in multiple room simulations. On the other hand, CONRAD1 must store 5 temperatures and 5 temperature gradients for each conduction node, while CONRAD2 must store 20 temperatures, so the

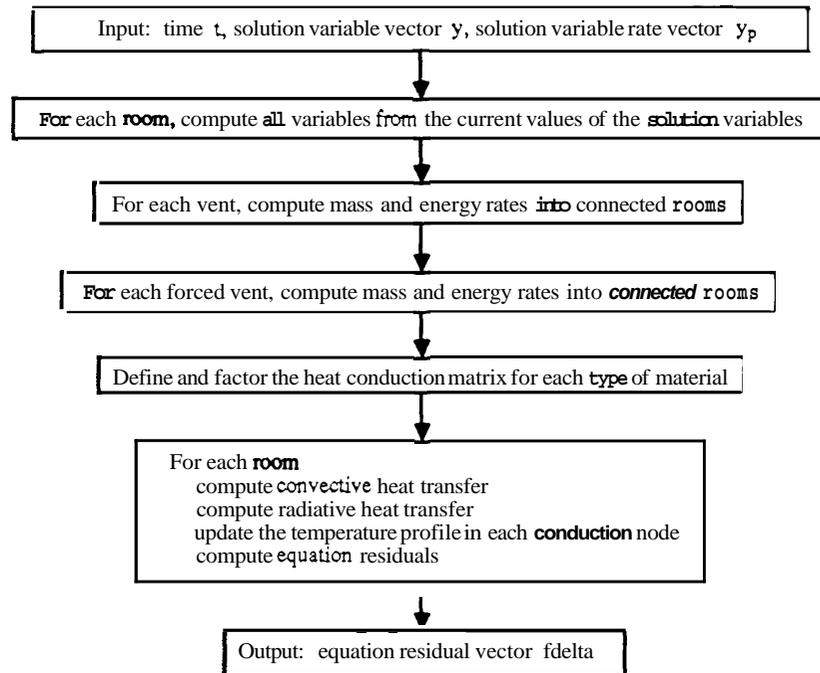


Figure 7: Subroutine F Details

memory requirements for **CONRAD2** are greater. In Section 6, we discuss methods for reducing this memory requirement.

A sample two room input data file and the corresponding output follow.

Input:

```

'RELATIVE ERROR TOLERANCES FOR P, Y, MU, ML, NODETEMP'
1.D-6 1.D-6 1.D-6 1.D-6 1.D-2
'ABSOLUTE ERROR TOLERANCES FOR P, Y, MU, ML, NODETEMP'
1.D-6 1.D-6 1.D-6 1.D-6 0.D0
'NUMBER OF ROOMS'
2
'FIRE WATTS, DIM: DATUM TO FLOOR, XROOM, YROOM, ZROOM, GXFIRE, GYFIRE, GZFIRE'
'FOLLOWED BY RADIATION, NODES, CODES (0=NULL, 1=CONCRETE, 2=GYPSUM, 3=KAOWOOL)'
0.1D+7 0. 3. 2. 3. 1. 1. 0.
.TRUE. 4 2 2 2 2
0.D0 0. 3. 2. 3. 1. 1. 0.
.TRUE. 4 2 2 2 2
'NUMBER OF VENTS'
2
'AREA, DIS: DATUM TO TOP, DATUM TO BOTTOM, ROOM NUMBERS ON EACH SIDE'
1.0 1.0 0.0 1 -1
1.0 1.0 0.0 1 2
'NUMBER OF FORCED VENTS'

```

```

0
'AREA, DIS: DATUM TO TOP, DATUM TO VENT, FLOW RATE, ROOM NUMBERS ON EACH SIDE'
'TFINAL, TPRINT'
300. 30.
'LOGICAL, TS, TF: DIAGNOSTICS AT INTERMEDIATE STEPS IN INTERVAL [TS, TF]'
.FALSE. 0.0 0.5
'NUMBER OF BREAKPOINTS NX'
20

```

output:

```

NRMNODE = 0 -- NULL
NRMNODE = 1 -- CONCRETE
NRMNODE = 2 -- GYPSUM
NRMNODE = 3 -- KAOYUOL

```

```

NODE = 1 -- CEILING
NODE = 2 -- UWALL
NODE = 3 -- LYALL
NODE = 4 -- FLOOR

```

MATERIALS USED

```

MT BK   H   CCP  RHOU THUALL  EPO  EPI      CHT  HCI  HCO
  2  20  0.16  900.  800. 0.0160  0.90  0.90  0.222D-06  30.0  5.0

```

```

ABSORPTION COEFFICIENT FOR UPPER LAYER = 1.0000000000000D-01
ABSORPTION COEFFICIENT FOR LOUER LAYER = 1.0000000000000D-02

```

NEQN = 24

```

RTOL =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-01
ATOL =
  0.1000D-05  0.1000D-05  0.1000D-05  0.1000D-05  0.0000D+00

```

SOLVER = DASSL

```

G = 9.8 CP = 1000.0 R = 285.7 GAMMA = 1.4
PDATUM = 101325.0
AMBIENT DENSITY = 1.20000 AMBIENT TEMP = 295.53
LAMDAT = .35 LAMDAT = .60
TFINAL = 300.00 TPRINT = 30.00
PRESSURE TRANS TIME = 1.6899661619836D-03
IGNORING INITIAL PRESSURE TRANSIENT

```

ROOM	FH	XROOM	YROOM	ZROOM	GXFIRE	GYFIRE	GZFIRE	QDTFIR
1	0.000	3.000	2.000	3.000	1.000	1.000	0.000	0.10D+07
2	0.000	3.000	2.000	3.000	1.000	1.000	0.000	0.00D+00

ROOM	RADIATION	NODES	HATERIALS
1	T	4	2 2 2 2

2	T	4	2	2	2	2
VENT	AVENT	HVTOP	HVBOT	FROM	TO	
1	1.00000	1.00000	0.00000	1	-1	
2	1.00000	1.00000	0.00000	1	2	

FICTITIOUS LAYER THICKNESS, NORMALIZED = 1.0000000000000D-05

TIME TO FIRST CALL OF DASSL = 3.0000001192093D-02

ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEWPL
1	30.00	-0.3552E+00	0.72	5.69	4.80	0.4157	1.1123	853.15	318.82
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	525.09	530.34	344.50	366.49					
	295.54	295.54	295.53	295.53					

ROOM	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEHPU	TEMPL
2	30.00	0.1462E+00	1.85	5.02	13.28	0.7288	1.1961	486.62	296.49
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	364.52	365.94	297.44	297.48					
	295.53	295.53	295.53	295.53					

ROOM	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEMPU	TEMPL
1	60.00	-0.6800E+00	0.73	5.28	4.55	0.3877	1.0397	914.61	341.10
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	625.77	632.03	388.57	422.22					
	296.02	296.03	295.62	295.68					

ROOM	TIME	PRESSURE	LAYER	HASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
2	60.00	-0.3521E-01	0.74	10.47	5.22	0.7728	1.1721	458.91	302.57
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	376.75	377.59	301.35	302.53					
	295.68	295.68	295.53	295.53					

ROOM	TIME	PRESSURE	LAYER	HASSU	MASSL	RHOU	RHOL	TEMPU	TEMPL
1	90.00	-0.1143E+01	0.72	5.03	4.05	0.3672	0.9415	965.83	376.67
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	698.07	704.37	439.46	482.77					
	298.96	299.03	296.24	296.60					

ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
2	90.00	-0.7599E+00	0.31	13.53	2.09	0.8381	1.1281	423.13	314.37
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	372.77	373.16	307.03	309.41					
	296.55	296.57	295.56	295.56					

ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
1	120.00	-0.1392E+01	0.74	4.80	3.90	0.3539	0.8774	1002.20	404.18
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	752.09	758.24	488.68	537.30					
	305.84	306.05	297.99	299.05					

ROOM	TIME	PRESSURE	LAYER	MASSU	HASSL	RHOU	RHOL	TEMPU	TEMPL
2	120.00	-0.1146E+01	0.12	14.63	0.83	0.8476	1.1266	418.42	314.79
INTERIOR NODE TEMPERATURES, THEN EXTERIOR									
	372.23	372.45	310.18	313.30					

298.39 298.44 295.65 295.67

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU MOL TEMPU TEMPL
1 150.00 -0.1583E+01 0.76 4.62 3.79 0.3438 0.8298 1031.51 427.39
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
795.28 801.23 534.65 585.67
316.39 316.79 301.24 303.40

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEMPL
2 150.00 -0.1422E+01 0.03 14.93 0.23 0.8392 1.1234 422.59 315.67
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
375.21 375.34 312.21 316.02
300.84 300.92 295.88 295.93

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEMPL
1 180.00 -0.1730E+01 0.77 4.49 3.68 0.3358 0.7936 1056.08 446.85
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
831.38 837.09 577.49 629.04
329.71 330.31 306.14 309.71

ROOM TIME PRESSURE LAYER MASSU HASSL RHOU RHOL TEMPU TEMPL
2 180.00 -0.1523E+01 0.01 14.89 0.04 0.8291 1.1103 427.72 319.39
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
379.39 379.47 315.11 319.45
303.54 303.63 296.24 296.37

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEWPL
1 210.00 -0.1844E+01 0.78 4.38 3.58 0.3291 0.7648 1077.47 463.70
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
862.46 867.95 617.03 667.92
344.81 345.60 312.72 317.88

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEMPL
2 210.00 -0.1571E+01 0.00 14.73 0.00 0.8184 1.0983 433.34 322.88
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
384.03 384.08 318.09 322.89
306.26 306.36 296.73 296.97

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEHPL
1 240.00 -0.1940E+01 0.79 4.29 3.50 0.3234 0.7405 1096.68 478.90
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
890.00 895.29 653.65 703.19
360.72 361.69 320.77 327.55

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEHPU TEMPL
2 240.00 -0.1607E+01 0.00 14.54 0.00 0.8077 1.0976 439.05 323.10
IBTERIOR NODE TEMPERATURES, THEN EXTERIOR
388.86 388.89 321.12 326.34
308.93 309.04 297.31 297.68

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEMPU TEMPL
1 270.00 -0.2022E+01 0.79 4.21 3.43 0.3182 0.7194 1114.32 492.95
IITERIOR NODE TEMPERATURES, THEN EXTERIOR
914.99 920.09 687.76 735.59
376.76 377.86 330.00 338.30

ROOM TIME PRESSURE LAYER HASSU HASSL RHOU RHOL TEWPU TEHPL

```

2 270.00 -0.1642E+01 0.00 14.35 0.00 0.7975 1.0753 444.68 329.80
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
393.76 393.77 324.21 329.84
311.54 311.65 297.98 298.50

```

```

ROOM TIME PRESSURE LAYER MASSU MASSL RHOU RHOL TEHPU TEMPL
1 300.00 -0.2094E+01 0.80 4.14 3.36 0.3136 0.7008 1130.72 506.07
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
937.94 942.87 719.57 765.54
392.44 393.65 340.15 349.82

```

```

ROOM TIME PRESSURE LAYER HASSU MASSL RHOU RHOL TEHPU TEHPL
2 300.00 -0.1675E+01 0.00 14.18 0.00 0.7878 1.0637 450.17 333.39
INTERIOR NODE TEMPERATURES, THEN EXTERIOR
398.65 398.64 327.39 333.42
314.10 314.20 298.74 299.41

```

CALL	ITERS	TIME	ACCUH TIME
1	611	13.010	13.010
2	79	1.460	14.470
3	168	1.650	16.120
4	28	0.430	16.550
5	38	0.680	17.230
6	83	0.960	18.190
7	19	0.460	18.650
8	15	0.350	19.000
9	60	3.900	22.900
10	32	0.210	23.110

TIME TO FINISH = 23.240000229329

5 Numerical Experiments

In this section we present an outline of the numerical experiments conducted during the development of CONRAD1 and CONRAD2. These numerical experiments were conducted on an Apple Macintosh II using the Absoft MacFortran 2.4 compiler and on a Sun Sparcstation 2 using the Sun Fortran 1.4 compiler. MCCFM, CONRAD1, and CONRAD2 were written using Fortran 77 together with two extensions that are available in almost all compilers. These extensions are the "IMPLICIT NONE" statement that forces the typing of all variables and the "INCLUDE" statement that allows various header files to be read in at the beginning of a program unit. CONRAD2 has been ported to and tested on an Apple Macintosh II, a Sun Sparcstation2, an SGI 4D-35, and an IBM Risc 6000 Model 320. MCCFM and CONRAD1 have been ported to and tested on an Apple Macintosh II and a Sun Sparcstation2. The only machine dependent parts of these codes are the default unit number for screen output, the timing routines, and the floating point constants. MCCFM, CONRAD1, and CONRAD2 are fully documented via comment statements which include porting instructions. All timings reported in this document are for a Sun Sparcstation

2.

We began the development of CONRAD1 with the construction of a trial **MOL** code to investigate the piecewise cubic Hermite **MOL** approach to solving the initial-value problem (8)–(11) with the radiation terms set to zero. Numerous possible time histories for the upper layer temperature T_U were considered. An application was coded for the Apple Macintosh which would produce graphs of the temperature profiles at each time step. This was a case of a "picture being worth a thousand words" because it was **far** easier to analyze these graphs than it was to examine tabular output. From this experimentation, it became clear that it would be possible to approximate the temperature profiles to desired accuracy (relative error of .01) using only about five well placed breakpoints. It also became clear that even **50 or** more equispaced breakpoints would not do nearly **as** well. This is due to the initially steep temperature gradients near $x = 0$ in a typical T_U scenario. Based on this experimentation, we produced the graded mesh heuristic which appears in CONRAD1.

Next, we added a single ceiling conduction node to MCCFM by merging the trial MOL code with MCCFM. The term T_U' was found by differentiating the ideal **gas** law and substituting equations (1)–(3). Once this code was up and working we added simple temperature to the fourth power radiation terms to the right hand sides of the boundary conditions. Now it was clear that this method of combining heat conduction with MCCFM would work; that is the ODE solver could handle this stiff problem. Next, we set out to produce the first version of CONRAD1. As described in Section 4, the user can "turn on" heat conduction on a **room** by room basis. A room with heat conduction turned on has four heat conduction nodes: ceiling, upper wall, lower wall, and **floor**. As previously noted, the stiff ODE solver DEBDF which **was** used in MCCFM was changed to the solver DDRIV2 because of the form of the ODE's.

Once Dr. Glenn Forney's radiation code became available, a new version of CONRAD1 was produced. In addition to incorporating this new radiation model, the ODE solver had to be changed once again. As was explained in Section 4, it was no longer possible to convert the heat **flux** boundary conditions into differential equations, so they were left **as** algebraic equations and DDRIV2 was replaced by the DAE solver DASSL. This completed the development of CONRAD1.

The idea underlying CONRAD2 was inspired by a visit by Dr. Glenn Forney to Clemson University to discuss this project, and by the author's study of DAE solvers. What was needed was some way to convert the heat conduction problem into a functional equation that could be defined implicitly in a subroutine. That this **was** theoretically possible was clear from the existence and uniqueness theory for such problems. Thus we began the development of subroutine CNDUCT. Since it was not clear that this idea would work, we decided to take the simplest approach which was to solve the heat equation by the MOL using a **central** difference approximation to the second spatial derivative. Again, we developed a graded mesh heuristic.

We found that about **20** well placed breakpoints were **all** that was required. The resulting system of ODE's was solved by the backward Euler method. We also tried the Crank-Nicholson method, but found it to be numerically unstable for nonuniform meshes. The backward Euler method is numerically stable independent of the mesh spacing, uniform or not. Instabilities usually exhibit themselves as oscillations in the temperature profile which are easy to spot, especially when looking at a graph.

As was described in Section 4, the temperature profile can be advanced by one time step by solving a tridiagonal system of linear equations. Subroutine CNDUCT as currently implemented takes as inputs the temperatures at the conduction node surfaces (predicted by DASSL) at the next time and returns **as** outputs the temperature gradients at the node surfaces at the next time. Two functional equations for each node are then produced by equating the gradients from CNDUCT with those required by convection and radiation. This approach works amazingly well for reasons that are still not fully understood.

We experimented with the time stepsize in CNDUCT, trying $\Delta t = \frac{k}{2}$ and $\Delta t = \frac{k}{4}$ where E is the time stepsize from the ODE solver DASSL; that is, we tried either **2** or **4** backward Euler steps to integrate from time t to time $t + k$. We found no significant improvement over the results obtained with $\Delta t = E$.

We tried a second approach which did not work at **all**. We took the node surface temperatures predicted by DASSL at the next time and substituted them into the heat flux boundary conditions to generate temperature gradients at the next time. These gradients were inputs for a subroutine, CNDUCT1, which output the temperatures at the node surfaces at the next time. We generated two equations by equating the predicted temperatures and the corresponding outputs from CNDUCT1.

We made use of the vector relative error feature of DASSL to specify relative errors of 10^{-6} for pressure offset, layer interface height, and upper and lower layer masses. For temperatures we specified a relative error of 10^{-2} . The report [1] discusses the requirement for this level of accuracy in the offset pressure, but it seemed unreasonable and inefficient to require this much accuracy for temperatures. CONRAD1 and CONRAD2 have been compared in a number of cases. The outputs from the two room case presented in Sections 3 and 4 are typical. The agreement is within the relative error tolerance for these simulations.

As a further check of CONRAD2, we developed a variant this code which checked for energy conservation in the ceiling conduction node; that is, it checked to see if the amount of heat being added to the ceiling conduction node was consistent with the rise in average temperature. Let T_{av} denote the average temperature in the ceiling conduction node. We computed T_{av} directly using the trapezoid rule and compared this estimate with the value predicted by integrating the ODE

$$\frac{dT_{av}}{dt} = \frac{H}{C_p \rho} \left[\frac{\partial u}{\partial x}(W, t) - \frac{\partial u}{\partial x}(0, t) \right]. \quad (27)$$

This **ODE** is found by integrating the heat equation (19) with respect to the spatial

variable over the ceiling node thickness W . Again, the comparison was within the error tolerance.

6 Future Work

In this section we propose future work to improve CONRAD2.

Using a DAE solver to solve a DFE system appears to be a new development in the application of mathematics. The theoretical basis and the limitations of this method need to be explored. **This** is a mathematical analysis issue, not a programming issue. It would be unwise to proceed without doing this first.

The subroutine CNDUCT in CONRAD2 that solves the heat equation for a conduction node can be rewritten and based on the piecewise cubic Hermite expansion approach to the **MOL** instead of the current central difference approximation approach. This change has several advantages. The order of accuracy of the spatial approximation is increased, and consequently, fewer breakpoints and less memory is required. The expansion method provides a natural method for interpolating the temperature profile between breakpoints, and it provides the gradients at the endpoints directly without the need for an additional approximation. **This** expansion method should be less sensitive to poorly chosen breakpoint spacing. The resulting linear system that must be solved is now pentadiagonal instead of tridiagonal, but the size of the system is cut in half.

The breakpoint heuristic in CONRAD2 can be improved. Because the heat equation solver is hidden in subroutine CNDUCT, it is possible to change the breakpoints without restarting the DAE solver. The current heuristic is based on the final time t_{final} . As an alternative, three different sets of breakpoints could be generated based on three times $t_{\text{short}} < t_{\text{mid}} < t_{\text{final}}$. Subroutine CNDUCT could be rewritten to use the set based on t_{short} for $0 < t \leq t_{\text{short}}$, the set based on t_{mid} for $t_{\text{short}} < t \leq t_{\text{mid}}$, and the set based on t_{final} for $t_{\text{mid}} < t \leq t_{\text{final}}$. **This** would amount to a crude moving mesh strategy. When the breakpoints are changed, the current temperatures at the new breakpoints must be computed. This is especially easy with the piecewise cubic Hermite approach since it provides a natural interpolation formula.

The variation of wall temperature profile in the vertical direction is clearly not correct in CONRAD1 or CONRAD2. These codes predict a steep vertical gradient in the wall temperature at the layer interface height. This is incorrect because the movement of heat down the wall will not necessarily match the movement of the layer interface height. The rate at which heat will be conducted down the wall will be different from the rate at which the layer height moves.

One way to achieve a better vertical approximation would be to divide the wall into a number of fixed area nodes. Instead of two wall nodes with variable area, 10 with fixed areas might be used. Each of these fixed nodes would be handled in the same way as the variable area nodes. When the layer interface lies in one of these nodes, the temperature used in the convective term of the boundary condition could

be based on a weighted (by area) average of T_U and T_L . A heuristic could **also** be developed so that these node could exchange energy vertically inside the wall. An advantage of this approach is that the radiation model could be reworked for fixed area nodes. A much more efficient version of the radiation model is possible when fixed area nodes are used instead of variable area nodes.

A **2-D** approach to heat conduction in the walls should be examined. Here the entire wall would be treated as a single heat conduction node and the a **2-D** heat equation would **be** solved. **If** the number of vertical breakpoints needed is comparable to the number of fixed area nodes mentioned in the previous paragraph, then both methods would have the same computational complexity. **An** advantage of the **2-D** approach would be that vertical heat exchange would not have to be handled by a heuristic.

Currently, the exterior surfaces of heat conduction nodes in **CONRAD1** and **CONRAD2** exchange heat to ambient. The model could be improved by allowing neighboring rooms to exchange heat via conduction through ceilings, walls, and floors. A simple first step would be to implement this for the room(s) of fire origin. The biggest problem here is one of user input. The ceilings, walls, and floors (room interfaces) would now be treated as vents are in that the user would have to specify in the input data **file** the rooms on each side of these “room interfaces.”

A Notation

A_{vent}	area of vent m^2
c_{vent}	vent coefficient, usually about .68
C	specific heat of conduction material
ρ	density of conduction material kg/m^3
ρ_L	density of lower layer
ρ_U	density of upper layer
m	mass kg
P	absolute pressure Pa
ΔP_∞	quasi-steady state pressure
γ	ratio of specific heats for air
\dot{q}	energy transfer rate
R	universal ideal gas constant $\frac{m^2}{s^2K}$
T	temperature in degrees Kelvin K
V	volume m^3

References

- [1] Glenn P. Forney and William F. Moss. Numerical characteristics of zone fire models. Internal Report **4763**, National Institute of Standards and Technology, **1991**.
- [2] Leonard. Y. Cooper and Glenn P. Forney. The consolidated compartment fire model (ccfm) computer application ccfm.vents - part i: Physical reference guide. Internal Report **4342**, National Institute of Standards and Technology, **1990**.
- [3] Glenn P. Forney and Leonard. Y. Cooper. The consolidated compartment fire model (ccfm) computer application ccfm.vents - part iv: Software reference guide. Internal Report **4343**, National Institute of Standards and Technology, **1990**.
- [4] Glenn P. Forney, Leonard. Y. Cooper, and William F. Moss. The consolidated compartment fire model (ccfm) computer application ccfm.vents - part iv: Users reference guide. Internal Report **4345**, National Institute of Standards and Technology, **1990**.
- [5] Leonard. Y. Cooper and Glenn P. Forney. Fire in a room with a hole: A prototype application of the consolidated compartment fire model (ccfm) computer code, **1987**. Presented at the **1987** National Bureau of Standards Annual Conference on Fire Research.
- [6] C. William Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice Hall, Englewood Cliffs, New Jersey, **1971**.
- [7] Walter W. Jones and Richard D. Peacock. Technical reference guide for fast version **18**. Technical Note **1262**, National Institute of Standards and Technology, **1989**.
- [8] Glenn P. Forney. Computing radiative heat transfer occurring in a zone fire model. Internal Report **4709**, National Institute of Standards and Technology, **1991**.
- [9] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. North-Holland, New York, New York, **1989**.
- [10] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software*. Prentice Hall, Englewood Cliffs, New Jersey, **1989**.
- [11] Linda R. Petzold. A description of dassl: A differential/algebraic system solver. technical report **8637**, Sandia National Laboratories, **1982**.

- [12] C. W. Gear. The simultaneous solution of differential-algebraic equations. *IEEE Trans. Circuit Theory*, 18:89–95, 1971.
- [13] T. Dupont. Mesh modification for evolution equations. *Math. Comp.*, 39(159):85–107, 1981.
- [14] D. C. Arney and J. E. Flaherty. A two-dimensional mesh moving technique for time-dependent partial differential equations. *JCP*, 67:124–144, 1986.
- [15] L. Demkowicz. Some remarks on moving finite element methods. *Comp. Meth. App. Mech. Eng.*, 46:339–349, 1984.
- [16] G. Hellwig. *Partial Differential Equations*. Blaisdell, New York, New York, 1964.

NIST-114A (REV. 3-90)		U.S. DEPARTMENT OF COMMERCE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY		1. PUBLICATION OR REPORT NUMBER NIST-GCR-92-605
BIBLIOGRAPHIC DATA SHEET				2. PERFORMING ORGANIZATION REPORT NUMBER
				3. PUBLICATION DATE March 1992
Numerical Analysis Support for Compartment Fire Modeling and Incorporation of Heat Conduction into a Zone Fire Model				
4. AUTHOR(S) William F. Moss				
5. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS) Clemson University Clemson, SC 29634			7. CONTRACT/GRANT NUMBER 60NANB8D0857	
6. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP) Building and Fire Research Laboratory National Institute of Standards & Technology U.S. Department of Commerce Gaithersburg, MD 20899			a. TYPE OF REPORT AND PERIOD COVERED Final 8/15/88 to 3/31/91	
8. SUPPLEMENTARY NOTES				
9. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE)				
<p>The research goal for the first year of the grant was to determine the best available numerical technology for use in zone fire modeling. The goal for the second year was to incorporate heat conduction into a zone fire model in a numerically robust and efficient manner. Three prototype zone fire models named MCCFM, CONRAD1 and CONRAD2 were constructed to test the numerical technology used to realize these goals. These zone fire models and their implementations as Fortran codes are presented. The code MCCFM, developed during the first year of the grant, demonstrates the advantages of using mass as a solution variable instead of density. CONRAD1 and CONRAD2 examine two strategies for coupling the heat conduction equation (a one dimensional partial differential equation) with the zone fire modeling ordinary differential equations. CONRAD1 performs this coupling via the method of lines by using standard piecewise cubic Hermite polynomial basis functions to represent the unknown temperature profiles in the ceiling, wall, and floor heat conduction nodes.</p> <p>CONRAD2 reduces the heat conduction problem to a set of implicitly defined functional equations, a strategy never before used in zone fire modeling. Both CONRAD1 and CONRAD2 use a differential-algebraic equation solver. Supporting numerical results are presented with timings for a Sun Sparcstation 2.</p>				
10. KEY WORDS (6 TO 12 ENTRIES: ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)				
Computer modeling, conductive heat transfer, fire modeling, numerical analysis, zone models				
11. AVAILABILITY			14. NUMBER OF PRINTED PAGES	
<input checked="" type="checkbox"/> UNLIMITED FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).			50	
<input type="checkbox"/> ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.			15. PRICE	
<input checked="" type="checkbox"/> ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.			A03	

ELECTRONIC FORM



