NISTIR 4763

Analyzing and Exploiting Numerical Characteristics of Zone Fire Models

Glenn P. Forney
William F. Moss

United States Department of Commerce
National Institute of Standards and Technology

NISTIR 4763

Analyzing and Exploiting Numerical Characteristics of Zone Fire Models

Glenn P. Forney
William F. Moss

March 1992

U.S. Department of Commerce
Rockwell A. Schnabel, *Acting Secretary*
Technology Administration
Robert M. White, *Under Secretary*
National Institute of Standards and Technology
John W. Lyons, *Director*
Building and Fire Research Laboratory
Gaithersburg, MD 20899

Contents

List of Figures

List of Tables

Analyzing and Exploiting Numerical Characteristics of Zone Fire Models

Glenn P. Forney* William F. Moss†

Abstract

In order to design robust and stable zone fire modeling algorithms, the numerical properties of computer arithmetic and the modeling differential equations must be understood. This report examines some of these properties and provides tools for their analysis. Many sets of differential equations for zone fire modeling can be derived using the laws of conservation of mass and energy. A comparison between various possible formulations is made in terms of numerical properties. One property that many formulations possess is the presence of multiple time scales. Pressures equilibrate much faster than other quantities such as density and temperature. Numerically, this property is known as stiffness. Stiffness, in the context of fire modeling, and numerical methods for handling it are discussed.

1 Introduction

1.1 Background

An understanding of the numerical properties of computer arithmetic and the differential equations used in modeling is required for the design of robust and stable zone fire modeling algorithms. The basic premise used to formulate a zone fire model is that an enclosure can be divided into a number of regions or zones each with approximately uniform conditions. These zones interact by exchanging mass and energy[1]. Mass and energy conservation along with expressions relating mass, density, volume, internal energy, 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 such as that for density or internal energy. Computationally, zone fire modeling is challenging due to the numerical

*Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899, U.S.A. (gpfrn@cfr6.cfr.nist.gov).

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

characteristics of the basic conservation equations used to simulate mass and energy exchange between various zones. The purpose of this report then is to provide a numerical foundation for the design of fire modeling algorithms. It is important to understand when differences in these algorithms are numerically significant so that the best possible fire modeling algorithms can be designed and implemented.

1.2 Overview

Fire modeling algorithms need to be designed with the limitations of floating point arithmetic in mind. The types of error encountered in a computational algorithm are described. Basic tools from numerical analysis are presented in order to understand how errors introduced in the modeling process propagate or grow.

Several zone fire modeling formulations are derived using conservation of mass and energy. The advantages and disadvantages of each formulation from a numerical perspective are indicated. Approximate formulations based on the rapid equilibration of pressure are discussed. Other approximate formulations based on a simplified treatment of the lower layer are also discussed.

The numerical characteristics of a few physical models for exchanging mass and energy between zones where interesting numerical problems have been observed are discussed. The behavior of the differential equations treated as a system is also discussed. The fact that pressure in a room equilibrates rapidly in response to changing fire size, vent area, and layer height causes difficulties in the solution of the differential equations. This phenomenon is called stiffness. Algorithms for solving stiff systems of ordinary differential equations are outlined.

Floating point numbers, model numbers and machine or computer numbers are all terms that refer to those numbers that are exactly representable on the computer. The appendix presents a model of floating point arithmetic which details how numbers are represented in computers and how operations with these numbers behave.

2 Sources and Analysis of Numerical Error

The introduction of error into a fire modeling algorithm is inevitable; however, the algorithm should be designed to minimize the impact of these errors. Some common sources of error are physical models which do not completely describe the phenomena of interest, imprecise data, limitations of the algorithms used to solve the modeling equations, and numerical errors introduced when these algorithms are implemented on a computer. This report is mainly concerned with the last two sources of error.

2.1 Sources of Numerical Error

Numerical error can be divided into three categories: roundoff, truncation and discretization error. **Roundoff error** occurs because computers represent real numbers

using a finite number of digits. The best that can be expected is that the computed result of an operation is the nearest floating point number to the true result. The appendix details the properties of computer arithmetic as implemented on most computers used today. These properties can be summarized by the statement that "subtractions can cause significant loss of accuracy in a numerical computation and this loss of accuracy can be greatly amplified in subsequent calculations, a condition termed catastrophic cancellation." **Truncation error** occurs when an infinite process is replaced by a finite one. This can happen, for example, when an infinite series is truncated after a finite number of terms or when an iteration is terminated after a convergence criterion has been satisfied. **Discretization error** occurs when a continuous process such as a derivative is approximated by a discrete analog such as a divided difference.

Consider the problem of finding the solution at $t > 0$ to the scalar initial value problem

$$\begin{aligned} \frac{dy}{d\tau} &= f(\tau, y) \\ y(0) &= b\,. \end{aligned} \tag{1}$$

Denote this solution by $y(t)$. A discrete analog of problem (1) is obtained by replacing the derivative by a divided difference. The simplest example is the Euler method. The advantages and disadvantages discussed below of using the explicit and implicit Euler methods are representative of more complicated methods for solving differential equations. These alternate methods differ in the number of terms in the Taylor expansion that are considered.

Choose a positive integer n (the number of steps) and define $\tau_i = ih$, $i = 1, \ldots, n$, where the stepsize $h = t/n$. Then $\tau_n = t$ and $y(t)$ is approximated by y_n which is defined by the recurrence

$$\begin{aligned} \frac{y_{i+1} - y_i}{h} &= f(\tau_i, y_i),\ i = 1, \ldots, n-1 \\ y_0 &= b\,. \end{aligned}$$

The quantity $y(\tau_i) - y_i$ is usually called the global error. Disregarding roundoff error, the final global error, $y(t) - y_n$, can be shown to be proportional to the stepsize h under fairly general assumptions about f (see [2]).

A problem such as (1) is called **stable** or **well-conditioned** if small changes in the data or input parameters produce small changes in the solution; otherwise, the problem is called **unstable** or **ill-conditioned**. It is important to note that this concept has *nothing* to do with numerical methods for solving the problem. If $f(\tau, y) = ay$ in (1) with $a > 0$, then this problem is unstable or ill-conditioned. This can be seen by examining the family of solutions to the differential equation that pass close to the initial point $(0, b)$. As illustrated in Figure 1, two of these solutions can be close at $t = 0$, but widely separated at $t > 0$. Similarly, if $f(\tau, y) = ay$ in (1) with $a < 0$, then this problem is stable or well-conditioned.

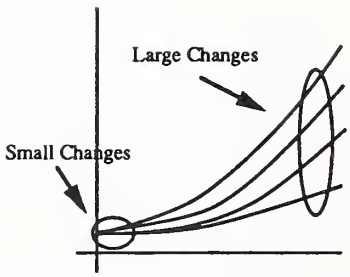

Figure 1: A Graphical Example of Instability

The **inherent error** of a problem is the maximum error that could occur in the solution if the data were perturbed by a relative amount no greater than machine epsilon where machine epsilon is the smallest positive number added to one that gives a result different than one. This and other floating point properties are discussed in more detail in the appendix. In the simplest case where $f(\tau, y) = ay$ in (1), the inherent error would arise from perturbations in a and b. An algorithm for solving a problem is called **numerically stable** if the error in the numerical solution is the same order of magnitude as the inherent error; otherwise, it is called **numerically unstable.** Roughly speaking, an algorithm is numerically unstable if small errors made at one stage are amplified in subsequent stages and seriously degrade the accuracy of the overall computation.

It is important to distinguish between ill-conditioned problems and numerically unstable algorithms. An ill-conditioned problem cannot be made well-conditioned by simply using stable algorithms. All algorithms will have difficulties for such problems. In extreme cases, when increasing the precision of the computation does not help, it may be necessary to modify the problem.

Consider the Euler method applied to (1) with $f(\tau, y) = -ay$ and $a > 0$. The analytic solution is $y(t) = y_0 \exp(-at)$ which decays with increasing t. The discrete solution using the explicit Euler method and assuming exact arithmetic is $y_i = y_{i-1}(1 - ah) = y_0(1 - ah)^i$. The growth of the global error, $e_i := |y(\tau_i) - y_i|$, will determine if the Euler method is numerically stable. The global error at step $i + 1$ can be expressed in terms of the global error at the previous step to be

$$\begin{aligned} e_{i+1} &= |y(\tau_{i+1}) - y_{i+1}| \\ &= |(e^{-ah} - (1 - ah))y(\tau_i) + (1 - ah)(y(\tau_i) - y_i)| \\ &\leq y_0 \frac{(ah)^2}{2} \left(e^{-ah}\right)^i + |1 - ah| e_i \ . \end{aligned}$$

The expression $|1-ah|$ is called the amplification factor. If $h > 2/a$, then $|1-ah| > 1$

and both e_i and y_i grow exponentially with the index i. In this case the Euler method will be numerically unstable (even on a computer that uses an infinite number of digits in its arithmetic).

If $a > 0$, the best option is to change to an implicit method such as the backward Euler method given by

$$\begin{aligned} \frac{y_{i+1} - y_i}{h} &= f(\tau_{i+1}, y_{i+1}),\ i = 1, \ldots, n-1 \\ y_0 &= b\,. \end{aligned}$$

The error bound for the backward Euler method is

$$e_{i+1} \leq y_0 \frac{(ah)^2}{2} \left(e^{-ah}\right)^i + \frac{e_i}{|1 + ah|}\,.$$

Here the amplification factor is $1/(1 + ah)$. For all positive h, the amplification factor is less than or equal to one and the backward Euler method is numerically stable.

This example says that an unstable algorithm can give the wrong result for a stable or well-conditioned problem, whereas a stable algorithm will give correct results. Further, ill-conditioned problems cannot be made well-conditioned by using stable algorithms. Strategies for improving the solution of ill-conditioned problems must center on methods for changing the **problem** to make it more stable.

2.2 Analyzing Numerical Error

Floating point error bounds, such as those presented in the appendix, derived using fundamental properties of computer arithmetic, are difficult to apply to complicated problems. Simpler tools, however, can be derived for analyzing error propagation properties of differentiable functions. Suppose that a fire algorithm can be expressed as a function $f(x)$. As an example, the input x could represent pressure and the value of f could represent enthalpy flow rate through a vent. If x is perturbed by an amount h, how is $f(x)$ affected? This question can be addressed using condition numbers. Suppose f has two continuous derivatives. From Taylor's Theorem, it follows that

$$\Delta f := f(x + h) - f(x) = f'(x)h + O(h^2) \tag{2}$$

where $O(h^2)$ denotes a term that is bounded by a constant times h^2. The absolute condition number of f, denoted $c_a(f)$, is defined to be the coefficient of h in an expansion of Δf in powers of h. To a first order approximation, the absolute condition number relates the absolute changes or error in x with the absolute changes in f according to

$$\Delta f = c_a(f)h.$$

Consequently

$$c_a(f) = f'(x) .$$

Similarly, the relative condition number of f, denoted $c_r(f)$, is defined to be the coefficient of h/x in an expansion of $\Delta f/f$ in powers of h/x. From (2) it follows that

$$c_r(f) = \frac{xf'(x)}{f(x)} .$$

Similarly, the relative condition number relates the relative error of x with the relative error of f according to

$$\frac{\Delta f}{f} = c_r(f)\frac{h}{x}$$

The absolute and relative condition numbers c_a and c_r can be used to analyze numerical properties of fire algorithms. These condition numbers give a quantitative measure of how input errors are magnified by a problem. For example, consider the expression

$$\dot{q}_{\text{vent}} = K\sqrt{P_1 - P_2}$$

for enthalpy flow rate through a vent where $K = c_p c_{\text{vent}} A_{\text{vent}} \sqrt{2\rho_{\text{vent}}} T_{\text{vent}}$, c_p is the constant pressure specific heat, A_{vent} is the area of the vent, c_{vent} is the vent flow coefficient, and ρ_{vent} and T_{vent} are the density and temperature of the gas flowing through the vent. The pressure difference $P_1 - P_2$ across the vent drives the flow of mass through the vent. The relative and absolute condition numbers for $\dot{q}_{\text{vent}}$ with respect to pressure P_1 are

$$\begin{aligned} c_a(\dot{q}_{\text{vent}}) &= .5\frac{\dot{q}_{\text{vent}}}{P_1 - P_2} , \\ c_r(\dot{q}_{\text{vent}}) &= .5\frac{P_1}{P_1 - P_2} . \end{aligned}$$

Both condition numbers show that problems can occur when P_1 and P_2 are close and in the case of the relative condition number, these problems are independent of K.

Condition numbers can be generalized to predict error propagation properties of vector-valued functions, for example see [3]. Table 1 gives the absolute and relative condition numbers for a few elementary functions.

Table 1: Absolute and Relative Condition Numbers for a Few Elementary Functions

f	$c_a(f)$	$c_r(f)$
$f(x)$	$f'(x)$	$\frac{xf'(x)}{f(x)}$
$x+a$	1	$\frac{x}{x+a}$
ax	a	1
$\sin(x)$	$\cos(x)$	$x\cot(x)$
x^a	ax^{a-1}	a
e^{ax}	ae^{ax}	ax

3 Zone Fire Modeling Formulations

The zone fire models presented here take the mathematical form of an initial value problem for a system of differential equations. These equations are derived using the conservation of mass or continuity equation, the conservation of energy or the first law of thermodynamics, the ideal gas law, and definitions of density and internal energy (for example, see [4]). The conservation of momentum is ignored. These conservation laws are invoked for each zone or control volume. A zone may consist of a number of interior regions (usually an upper and a lower gas layer), and a number of wall segments. The basic assumption of a zone fire model is that properties such as temperatures can be uniformly approximated throughout the zone. It is remarkable that this assumption seems to hold for as few as two gas layers.

Many differential equation formulations based upon these assumptions can be derived. One formulation can be converted into another using definitions of density, internal energy and the ideal gas law. Though equivalent analytically, these formulations differ in their numerical properties. One property that many share is the presence of multiple time scales. Physically, the pressure in a compartment equilibrates much quicker than densities and temperatures. Numerically, this property is known as stiffness and requires the use of special differential equation solvers. The physical origins and numerical consequences of stiffness are discussed in more detail in section 4.2.

Each differential formulation can be expressed in terms of mass and enthalpy flow rates. These flow rates represent the exchange of mass and/or energy between zones due to physical phenomena or sub-models such as fire plumes, natural and forced vents, convective and radiative heat transfer *etc.* For example, a vent exchanges

mass and energy between zones in connected rooms, a fire plume typically adds heat to the upper layer and transfers entrained mass and energy from the lower to the upper layer, and convection transfers energy from the gas layers to the surrounding walls.

Using the formalism developed by Cooper for CCFM.VENTS [4] the mass flow rate to the upper and lower layers is denoted $\dot{m}_U$ and $\dot{m}_L$ and the enthalpy flow rate to the upper and lower layers is denoted $\dot{q}_U$ and $\dot{q}_L$. It is tacitly assumed that these flow rates may be computed in terms of zone properties such as temperatures, densities, *etc.* These rates represent the net sum of all possible sources of mass and energy due to phenomena like those listed above. The numerical characteristics of the differential equation formulations are easier to identify if the underlying physical phenomena are decoupled in this way.

Many approximations are obviously necessary when developing physical submodels for the mass and enthalpy flow rate terms. For example, most fire models assume that 1) the specific heat terms c_p and c_v are constant even though they depend upon temperature, 2) hydrostatic terms can be ignored in the equation of state (the ideal gas law) relating density of a layer with its temperature. We wish to distinguish between various formulations according to whether they are mathematically equivalent to the conservation laws of mass and energy. A formulation which is equivalent to the conservation laws will be denoted **conservative** otherwise it will be identified as **approximate**. Conservative formulations in this sense are not necessarily better than approximate ones. The next two sections discuss formulations which are conservative and approximate. Again, two conservative formulations that are equivalent mathematically need not be equivalent numerically.

3.1 Conservative Formulations

A compartment can be divided into two control volumes, an upper layer of hot gases and smoke and a lower layer of air as illustrated in Figure 2. The gas in each layer has attributes of mass, internal energy, density, temperature, and volume denoted respectively by m_i, E_i, ρ_i, T_i, and V_i where $i = L$ for the lower layer and $i = U$ for the upper layer. The compartment as a whole has the attribute of pressure P. These eleven variables are related by means of the following seven[1] constraints

$$\rho_i = \frac{m_i}{V_i} \text{ (density)} \tag{3}$$

$$E_i = c_v m_i T_i \text{ (internal energy)} \tag{4}$$

$$P = R\rho_i T_i \text{ (ideal gas law)} \tag{5}$$

$$V = V_L + V_U \text{ (total volume)} . \tag{6}$$

[1] We get seven by counting density, internal energy and the ideal gas law twice (once for each layer).

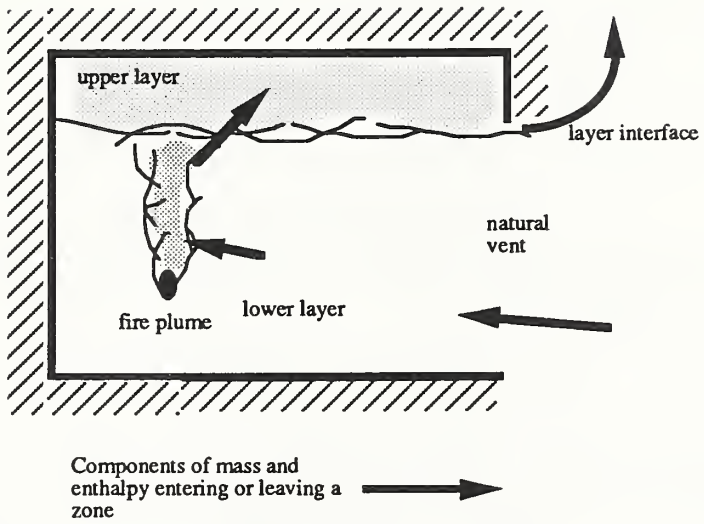

Figure 2: Two Layer Zone Model Setup

The specific heats at constant volume and at constant pressure c_v and c_p, the universal gas constant, R, and the ratio of specific heats, γ, are related by

$$\begin{aligned} \gamma &= \frac{c_p}{c_v}, \\ R &= c_p - c_v . \end{aligned}$$

For air, $c_p \approx 1000$ kJ/kg K and $\gamma = 1.4$.

The differential equations for mass in each layer are trivially

$$\begin{aligned} \frac{dm_L}{dt} &= \dot{m}_L \\ \frac{dm_U}{dt} &= \dot{m}_U . \end{aligned}$$

The first law of thermodynamics states that the rate of increase of layer internal energy plus the rate at which the layer does work by expansion is equal to the rate at which enthalpy is added to the gas. In differential equation form this is

$$\overbrace{\frac{dE_i}{dt}}^{\text{internal energy}} + \overbrace{P\frac{dV_i}{dt}}^{\text{work}} = \overbrace{\dot{q}_i}^{\text{enthalpy}} . \qquad (7)$$

A differential equation for pressure can be derived by adding the upper and lower

layer versions of equation (7), noting that $\frac{dV_U}{dt} = -\frac{dV_L}{dt}$ and

$$\frac{dE_i}{dt} = \frac{d(c_v m_i T_i)}{dt} = \frac{c_v}{R}\frac{d}{dt}(PV_i) \tag{8}$$

to obtain

$$\frac{dP}{dt} = \frac{\gamma - 1}{V}(\dot{q}_L + \dot{q}_U) \, . \tag{9}$$

Differential equations for the layer volumes can be obtained by substituting equation (8) into equation (7) to obtain

$$\frac{dV_i}{dt} = \frac{1}{P\gamma}((\gamma - 1)\dot{q}_i - V_i\frac{dP}{dt}) \, . \tag{10}$$

Equation (7) can be rewritten to eliminate the $\frac{dV_i}{dt}$ term to obtain

$$\frac{dE_i}{dt} = \frac{1}{\gamma}(\dot{q}_i + V_i\frac{dP}{dt}) \, .$$

Differential equations for the densities can be derived by applying the quotient rule to $\frac{d\rho_i}{dt} = \frac{d}{dt}\left(\frac{m_i}{V_i}\right)$ and using equation (10) to eliminate $\frac{dV_i}{dt}$ to obtain

$$\frac{d\rho_i}{dt} = \frac{-1}{c_p T_i V_i}((\dot{q}_i - c_p \dot{m}_i T_i) - \frac{V_i}{\gamma - 1}\frac{dP}{dt}) \, . \tag{11}$$

Differential equations for temperatures can be derived by applying the quotient rule to $\frac{dT_i}{dt} = \frac{d}{dt}\left(\frac{P}{R\rho_i}\right)$ and using equation (11) to eliminate $\frac{d\rho_i}{dt}$ to obtain

$$\frac{dT_i}{dt} = \frac{1}{c_p \rho_i V_i}((\dot{q}_i - c_p \dot{m}_i T_i) + V_i\frac{dP}{dt}) \, . \tag{12}$$

Differential equations for each of the eleven variables are summarized in Table 2. Notice that a $\frac{dP}{dt}$ term occurs in all but the mass equations. For many fire scenarios this term can be set to zero. Section 3.3 discusses approximations to the zone fire modeling equations derived by dropping the pressure transient terms.

Using the constraint equations (3) to (6), it can be shown that four of the eleven variables can be chosen as solution variables. The time evolution of these solution variables can be computed by solving the corresponding differential equations together with appropriate initial conditions. The remaining seven variables can be determined from the solution variables. There are, however, many possible differential equation formulations. The numerical characteristics of some of these formulations will be discussed in the next section.

Table 2: Conservative Zone Modeling Differential Equations

Equation Type	Differential Equation
i'th layer mass	$\frac{dm_i}{dt} = \dot{m}_i$
pressure	$\frac{dP}{dt} = \frac{\gamma-1}{V}(\dot{q}_L + \dot{q}_U)$
i'th layer energy	$\frac{dE_i}{dt} = \frac{1}{\gamma}(\dot{q}_i + V_i\frac{dP}{dt})$
i'th layer volume	$\frac{dV_i}{dt} = \frac{1}{\gamma P}((\gamma-1)\dot{q}_i - V_i\frac{dP}{dt})$
i'th layer density	$\frac{d\rho_i}{dt} = \frac{-1}{c_p T_i V_i}((\dot{q}_i - c_p\dot{m}_i T_i) - \frac{V_i}{\gamma-1}\frac{dP}{dt})$
i'th layer temperature	$\frac{dT_i}{dt} = \frac{1}{c_p \rho_i V_i}((\dot{q}_i - c_p\dot{m}_i T_i) + V_i\frac{dP}{dt})$

Table 3: Conservative Zone Model Equation Selections

Zone Fire Model	Equations	Substitutions
CFAST, FAST [5]	$\frac{d\Delta P}{dt}, \frac{dV_U}{dt}, \frac{dT_U}{dt}, \frac{dT_L}{dt}$	$\Delta P = P - P_{\text{ref}}$
CCFM.HOLE [6]	$\frac{d\Delta P}{dt}, \frac{dy}{dt}, \frac{d\rho_U}{dt}, \frac{d\rho_L}{dt}$	$\Delta P = P - P_{\text{ref}}, y = V_L/A_{room}$
CCFM.VENTS [4]	$\frac{d\Delta P}{dt}, \frac{dy}{dt}, \frac{dm_U}{dt}, \frac{dm_L}{dt}$	$\Delta P = P - P_{\text{ref}}, y = V_L/A_{room}$
FIRST, HARVARD V [7]	$\frac{dE_U}{dt}, \frac{dE_L}{dt}, \frac{dm_U}{dt}, \frac{dm_L}{dt}$	

3.2 Numerical Characteristics of Several Zone Fire Modeling Differential Equations

There are 330 different ways to select four variables from eleven to form a system of differential equations. Many of these systems are incomplete due to the relationships that exist between the variables given in equations (3) to (6). For example the variables, ρ_U, V_U, m_U and P form a dependent set since $\rho_U = m_U/V_U$. Table 3 shows the solution variable selection made by a few zone fire models. The variable y that appears in this table is the height of the upper layer above the compartment floor.

The number of differential equation formulations can be considerably reduced by not mixing variable types between layers; that is, if upper layer mass is chosen as a solution variable, then lower layer mass must also be chosen. For example, for

two of the solution variables choose m_L and m_U, or ρ_L and ρ_U, or T_L and T_U. For the other two solution variables pick E_L and E_U or P and V_L or P and V_U. This reduces the number of distinct formulations to nine. Since the numerical properties of the upper layer volume equation are the same as a lower layer one, the number of distinct formulations can be reduced to six.

The next several subsections discuss the numerical implications of using these formulations. Some of the problems discussed can be solved by ignoring the pressure equation. The resulting approximate equations and their implications are discussed in section 3.3.

3.2.1 Pressure

Some of the numerical problems that arise in zone fire modeling are due to the difficulty of computing accurate pressure differences across vent openings. When adjacent room pressures are close, a catastrophic cancellation will lead to a loss of significant digits. If the pressures are too close, the result of the subtraction is roundoff noise. This can cause problems if the noise is amplified in the next stage of the computation; that is, the noise is propagated and may dominate some later stage of the calculation. This problem is compounded by the fact that the base pressure in a room at 1 atmosphere is about 10^5 pascals (Pa). Pressure drops across a vent as low as .1 Pa can cause significant mass flow through a vent. Therefore, if pressure is used as a solution variable, seven accurate digits must be carried in order to have one significant digit in the vent flow calculation. One way around this problem is to solve for an offset pressure, ΔP, where for some reference pressure, P_{ref}, the room pressure is given by

$$P = P_{ref} + \Delta P \, .$$

The differential equation for relative pressure, ΔP is given by

$$\begin{aligned} \frac{d\Delta P}{dt} &= \frac{dP}{dt} - \frac{dP_{ref}}{dt} \\ &= \frac{dP}{dt} \text{ if } P_{ref} \text{ is constant} \, . \end{aligned}$$

3.2.2 Internal Energy

The problem with using internal energy in a formulation is the difficulty in accurately determining the offset pressure, ΔP, needed for accurate vent flow calculations. In large part this is due to the fact that energy is not an intensive variable, it is proportional to layer volume as well as temperature. To illustrate, consider that the total room pressure can be expressed in terms of lower and upper layer internal energy and room volume by using equations (3) to (6) to obtain

$$P = \frac{\gamma - 1}{V}(E_L + E_U) \, .$$

Substituting $E_L = E_{ref} + \Delta E_L$, $E_U = E_{ref} + \Delta E_U$ and $P = P_{\text{ref}} + \Delta P$ into the above equation and solving for E_{ref} and ΔP we obtain

$$\begin{aligned} \Delta P &= \frac{\gamma - 1}{V}(\Delta E_L + \Delta E_U) \\ E_{\text{ref}} &= \frac{P_{\text{ref}} V}{2(\gamma - 1)} \cdot \end{aligned}$$

The term, ΔP can be small while the term $|\Delta E_L| + |\Delta E_U|$ is large. This will result in cancellation errors. The term, ΔP, will not be zero in general, therefore we can not assume that $\Delta E_L = -\Delta E_U$. Section 3.3.2 discusses the approximation where ΔP is assumed to be zero.

3.2.3 Temperature and Density

The temperature and density differential equations have several advantages and disadvantages in common. As seen in Table 2, both the density and temperature equations have the term $\dot{q}_i - c_p \dot{m}_i T_i$. The vent flow component of this term is identically zero for flows leaving a zone since the enthalpy flow rate for a vent flow is $\dot{q}_{\text{vent}} = c_p \dot{m}_{\text{vent}} T_{\text{vent}}$. An unnecessary subtraction can be avoided by setting this vent flow component to zero analytically, thereby avoiding a loss of significant digits. This property of the density and temperature equation is due to the fact that, ignoring pressure transients, the temperature of a room is not affected by flows leaving it. Similar cancellations must also be eliminated in the pressure equation for this strategy to be useful. Unfortunately, this is not possible unless the pressure equation is dropped from the equation set.

The density and temperature equations also have a problem in common. Both have layer volume terms in the denominator which may vanish. Although these singularities are removable (if they were not the derivatives $\frac{d\rho}{dt}$ and $\frac{dT}{dt}$ would be infinite), they cause numerical problems since it is difficult to determine proper values for $\frac{d\rho}{dt}$ and $\frac{dT}{dt}$ in this case. One method used to solve this problem is not to allow layers to vanish.

3.2.4 Mass

The mass equation does not have the vanishing denominator problem of the density or temperature equation. Using the mass equation allows sensible initial conditions. The mass for a layer with zero height is just zero. However, quantities derived from mass such as density are only valid when a layer volume has a significant number of accurate digits.

3.3 Approximate Formulations

The formulations discussed in this section are approximate in the sense that certain terms deemed negligible are removed from the modeling differential equations

Table 4: Approximate Zone Model Equation Selections

Zone Fire Model	Equations	Substitutions
ASET [8, 9]	$P = P_{\text{amb}}, \frac{dy^*}{dt}, \frac{d\rho_U^*}{dt}, T_L = T_{\text{amb}}$	$y = V_L/A_{room}$
BRI [10, 11]	$\frac{dP}{dt} = 0, \frac{dV_U}{dt}, \frac{dT_U}{dt}, \frac{dT_L}{dt}$	
LAVENT [12, 13]	$P = P_{\text{amb}}, \frac{dy}{dt}, \frac{dm_U}{dt}, T_L = T_{amb}$	$y = V_L/A_{room}$

derived in Section 3.1. Three types of approximations and their error behavior are discussed. Two involve the elimination of the pressure transient, ($\frac{dP}{dt}$), terms. A third way approximates the differential equations by assuming that the conditions in the lower layer remain at ambient. Some fire models along with their variable choices that use some of the approximation techniques discussed in this section are listed in Table 4. ASET assumes the pressure remains at ambient and uses a non-dimensional form of the layer height and upper layer density equations. BRI assumes that the pressure relaxes to quasi-steady values instantly. BRI solves a non-linear algebraic equation equivalent to equation (9) with $\frac{dP}{dt}$ set to zero. First, we examine the behavior of pressure as motivation for making these approximations.

3.3.1 Behavior of the Pressure Equation for a Heated Enclosure with a Small Leak

The pressure in a compartment approaches steady state rapidly if other compartment properties such as layer temperatures, fire size, and vent sizes are constant. The equilibrium pressure value depends on the fire size and vent areas or more generally on the sources and sinks of enthalpy in the room. To characterize the equilibrium pressure value and the time required to reach equilibrium, consider a room with a fire that is vented to the outside as illustrated in Figure 3.

To simplify the analysis assume that the vent is a slit located at the floor so that Bernoulli's law,

$$\dot{m}_{\text{vent}} = c_{\text{vent}} A_{\text{vent}} \sqrt{2\rho_{\text{vent}} \Delta P},$$

can be used to model the mass flow through the vent. The conclusions found here hold for more general vent algorithms since they all use or compute pressure differences. Here A_{vent} denotes the area of the vent, c_{vent}, the vent coefficient, while ρ_{vent} denotes the density of the vent flow gas. Further, assume that the density and temperature of the gas flowing through the vent is constant over the time period required to reach pressure equilibrium. This is reasonable since this time period is typically rather short.

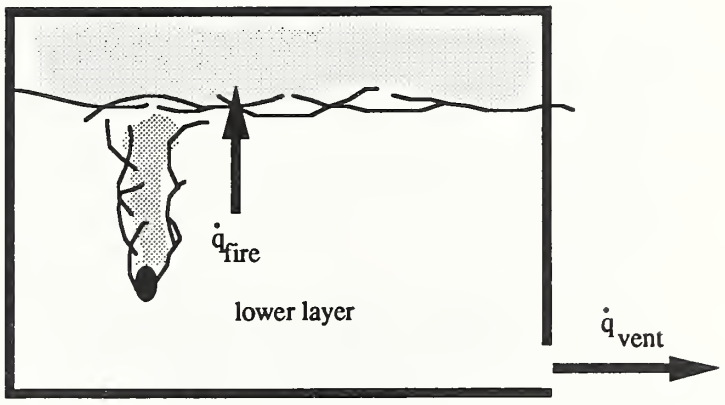

Figure 3: One Room Test Case

The source of enthalpy in the room is the fire and is denoted by $\dot{q}_{\text{fire}}$[2]. The enthalpy flow rate out of the room is due to the vent and is denoted by $\dot{q}_{\text{vent}}$. The initial value problem for the pressure drop across the vent is

$$\begin{aligned} \frac{d\Delta P}{dt} &= \frac{\gamma - 1}{V}(\dot{q}_{\text{fire}} - \dot{q}_{\text{vent}}) \\ \Delta P(0) &= 0 \end{aligned} \tag{13}$$

where ΔP is the pressure drop across the vent, V is the volume of the room, and the other terms have been defined previously. Setting

$$\begin{aligned} a &= \frac{\gamma - 1}{V}\dot{q}_{\text{fire}} \\ b &= \frac{\gamma - 1}{V}c_p c_{\text{vent}} A_{\text{vent}} T_{\text{vent}} \sqrt{2\rho_{\text{vent}}}\,, \end{aligned}$$

problem (13) simplifies to

$$\begin{aligned} \frac{d\Delta P}{dt} &= a - b\sqrt{\Delta P} \\ \Delta P(0) &= 0\,. \end{aligned}$$

This differential equation is separable and can be integrated to obtain the implicit solution

$$t = \frac{2\Delta P_\infty}{a}\left(ln\left(\frac{1}{1 - \sqrt{\Delta P/\Delta P_\infty}}\right) - \sqrt{\Delta P/\Delta P_\infty}\right)\,, \tag{14}$$

[2]Most entrainment models used by fire plume models do not affect the calculation of the pressure rise or transient time since the enthalpy entrained from the lower layer cancels with the enthalpy added back into the upper layer.

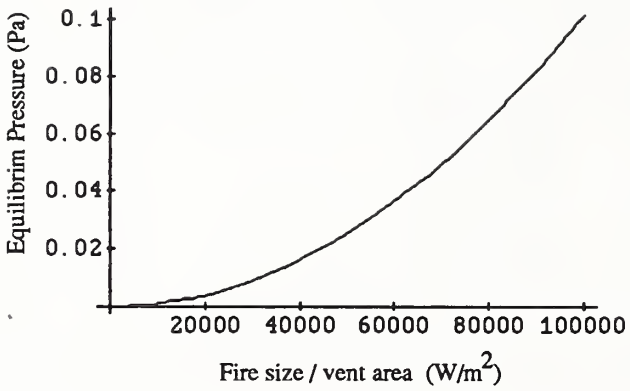

Figure 4: Pressure Equilibrium Values for a One Room Case

or equivalently

$$\Delta P = \Delta P_\infty \left(1 - e^{\frac{-at}{\Delta P_\infty}} e^{-\sqrt{\Delta P/\Delta P_\infty}}\right)^2 \tag{15}$$

with the equilibrium pressure ΔP_∞ given by

$$\Delta P_\infty = \left(\frac{a}{b}\right)^2 \approx 1.00 \times 10^{-11} \left(\frac{\dot{q}_{\text{fire}}}{A_{\text{vent}}}\right)^2 . \tag{16}$$

where $T_{\text{vent}} = 300K$,$\rho_{\text{vent}} = 1.2kg/m^3$,$c_{\text{vent}} = .68$ and $c_p = 1000kJ/kgK$. Taking the limit as $t \to \infty$, equation (15) verifies that ΔP_∞ is the equilibrium pressure and equation (14) shows how long it takes to achieve it. Substituting $\Delta P/\Delta P_\infty = .99$ into equation (14) gives the time required for the pressure to reach 99 per cent of ΔP_∞ or

$$t_{.99} \approx 21.5 \frac{\Delta P_\infty}{\dot{q}_{\text{fire}}/V} .$$

For a room with volume $18m^3$, a 100 kW fire, and an equilibrium pressure of .1 Pa, the time required to reach 99 per cent of the equilibrium value is approximately .00038 seconds.

Figure 4 shows the dependence of equilibrium pressure on fire size and vent area. Figure 5 shows how the time required to reach equilibrium is affected by these same two quantities.

This analysis shows that pressures tend towards equilibrium values very quickly. The equilibrium pressure value is a function of quantities that affect the sources or sinks of enthalpy in a room. The flow of enthalpy through a vent into and out

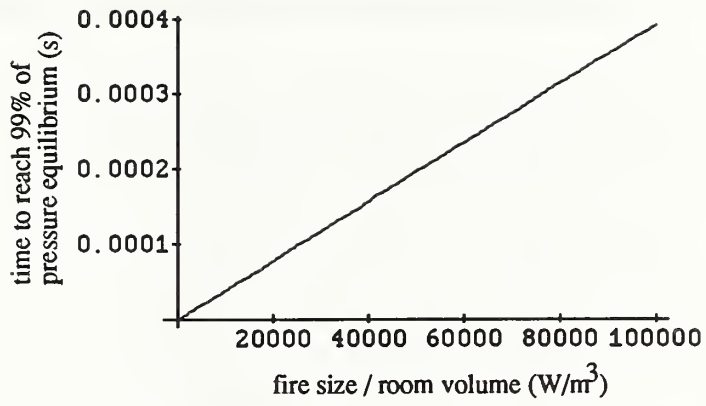

Figure 5: Pressure Equilibrium Times for a One Room Case

of a room is affected by vent area, layer heights and temperatures *etc.* The fire also contributes to the enthalpy gain in a room. As these quantities (vent area, layer heights, fire size, *etc*) change, the room pressure adjusts almost instantly to a new quasi-steady state value. Numerically, this property is known as stiffness. A numerical challenge of zone fire modeling is to determine pressures accurately and efficiently.

For a one room model the algebraic equation (16) can be used to determine the equilibrium pressure instead of the pressure differential equation (13). Furthermore, the $\frac{dP}{dt}$ term in the other zone modeling differential equations (as expressed in Table 2) can be dropped in cases when $\frac{dP}{dt}$ goes to zero almost instantly.

3.3.2 Constant Pressure Approximation

The constant pressure approximation is based on the assumption that $\frac{dP}{dt} = 0$ and that the compartment pressure is given by the pressure at some particular elevation usually the floor. It is further assumed that P_{floor} is hydrostatically related to a reference pressure using

$$P_{\text{floor}} = P_{\text{ref}} - \rho_{\text{ref}} g y_{\text{floor}}$$

where P_{ref} and ρ_{ref} are the reference pressure and density, g is the acceleration of gravity, and y_{floor} is the distance between the floor and the reference elevation. Usually, P_{ref} is the atmospheric pressure at the reference elevation under ambient conditions and ρ_{ref} is the ambient density. In the simplest case, when each room's floor is at the reference elevation, the floor pressure is the same as the reference

Table 5: Approximate Zone Modeling Differential Equations

Equation Type	Differential Equation
i'th layer energy	$\frac{dE_i}{dt} = \frac{\dot{q}_i}{\gamma}$
i'th layer volume	$\frac{dV_i}{dt} = \frac{(\gamma-1)\dot{q}_i}{\gamma P}$
i'th layer density	$\frac{d\rho_i}{dt} = -\frac{\dot{q}_i - c_p \dot{m}_i T_i}{c_p T_i V_i}$
i'th layer temperature	$\frac{dT_i}{dt} = \frac{\dot{q}_i - c_p \dot{m}_i T_i}{c_p \rho_i V_i}$

pressure for all rooms. Table 5 gives the approximate zone modeling differential equations corresponding to the conservative ones given previously in Table 2. The equations in this table were obtained by neglecting the dP/dt term in Table 2 where P is the total pressure in a room.

These approximate equations are reasonable if the true pressure offsets (with respect to P_{floor}) are small compared to the hydrostatic head, the pressure drop between the floor and the ceiling. This pressure drop is about 35 Pa for a room with a 3m ceiling containing air at ambient conditions ($\rho = 1.2kg/m^3$). Equation (16) shows that for a simple one room model an equilibrium pressure offset of .1 pa corresponds to a $\dot{q}_{\text{fire}}$ of 100 kw, an A_{vent} of $1m^2$, and ambient ρ_{vent} and T_{vent}.

The absolute errors generated by using the approximate equations in Table 5 can be estimated by defining an error differential equation for each variable. Assuming that the mass and enthalpy sources are the same for the conservative and approximate differential equations, define the approximation error for the temperature equation using

$$e_i^T = \hat{T}_i - T_i \,,$$

where $\hat{T}_i$ denotes the approximate temperature. An initial value problem for e_i^T can be derived using

$$\begin{aligned} \frac{de_i^T}{dt} &= \frac{d\hat{T}_i}{dt} - \frac{dT_i}{dt} \\ &\approx \frac{-1}{c_p \rho_i} \frac{dP}{dt} \\ e_i^T(0) &= 0 \,. \end{aligned}$$

This equation can be solved assuming that the density is constant over the time period required to achieve equilibrium. The result is

$$\text{temperature error} \approx \frac{-\Delta P_\infty}{c_p \rho_i} \,. \tag{17}$$

Table 6: Approximate Zone Modeling Differential Equation Error

Equation Type	Error Estimate
i'th layer energy	$-\frac{V_i \Delta P_\infty}{\gamma}$
i'th layer volume	$\frac{V_i \Delta P_\infty}{\gamma P}$
i'th layer density	$-\frac{\Delta P_\infty}{c_p T_i}$
i'th layer temperature	$\frac{-\Delta P_\infty}{c_p \rho_i}$

The error in temperature resulting from this approximation is related to the magnitude of the equilibrium pressure. The approximate equations are then valid as long as the pressure rise is not significant. Another way of looking at this is that the conservative equations in Table 2 are essentially the same as the approximate equations in Table 5 except during a transient period when $\frac{dP}{dt}$ decays to zero.

Approximate errors for the other zone modeling equations are derived similarly and are presented in Table 6. The error defined by equation (17) underestimates the actual error since the enthalpy flow rate out of a room is highly sensitive to the pressure drop across the vent.

3.3.3 Algebraic Pressure Approximation

The pressure differential equation (9) can be approximated algebraically by determining the pressure that satisfies the non-linear equation

$$\dot{q}_L(P) + \dot{q}_U(P) = 0 \,. \tag{18}$$

Unlike the pressure approximation in the previous section, this pressure will not in general be constant in time. For simple cases, such as a one room model with one fire and layer height above a thin slit vent, this equation can be solved analytically for P. For example, using equation (16) for the equilibrium pressure offset ΔP_∞, this equation has the solution

$$P = P_{\text{ref}} + \left(\frac{\dot{q}_{\text{fire}}}{c_p c_{\text{vent}} A_{\text{vent}} T_{\text{vent}} \sqrt{2 \rho_{\text{vent}}}} \right)^2 .$$

Rehm and Baum in [14] similarly calculate the equilibrium pressure rise in the context of field modeling.

For multiple room simulations the approximation is more complicated. An equilibrium pressure offset must be calculated for each room using an equation similar

to (18). This results in a system of equations since the equilibrium pressure in one room depends on the equilibrium pressure in another *via* vent connections. Differential equation solvers such as Petzold's DASSL [15, 16] can solve algebraic equations simultaneously with differential equations. This was tried with CCFM.VENTS. It was found that there was no advantage to doing this. The algebraic pressure equation version could not track rapidly changing pressures and run times were not significantly shorter.

3.3.4 A One Zone Approximation

Conditions in the lower layer are essentially ambient for many fire scenarios of interest. This observation may be exploited by replacing the differential equation for lower layer density (equation (11)) with $\rho_L = \rho_{\text{amb}}$ or the differential equation for lower layer temperature (equation (12)) with $T_L = T_{\text{amb}}$. This is equivalent to assuming that the rate of mass and enthalpy additions to the lower layer satisfy $\dot{q}_L/(c_p\dot{m}_L) = T_{\text{amb}}$ which will be true as long as the temperature of flows added to the lower layer are at ambient.

4 Zone Fire Modeling Numerical Characteristics

Physical models of natural and forced vent flow, fire plumes, radiation, conduction, convection *etc* are used to exchange or transfer mass and/or energy between zones. As with the differential equation formulations discussed in the previous section, two physical models for computing these phenomena may be identical physically but be different numerically. This section addresses some of the numerical issues important in the design of physical algorithms, some miscellaneous numerical considerations appropriate for any fire model and finally discusses the numerical properties of systems of differential equations.

4.1 Numerical Characteristics of Some Physical Models

4.1.1 Natural Vent Flow

Numerical difficulties can arise when calculating vent flow because of its dependence on the square root of pressure differences. This is especially a problem when the pressure drop across the vent is small relative to the pressure computed in each room adjacent to the vent. To illustrate this phenomena consider two rooms. Suppose the first room has a fire and is connected to the outside and to a second room which does not have a fire. This second room is assumed to be connected only to the first room and not to the outside. To simplify the analysis assume that the vents are narrow slits located at the floor. This configuration is depicted in Figure 6.

Suppose that the pressure above ambient in rooms 1 and 2 are denoted ΔP_1 and ΔP_2. Theoretically, these pressure offsets will be the same after the initial

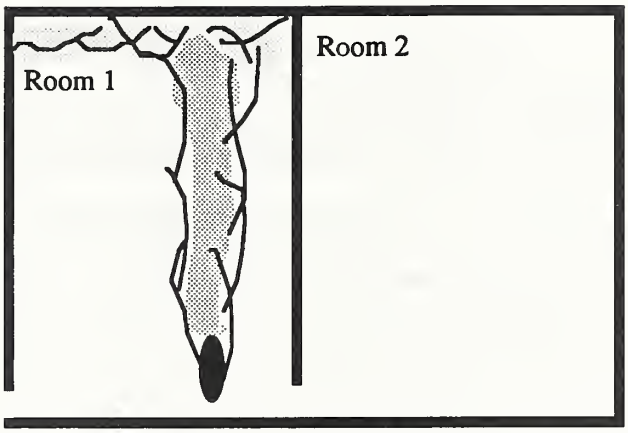

Figure 6: Two Room Test Case Configuration

pressure transient dies away according to the analysis in Section 3.3.1. Numerically, however, these two offsets will be different. Though this difference may be insignificant physically, the magnification of the cancellation error occurring in the vent flow computation may be significant numerically. Unfortunately, simply setting the pressure drop to zero for physically insignificant flows is not an adequate solution to this problem of unwanted error propagation as will be explained next.

In this test problem, the pressure in room 1 will rise to its equilibrium value as given by equation (16) in a time period given by equation (14). If a cut off pressure drop ΔP_{cut} is chosen so that $\frac{d\Delta P_2}{dt} = 0$ for $|\Delta P_2 - \Delta P_1| < \Delta P_{\text{cut}}$, then ΔP_1 will continue to rise (since $\frac{d\Delta P_1}{dt}$ is not zero) while ΔP_2 remains fixed. Eventually $|\Delta P_2 - \Delta P_1|$ will exceed ΔP_{cut} so that $\frac{d\Delta P_2}{dt}$ will not be zero. Again, ΔP_2 will rapidly approach ΔP_1 until $|\Delta P_2 - \Delta P_1| < \Delta P_{\text{cut}}$. This type of algorithm will result in a drastic reduction in time step size as the differential equation solver tries to track the solution due to the "stair stepping" behavior of ΔP_2 illustrated in Figure 7. Of course, for this test problem special methods may be used to resolve the problem such as eliminating the differential equation for $\frac{d\Delta P_2}{dt}$ by setting $\Delta P_2 = \Delta P_1$. This will not work for the general case.

Denote the pressure difference across the vent by $\Delta P = \Delta P_1 - \Delta P_2$. This section establishes a criterion for choosing ΔP_{cut} such that pressure differences satisfying $\Delta P > \Delta P_{\text{cut}}$ contain at least some accurate digits and gives a method for smoothly damping ΔP to zero when $\Delta P < \Delta P_{\text{cut}}$.

Although the Bernoulli law for computing vent flow is known to break down for small pressure differences, the problems discussed here are solely numerical and are a consequence of the fact that only a finite number of digits are used to compute

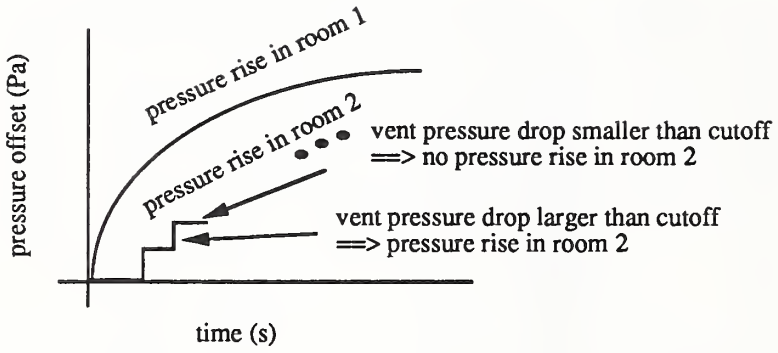

Figure 7: Numerical Pressure Behavior for Two Room Test Case

and represent the pressure offsets ΔP_1 and ΔP_2.

Suppose that ΔP_1 is an $o(1)$ quantity and is known to an absolute error tolerance of 10^{-3}. The binary representation of ΔP_1 then has about 10 base 2 digits of information. This number is obtained by solving the equation $2^{-n} = 10^{-3}$ for n. The other 14 base 2 digits (assuming a 24 bit mantissa) contain roundoff noise. If the first 13 base 2 digits of two $o(1)$ pressure offsets ΔP_1, ΔP_2 agree, then the subtraction $\Delta P_1 - \Delta P_2$ will result in loss of all significant digits. It is therefore important to know when a subtraction will result in total cancellation.

The number of *decimal* digits lost in the subtraction $\Delta P_1 - \Delta P_2$ can be estimated using

$$n_{\text{lost}} = \log_{10} \frac{\max(1, \Delta P_1, \Delta P_2)}{|\Delta P_1 - \Delta P_2|} \ . \tag{19}$$

If $\log_{10}$ is replaced by $\log_2$ then n_{lost} estimates the number of *binary* digits lost. Suppose that ΔP_1 and ΔP_2 each have m significant digits and suppose that

$$n_{\text{lost}} < m \ .$$

Substituting equation (19) and exponentiating, it follows that

$$|\Delta P_1 - \Delta P_2| > \epsilon_p \max(1, \Delta P_1, \Delta P_2)$$

where $\epsilon_p = 10^{-m}$. This inequality gives a criterion for deciding if $\Delta P_1 - \Delta P_2$ has any accurate digits. To smoothly damp the roundoff noise present in the computation $\Delta P = \Delta P_1 - \Delta P_2$, ΔP can be replaced by

$$\Delta \hat{P} = \Delta P \left(1 - e^{-\frac{\Delta P}{\epsilon_{\text{cut}}}}\right)$$

where ϵ_{cut} defined by

$$\epsilon_{\text{cut}} = 10\epsilon_p \max(1, \Delta P_1, \Delta P_2) .$$

When ΔP is large relative to ϵ_{cut}, ΔP and $\Delta \hat{P}$ are essentially the same. On the other hand, when ΔP is small relative to ϵ_{cut}, $\Delta \hat{P}$ is essentially zero.

4.1.2 Forced Vent Flow

A forced ventilation system consists of a network of ducts and supply and exhaust fans. In general, the volume flow rate delivered by a fan depends on the pressure drop across the fan. This relation is known as the fan curve. Numerical problems can occur even in simple systems consisting of one fan with a constant volume flow rate and one duct connecting two rooms. For example, consider a fan exhausting a room where sources of enthalpy such as fires and natural vents are small relative to the enthalpy removed by the exhaust fan. Numerical problems occur when the fan is modeled by removing flow at a single elevation. In this case, the fan is either in the lower layer or the upper layer. While the fan is in the lower layer, the layer interface drops since the fan is "stronger" than the fire. Similarly, while the fan is in the upper layer the layer interface rises. This problem is ill-conditioned since the total enthalpy flow rate, $(\dot{q}_L + \dot{q}_U)$, is a discontinuous function of layer height. When the layer interface is near the fan elevation, small changes in the layer height produce large changes in the layer height derivative. A differential equation solver detecting this jump in the layer height derivative (or layer volume derivative) will reduce the time stepsize drastically to try and track the layer height.

The solution to this problem is to model the fan by withdrawing air over a finite but nonzero height in the wall. When the layer is within a forced fan duct flow is withdrawn from both layers. This results in smoother transition in the layer height derivative.

4.1.3 Fire Plumes

A fire deposits energy denoted $\dot{q}^L_{\text{fire}}$, $\dot{q}^U_{\text{fire}}$ into the lower and/or upper layer. This lower layer contribution is set to zero in most plume models. A fire also transfers mass and energy from the lower layer to the upper layer by entraining a portion of the lower layer into the hot plume which in turn flows due to buoyancy into the upper layer. Let the energy entrainment terms be denoted $\dot{q}^L_{\text{ent}}$ and $\dot{q}^U_{\text{ent}}$. The total energy contribution to the lower and upper layers due to both mechanisms is then

$$\dot{q}^L_{\text{plume}} = \dot{q}^L_{\text{fire}} + \dot{q}^L_{\text{ent}}, \tag{20}$$

$$\dot{q}^U_{\text{plume}} = \dot{q}^U_{\text{fire}} + \dot{q}^U_{\text{ent}} . \tag{21}$$

It is usually assumed that all of the energy taken out of the lower layer due to entrainment is added back into the upper layer so that $\dot{q}^L_{\text{ent}} = -\dot{q}^U_{\text{ent}}$. Therefore

$$\dot{q}^L_{\text{plume}} + \dot{q}^U_{\text{plume}} = \dot{q}^L_{\text{fire}} + \dot{q}^U_{\text{fire}} . \tag{22}$$

To avoid a catastrophic cancellation, a plume model should calculate the total energy contribution using equation (22) rather than summing the terms as calculated in equations (20) and (21). This is especially important when $\dot{q}^L_{\text{ent}}$ is large compared to $\dot{q}^U_{\text{fire}}$ or $\dot{q}^L_{\text{fire}}$. The total energy term is required by zone fire models which use the pressure differential equation or its equivalent.

4.2 Numerical Characteristics of the Zone Fire Modeling Differential Equations

In Section 3.2 the numerical properties of individual differential equations were discussed. It is important, however, to understand how these equations behave together as a system in order to implement accurate and efficient algorithms for their solution. The key property that zone modeling differential equations possess is called stiffness. Differential equation solvers not taking this property into account will at best be grossly inefficient and at worst give wrong answers.

4.2.1 Comparison of the Physical and Numerical Behavior of the Differential Equations

Stiff systems of ordinary differential equations (ODE's) are an important class of problems that can occur when the modeled phenomenon possesses characteristic time scales that vary by several orders of magnitude. Physically, the system of ODE's used in zone fire modeling are stiff because the pressure adjusts to changing conditions much faster than other quantities such as upper and lower layer temperatures or layer heights.

The numerical difficulties encountered because of stiffness can not be avoided by exchanging the pressure equation for some other equation such as temperature, density, or internal energy. As shown in Table 2, each zone modeling differential equation contains a $\frac{dP}{dt}$ term. If the pressure is computed using one of the approximations discussed in sections 3.3.2 or 3.3.3 and $\frac{dP}{dt}$ is removed from the modeling differential equations, the resulting approximate ODE's are not stiff and a standard nonstiff solver may be used. However, the class of problems that can be solved is reduced since large pressure fluctuations can not be modeled properly.

The curious aspect of stiff ODE's is that the solution appears to be changing slowly and yet the computational costs of computing the solution are enormous when using standard nonstiff algorithms such as Runge-Kutta methods or predictor-corrector methods such as Adams-Bashforth. The question then is why does it cost so much to solve a problem whose solution changes slowly? To maintain stability, a nonstiff solver must use a stepsize that is small enough to track the part of the solution corresponding to the shortest time scale even when this solution component decays rapidly to some quasi-steady value. This stepsize is much smaller than required to accurately track the desired part of the solution which corresponds to one of the longer time scales. So for stiff problems the choice of stepsize is dominated

by considerations of stability, not accuracy. These concepts and their significance will be discussed in section 4.2.2.

Stiff differential equation solvers are expensive to use. This is because a nonlinear set of simultaneous equations involving the solution variables must be solved at each time step. Newton's method is typically used for this and requires the solution of a system of linear equations at each Newton iteration. Consequently, it is inefficient to use stiff methods for nonstiff problems. Stiff methods work because their choice of stepsize is dominated by considerations of accuracy, not stability. As a result, they allow larger time steps to be taken than nonstiff methods. Even though the work per step is greater, the number of steps is sufficiently smaller that the total work is smaller.

Several approaches can be taken to decide whether a given system of differential equations is stiff. One approach is to let the solver decide. Many modern ODE solvers have built-in heuristics to determine if a problem is stiff. To analyze the numerical character of the ODE's in the zone fire model CCFM.VENTS, DEPAC was used. DEPAC is a set of three ODE solvers DERKF, DEABM, and DEBDF designed by Shampine and Watts. DERKF is a fifth order variable step size Runge-Kutta code. It can be used for nonstiff and mildly stiff ODE's when derivative evaluations are not expensive. It should not be used for high accuracy, nor for answers at a great many points. DEABM is a variable order, variable step size Adams code. It can be used for nonstiff and mildly stiff ODE's when high accuracy is required. DEBDF is a variable order, variable step size backward differentiation formula code which can be used on stiff ODE's when moderate accuracy is required. DERKF and DEABM attempt to determine when their use is not suitable by performing diagnostics for stiffness. When used in CCFM.VENTS both DERKF and DEABM reported that the problem might be stiff. This occurred for a wide range of fire scenarios.

A second approach is to analyze the Jacobian of the right hand side of the system of ODE's. A subroutine, EIGF, was written to approximate this Jacobian and its eigenvalues. The characteristic local time scales of the solution were determined from the eigenvalues. It was found that the characteristic time scales for the pressure variables were much smaller than for the other solution variables. For some problems, the variation in time scales was over five orders of magnitude. As a result of this analysis, the stiff ODE solver DEBDF was chosen.

4.2.2 Some Numerical Considerations of Solving Stiff Differential Equations

A zone fire modeling initial value problem can be expressed in vector form as follows. At time $t > t_0$ find the value to the solution y to

$$\begin{aligned} \frac{dy}{d\tau}(\tau) &= f(\tau, y) \\ y(t_0) &= y_0 \end{aligned} \tag{23}$$

where y and f are a real valued N-vector functions. The vector y contains the solution variables chosen and the right hand side f gives the rate at which these variables change with time.

Modern ODE solvers use a variable stepsize strategy; that is, the solver takes a number of steps to integrate (23) from t_0 to t, but these steps are not generally all of the same size. Suppose that K steps are taken and approximations y_i are found to $y(\tau)$ at $\tau = t_1, \ldots, t_K = t$. Consider the "local" initial value problem: find the solution at t_i to

$$\begin{aligned} \frac{dz}{dt}(t) &= f(z,t) \\ z(t_{i-1}) &= y_{i-1} \, . \end{aligned}$$

The local error is defined by $z(t_i) - y_i$, for $i = 0, \ldots, K$. Most solvers control the local error at each step. The true or global error is defined by $e_i = y(t_i) - y_i$, $i = 0, \ldots, K$. Note that $e_K = y(t) - y_K$ is the error in the approximate solution at time t. Consider the decomposition

$$e_i = (y(t_i) - z(t_i)) + (z(t_i) - y_i) \, . \tag{24}$$

A linearization shows that to terms of first order

$$(y(\tau) - z(\tau))' = f_y(\tau, y(\tau))(y(\tau) - z(\tau)) \, .$$

The $N \times N$ matrix f_y is called the Jacobian matrix. If the initial value problem (23) is stable (all the eigenvalues of f_y have negative real parts), then the first term in (24) will not grow with index i. The second term is the local error and will be dependent on the accuracy and stability of the numerical method employed. If the numerical method is stable, then the growth of e_i with i will be at most linear (and not exponential).

Let $\tilde{y}_i$ denote the numerical solution determined from exact data $y(t_{i-1}), y(t_{i-2}), \ldots$. The following alternative decomposition of the global error is also useful:

$$e_i = \overbrace{(y(t_i) - \tilde{y}_i)}^{\text{local truncation error}} + \overbrace{(\tilde{y}_i - y_i)}^{\text{propagation error}} \, . \tag{25}$$

The propagation error can be written as a product of an amplification factor and the error term e_{i-1}. If the numerical method is stable, the absolute value of the amplification factor will be less than or equal to one and the propagation error term will not grow with i. The local truncation error depends on the accuracy of the numerical method. These errors at $i = 2$ are illustrated in Figure 8.

There is no one definition of stiffness that is universally applied to initial value problem (23). One that is commonly applied is the following (see [17]). **Definition:** The initial value problem (23) is called stiff (oscillatory) if the eigenvalues, $\lambda_j = u_j + iv_j$, $j = 1, \ldots, N$ of the Jacobian, f_y, satisfy

$$u_j < 0, \; j = 1, \ldots, N \, ,$$

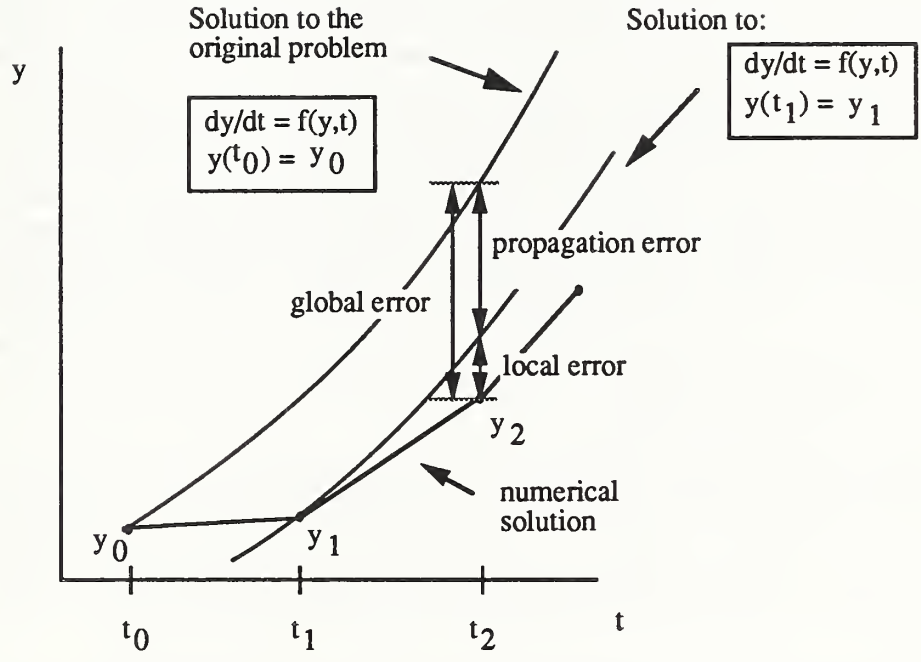

Figure 8: Differential Equation Error Terms

and

$$\max_{1\le j\le N}(|u_j|) \gg \min_{1\le j\le N}(|u_j|);.$$

A physical interpretation of stiffness, which is imprecise mathematically, is to say that the initial value problem (23) is stiff if the process being modeled possesses multiple time scales which vary widely. As has been discussed earlier, this is the case with zone fire models because pressures equilibrate much quicker than do layer volumes, masses, densities, or temperatures. Yet another (imprecise) way to characterize stiffness would be to say a problem is stiff if explicit solvers such as DERKF or DEABM cannot handle it. ODE solvers are either explicit or implicit. Implicit solvers are generally stable for a much wider range of stepsizes than are explicit solvers. If a solver were capable of performing the decomposition (25), it would regard a problem as stiff if the stepsize required to maintain stability (keep the propagation error term from growing) is much smaller than that required to keep the local truncation error term small. In this case, the stepsize is said to be restricted by stability, rather than by accuracy. Unfortunately, there is no practical way to make this characterization quantitative. What is estimated at each step is the local error. Stepsize choice is based on this estimate. Diagnostic messages are issued when the stepsize is extremely small or the number of steps taken has exceeded a predetermined bound. These messages typically suggest that the user try a stiff solver such as DEBDF.

Implicit methods are more suitable for stiff problems. In the case of the backward differentiation formulas used in DEBDF, the stepsize choice for a stiff problem will generally be restricted by accuracy, not by stability. Stepsize can be increased at a fixed accuracy level by increasing the order (of accuracy) of the method. DEBDF provides formulas of orders one through five, all of which have excellent stability properties.

The simulation time interval $[t_0, t]$ for a zone fire model can be broken into two types of subintervals, stiff transient and stiff. In the stiff transient subintervals, the pressure is rapidly moving toward a quasi-steady state value. The simulation generally begins with a stiff transient. Each time layer height passes a vent boundary or the fire output takes a jump, a new stiff transient begins. During a stiff transient, stepsize will generally be small because it will be restricted by accuracy; that is, a small stepsize will be required to accurately track the rapidly changing pressure. Nonstiff solvers can generally integrate over the stiff transient time subinterval. Outside of these very short time intervals, a stiff solver is required. Since there is a large overhead associate with switching solvers, it is more efficient to use the stiff solver throughout the computation.

5 Conclusions

Tools were presented for analyzing the numerical behavior of fire algorithms. These tools show that natural vent flow is particularly susceptible to numerical problems due to the loss of significant digits that can occur when computing pressure differences. One approach for reducing this problem is to modify the subtraction $|\Delta P_1 - \Delta P_2|$ in order to damp out unwanted error that occurs when ΔP_1 and ΔP_2 are nearly equal.

Many model formulations can be derived from the basic mass and energy conservation equations some of which are analytically equivalent; that is, the equations in one formulation can be converted into those of another using the ideal gas law and definitions of density and internal energy. These formulations, however, are not equivalent numerically. When pressure changes are significant, the pressure equation should be included in a multi-room model. The significance of pressure changes can be evaluated in terms of flow through a vent since vent flow is quite sensitive to small pressure changes. When it is valid to assume that pressures are constant, the modeling differential equations can be simplified by setting $\frac{dP}{dt}$ to zero. The resulting ODE's have the added advantage that they are easier to solve since they are no longer stiff.

Nomenclature

a, b	miscellaneous constants used in the equilibrium pressure relation, equation (16)
A_{vent}	area of vent m^2
b	base used in the Wilkinson floating point model
$c_a(f)(c_r(f))$	absolute (relative) condition number of f
$c_p(c_v)$	specific heat at constant pressure (volume) kJ/kg K
c_{vent}	vent coefficient, usually has a value of around .68
e_i^T	absolute error of temperature K
e_i	global error of the differential equation solution
E	internal energy kJ
E_{ref}	reference internal energy kJ
e	Exponent in Wilkinson floating point model
$e_{\min}$ ($e_{\max}$)	smallest (largest) exponent in the Wilkinson floating point model
f	a vector valued function which is the right hand side of the differential equation to be solved
f_y	Jacobian of the right hand side vector function f
fl()	The operator fl maps a real number to its floating point representation.
h	Step size
K	Vent flow coefficient kJ$\sqrt{\text{m/kg}}$
m	mass kg
n_{lost}	number of digits (decimal) lost due to cancellation in a subtraction
P	absolute pressure Pa
P_{cut}	pressure below which flows (through vents) are deemed to be negligible Pa
P_{ref}	reference pressure Pa
P_{amb}	ambient pressure usually about 101325 Pa
$\dot{q}_L, (\dot{q}_U)$	total enthalpy flow rate into the lower (upper) layer W
$\dot{q}_{\text{fire}}$	energy release rate of a fire W
$\dot{q}_{\text{vent}}$	energy flow rate through a vent W
R	universal ideal gas constant $\frac{m^2}{s^2 K}$

T	temperature K
t_0	initial time s
$t_{.99}$	time for pressure in a room to reach 99 percent of its final value s
T_{vent}	temperature of gas flowing through vent K
t	number of digits in the Wilkinson floating point model
t	independent variable, time, in the differential equations
T_{amb}	ambient temperature
V	volume m^3
y	layer height m
y	solution of the differential equation
y_i	numerical solution of a differential equation at step i
y_0	solution of a differential equation at time t_0
ΔP	pressure drop across a vent Pa
$\Delta \hat{P}$	pressure drop across a vent modified to account for catastrophic cancellation errors Pa
ΔP	Pressure offset satisfying $P = P_{ref} + \Delta P$ Pa
$\Delta P_1, (\Delta P_2)$	pressure rise above floor pressure in room 1 (2) Pa
ΔP_{cut}	pressure below which flow is insignificant numerically Pa
ΔP_∞	Equilibrium pressure Pa
ϵ_m	machine epsilon, smallest number added to 1 on the computer that gives a result different than 1.
ϵ_p	error tolerance of pressure variables used in the calculation vent flows
γ	ratio of specific heats c_p, c_v
ρ	density kg/m^3
ρ_{amb}	ambient density kg/m^3
ρ_{vent}	density of gas flowing through vent
τ	time s

References

[1] J. Quintiere. Fundamentals of enclosure fire 'zone' models. *Journal of Fire Protection Engineering*, 1:99–119, 1989.

[2] C. William Gear. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice Hall, Englewood Cliffs, New Jersey, 1971.

[3] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis.* Springer-Verlag, New York, 1980.

[4] 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.

[5] Walter W. Jones and Richard D. Peacock. Technical reference guide for FAST version 18. Technical Note 1262, National Institute of Standards and Technology, 1989.

[6] 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.

[7] Henri Mitler. Mathematical modeling of enclosure fires. Internal Report 4294, National Institute of Standards and Technology, 1990.

[8] L. Y. Cooper. A mathematical model for estimating available save egress time in fires. *Fire and Materials*, 6:135, 1982.

[9] L. Y. Cooper. The development of hazardous conditions in enclosures with growing fires. *Combustion Science and Technology*, 33:279–297, 1983.

[10] T. Tanaka. A model of multiroom fire spread. Internal Report 2718, National Bureau of Standards, 1983.

[11] T. Tanaka. A model of multiroom fire spread. *Fire Science and Technology*, 3:105–121, 1983.

[12] Leonard. Y. Cooper. Estimating the environment and the response of sprinkler links in compartment fires with draft curtains and fusible link-actuated ceiling vents-part i: Theory. Internal Report 3734, National Institute of Standards and Technology, 1988.

[13] William D. Davis and Leonard. Y. Cooper. Estimating the environment and the response of sprinkler links in compartment fires with draft curtains and fusible

link-actuated ceiling vents-part ii: User guide for the computer code LAVENT. Internal Report 4122, National Institute of Standards and Technology, 1989.

[14] Ronald G. Rehm and Howard R. Baum. The equations of motion for thermally driven buoyant flows. *Journal of Research of the National Bureau of Standards*, 83:297–308, 1978.

[15] Linda R. Petzold. A description of DASSL: A differential/algebraic system solver. technical report 8637, Sandia National Laboratories, 1982.

[16] 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.

[17] S. O. Fatunla. *Numerical Methods for Initial Value Problems in Ordinary Differential Equations.* Academic Press, New York, New York, 1988.

[18] J. H. Wilkinson. Error analysis of floating-point computation. *Numer. Math.*, 2:219–340, 1960.

[19] J. H. Wilkinson. *Rounding Errors in Algebraic Processes.* Wiley, New York, 1963.

[20] J. Coonen, W. Kahan, J. Palmer, T. Pittman, and D. Stevenson. A proposed standard for floating point arithmatic. *ACM SIGNUM Newsletter*, pages 4–12, October 1979.

[21] W. Kahan and J. Palmer. On a proposed floating point standard. *ACM SIGNUM Newsletter*, pages 13–21, October 1979.

[22] ANSI/IEEE. IEEE standard for binary floating-point arithmetic. *IEEE Std. 754-1985*, 1985.

[23] W. J. Cody. Analysis of proposals for the floating-point standard. *Computer*, 14(3):63–69, 1981.

[24] David Stevenson. A proposed standard for binary floating-point arithmetic. *Computer*, 14(3):51–62, 1981.

[25] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software.* Prentice Hall, Englewood Cliffs, New Jersey, 1989.

A A Description of Floating Point Arithmetic

A model of floating point arithmetic consists of a description of how floating point numbers are represented (word length and bits for sign, exponent, and mantissa), how operations with these numbers behave (error properties), and what happens when an arithmetic fault occurs such as division by zero.

Properties of the real number line such as closure, distributivity and associativity do not hold for "computer" numbers. Consequently, two algorithms may be equivalent analytically but not the same numerically. The methods used for implementing equations that arise in fire modeling are therefore important.

An early problem in scientific computation was the lack of a uniform floating point environment. Each computer manufacturer had their own method for representing and using floating point numbers. As a result, scientific programs running on different computers would occasionally produce significantly different numerical results.

Wilkinson, considered the father of floating point error analysis describes [18, 19] a simple model of floating point arithmetic. To address the problem of lack of uniformity in floating point implementations, Kahan [20, 21] extended the Wilkinson model and proposed a standard for floating point arithmetic. Kahan's proposal along with features of several other proposals became the IEEE Standard for Binary Floating Point Arithmetic [22, 23, 24].

The IEEE Standard uses three formats to represent floating point numbers, single, double and extended precision. Single uses 32 bits, double uses 64 bits. The format of extended arithmetic is left open. Two special bit patterns designated NaN (not a number) and **infinity** are used to represent the results of special operations. When the result of a mathematical operation is undefined such as zero divided by zero a result of NaN is returned. Some compilers initialize variables with NaN to detect the use of undefined variables. The standard also specifies that the operations addition, subtraction, multiplication, division, square root, remainder and comparison be provided. The **default rounding** behavior takes the result of an operation as if correct to infinite precision and rounds it to the nearest machine number. Rounding the result toward or away from zero is also provided. Rounding towards zero is sometimes called chopped or truncated arithmetic.

The real number line is modeled on the computer as a **finite** collection of floating point numbers where each floating point number consists of a mantissa or fraction, an exponent, and a sign. These numbers are sometimes called model or machine numbers. Wilkinson [18, 19] parameterizes the mantissa and exponent in terms of the four parameters: b, t, $e_{\min}$ and $e_{\max}$ where b represents the base ($b = 2$ for binary, $b = 16$ for hexa-decimal), t is the precision or number of digits in the mantissa, $e_{\min}$ is the smallest exponent and $e_{\max}$ is the largest. Values of these parameters for various computers are presented in Table 7.

The IEEE Standard is a model of *binary* floating point arithmetic. Most comput-

Table 7: Single Precision Floating Point Characteristics for Various Computers

Computer	b	t	$e_{\min}$	$e_{\max}$	$\epsilon_m = b^{1-t}$
Univac 1100	2	27	-128	127	1.49×10^{-8}
PDP-11/Vax 11/780	2	24	-127	127	1.19×10^{-7}
Cray - 1	2	47	-8189	8190	1.42×10^{-14}
Apple Macintosh, IBM PC	2	24	-128	127	1.19×10^{-7}
IBM 360/370, Concurrent 7/32	16	6	-64	63	9.54×10^{-7}
CDC 200 (Cyber 205)	2	47	-28625	28718	1.42×10^{-14}
IEEE 754 Standard	2	24	-128	127	1.19×10^{-7}

ers built today use this standard or make it available as an option. The Apple Macintosh, the IBM PC and clones, and UNIX workstations from Sun, Silicon Graphics, DEC, IBM, and HP all use the IEEE Standard for floating point arithmetic. Implementations differ from vendor to vendor, but these vendors claim to adhere to the IEEE Standard. Other manufacturers such as Convex make it available as an option. The Cray series of computers is a notable exception. Programs that run on computers that use IEEE Standard arithmetic should give similar floating point results.

The floating point numbers of the Wilkinson model, on which the IEEE Standard is based, can be generated from the four parameters b, t, $e_{\min}$, $e_{\max}$ using the following representation

$$\pm (b_1 \times b^0 + b_2 \times b^{-1} + ... + b_t \times b^{1-t}) \times b^e \tag{26}$$

where

$$1 \leq b_1 \leq b - 1 \tag{27}$$

$$0 \leq b_i \leq b - 1, \; i = 2, \ldots, t \tag{28}$$

$$e_{\min} \leq e \leq e_{\max}. \tag{29}$$

The numbers $b_1, \ldots, b_t$ form the mantissa and e denotes the exponent. If the first bit of the mantissa, b_1, is never 0 then the set of floating point numbers is said to be **normalized**. For normalized **binary** systems the first bit is usually not stored (since it is always 1). Then 32 bits can then be used to represent 1 sign bit, an 8 bit exponent and a 24 bit mantissa.

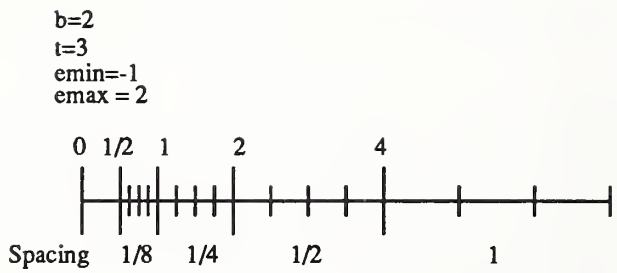

Figure 9: Example Floating Point Model

The real number line and the floating point numbers which approximate it for $b = 2$, $t = 3$, $e_{\min} = -1$, and $e_{\max} = 2$ are depicted in Figure 9. The set of positive floating point numbers can be easily counted. For each of the $e_{\max} + 1 - e_{\min}$ values of the exponent e, there are $(b-1)b^{t-1}$ positive floating point numbers. For the example of Figure 9, there are 33 floating point numbers; 16 are positive, 16 are negative and one is zero. These numbers are not evenly spaced. Numbers with small exponents are near the origin and closely spaced, while numbers with large exponents are far from the origin and widely spaced. In fact, the *absolute spacing* between consecutive positive floating point numbers with exponent e is $b^e b^{1-t}$ which depends on e. On the other hand, the largest *relative spacing* for positive floating point numbers with exponent e is b^{1-t} which is independent of e. Here, the spacing has been divided by b^e, the smallest positive floating point number with exponent e.

One particular parameter of interest, machine epsilon, can be derived from the base b and the precision t. The smallest floating point number that can be added to 1.0 so that the result is a floating point number larger than 1.0 is called **machine epsilon**. If default rounding is used, this number is $\epsilon_{\mathrm{m}} = \frac{b^{1-t}}{2}$ since b^{1-t} is the spacing between consecutive positive floating point numbers with exponent 0. For the floating point system illustrated in Figure 9, machine epsilon is 1/8.

The error in representing a nonzero real number, x, on the computer can be expressed in terms of machine epsilon. This error is sometimes called **rounding error**. Suppose that x falls in the range of floating point numbers; that is, $b^e \leq |x| \leq (1 - b^{-t})b^{e+1}$ where $e_{\min} \leq e \leq e_{\max}$. Using default rounding, the maximum distance (or error) between x and the closest floating point number is $b^e \epsilon_{\mathrm{m}}$. The relative distance between x and the closest floating point number is less than or equal to ϵ_{m}. If $|x| < b^{e_{\min}}$, x is said to **"hard" underflow**. The IEEE Standard also defines a "soft" underflow by allowing the floating point numbers with exponent $e_{\min}$ to be denormalized; that is, b_1 is allowed to be zero. In this case **"soft" underflow**

occurs if $|x| < b^{e_{\min}+1-t}$. If $|x| > (1-b^{-t})b^{e_{\max}+1}$, x is said to **overflow**. Compilers often provide options for dealing with underflows and overflows. The default is typically to set underflows to zero and to stop execution when overflows occur.

A standard approach for obtaining the correct floating point characteristics of a computer is to use the three Fortran function subprograms, I1MACH, for integer parameters, R1MACH, for real parameters, and D1MACH, for double precision parameters. These routines were originally written at Bell Labs by P. A. Fox, A. D. Hall, and N. L. Schryer and are currently widely available [25]. Many scientific computer applications written in Fortran now have these three function subprograms as part of their distribution. When such an application is installed on a particular computer, the installer must uncomment the data statements in these routines that correspond to the appropriate computer before compilation.

Numerical errors produced by an algorithm arise from errors in representing the data, errors in floating point arithmetic, and amplification of errors due to the sensitivity of various parts of the algorithm to perturbations. The notation fl() is often used to indicate the floating point result of a machine computation. If z is a real number in the range of the floating point numbers, then

$$\mathrm{fl}(z) = z(1+\epsilon),\ \epsilon \le \epsilon_{\mathrm{m}}$$

This means that the relative error in representing z on the machine is bounded by machine epsilon.

Let x and y denote nonzero floating point numbers. The IEEE Standard requires that $\mathrm{fl}(x+y)$ equal the actual result, $x+y$, rounded to the nearest floating point number, and similarly for subtraction, multiplication, and division. In practice, this is accomplished by using extended length registers for the arithmetic. The extra bits in these registers are called "guard" bits. The number of guard bits used varies with vendor. It follows that

$$\mathrm{fl}(x \pm y) = (x \pm y)(1+\epsilon),\ \epsilon \le \epsilon_{\mathrm{m}} \tag{30}$$

$$\mathrm{fl}(xy) = xy(1+\epsilon),\ \epsilon \le \epsilon_{\mathrm{m}} \tag{31}$$

$$\mathrm{fl}(x/y) = (x/y)(1+\epsilon),\ \epsilon \le \epsilon_{\mathrm{m}}. \tag{32}$$

Suppose on the other hand that x and y are the result of a machine computation whose exact answer is given by the nonzero numbers $\tilde{x}$ and $\tilde{y}$. Let ϵ_x and ϵ_y denote the relative errors in approximating $\tilde{x}$ and $\tilde{y}$ by x and y; that is, $x = \tilde{x}(1+\epsilon_x)$ and $y = \tilde{y}(1+\epsilon_y)$. Then substituting these relations for x and y into the right hand side of equations (30), (31), and (32) it follows that

$$\frac{|\mathrm{fl}(x \pm y) - (\tilde{x} \pm \tilde{y})|}{|\tilde{x} \pm \tilde{y}|} \le \frac{|\tilde{x}|}{|\tilde{x} \pm \tilde{y}|}|\epsilon_x|(1+\epsilon_{\mathrm{m}}) + \frac{|\tilde{y}|}{|\tilde{x} \pm \tilde{y}|}|\epsilon_y|(1+\epsilon_{\mathrm{m}}) + \epsilon_{\mathrm{m}} \tag{33}$$

$$\frac{|\mathrm{fl}(xy) - \tilde{x}\tilde{y}|}{|\tilde{x}\tilde{y}|} \le |\epsilon_x| + |\epsilon_y| + |\epsilon_x\epsilon_y| + |(1+\epsilon_x)(1+\epsilon_y)|\epsilon_{\mathrm{m}} \tag{34}$$

$$\frac{|\mathrm{fl}(x/y) - \tilde{x}/\tilde{y}|}{|\tilde{x}/\tilde{y}|} \le \left|\frac{\epsilon_x}{1+\epsilon_y}\right| + \left|\frac{\epsilon_y}{1+\epsilon_y}\right| + \left|\frac{1+\epsilon_x}{1+\epsilon_y}\right|\epsilon_{\mathrm{m}}\,. \tag{35}$$

Equations (34) and (35) show that the error in x and y is not amplified by multiplication or division. Also, equation (33) shows that if cancellation does not occur in $\tilde{x} \pm \tilde{y}$, then again the error in x and y is not amplified. But if cancellation does occur in $\tilde{x} \pm \tilde{y}$ and this number is close to zero, then the errors in x and y can be greatly amplified. This condition is usually referred to as "catastrophic cancellation." Whenever possible, such cancellations should be avoided in the construction of an algorithm although it is often difficult to anticipate where they will occur.

Equations (33) and (34) can be used to show that the equivalent expressions $a(b - c)$ and $ab - ac$ are also equivalent numerically (both expressions have the same error bounds). In fire modeling this may arise when computing enthalpy differences. There is no numerical advantage to computing an enthalpy difference using $c_p T(\dot{m}_1 - \dot{m}_2)$ instead of $c_p T\dot{m}_1 - c_p T\dot{m}_2$.

Figure 10 illustrates the steps involved in multiplying two numbers on the computer using the example floating point model given in Figure 9. For this example,$\epsilon_x = .05$, $\epsilon_y = .1$, $\epsilon_{\mathrm{m}} = 1/8$ and

$$|\mathrm{fl}(xy) - \tilde{x}\tilde{y}|/|\tilde{x}\tilde{y}| \approx .1453$$

which is indeed smaller than

$$|\epsilon_x| + |\epsilon_y| + |\epsilon_x\epsilon_y| + |(1 + \epsilon_x)(1 + \epsilon_y)\epsilon_{\mathrm{m}}| \approx .2994 \ .$$

Some vendors provide additional extended length registers for storage of intermediate results. In this case, intermediate results are not rounded to working precision before they are used.

The issue of loss of accurate digits due to cancellation arises in the choice of unknowns in a zone fire model. The presence of a fire in a building causes a pressure change in each room. This pressure change is generally small (a few Pascals) compared with the ambient pressure P_{amb} (usually one atmosphere or about 10^5 Pascals). The pressure differential across a door or other opening drives the transfer of mass and energy between connected rooms. Accurate mass and energy transfer computations require accurate pressure differentials as input. Here is a case where reasonable accuracy is desirable in a computation which is subject to catastrophic cancellation. Let a and b denote pressure changes in adjoining rooms. If the total pressure in each room is chosen as an unknown, then the pressure differential will be computed as $(P_{\mathrm{amb}} + a) - (P_{\mathrm{amb}} + b)$. On the other hand, if the pressure rise in each room is chosen as an unknown, the pressure differential will be computed as $a - b$. Noting that generally $|a|, |b| \ll P_{\mathrm{amb}}$, examination of equation (33) shows that this second choice is numerically superior to the first.

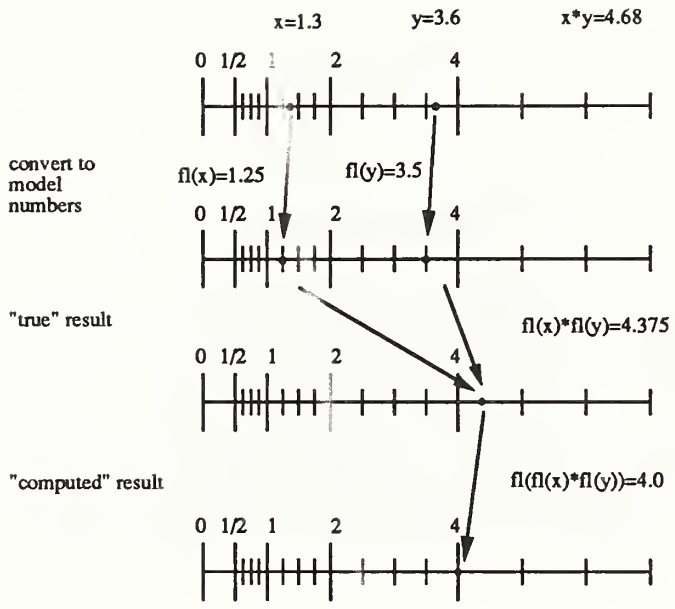

Figure 10: Sample Floating Point Multiplication

NIST-114A
(REV. 3-90)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NISTIR 4763

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
March 1992

4. TITLE AND SUBTITLE
Analyzing and Exploiting Numerical Characteristics of Zone Fire Models

5. AUTHOR(S)
Glenn P. Forney and William F. Moss

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)
U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

10. SUPPLEMENTARY NOTES

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

In order to design robust and stable zone fire modeling algorithms, the numerical properties of computer arithmetic and the modeling differential equations must be understood. This report examines some of these properties and provides tools for their analysis. Many sets of differential equations for zone fire modeling can be derived using the conservation of mass and energy. A comparison between various possible formulations is made in terms of numerical properties. One property that many formulations possess is the presence of multiple time scales. Pressures equilibrate much faster than other quantities such as density and temperature. Numerically, this property is known as stiffness. Stiffness, in the context of fire modeling, and numerical methods for handling it are discussed.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)
Computer modeling, fire modeling, numerical analysis, zone models

13. AVAILABILITY

X	UNLIMITED
	FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).
	ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.
X	ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
44

15. PRICE
A03

ELECTRONIC FORM