A Numerical Model of the Late (Ascending) Stage of a Nuclear Fireball

I. E. Galbally, P. C. Manins, L. Ripari and R. Bateup

DIVISION OF ATMOSPHERIC RESEARCH TECHNICAL PAPER No. 16
COMMONWEALTH SCIENTIFIC AND INDUSTRIAL RESEARCH ORGANISATION, AUSTRALIA 1987

A Numerical Model of the Late (Ascending) Stage of a Nuclear Fireball

By I. E. Galbally, P. C. Manins, L. Ripari and R. Bateup

Division of Atmospheric Research Technical Paper No. 16

Commonwealth Scientific and Industrial
Research Organisation, Australia
1987

A NUMERICAL MODEL OF THE LATE (ASCENDING) STAGE OF A NUCLEAR FIREBALL

I.E. Galbally, P.C. Manins, L. Ripari and R. Bateup

ABSTRACT

A numerical model of the late ascending stage of a nuclear fireball is presented. The model is based on five equations covering the conservation of mass including entrainment, buoyancy, radiative loss, the energy balance of the fireball and the velocity distance relationship. The ideal gas law is used to relate pressure, volume and temperature and Eriksson's (1971) free energy minimization scheme is used to calculate the molecular composition and enthalpy of the fireball air. The model simulations of the time dependent behavior of selected parameters and the final rise heights compare favourably with available data from mid latitude explosions. A simplification of this model, suitable only for physical calculations, is discussed in an appendix.

CONTENTS

1. INTRODUCTION

Recently there has been renewed interest in the study of nuclear explosions and their impact on the atmosphere (Ambio 1982). There have been new discoveries about the effects of post explosion fires releasing pollutants into the atmosphere, and the more well known questions concerning nitrogen oxides, radioactivity, electromagnetic pulse, flash burning and shock wave damage have been reexamined (Pittock et al. 1986). However while these processes are generated by the nuclear fireball very little is known, within the open literature, about these fireballs.

As an aid to analyses of the environmental impact of nuclear explosions a numerical model of the late, ascending, stage of a nuclear fireball has been developed. This model commences after the shock wave has separated from the fireball and thus cannot address questions concerning the shock wave. The production of nitrogen oxides, radiation flash intensity, and height of rise of the radioactive debris can all be examined with the model presented here. Furthermore other questions concerning fireball composition can be addressed with this model.

Complex numerical models do exist (e.g. Brode and Bjork, 1960) and these describe nuclear explosions in great detail. However, according to Bethe (1964) and Brode (1968), only the early and mid time histories of fireball development - before buoyant rise of the hot vapour begins - are considered by these models. The behaviour at late times has apparently been studied at length (Glasstone and Dolan 1977, Fig.10,p158) but no information about the models which provide the basis for presented results is forthcoming.

As far as we know, the only accessible report dealing with the rising fireball is by Waltman (1975) and the model presented is singularly unsuccessful in simulating observed behaviour. The dynamic model presented here (i.e., the model without the chemical reactions and associated thermodynamic processes) is superficially similar to that presented by Waltman (1975). However when the two models with identical input parameters are compared, widely differing results are obtained. After detailed examination we are still unable to understand the numerical solutions presented in Waltman (1975).

The purpose of this Report is to document the basis of the new model of nuclear fireball rise, to provide a detailed description of the code which implements the model so that others may utilize it, to present limited comparisons of model output with available data, and to demonstrate the sensitivity of the model to various assumptions. It is not intended that the Report should deal with significant new research problems: these are the subject of papers to be presented elsewhere.

2. INITIAL AND AMBIENT CONDITIONS

A near-surface nuclear explosion represents the case where the fireball does not touch the earth's surface and no surface material is entrained into the fireball. This will subsequently be described as a clean fireball and the only constituents present will be those from ambient air.

Another case, not considered here, is that of a surface nuclear explosion. It may be described as a dirty fireball and the composition of the fireball would include materials entrained from the surface during the early stage of the explosion.

2.1 Ambient atmospheric conditions

The ICAO (1964) standard atmospheric conditions used by Manins (1985) are employed here. The surface is taken to be at mean sea level and in mid latitudes (±30° to ±70° latitude) the tropopause is at an altitude of 11km. The ICAO standard atmosphere assumes hydrostatic conditions prevail so

$$\frac{dp}{dz} = - \rho_\infty g \qquad (1)$$

where p is atmospheric pressure, g is acceleration due to gravity, ρ is the density at level z and the subscript denotes ambient conditions. The standard vertical distributions of temperature, pressure and density from the surface to an altitude of 40km as given in ICAO (1964) are used. The composition of the atmosphere is given in Table 1 and has been adapted from Valley (1965) taking into account recent data for CO_2 and water vapour.

The rate of rise of thermonuclear fireballs is so large (up to 200 m s^{-1}) that the influence of ambient winds on growth of the fireball may be ignored. Winds will determine the subsequent spread and transport of fireball material at the ultimate height of rise.

2.2 Energy partitioning following a near-surface nuclear explosion

The energy partitioning in a nuclear explosion is uncertain. The total energy release can be calculated from the amount of fission (and fusion if appropriate) that has taken place. According to Glasstone and Dolan (1977) p.7, for a fission nuclear explosion below 40,000 ft, the energy is partitioned into thermal radiation 35%, air shock 50%, initial nuclear radiation 5%, and residual nuclear radiation 10%. The residual nuclear radiation is not included when the energy yield of a nuclear explosion is stated, e.g., in terms of the TNT equivalent. Hence the yield is 90% of the total fission energy[1] (or 95% of the total fission-fusion energy for a thermonuclear explosion). Thus the energy partitioning as a function of yield, W, for a fission explosion would be thermal radiation 39%, air shock 56%, and initial nuclear radiation 5%. However two details complicate this picture. Firstly there is another source of energy loss, this is the energy dissipated via entrainment and buoyant rise of this fireball and this will be discussed later. Furthermore according to Glasstone and Dolan (1977) p313, the thermal partition for various explosive yields at sea level is 0.35, in contradiction to the 0.39 (39%) derived above.

---

1. The total energy, or 'yield', W, of a thermonuclear explosion is expressed in Mt of TNT equivalent. 1 Mt of TNT equivalent releases 4.18×10^9 MJ of energy.

Table 1: The composition of ambient air

(a) ATMOSPHERIC MIXING RATIO FOR DRY AIR (V/V)

Species	Mixing Ratio (V/V)
N_2	0.780836
O_2	0.209481
CO_2	0.000340
Ar	0.009343

(b) WATER VAPOUR MIXING RATIO AS FUNCTION OF HEIGHT

Height (km)	Mixing Ratio (V/V)	Height (km)	Mixing Ratio (V/V)
0.0	0.010129	2.0	0.006110
4.0	0.003055	6.0	0.001447
8.0	0.000434	10.0	0.000059
12.0	0.000037	14.0	0.000016
16.0	0.000014	18.0	0.000010
20.0	0.000010	22.0	0.000010
24.0	0.000010	26.0	0.000010
28.0	0.000010	30.0	0.000010
31.0	0.000010		

Actual mixing ratio of species (apart from H_2O) is:
Dry mixing ratio x (1 - Water vapour mixing ratio)

While acknowledging that the efficiency of a blast depends (weakly) on bomb design, Brode (1968) in a review of nuclear weapons effects gives a formula for thermal yield/total yield $\{= 0.4 + (0.06W^{\frac{1}{2}})/(1 + 0.5W^{\frac{1}{2}})\}$ where W is in Mt. This indicates that the thermal partition fraction is generally greater than 0.4 and equals 0.44 for a 1 Mt yield explosion. Some early measurements of the thermal radiation from a nuclear detonation (Broido et al. 1953) indicate thermal efficiencies in the range 0.34 to 0.45, decreasing with increasing yield over the range 1 kt to 30 kt.

There is some uncertainty about the thermal yield, but values around 0.40 ± 0.05 are appropriate. Theoretical studies of large explosions (Taylor 1950a) suggest that half the energy is partitioned into the blast wave and the other half remains behind the shock wave as wasted hot air that in the case of nuclear explosions is the fireball.

This energy partitioning is essentially the time integral over the lifetime of the explosion (∿1 min) of the energy used in a particular process as a fraction of the total energy released by the weapon at the time of the explosion. The sequence of events in such an explosion is described. Initially, a nuclear explosion generates a massive quantity of soft x-rays which are immediately absorbed by the surrounding few metres of atmosphere (Glasstone and Dolan 1977). A hot spherical isothermal bubble of dissociated air and bomb products at very high pressure is formed. It expands primarily by radiative transfer. A blast wave propagates outward, shocking surrounding air to high temperature and pressure and for a short period of time this shock wave is opaque. For definiteness, consider a 1 Mt explosion. As witnessed from afar, the temperature of the explosion decreases rapidly but within a few tenths of a second the blast wave has expanded so far away from the directly heated gases (the fireball) that the air in the shock wave is no longer shocked to high enough temperature to be opaque. The fireball is then visible through the shocked air and begins to cool rapidly by emission of radiation to the far field (Brode 1968). The supply of radiant energy can be thought of as coming from a cooling wave propagating into the fireball, and is manifest by a slow shrinking of the radiating surface (Bethe 1964). However, to an observer, the apparent temperature of the fireball at first begins to rise as the much hotter fireball becomes visible through the clearing shocked air. The shock wave is then moving ahead of the central fireball and this stage is termed 'breakaway'. By about 0.9s after detonation of the device the 'second thermal maximum' is reached. The temperature of the fireball stabilizes at a value between 6000K and 10,000K, independent of device yield, before falling again. Bethe (1964) indicates that the second thermal maximum for a near surface explosion occurs when the fireball is at a pressure of 500 kPa. However the shock wave expands quickly and the fireball rapidly relaxes to ambient pressure. Model calculations presented in Brode (1968) show that while a fireball from a 1 Mt explosion in near surface air reaches its second thermal maximum at 0.9 seconds (±20%) the fireball has relaxed to ambient pressure at some time around 1.3 seconds. The maximum radius observed around this time, a_1, (a parameter to be used in the model presented here) will occur when the fireball has relaxed to ambient pressure. At this stage the radiation to the far field is most intense, but it decays rapidly to zero thereafter as the temperature decreases to below a few thousand K. Also at this time the fireball has temporarily ceased to expand and even begins to shrink. Soon after, hydrodynamic forces start to dominate, the fireball begins to rise and grow again by entrainment of ambient air.

We take the time just after the second thermal maximum when the pressure has relaxed to ambient as the initial state (time t=0) for the model to be presented here. The fireball is treated as a hot isothermal spherical thermal of radius a and characteristic density ρ. Actually the fireball is distorted initially by its proximity to the ground and by reflected blast waves, and later by the development of a strong internal circulation and stem-cloud. Its density also varies due to radiative effects and incomplete mixing. The thermal is assumed to be at ambient pressure throughout the model calculations. The temperature of the initial state, T_i, is set at 8000K in conformity with the data of Glasstone and Dolan (1977). Independence of this initial temperature on device yield implies that the volume of the fireball then is proportional only to the yield.

The initial excess energy of the whole fireball or thermal, E_i, (above ambient conditions) is

$$E_i = \beta \times 4.18 \times 10^{15} \times W \quad (J) \quad , \qquad (2)$$

where β is the fraction of the initial explosive yield, W, (in Mt) remaining in the fireball at time t=0. If at t=0, ρ is density, and $V = 4/3\pi\, a_i^3$ is the volume of the sphere, E_i is given by the enthalpy excess of the fireball, $\rho_i V\{h(T_i)-h(T_\infty)\}$. The specific enthalpy of the fireball is represented by $h(T)$ where T is temperature and the subscripts i and ∞ represent initial (fireball) and ambient conditions and a variable without a subscript is that variable for the fireball. Note that initially, at least, some of the energy is stored as chemical energy in the dissociated gases making up the fireball. This is discussed further when the thermodynamic equation for the thermal is presented. We have

$$a_i^3/W = \beta \times 4.18 \times 10^{15} / \frac{4}{3}\pi\rho_i \{h(T_i)-h(T_\infty)\} \quad . \qquad (3)$$

This scaling is the same as that used by Taylor (1950a). Observational evidence about fireball characteristics is presented in Taylor (1950b), Glasstone (1962), Brode (1968) and Waltman (1975). The available data on initial fireball radius for bombs ranging from 1kt to 1Mt are summarised in Table 2. From this table a representative value of a_i^3/W is approximately 0.5×10^{9} m^3Mt^{-1} and by application of Eq.(3) this corresponds to a β of 0.44. (To calculate the enthalpy at T_i = 8000K does require the full procedure discussed in Section 3.3).

The constraint that β = 0.44 is used as the standard case in model calculations. As time increases most of the energy initially in the fireball is subsequently radiated to the far field and the rest is dissipated by mixing of ambient air into the fireball as it rises. We note that 20% of the thermal radiation lost to the far field is radiated before the second thermal maximum (Glasstone and Dolan 1977). As the model presented here commences just after the second thermal maximum, then the calculated total radiant loss must be increased by 25% to account for this early radiant loss.

3. FIREBALL DESCRIPTION

There are five processes which describe the fireball. These are:

- conservation of mass including entrainment
- conservation of momentum
- variation of chemical composition and enthalpy with temperature
- radiative loss
- conservation of energy

These processes are described in the following sections:

3.1 Conservation of mass and entrainment

Following Baines and Hopfinger (1984), we assume that the rate of growth of the thermal depends only on its gaseous mass, m, momentum (which implicitly contains the speed and density of the thermal) and the density of the ambient air, ρ_∞. To lowest order this gives

$$\frac{dm}{dt} = \alpha \rho_\infty A |u| \left(\frac{\rho}{\rho_\infty}\right)^{2/3} . \qquad (4)$$

The growth due to influx of ambient air occurs at a rate proportional to the surface area, A, the mean speed, u, of the rising thermal, and the 2/3 power of the ratio of densities of thermal and ambient air. This latter term accounts for the difficulty of effecting turbulent growth when the density across the sheared interface of the very light thermal is large (Batchelor 1954). The constant of proportionality α, is an entrainment coefficient: its constancy depends on an assumption of self-similarity of the thermal's structure which takes a few thermal radii of rise height to develop. Since the density difference is still large during that time, entrainment, from Eq.(4), is small and little error can arise from the assumption.

The initial organisation of the thermal is an important determinant of the value of the entrainment coefficient (Turner 1957). A disorganized thermal with little initial density difference has an α of 0.25, while a buoyant vortex ring - a very organized thermal - has an entrainment coefficient which, while almost always less than 0.25, increases with buoyancy force and decreases as the square of the imparted circulation. Escudier and Maxworthy (1973), reviewing the situation, come to the conclusion that α is rarely subject to precise specification, even under laboratory conditions. Values in the range 0.01 and 0.35 can be expected.

Mantrom and Haigh (1973) present data for a "low yield, low altitude event" from which a value of α can be derived. Probably based on these same data, Waltman (1975) quotes values of α in the range 0.13 to 0.26 from nuclear tests. The data of Mantrom and Haigh are plotted in Fig.1 as height of the centre of the fireball, z, vs. lateral half-width of the cloud, a, both nondimensionalized by the half-width of the fireball when it commences buoyant rise, a_i. By considering available evidence on rise heights, time to reach altitude and initial fireball diameters (e.g.,

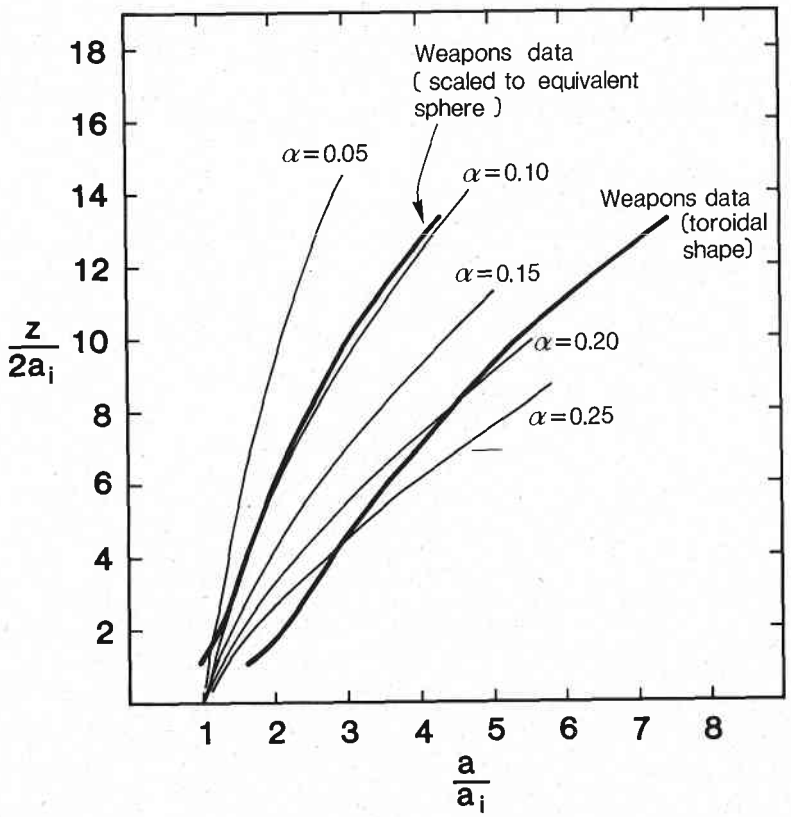


Figure 1: Comparison of model calculations with varying entrainment coefficients, α, with weapons data from Mantrom and Haigh (1973). Height of centre is plotted against half-width of fireball - both scaled by the initial half-width.

Glasstone and Dolan 1977), we conclude that the data presented by Mantrom and Haigh are actually for an event with yield near 1 Mt. Thus in Figure 1 we compare the prototype observations with results from the model developed here for several values of α for a 1 Mt explosion. Note that the range of α quoted by Waltman (1975) is in agreement with our simulations of the Mantrom and Haigh (1973) data.

However, closer study of Figure 1 shows that the fireball's lateral extent at first increases more strongly than the diameter of model thermals, and this is due to the development of a toroidal shape, probably with a hollow centre and certainly with a strong circulation, in the prototype (Glasstone and Dolan 1977) and laboratory analogue (Mantrom and Haigh 1973). In view of the discussion above it would be quite surprising to find an entrainment coefficient near 0.25 since this characterises a disorganized thermal, and at first glance crudely fits the observations presented in Figure 1. Recognizing that, for a sphere of equivalent surface area to the prototype toroidal structure, the radius will be significantly less than the lateral extent of the toroid, we find that the best agreement throughout the range of heights is obtained if the observed values of a are reduced by a factor of 1.75 to fall along the line characterised by $\alpha = 0.1$ in the model. The correspondence shown in Figure 1 is then particularly close. The factor of 1.75 was determined by trial and error. It incorporates a deal of information about toroid structure which is unavailable for prototype fireballs. Its constancy implies self-similarity in the fireball structure and this is required by the model. We take $\alpha = 0.1$ as the standard case in model calculations.

3.2 Conservation of momentum

Escudier and Maxworthy (1973) presented the momentum equation for a thermal:

$$\frac{d}{dt} [(\rho + k\rho_\infty)uV] = V(\rho_\infty - \rho)g \quad . \qquad (5)$$

In setting down equation (5), Escudier and Maxworthy have made the relatively minor assumption that there is no loss of momentum from the thermal to a wake, i.e., that there is no drag force on the thermal. The inertia of the thermal in Eq.(5) is enhanced by a factor $k\rho_\infty$ due to displacement of the surrounding air. The added mass coefficient, k, is assumed to be that for a sphere in potential flow, viz 1/2 (Lamb 1952). The buoyancy of the thermal relative to its surroundings changes the momentum of the upward-moving air.

3.3 Composition and enthalpy

The elemental composition of the fireball has two components. Firstly at the moment of commencement of the model we define a fireball composition, which for a clean explosion consists of moist air of composition appropriate at the earth's surface; see Table 1. The fireball composition is specified according to the number of atoms (in mole amounts) of each element per kg of fireball material. During the running of the model additional mass of ambient air is entrained into the fireball. The

composition of this air is defined according to Table 1. A mass balance is kept for each element, so that the elemental composition of the fireball is known at any time.

Accompanying this elemental composition record, there are calculations of the likely chemical composition of the fireball and its enthalpy for the appropriate range of fireball temperatures. A limited set of chemical species - less than 50 - are chosen according to their likelihood of occurring in the fireball for the range of elemental compositions being considered. A free energy minimization scheme of Eriksson (1971) is used to calculate the chemical composition and enthalpy of an equilibrium mixture having the elemental composition of the fireball for a given temperature and pressure. This scheme can handle solid, liquid and gaseous phases and charged species, of up to ten elements in each calculation. Thermochemical data for the various species is drawn from a computer-based library of information (Turnbull and Wadsley 1984).

The efficacy of this version of Eriksson's (1971) procedure for calculating high temperature chemical equilibria is tested in two ways. A calculation was performed to duplicate that of Gilmore (1955) of the equilibrium composition of dry air at 8000K. The comparison is presented in Table 3(a). The results are all in good agreement, though slightly different due to changes in the accepted values of thermodynamic properties of the species. The one exception to this is O^+, where the currently accepted heat of formation (JANAF 1971) is approximately 70 kcal/mole higher than the value used by Gilmore (1955).

The second test involves comparing the dissociation coefficients Z ($\equiv$ moles of air in heated state/moles of air at STP) and the specific enthalpy h(T) as calculated in the present program with those presented by Reynolds (1965). Again the results (Table 3(b)) are closely comparable, with differences that might be due to changes in accepted values of thermodynamic properties.

Because the enthalpy of the fireball is predicted but the temperature is unknown at each time step in the model, an iterative procedure is used to calculate temperature. In this procedure the fireball elemental composition, pressure and an estimated temperature are used to calculate enthalpy. The calculated enthalpy is compared against the fireball model enthalpy and if necessary the calculation is repeated with a further estimate of temperature until the two enthalpies agree. A quadratic fit of enthalpy versus temperature in the desired range is calculated and used to facilitate this process.

The volume of the fireball is calculated from the equation of state for an ideal gas

$$pV = nR_oT \qquad (6)$$

where R_o is the gas constant for air and n is the total number of moles of gaseous chemical species in the fireball.

The density of the fireball is calculated using this volume but considering all species - solid, liquid and gaseous - for the calculation of mass.

Table 3(a): Comparison of equilibrium composition at 8000 Kelvin and varying density (ratioed to density of air at STP) for the Eriksson procedure and the RAND calculations of Gilmore (1955). Units moles/(mole air at STP). The symbol e represents free electrons.

Species	log10(density ratio) -1.5		log10(density ratio) -2.0	
	Eriksson procedure	RAND	Eriksson procedure	RAND
CO_2	9.08E-10	1.32E-09	9.59E-11	1.40E-10
NO_2	3.39E-09	3.23E-09	3.95E-10	3.74E-10
O_3	5.72E-12	4.62E-12	5.80E-13	4.66E-13
CO	4.16E-05	4.10E-05	1.38E-05	1.37E-05
CO+		1.46E-07		9.06E-08
N_2	1.65E-01	1.65E-01	6.99E-02	6.97E-02
N_2+	1.07E-04	1.11E-04	8.30E-05	8.64E-05
NO	2.45E-03	2.37E-03	9.00E-04	8.72E-04
NO+	7.87E-04	8.40E-04	5.31E-04	5.69E-04
O_2-	1.00E-10	2.27E-10	1.74E-11	3.92E-11
O_2	3.11E-05	3.05E-05	9.92E-06	9.72E-06
O_2+	7.75E-07	5.55E-07	4.54E-07	3.27E-07
e	3.22E-03	3.54E-03	5.53E-03	6.03E-03
C	2.72E-04	2.63E-04	2.85E-04	2.77E-04
C+	1.65E-05	1.47E-05	3.16E-05	2.85E-05
N	1.23E+00	1.23E+00	1.42E+00	1.42E+00
N+	2.29E-03	2.03E-03	4.86E-03	4.34E-03
N++		3.29E-16		1.31E-15
N+++		8.36E-41		6.13E-40
O-	1.78E-06	1.94E-06	9.72E-07	1.05E-06
O	4.16E-01	4.16E-01	4.18E-01	4.17E-01
O+	9.04E-06	5.38E-04	1.67E-05	9.99E-04
O++		9.48E-20		3.27E-19
O+++		1.71E-48		1.09E-47
Ne		3.98E-05		3.98E-05
Ne+		6.81E-12		1.26E-11
Ar	9.33E-03	9.34E-03	9.33E-03	9.34E-03
Ar+	7.72E-06	7.05E-06	1.42E-05	1.30E-05
Ar++		4.96E-17		1.69E-16

Table 3(b): Comparison of dissociation coefficients, Z, and specific enthalpy, h(T), of heated dry air from this study and Reynolds (1965; Fig.B.9) at 1 atm. Units of h(T) are MJ/kg.

	This study		Reynolds (1965)	
Temp. K	Z	h(T)	Z	h(T)
8000	1.88	37.3	1.90	38.5
7000	1.60	25.8	1.58	26.0
6000	1.32	15.0	1.36	15.7
5000	1.21	9.9	1.21	10.3
4000	1.15	7.5	1.12	7.0
3000	1.02	3.8	1.06	4.3
2000	1.00	2.0	1.03	2.3
1000	1.00	0.7	1.00	1.1
288	1.00	0.0	1.00	0.3

3.4 Radiative loss

Radiation from the surface of a thermal is expressed in terms of the Stefan-Boltzmann law which can be written as

$$\dot{R} = \varepsilon\sigma A(T^4 - T_\infty^4) \qquad (7)$$

where ε is the effective emissivity, or more correctly opacity, of the thermal: it is a function of fireball composition and temperature. The Stefan-Boltzmann constant, σ, has the value 5.67×10^{-8} W m^{-2} K^{-4} and A is the surface area of the fireball. In Eq.(7) the environmental radiation temperature is taken as T_∞, the ambient temperature at the level of the thermal. This is an unimportant approximation to a complex situation.

There are detailed discussions of the opacity of heated air (Meyerott et al., 1960, Armstrong et al., 1961, Gilmore 1964) and radiation loss from nuclear fireballs (Bethe 1964, Brode 1968). Certain details are well known. The air within the fireball will be opaque at temperatures of 6000 K and above and transparent at lower temperatures around 1000 to 2000 K. The bomb debris within the fireball will probably be opaque at all temperatures and thus contributes to the opacity of the fireball (Gilmore 1964). To accommodate both these effects for a near-surface burst we use the simple assumption that the fireball acts as a opaque sphere with an emissivity at the outer surface calculated by the following procedure.

Gilmore (1964) provides a table of Planck mean absorption coefficient of heated air as a function of temperature and air density. The Planck mean absorption coefficient, P, is defined by Gilmore (1964) as

$$P = \dot{R}/8\sigma A T^4 a \qquad (8)$$

Thus we use Gilmore's (1964) table of P to derive the effective emissivity ε in Eq.(7) where

$$\varepsilon = 1-e^{-8Pa} \quad . \qquad (9)$$

The values of P are derived by linear interpolation of a transformed table involving logarithmic scales of Planck mean absorption coefficient and density and a linear scale for temperature. The value of P is extrapolated down to 1000K using the same transformed scales as for interpolation. The value of P is held at that for 1000K for all lower temperatures. These data are presented in Table 4(a).

There is a further factor which affects the loss of energy by radiation from the fireball. The atmosphere surrounding the fireball must be sufficiently transparent so that the emitted radiation is lost from the fireball and its immediate surrounds. If the radiation is absorbed in the immediate vicinity of the fireball (i.e., distances << a fireball radius from the surface) then the air which has been heated by the radiation will rise with the fireball and be entrained into it. In this case the radiative energy is not lost from the fireball system and contributes to the buoyant rise of the fireball.

Table 4(a): Planck mean absorption coefficient for high-temperature air (Gilmore 1964) Units: cm^{-1}.

Temperature	Density Ratio*					
(K)	10	1	10^{-1}	10^{-2}	10^{-3}	10^{-4}
2000	5.4E-05	1.7E-06	5.4E-08	1.7E-09	5.3E-11	1.7E-12
3000	1.3E-03	4.4E-05	1.6E-06	7.0E-08	2.7E-09	7.1E-11
4000	1.0E-02	5.7E-04	2.6E-05	7.7E-07	2.1E-08	6.7E-10
6000	1.8E-01	6.1E-03	1.9E-04	6.6E-06	3.3E-07	1.8E-08
8000	4.5E-01	1.4E-02	5.1E-04	1.9E-05	5.9E-07	1.8E-08

* : Density ratio = density heated air/density air at STP

Table 4(b): Fraction, Tr, of black body radiation emitted in the 0.25 to 2.5 μm window (Paltridge & Platt 1976).

Temperature (K)	$Tr = \int_{0.25\mu m}^{2.5\mu m} B_\lambda d\lambda / \sigma T^4$
288	0
1000	0.161
2000	0.634
3000	0.834
4000	0.914
5000	0.946
6000	0.954
7000	0.944
8000	0.919

The factor to be considered is what fraction of the radiation from the fireball can be lost to the far field?. We define this factor as T_r, the transmission of fireball radiation through air at ambient temperature in its immediate surrounds to the far field.

Ultraviolet radiation is strongly absorbed at short distances by the Schumann-Runge bands of O_2 at all temperatures. Complete absorption within a distance of several metres of the thermal is assumed for all wavelengths λ less than $0.25\mu m$.

At longer wavelengths there are many absorption processes but the most important of these insofar as determining the transmission of radiation to some distance from the thermal is due to water vapour and carbon dioxide in the atmosphere. There are many absorption bands for wavelengths greater than $2.5\mu m$ (Paltridge and Platt 1976). We assume complete absorption for $\lambda > 2.5\mu m$.

In the range $0.25\mu m < \lambda < 2.5\mu m$ the atmosphere surrounding the fireball is assumed to be transparent. The transmission factor for black body radiation from the fireball is computed as a function of temperature using the Plank function parameters tabulated in e.g., Paltridge and Platt (1976, Table 2.2) to evaluate the integral

$$Tr = \int_{0.25\mu m}^{2.5\mu m} B_\lambda d\lambda / \sigma T^4 \quad . \qquad (10)$$

The results are presented in Table 4(b). It is evident that the transmission is practically constant at a value of 0.95 for fireball temperatures above 5000 K but below that temperature it falls rapidly as the peak thermal emission moves to longer wavelengths. Approximately, $Tr \propto T^5$ at the lower temperatures.

3.5 Energy balance of the thermal

A two-stage process for thermal growth and rise over a time increment dt is considered. Firstly, the thermal entrains a mass dm of ambient air which is at the same pressure p as the thermal at the level z. The thermal then rises further, expanding against the enveloping air. The only heat loss from the thermal, dQ, is due to radiation to the far field.

So we have

$$dQ = - \dot{R}\, dt$$

and

$$dH = dQ + Vdp + dH' \quad .$$

Here H is the enthalpy of the thermal, dH' is the change in enthalpy of the thermal due to entrainment of ambient air of specific enthalpy h_∞. Thus $dH' = h_\infty\, dm$ and so

$$\frac{dH}{dt} = h_\infty \frac{dm}{dt} + V \frac{dp}{dt} - \hat{R}$$

or

$$m \frac{dh}{dt} = - (h-h_\infty) \frac{dm}{dt} + V \frac{dp}{dz} u - \hat{R} \qquad (11)$$

since

$$\frac{dz}{dt} = u \quad . \qquad (12)$$

The thermodynamic relations described in Section 3.3 provide a unique solution for the relationship between fireball enthalpy and temperature for any given fireball elemental composition, (provided the likely chemical species are known and ideal behaviour is assumed).

Theories of (non-radiating) thermals rising into stably stratified atmospheres are usually expressed in terms of the buoyancy of the thermal, $gV(\rho_\infty - \rho)$, rather than the thermal energy (e.g., Morton et al. 1956). Eq.(11) can be rewritten in terms of the buoyancy utilizing the equation of state, Eq.(6). If dissocation and temperature dependence of the specific heat, $(c_p = (\partial h/\partial T)_p)$, could be neglected, Eq.(11) would reduce to a familiar form:

$$\frac{d}{dt} [gV(\rho_\infty - \rho)] = -\rho_\infty VN^2 u - \frac{g\hat{R}}{c_p T_\infty} \quad . \qquad (13)$$

The buoyancy of the thermal changes due to its motion relative to the ambient thermal stratification given in terms of N(z), the Brunt-Väisälä frequency, and by radiative loss of heat to its surroundings.

In both the thermodynamic and buoyancy formulations of the above equation it has been assumed that there is no loss of internal energy to a wake.

3.6 Summary of model equations

Put $m^* = \rho a^3$, $V^* = \rho_\infty a^3$, $\rho' = \rho/\rho_\infty = m^*/V^*$, $M^* = \rho_\infty a^3 u$ and $F^* = gV^*(1-\rho')$. Then the working equations can be written as

$$\frac{dz}{dt} \equiv u = M^*/V^* \qquad (14)$$

$$\frac{d}{dt} [(\rho' + k)M^*] = F^* \qquad (15)$$

$$\frac{dm^*}{dt} = 3\alpha \left|\frac{M^*}{a}\right| (\rho')^{2/3} \qquad (16)$$

$$\frac{dh(T)}{dt} = -\{h(T)-h(T_\infty)\} \frac{3\alpha}{a} |u| \rho'^{-1/3}$$

$$- \frac{3 T_r \varepsilon \sigma}{a\rho} \{T^4-T_\infty^4\} - gu/\rho' \quad . \qquad (17)$$

These four equations are numerically integrated and a new specific enthalpy and elemental composition are predicted for the fireball after each time step. An iterative procedure is then used in the thermodynamic equilibrium module (see Section 3.3), to calculate the fireball temperature, chemical composition and volume corresponding to the enthalpy and elemental composition. This completes one time step and the process of numerical integration is commenced again. The numerical integration is performed with a fourth order Runge Kutta method (Margenau and Murphy 1956). The integration scheme was tested for numerical stability and it was found that a factor of 8 increase in timesteps produced on average an increase of 4% in the final variable values. The scheme was also tested by, on separate runs, setting the constants α, g and σ to zero. The model behaviour was in agreement with that predicted from the simplified set of equations for these tests.

The thermal commences to rise from rest from the level $z = 0$ at $t = 0$. The standard conditions are an initial temperature T_i of 8000K, and an initial radius a_i given by Eq.(2). The entrainment coefficient $\alpha = 0.1$, the added mass coefficient $k = 1/2$, and the enthalpy, opacity and transmission function are specified by the calculations described in sections 3.3 and 3.4. The ambient conditions are those of the mid-latitude ICAO standard atmosphere (section 2.1).

4. CHARACTERISTICS OF THE FIREBALL MODEL

Most of the published information about fireballs comes from 'clean' 1 Mt explosions. These data will form the standard against which the model will be judged. It is not sufficient that the model merely reproduce correct rise heights. The energy balance of the fireball, particularly during the critical high-temperature period, must be reasonably accurate so that the temperature history (and hence chemical composition) is well simulated.

Figure 2 presents the simulated temporal behaviour of a 1 Mt explosion. The thermal initially accelerates from rest at a rate very close to $2 \cdot g$, the theoretical rate for a weightless spherical bubble in a dense fluid (Milne-Thompson 1967). The thermal builds up speed until, by approximately 19 s, it is rising at a rate of 164 m s^{-1}. Thereafter the speed decreases as entrainment and acceleration of ambient air dominate. Eventually, by about 210 s, the thermal overshoots its neutral buoyancy rise height and begins to fall back, overshooting again. A damped oscillation in velocity and height ensues with period of 370 s, close to the buoyancy period of the atmosphere at that altitude. We take the final rise height to be equal to the first maximum and the vertical extent of fireball material to span this level and the subsequent minimum.

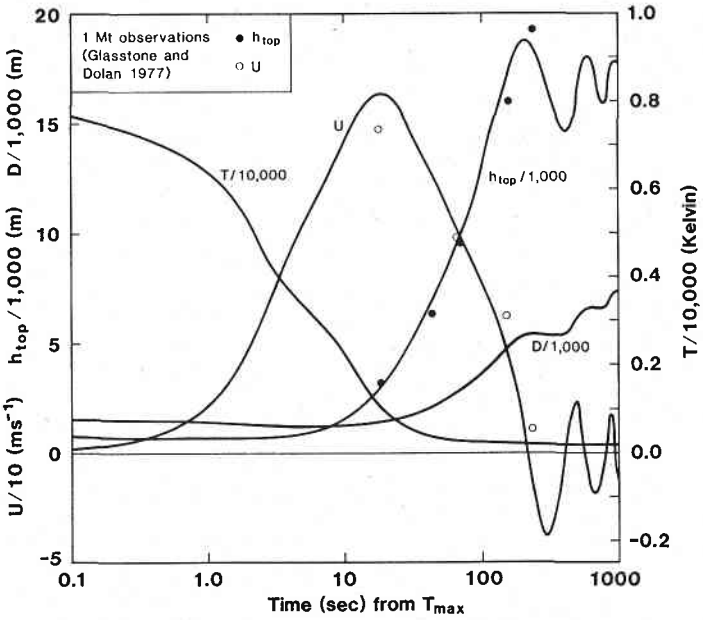


Figure 2(a): The height of the top of the fireball, h_{top}, its rate of rise, u, its temperature T and its diameter D(≈2a), as a function of time for a 1 Mt - yield model simulation compared with observations from Glasstone and Dolan (1977).

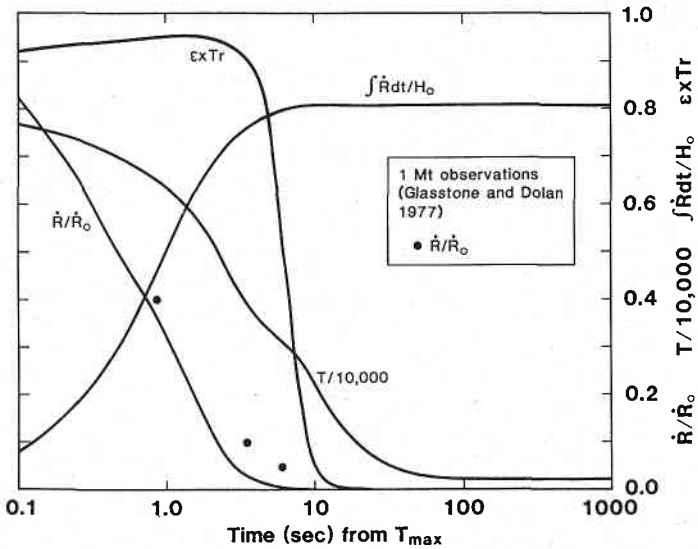


Figure 2(b): The temperature, radiant loss scaled to radiant loss at time=0, $\dot{R}/\dot{R}_o$, integrated radiant loss scaled to fireball energy at time t=0, $\int \dot{R}dt/H_o$ and the fraction of the black body radiance lost from the fireball at each stage, εxTr, see text.

At first the thermal cannot grow by entrainment because this requires relative velocity between thermal and atmosphere. However the thermal does radiate strongly, cooling the sphere, which responds by decreasing in diameter, Figure 2(a). The shrinkage is slight but it takes several seconds for entrainment growth to dominate over radiative cooling which rapidly decreases in importance as the temperature drops. As maximum rise height is approached the thermal grows rapidly and in practice would spread horizontally, carried by the ambient wind.

After approximately 3 s from the second thermal maximum the temperature of the thermal has dropped to 4200 K. Ninety percent of the total energy radiated from the thermal has by then been lost, and the radiative flux is only 4% of its initial strength.

The predicted radiative decay may be compared with data presented by Glasstone and Dolan (1977, Ch.7). As shown by Figure 2(b) the agreement is good. In particular it may be noted that the timescale is well simulated so the temperature variation must also be satisfactory. The predicted rise as a function of time is also in good agreement with observations reported by Glasstone and Dolan, but the ultimate rise height is slightly underestimated. The model agrees well with reported data on the rate of rise at later times, but perhaps overestimates the maximum velocity by as much as 12%.

Ultimately, 81% of the energy in the fireball at $t = 0$ is predicted to be radiated to the far field. Since the fraction of the device yield initially in the thermal is predicted to be 44% (see discussion after Eq.3 above), and 20% of the total thermal radiation emitted is expected to have been lost before the model commences to be relevant (Glasstone and Dolan 1977, Ch.7), the total radiated energy in this model description is 45% of the device yield, W. This is higher than the thermal partition fraction of $f = 35\%$, quoted by Glasstone and Dolan and is in good agreement with the fraction $f = 44\%$, quoted by Brode (1968).

About 19% of the energy in the fireball at $t=0$ is dissipated in turbulent mixing via the buoyant rise of the fireball: this corresponds with about 8% of the weapons yield being dissipated in this way. This loss of energy is not discussed in classical texts (Glasstone and Dolan 1977, Brode 1968). We can only presume that this energy is erroneously accounted for in the shock wave or in the "thermal partition". The latter is unlikely as it would create a serious discrepancy between our modelling results and the observations, whereas as far as we know the exact fraction of the energy in the shock wave is less accurately known.

It would be a simple matter to fine-tune the model by varying parameters but in view of the lack of published detail of fireball behaviour, this is not warranted. Instead, results of a sensitivity study are presented in Table 5. From the Table it can be seen that setting the opacity of the fireball and transmission of the air to a constant value of unity instead of allowing it to vary with temperature does not have a great effect on the major indicators of fireball behaviour. The thermal radiates more energy to the far field and so cools a little more rapidly than the standard case. The result is a lower height of rise, by 5%. Neglecting

Table 5: The sensitivity of the present fireball model to variation of parameters from the standard case. (T_i = 8000K, α = 0.1, β = 0.44, molecular composition, Tr and ϵ all specified functions of temperature). Units are MKS.

Case	T @ 3.5s	u_{max}	@ t	h_{top}	@ t	D_{min}	$\int \dot{R}\, dt/H_o$
Standard	3878	164	19	18827	211	575	0.81
ϵ x Tr = 1.0	3812	161	16	17925	215	570	0.84
ϵ x Tr = 0.0	7451	232	35	30567	201	796	0.00
β = 0.33	3761	155	17	17695	217	521	0.81
β = 0.55	3982	171	19	19814	207	621	0.81
T_i = 10000K	3826	159	17	18054	215	537	0.84
T_i = 6000K	3804	185	22	22379	201	748	0.58
α = 0.05	3998	229	26	27354	189	555	0.84
α = 0.15	3778	136	15	16277	243	587	0.79
Composition fixed	3373	175	21	21380	211	781	0.61

radiative losses entirely (by setting $\varepsilon = 0$ or $T_r = 0$) is quite unacceptable. The thermal does not cool correctly and the maximum velocity and rise height are far too high. The physically realistic lower bound to the radiative losses involves using the opacity of heated air alone (Gilmore 1964) as is done here. In fact the opacity may be higher at low temperatures due to the presence of bomb debris (Gilmore 1964).

Variation of the fraction of device yield initially in the energy of the thermal (equivalent to varying the initial radius of the thermal at constant temperature) causes an almost directly proportional response in ultimate rise height. It also has a large effect on the fraction of energy radiated. (Note the fraction of the device yield radiated is equal to $1.25\ \beta \int R dt / H_o$.) This comes about because the fraction of device yield in the thermal can be partitioned only between energy radiated and the dissipation of kinetic energy by the rising thermal. Reducing β by 25% drops the radiated energy to only 33% of device yield, while increasing it by 20% increases the radiated energy to 56% of total yield. Both extremes are unacceptable, so the energy partitioning in the initial fireball (and thus its radius) is constrained to be close to that proposed here. Note that the ratio of radius to bomb yield derived for the standard case is that observed from nuclear explosions (Section 2.2).

The initial temperature of the thermal is not a sensitive parameter, but a value as low as 6000 K does lead to a rise height which is too large, primarily because not enough energy is lost by radiation (only 32% of device yield). The initial temperature used in the standard case is that observed from nuclear explosions.

A 50% lower entrainment coefficient is much too low, with maximum velocity and rise height too high due to the reduced retardation. Increasing α to 0.15 is less drastic but the enhanced mixing does lead to a significantly lower rise height than observed. The comparison of fireball diameter to rise height for the model, and a real event, have been presented in Fig.1.

Finally, if the composition of the fireball was held at that for ambient conditions, i.e. N_2, O_2, Ar, CO_2, H_2O, and no dissociation allowed, then when the initial energy partitioning is maintained, the fireball has quite different behaviour to the standard case. The initial fireball has a 36% larger diameter, rises 14% higher and loses only 34% of the total explosive energy as radiant loss. Initially the fixed composition fireball falls in temperature more rapidly than the dissociated one. This arises because the specific enthalpy at 8000K is nearly four times higher for dissociated compared with the undissociated air, whereas by 3000K the difference is only 35%.

We conclude from the sensitivity study that it is not possible to have a great deal of variation from the proposed parameter values in the standard model if realistic major indicators of fireball behaviour are to be obtained. This is very satisfying since all parameters except the entrainment coefficient were chosen independently of the necessity to predict ultimate fireball performance.

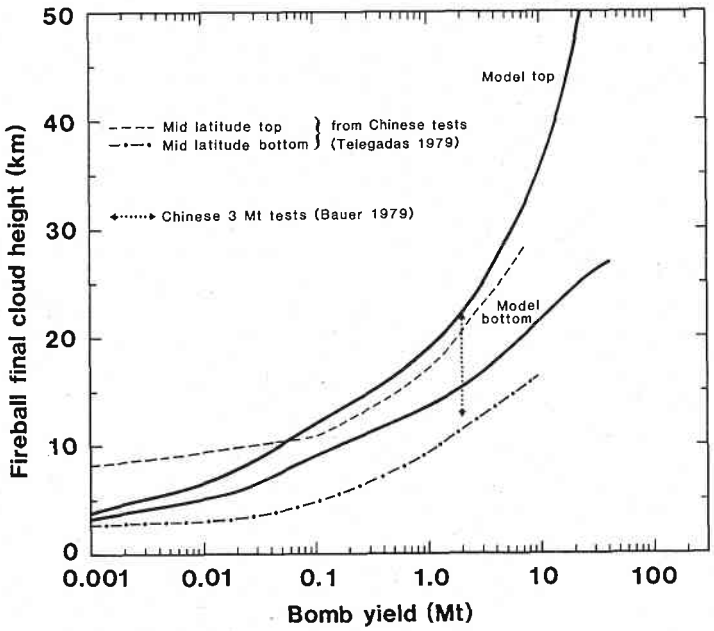


Figure 3(a): Fireball rise heights from the model simulations and from various atmospheric explosions in mid latitudes as described in the text, where the explosions were near mean sea level.

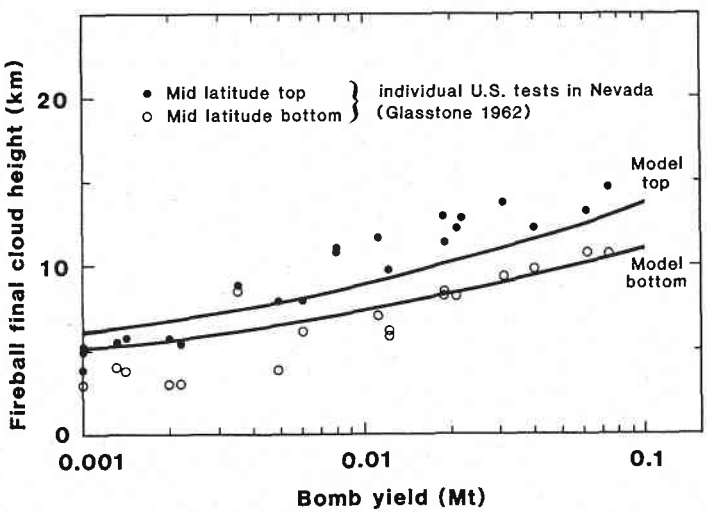


Figure 3(b): Fireball rise heights from the model simulations and from atmospheric explosions in Nevada, USA, (Glasstone 1962) where the model calculations are referenced to, and the atmospheric explosions took place at, approximately 2 km above mean sea level.

Figures 3(a) and 3(b) show the variation of rise height with device yield for the model. Most of the atmospheric nuclear testing has been conducted in equatorial latitudes (2-17°N) by the USA and in polar latitudes (75°N) by the USSR (Bauer 1979). The main data from mid latitudes comes from Chinese tests. There appears to be good agreement between this model and the analysis of the Chinese tests by Telegadas (1979) and Bauer (1979) as shown in Figure 3(a), with overlapping ranges from 0.001 to 10 Mt. Glasstone (1962) presents cloud heights from other mid latitude nuclear tests in north-eastern Nevada. The heights of burst in these tests were approximately 300 to 400m above the local surface, which in turn is on average slightly less than 2 km above mean sea level, leading to an approximate height of burst of around 2 km above sea level. These data are compared in Figure 3(b) with cloud heights from model calculations where the height of explosion is 2 km above sea level. It appears that the observations bracket the model calculations, but the cloud top is underestimated in the model for bomb yields of 0.01 to 0.1 Mt whereas it is slightly overestimated for bomb yields of 0.001 Mt.

Overall the predictions of cloud heights from this simple model lie within the scatter of the actual observations for most bomb yields from 0.001 Mt to 10 Mt, a dynamic range of 10^4.

5. CONCLUSION

A model of the late (ascending) stage of a nuclear fireball has been presented and its sensitivity to variations in basic parameters has been presented. The model has been shown to be robust, permitting little variation of parameters away from values established by comparison with independent observations.

The model is sufficiently complex to be useful as the basis for many extensions - to address issues such as:

> nitric oxide injection into the stratosphere by fireballs, soot and elemental carbon injections from "dirty" explosions, injection of radiactive isotopes into the atmosphere, radar signature of air bursts.

6. ACKNOWLEDGEMENTS

We thank Dr. Alan Turnbull and Mr. Mike Wadsley for supplying to us the code implementing the Eriksson (1971) chemical composition and enthalpy calculation procedures. This work would not have been possible without it. We thank Professor Paul Crutzen for discussions that stimulated this work. We also thank the Committee of the International Year of Peace, Department of Foreign Affairs for financial support for Lou Ripari and the project.

7. REFERENCES

Ambio advisers (1982): Reference scenario: How a nuclear war might be fought. Ambio 11, 94-99.

Armstrong, B.H., J. Sokoloff, R.W. Nicholls, D.H. Holland and R.E. Meyerott (1961): Radiative properties of high temperature air. J.Quant. Spectrosc.Radiant Transfer, 1, 143-162.

Baines, W.D. and E.J. Hopfinger (1984): Thermals with large density difference. Atmos.Environ., 18, 1051-1057.

Batchelor, G.K. (1954): Heat convection and buoyancy effects in fluids. Q.J.Roy.Met.Soc., 80, 339-358.

Bauer, E. (1979): A catalogue of perturbing influences on stratospheric ozone, 1955-1975. J.Geophys.Res., 84, 6929-6940.

Bethe, H.A. (1964): Theory of the fireball. Los Alamos Scientific Laboratory, University of California, Los Alamos, New Mexico, LA-3064, UC-34, PHYSICS, TID-4500 (30th Ed.), 85pp.

Brode, H.L. (1968): Review of nuclear weapons effects. Ann.Rev. Nuclear Sci., 18, 153-202.

Brode, H.L. and Bjork, R.L. (1960): Cratering from a megaton surface burst. Research Memorandum RM2600, The Rand Corporation, Santa Monica, USA, 60pp.

Broido, A., C.P. Butler, R.P. Day, R.W. Hillendahl, S.B. Martin and A.B. Willoughby (1953): Thermal Radiation from a Nuclear Detonation. Operation Tumbler-Snapper [Project 8.3]. U.S. Naval Radiological Defence Laboratory, San Francisco. NTIS AD-B951 585. 109pp.

Eriksson, G. (1971): Thermodynamic studies of high temperature equilibria, ACTA Chemica Scandinavica, 25, 2651-2658.

Escudier, M.P. and T. Maxworthy (1973): On the motion of turbulent thermals. J.Fluid Mech. 61, 541-552.

Gilmore, F.R. (1955): Equilibrium composition and thermodynamic properties of air to 24,000K. RM-1543 ASTIA Document No. AD84052, The Rand Corporation, Santa Monica. 68pp.

Gilmore, F.R. (1964): Approximate radiation properties of air between 2000 and 8000K. Memorandum RM-3997-ARPA (ARPA Order No.189-61). The Rand Corporation, Santa Monica.

Glasstone, S. (1962): The effects of nuclear weapons. US Government Printing Office, 730pp.

Glasstone, S. and Dolan, P.J. (1977): The effects of nuclear weapons. US Government Printing Office, 653pp.

ICAO (1964): Manual of the ICAO standard atmosphere extended to 32 kilometres (2nd Edn). International Civil Aviation Organization, Montreal.

JANAF (1971), JANAF Thermochemical Tables 2nd ed. D.R. Stull and H. Prophet, Office of Standard Reference Data, National Bureau of Standards, Washington D.C., Nat.Stand.Ref.Data Ser., Nat.Bur. Stand. (U.S.), 37, 1141p. plus 1976 and 1982 supplements.

Lamb, Sir H. (1952): Hydrodynamics. 6th Edn. Cambridge University Press. 738pp.

Manins, P.C. (1985): Cloud heights and stratospheric injections resulting from a thermonuclear war. Atmos.Env. 19, 1245-1255.

Mantrom, D.D. and Haigh, W.W. (1973): Fireball entrainment study. TRW Systems Group Redondo Beach, Calif. Final Report No. 18895-6004-RU-00. September 1973, NTIS-AD-780 192. 111pp.

Margenau, H. and G.M. Murphy, (1956): The Mathematics of Physics and Chemistry. Van Nostrand Co., Princeton, USA, 604pp.

Meyerott, R.E., J. Sokoloff and R.W. Nicholls (1960): Absorption coefficients of air. Geophysics Research Directorate, Air Force Research Division, Air Research and Development Command. USAF, Bedford, Massachusetts.

Morton, B.R., G.I. Taylor and J.S. Turner (1956): Turbulent gravitational convection from maintained and instantaneous sources. Proc.Roy. Soc. A., 234, 1-23.

Milne-Thompson, L.M. (1967): Theoretical hydrodynamics. 5th Edn. Macmillan. 660pp.

Paltridge, G.W. and C.M.R. Platt (1976): Radiative process in meteorology and climatology. Amsterdam. 318pp.

Pittock, A.B., T.P. Ackerman, P.J. Crutzen, M.C. MacCracken, C.S. Shapiro and R.P. Turco (1986): The environmental consequences of nuclear war, Volume 1, Physical and atmospheric effects, John Wiley and Sons, Chichester, UK, 359pp.

Reynolds, W.C. (1965): Thermodynamics. McGraw-Hill, New York, 458pp.

Taylor, G.I. (1950a): The formation of a blast wave by a very intense explosion. I. Theoretical discussion. Proc.Roy.Soc., A. 201, 159-174.

Taylor, G.I. (1950b): The formation of a blast wave by a very intense explosion. II. The Atomic Explosion of 1945. Proc.Roy.Soc., A. 201, 175-186.

Telegadas, K. 1979: Estimation of maximum credible atmospheric radioactivity concentrations and dose rates from nuclear tests. Atmos.Env., 13, 327-334.

Turnbull, A.G. and M.W. Wadsley (1984): Thermodynamic Modelling of Metallurgical Processes by the CSIRO-SGTE Thermodata System. Proc.Symp. on Extractive Metallurgy, Aus.I.M.M. Nov. 1984. p79-85.

Turner, J.S. (1957): Buoyant vortex rings. Proc.Roy.Soc., A. 239, 61-75.

Valley, S.L. (ed.) 1965, Handbook of geophysics and space environments. McGraw-Hill, N.Y., USA, 683pp.

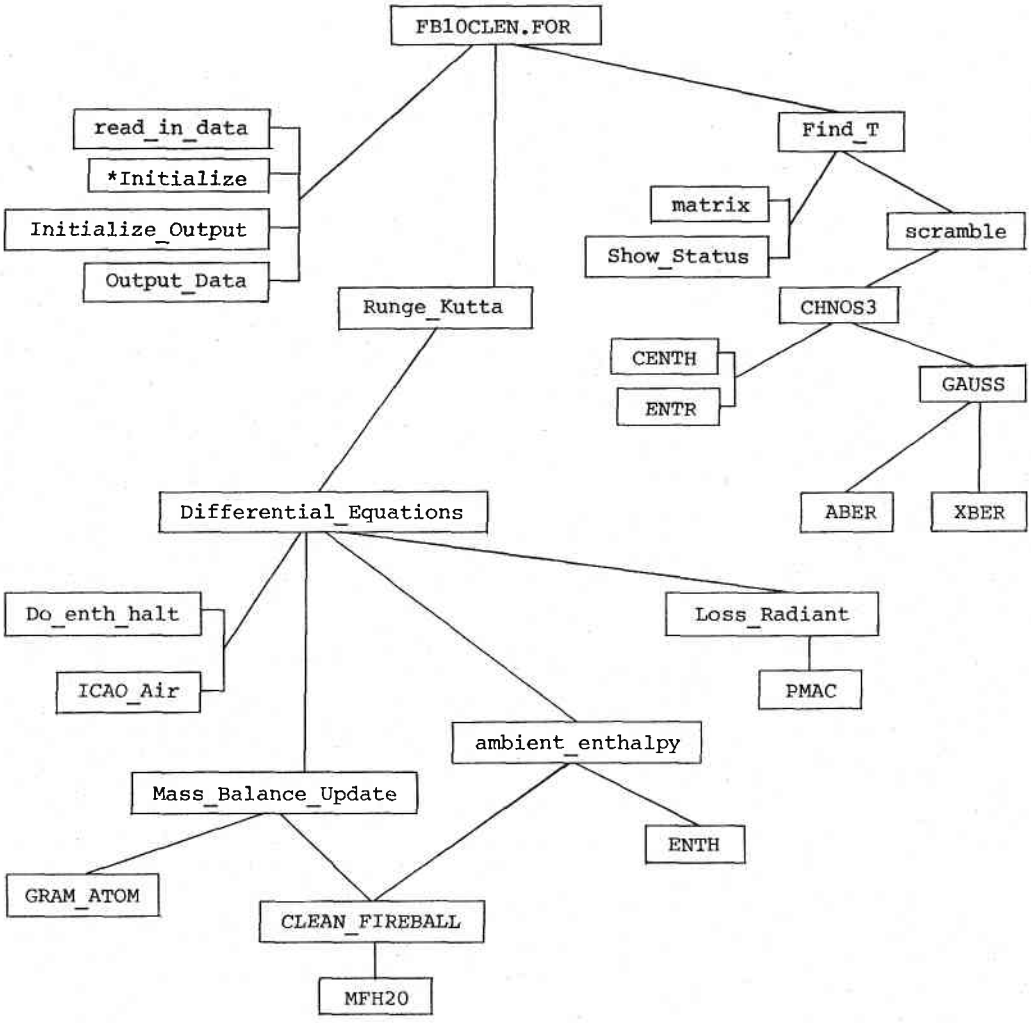
Waltman, D.D. (1975): A simple description of an ascending nuclear fireball and a fortran solution. M.Sc. thesis, Air Force Institute of Technology, Ohio, USA. NTIS AD-A017 180/1. 78pp.

Appendix A: A simpler model of the late (ascending) stage of a nuclear fireball.

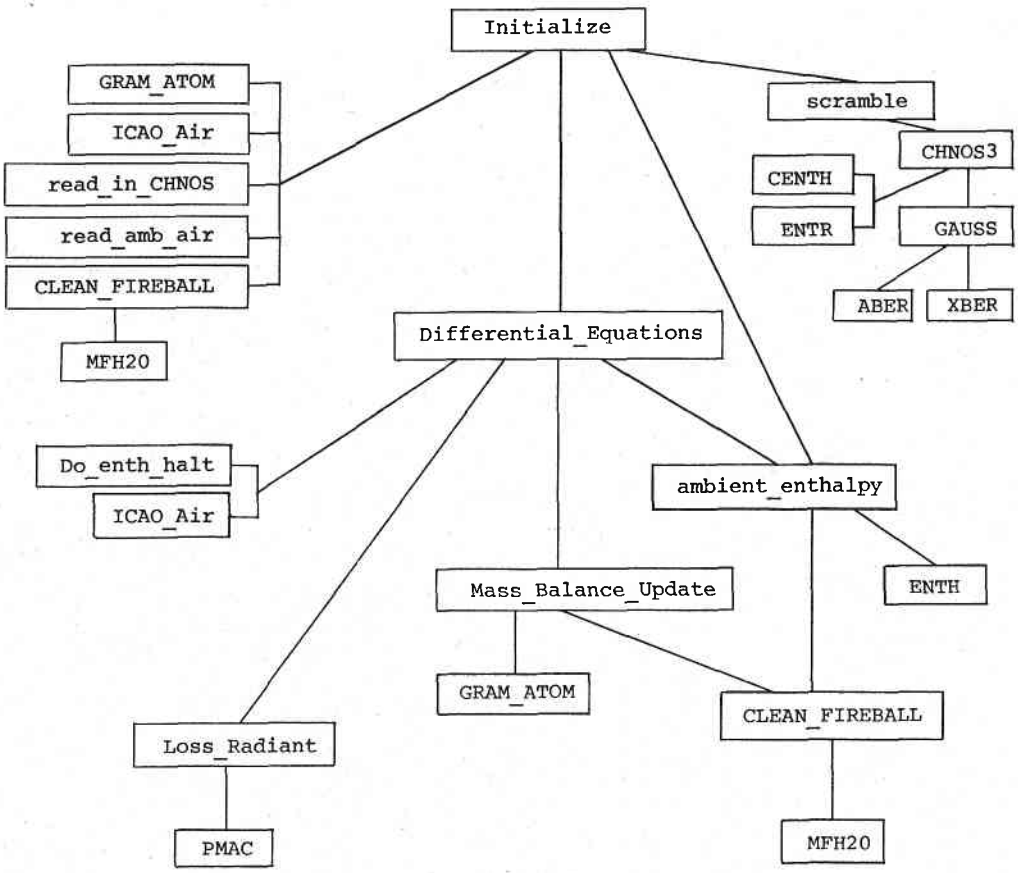
If the only purpose of the fireball model is to examine physical behaviour and no detail of the chemical model is required then the computation can be greatly simplified by using an enthalpy/temperature relationship for heated air, such as may be obtained from Reynolds (1965; Fig.B.9) in place of the free energy minimization scheme. This leads to a substantial reduction in code ($\sim$ 30%) and the running time is reduced by a factor of 2 to 10 depending on the number of species evaluated in the chemical equilibria scheme.

If the enthalpy/temperature relationship is computed for the same species and air of the same elemental composition in the simple and the full model then the model predictions of other physical parameters are (by definition) indistinguishable.

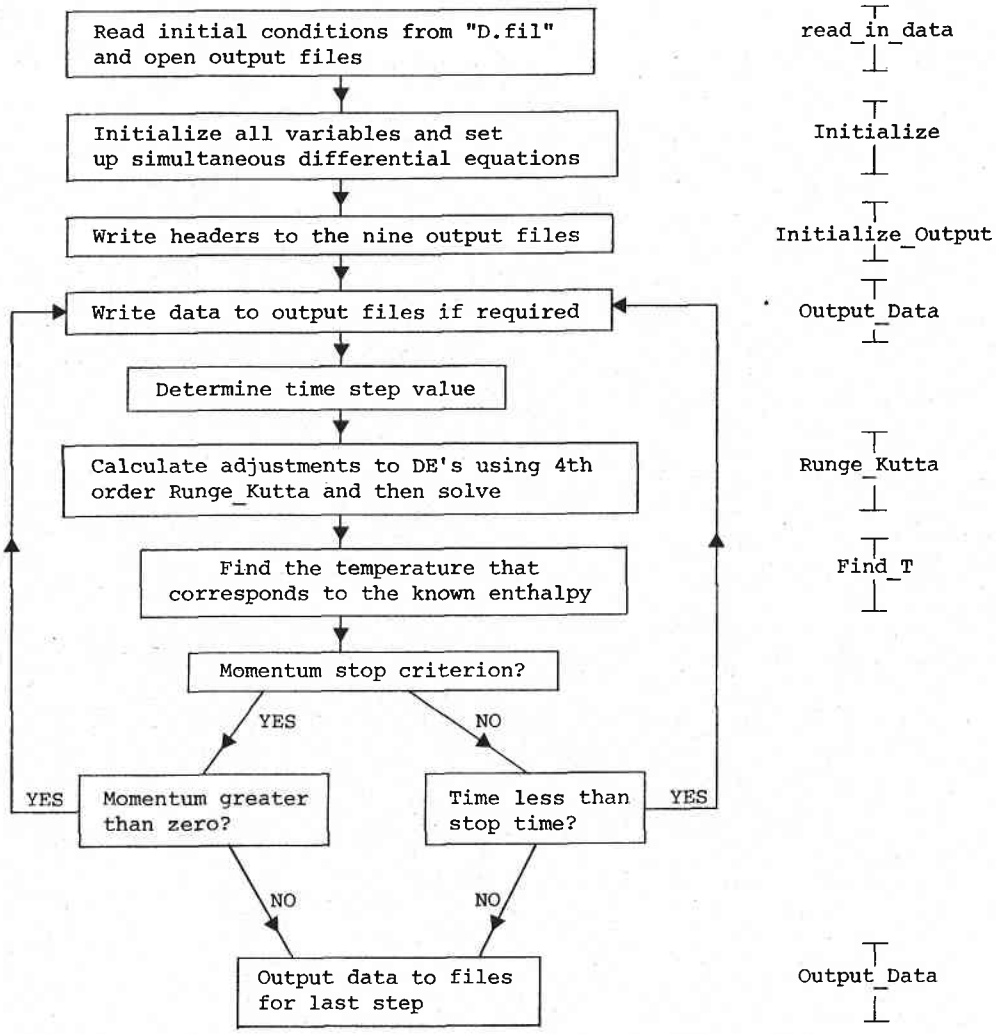
Appendix B: Fireball Model Flow Diagrams



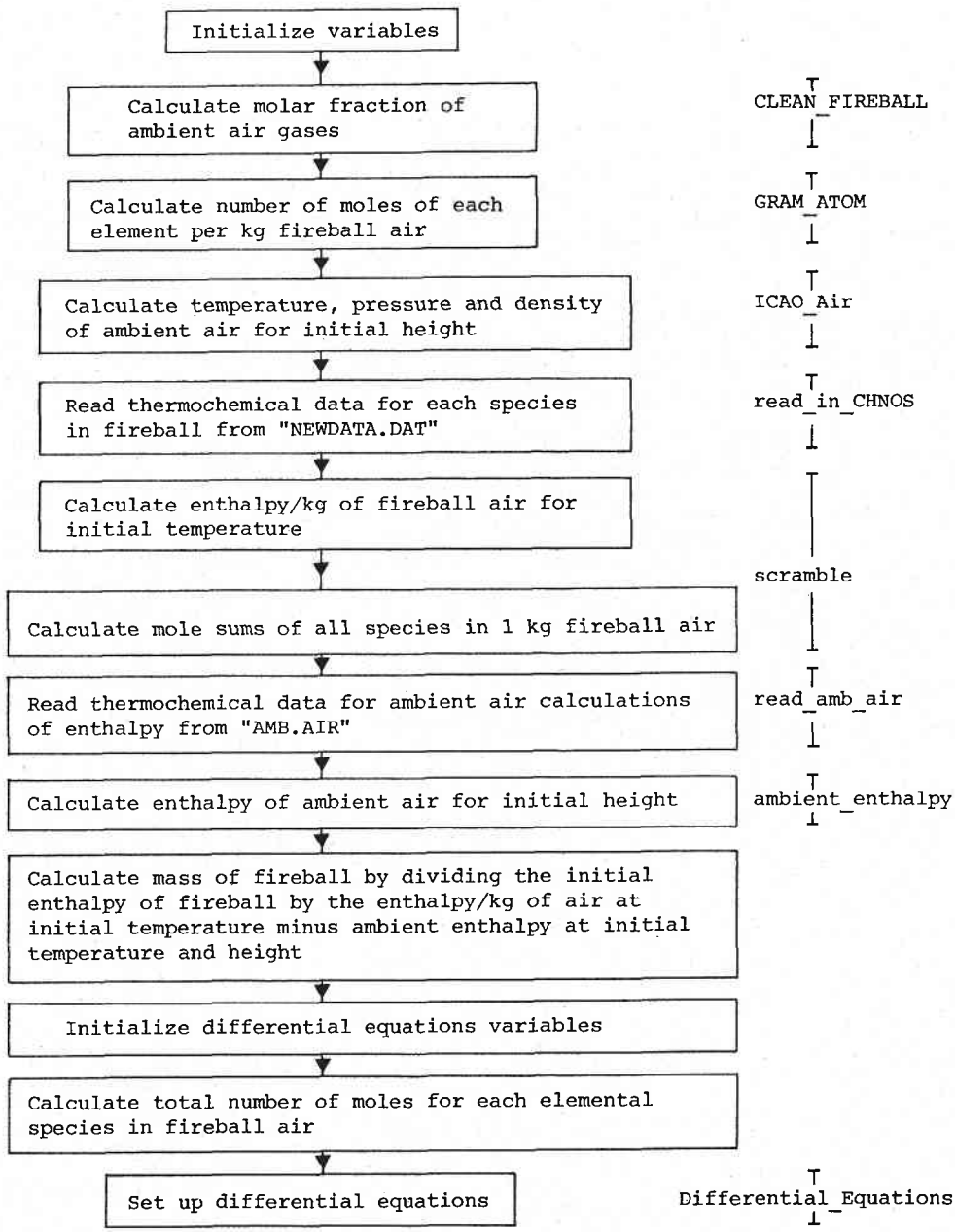
*see next page


Initialize
GRAM_ATOM
ICAO_Air
read_in_CHNOS
read_amb_air
CLEAN_FIREBALL
MFH20
scramble
CHNOS3
CENTH
ENTR
GAUSS
ABER
XBER
Differential_Equations
Do_enth_halt
ICAO_Air
ambient_enthalpy
ENTH
Mass_Balance_Update
GRAM_ATOM
CLEAN_FIREBALL
Loss_Radiant
PMAC
MFH20

PROGRAM OUTLINE

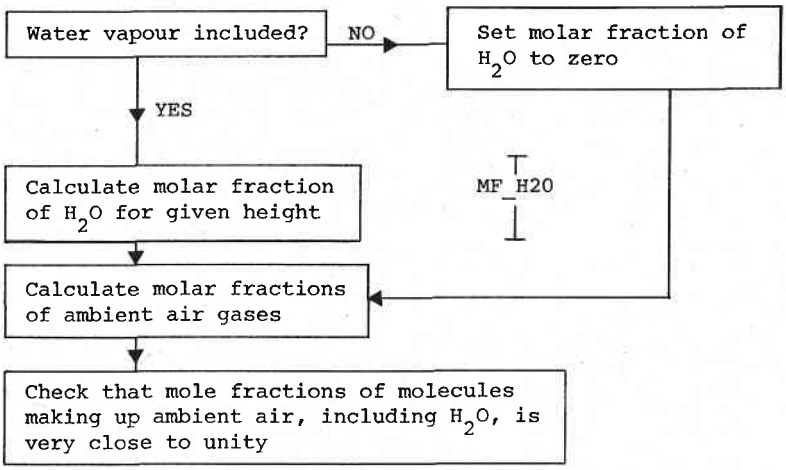


Subroutine Initialize



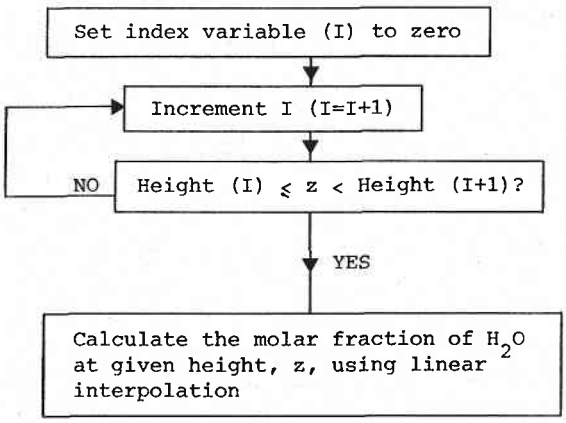
Subroutine CLEAN FIREBALL (hite, vapour, MFH2O, MFN2, MFO2, MFCO2, MFAC)

- calculates the molar fractions of ambient air gases



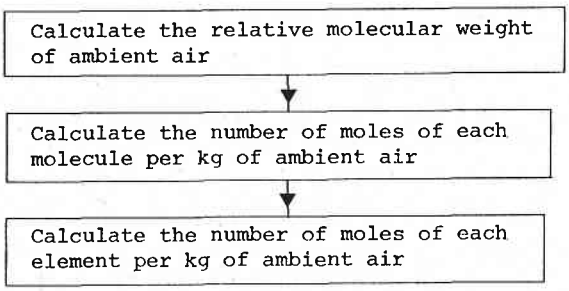
Function MF H2O(z)

- returns the molar fraction of H_2O at the given height



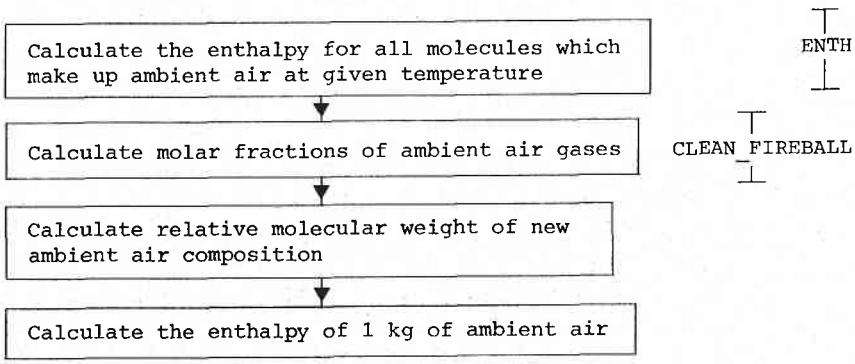
Subroutine GRAM_ATOM (MFH2O,MFN2,MFO2,MFCO2,MFAr,nN,nO,nH,nC,nAr)

- calculates the number of moles of each element given the molar fractions of the molecoles per kg of ambient air



Subroutine ambient_enthalpy (z,temp)

- calculates the enthalpy of new ambient air composition



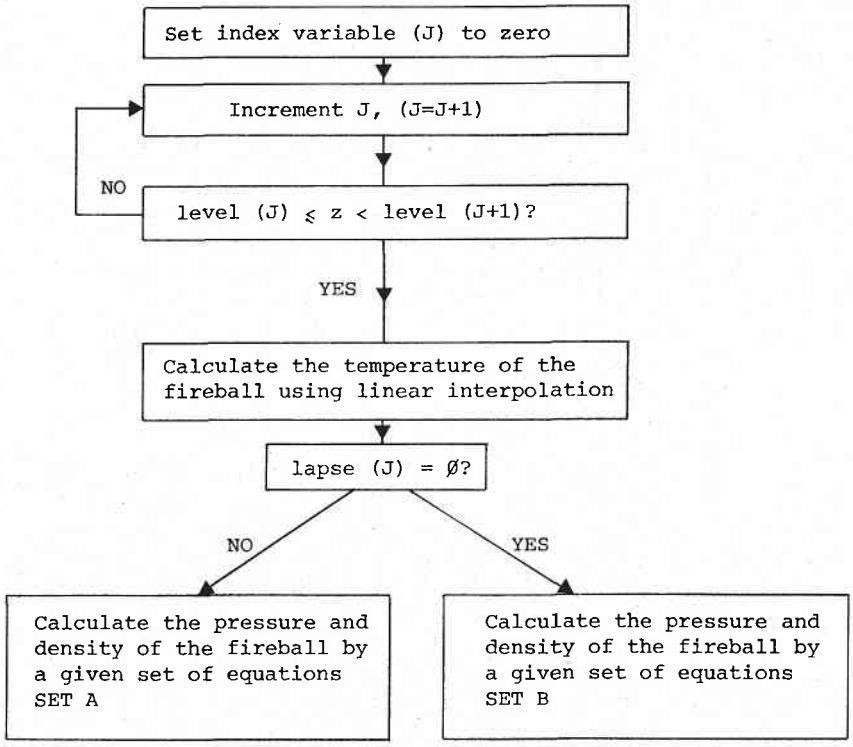
Function ENTH (I,T)

This function calculates the enthalpy of the ambient air species indexed by I for the temperature T where T is up to 6000°K.

The thermodynamic data used for this has already been read in from the file 'amb.air' and stored into the arrays ER(6), which is the heat of formation at 298.15°K for each of the six species, and C(6,6) which contains the numeric coefficients, that describe the variation of specific heat at constant pressure as a function of temperature, needed for the calculation of enthalpy.

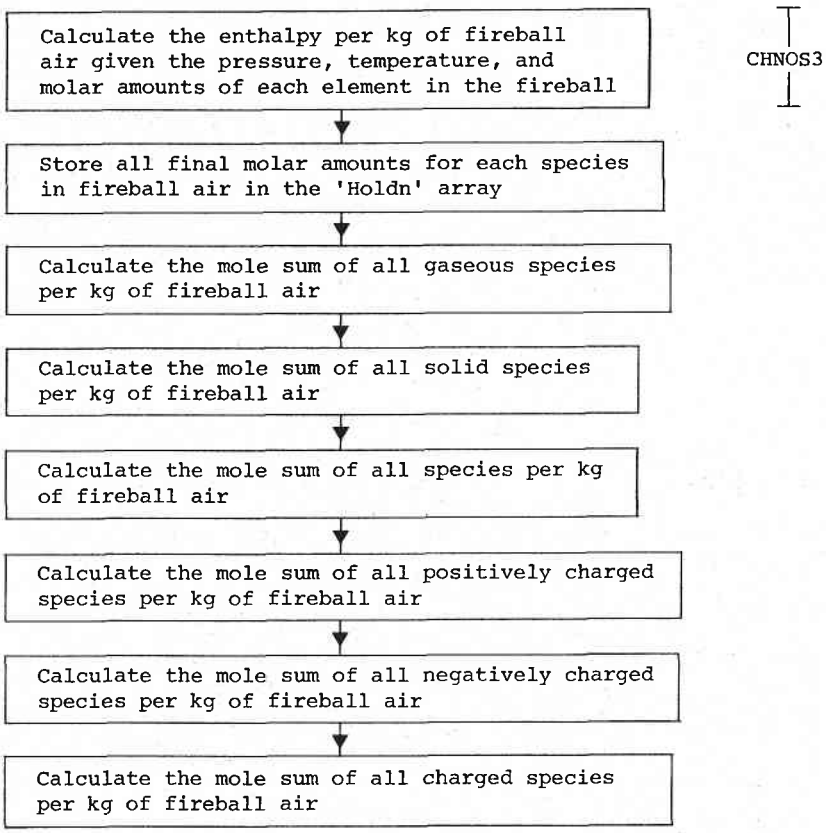
Subroutine ICAO_Air (z,temp,pressure,density)

- calculates temperature, pressure and density of the ambient air for the given height of the fireball



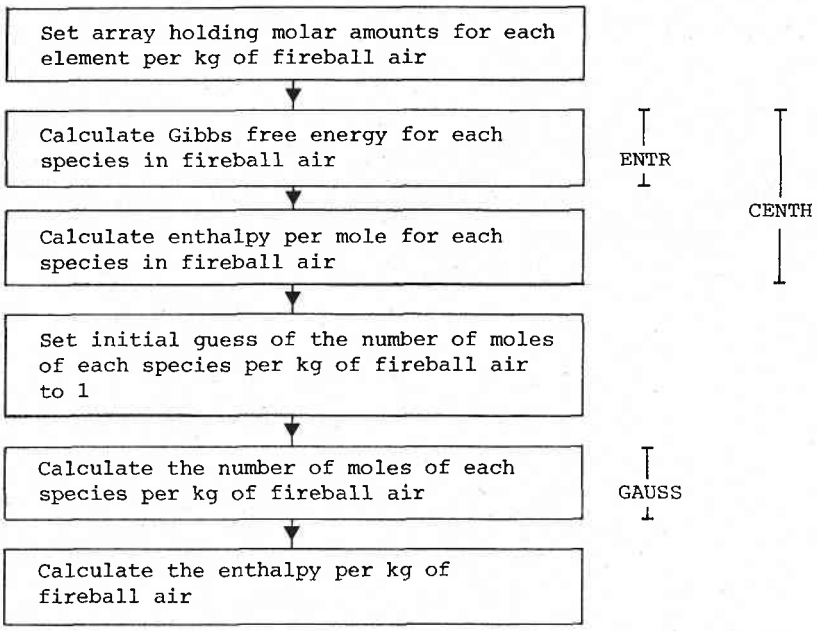
Subroutine scramble (temp,pressure)

- uses the output of CHNOS3, stores all final molar amounts for species in fireball in 'Holdn' array and calculates various properties of the composition of the fireball



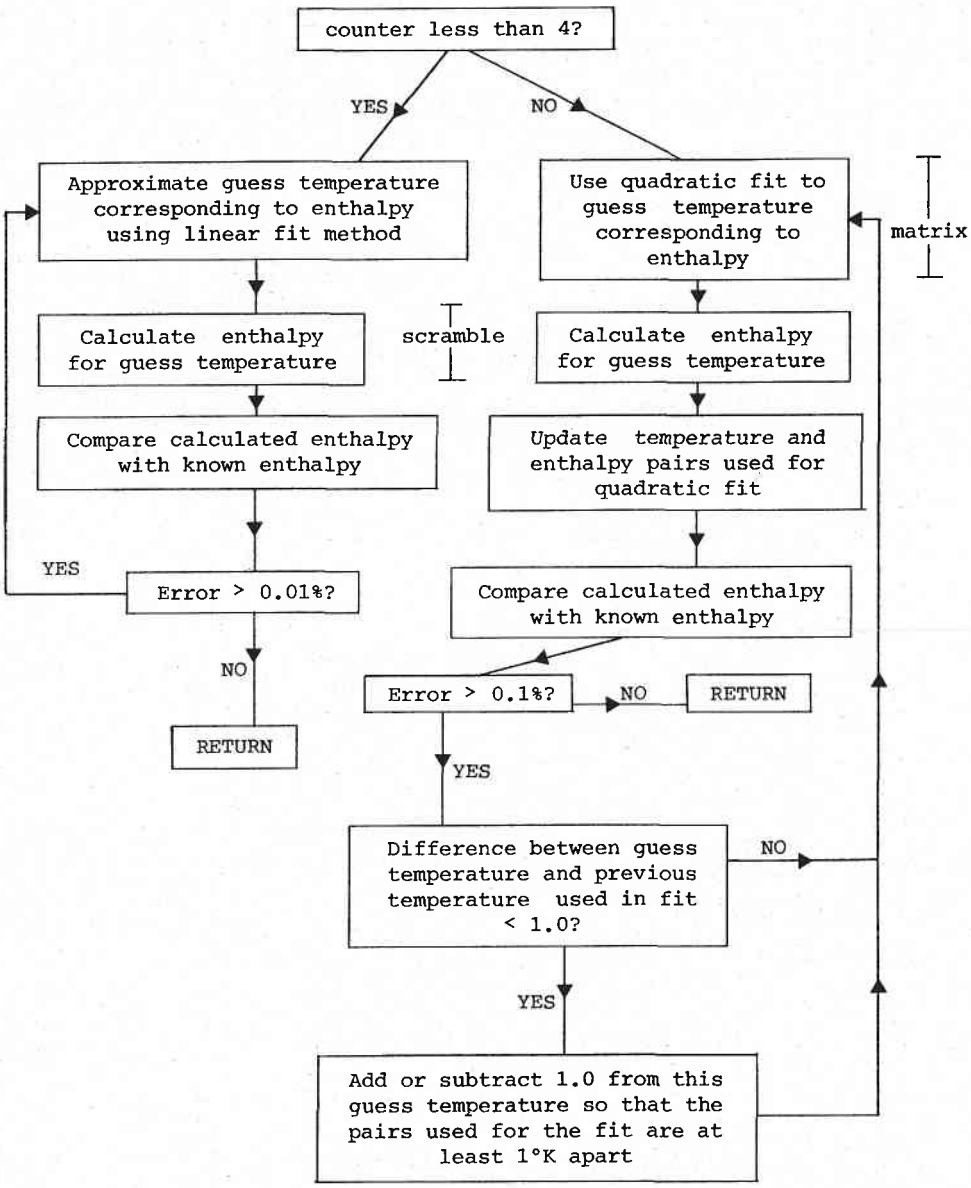
Subroutine CHNOS3 (temp,pressure,aN,aO,aH,aC,aAr,aS,aE,Enthalpy_per_kg)

- calculates the enthalpy per kg of fireball air



Subroutine Find_T

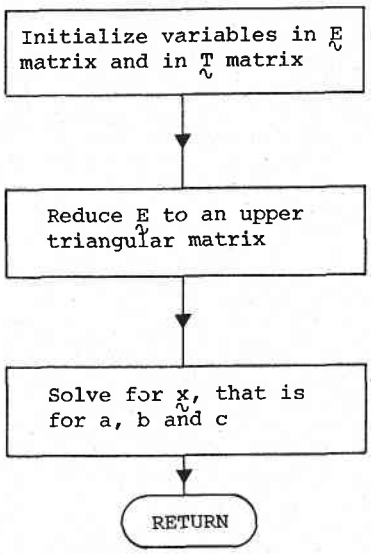
- finds the temperature that corresponds to a known enthalpy



Subroutine matrix

- A 2nd order polynomial fit used by Find_T to approximate guess temperature corresponding to enthalpy

Have $\begin{bmatrix} E_1^2 & E_1 & 1 \\ E_2^2 & E_2 & 1 \\ E_3^2 & E_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$ ie: $\underset{\sim}{E}\,\underset{\sim}{x} = \underset{\sim}{T}$ need to solve for $\underset{\sim}{x}$



Function CENTH (I,T) and Function ENTR (I,T)

The functions CENTH and ENTR calculate the enthalpy and entropy respectively and any fireball species, I, for any temperature T.

The thermodynamic data required for these calculations has already been read from the file 'newdata.dat' and stored in the arrays HR(50), the heat of formation of species at 298.15K, SR(50), the enthalpy of the species at 298.15K, C1(50,5), C2(50,5), C3(50,5), C4(50,5), TM(50,5) and HH(50,5) which contain the numeric coefficients that describe the variation of specific heat at constant pressure as a function of temperature.

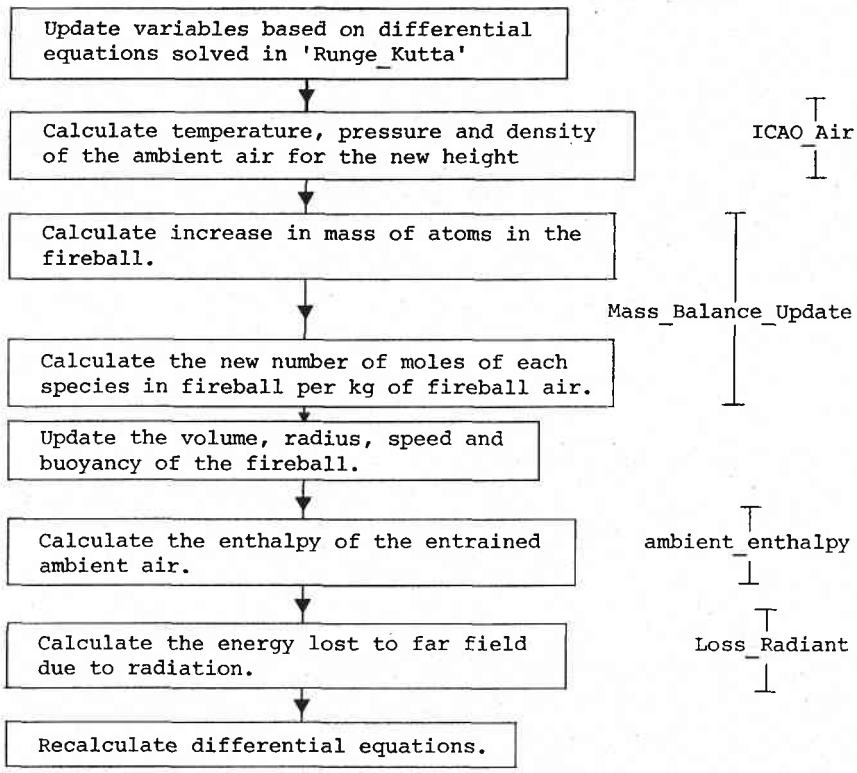
Subroutine Runge_Kutta (N,t)

-A 4th order Runge_Kutta numerical approximation to the solution of the N differential equations.

This subroutine increments the time by the timestep.

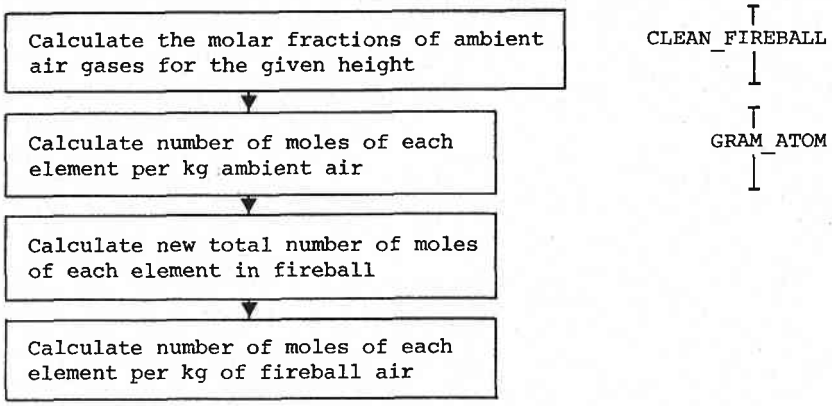
Subroutine Differential_Equations

- recalculates differential equations



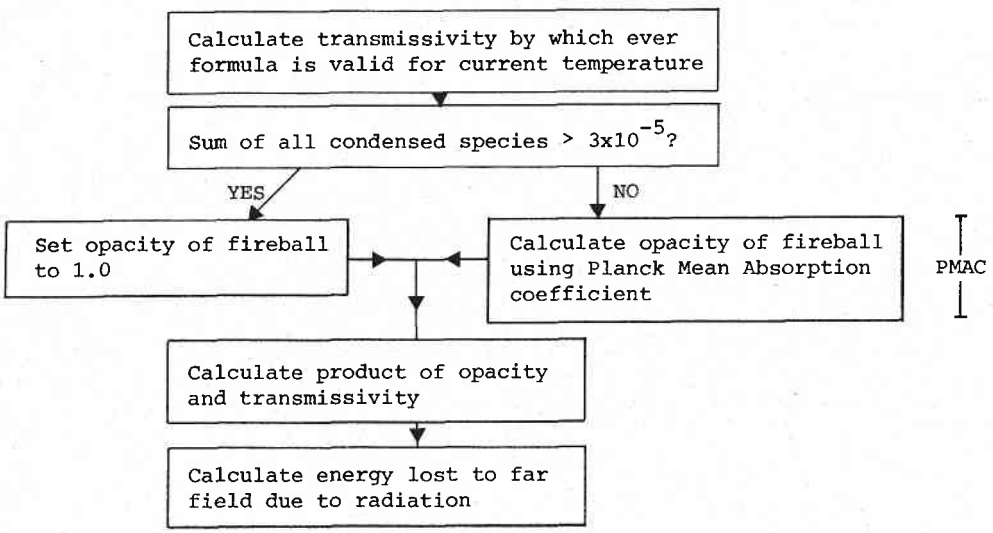
Subroutine Mass_Balance_Update (OLD_MASS, BMass, hite)

- calculates the increase in mass of atoms in fireball and updates for present time



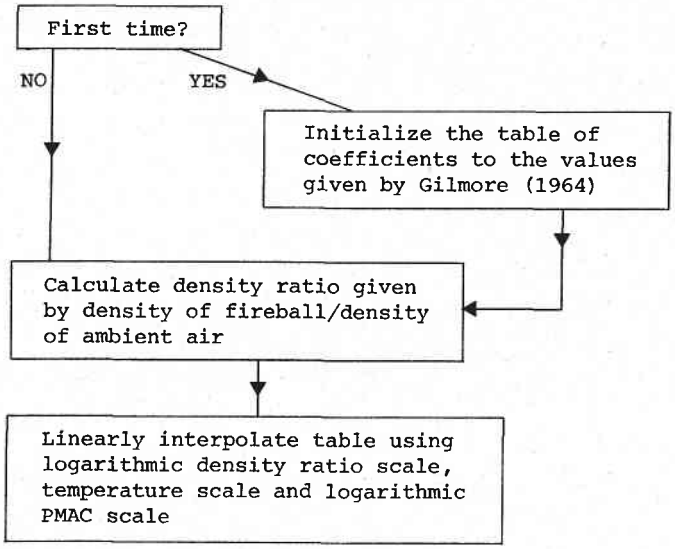
Subroutine Loss_Radiant (T, rad, T_amb)

- calculates the energy lost to far field due to radiation



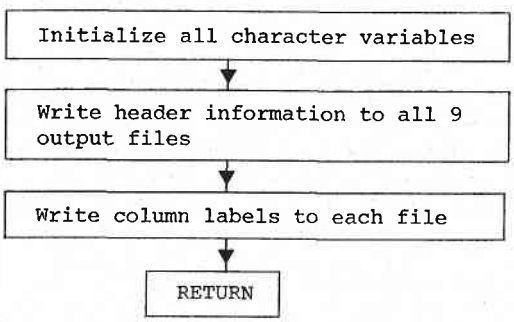
Function PMAC (T_fb)

- returns the Planck Mean Absorption Coefficient



Subroutine Initialize_Output

- sets initial information in headers for data output to screen and output files



Subroutine Output_Data

- outputs data to the screen and output files. It is only called every five timesteps and for the final timestep

Subroutine read_amb_air

- reads and stores the thermochemical data for the ambient air gases from the file "AMB.AIR"

Subroutine read_in_CHNOS

- reads the species in the fireball from "D.FIL" and all the thermochemical data for these species from "NEWDATA.DAT". Counts the number of elements, gases and condensed species and sets up the atoms matrix for use in the Eriksson procedure

Subroutine read_in_data

- reads the initial conditions and other input data from "D.FIL" and opens the output files

Subroutine DO_enth_halt

- this is an error message subroutine, it is called from the subroutine Differential_Equations if the enthalpy per kg of fireball mass is greater than the initial fireball enthalpy. It prints an error message and stops execution of the program

Subroutine Show_Status

- another error message subroutine which is called from the subroutine Find_T if the calculated temperature of the fireball is less than zero or greater than 10,000°K. It prints out an error message and stops execution of the program

Appendix C: Fireball Model Code

LIST OF VARIABLES USED IN PROGRAM

VARIABLE	TYPE	DESCRIPTION
aXX	Real	- Used in "CHNOS3". Molar amount value for entity XX per kg fireball. XX : Ar,C,E,H,N,O,S
ANF	Character*80	- Used in "read_in_CHNOS3". Reads a line of data from "D.FILe".
anXX	Real	- Calculated in "GRAM_ATOM" and used in "Mass_Balance_Update". Molar amount of XX per kg of ambient air. XX : Ar,C,E,H,N,O,S
atime	Real	- Used in "Output_Data". Program calculated time.
Atmos	Character*20	- Used in "read_in_data" and "Initialize_Output". Type of atmosphere used.
b(0:3)	Real Array	- Used in "Runge_Kutta". Coefficient array used in Runge Kutta integration scheme.
beta	Real Constant	- Read from "D.FILe" and used in "Initialise". Fraction of bomb yield in fireball at start time. For 8000 Kelvin at start temp, beta has value of 0.443942345.
BMass	Real	- Dummy variable (same as Mass), used in "Mass_Balance_Update". Mass of fireball air. Unit : kg.
c(0:3)	Real Array	- Used in "Runge_Kutta". Coefficient array used in Runge Kutta integration scheme.
C(6,6)	Real Array	- Used in "read_amb_air" to store the numeric coefficients from "AMB.AIR" allowing the calculation of entropy and enthalpy.
cala	Real	- Used in "Find_T" and "Matrix". Coefficient of enthalpy/temperature polynomial calculated in "Matrix", used in "Find_T" to estimate temperature given enthalpy.
calb	"	- " " "
calc	"	- " " "
ch	Character*1	- Used in "read_in_CHNOS" and "read_in_data". Reads a character from "D.FILe".

chan_z	Real	- Used in "Differential_Equatons". Difference in height between current and previous time step.
Charged	Logical	- set in "read_in_CHNOS" used in "CHNOS3" and "Output_Data".
count	Integer	- Used in Main program and "Output_Data". It counts the number of times that the main program loop is cycled and allows the output data to be sent to the output file every five cycles.
counter	Integer	- Used in "Initialise" and "Find_T". Holds the number of Enthalpy/Temperature calculations done.
cVol	Real	- Used in "Differential_Equations". Volume of the fireball. Unit : cu metre.
D_ambient	Real	- Used in "Differential_Equations". Ambient Density. Unit : kg / m**3.
D_ground	Real	- Used in "PMAC". Density ambient at ground level. Unit : kg / m**3.
D_log	Real	- Used in "PMAC". Log to the base ten of the ratio of required density to ground density.
d.file	File Name	- The name of the file which contains the fireball specifications to be entered into the "Bombs" program and run.
D1	Real	- Used in "Loss_Radiant". Interpolated log base ten density for lower temperature.
D2	Real	- Used in "Loss_Radiant". Interpolated log base ten density for upper temperature.
Ddash	Real	- Used in "Differential_Equations". Ratio of fireball density to D_ambient.
Delta_Enth	Real	- Used in "Find_T". Is a messure of the closeness of the enthalpy, calculated for a given temperature, to the required enthalpy.
delta_mass	Real	- Used in "Mass_Balance_Update". Difference in mass between current and previous time steps.

Den	Real	- Used in "PMAC". Ratio of required density to ground density.
density	Real	- Dummy variable (same as D_ambient) used in "ICAO_Air". Density of ambient air. Unit : kg/m**3.
Density	Real	- Used in "Differential_Equations". Density of fireball. Unit : kg/m**3.
df	Real Array	- Used in "Differential_Equations". Array of differential equations which describes fireball behaviour.
DL	Real Array	- Used in "PMAC". Log to the base ten of the density ratio taken from Gilmore.
dm	Real	- Used in "Differential_Equations". Increment in mass of Fireball. Unit : kg.
DMFXX	Real	- Read from "D.FILe" and used in "Clean_Fireball". Dry mole fraction of ambient air XX. XX : Ar,CO2,N2,O2
Dratio	Real	- Used in "PMAC". Ratio which will allow interpolation of log to the base ten density.
dt	Real	- Used in Main program and "Runge_Kutta". The time increment. Unit : second
E1	Real	- Used in "Matrix". Variables used in matrix inversion.
E2	"	- " " "
E3	"	- " " "
E1sqr	"	- " " "
E2sqr	"	- " " "
E3sqr	"	- " " "
E1nul	"	- " " "
E2nul	"	- " " "
E3nul	"	- " " "
ENTH	Real	- Used in "ENTH". Enthalpy. Unit : cal / mole
enth_ambient	Real	- Calculated in "ambient_enthalpy_". Enthalpy of ambient air. Unit : Joule / mole.
enth_fb_per_Mass	Real	- Used in "Differential_Equations". Enthalpy per kg fireball mass.
Enthal	Real Array	- Used in "Intialize" and "Find_T". Enthalpy per kg of fireball air. Unit : Joule / mole.

ENTHAL	Real Array	- Read in "read_amb_air" and used in "ambient_enthalpy". Holds enthalpy per mole of given ambient gas. Unit : Joule / mole.
enthalpy	Real Array	- Used in "CHNOS3". Array of enthalpy per mole values for species in fireball air. Unit : Joule / mole.
Enthalpy_per_kg	Real	- Used in "Initialize","CHNOS3" and "Find_T". Enthalpy per kg of fireball air. Unit : Joule / kg
entrain	Real Constant	- Read from "D.FILe" and used in "Differential_Equations". Normally equal to 1.0, it is fireball entrainment coefficient or the fraction of the fireball mass entrained.
ER	Real Array	- Read from "AMB.AIR" and holds heat of formation at 298.15 Kelvin.
f_new	Real Array	- Used in "Runge_Kutta". Equations used to describe different fireball behaviour after Runge_Kutta processing.
f_old	Real Array	- Used in "Runge_Kutta". Equations used to describe different fireball behaviour before Runge_Kutta processing.
f	Real Array	- Used in "Diferential_Equations". Equations used to describe different fireball behaviour.
Fb	Real	- Used in "Diferential_Equations". Fireball Buoyancy / (4 / 3 x pi). Unit : kg m / s**2
file_name	Character*22 Array	- Read from "D.FILe" and holds output file names.
G	Real Constant	- Read from "D.FILe" and used in "Differential_Equations". Geopotential constant equals 9.80665 m / s**2.
G	Real Array	- Used in "CHNOS3". Gibbs minimization energy of species.
GdivR	Real Constant	- Used in "ICAO_Air". = 0.034164d0.
HEIGHT	Real Array	- Used in function "MF_H2O". Array of height in metre above ground corresponding to water vapour WATERp, which contains mole fraction of H2O in ambient air.
Holdn	Real Array	- Used in "scramble" as a holder for the mole number of species per kg fireball.

IN	Integer Array	- Read from "D.FILe". IN(J) contains the position number in the "NEWDATA.DAT" file of the thermochemical data for species J.
initial_dt	Real Constant	- Read from "D.FILe" and used in "Initialize". Initial timestep Unit : Second
Initial_fb_enthalpy	Real	- Used in "Initialize". Equal to 44% of energy in fireball at t = 0 for initial fireball temperature equal to 8000.0 Kelvin. Reduces to 22% for initial fireball temperature equal to 6000 kelvin. Unit : Joule.
initial_fb_temp	Real	- Read from "D.FILe" and used in "Initialize". Start time temperature. Unit : Kelvin.
initial_momentum	Real	- Used in "Initialize". Start time momentum. Unit : kg m/s.
initial_radius	Real	- Used in "Initialize". Start time radius. Unit : Metre.
initial_z	Real	- Read from "D.FILe" and used in "Initialize". Start time height. Unit : Metre.
ISN	Integer Array	- Numeric order representation of ambient gases as present in "AMB.AIR".
JI	Integer	- Read from "D.FILe". JI holds a value which tells "CHNOS3" the Ith order that an entity was read. This allows the array variable "B" in "CHNOS3" to read the correct entity mole value. "I" ranges from 1 to 10.
Junk	Character*6	- Used to read unused data from "AMB.AIR" to allow reading of useful data on same line.
k	Real	- Read from "D.FILe" and used in "Differential_Equations". Added-mass coefficient.
l_density	Real Array	- Used in "ICAO_Air". Density at given level. Unit : kg/m**3.
l_press	Real Array	- Used in "ICAO_Air". Pressure at given level. Unit : Pascal.
l_temp	Real Array	- Used in "ICAO_Air". Temperature at given level. Unit : Kelvin.
lapse	Real Array	- Used in "ICAO_Air".
last_count	Integer Constant	- Read from "D.FILe" and used in Main program and "Output_Data".Maximum number of time steps.
level	Real Array	- Used in "ICAO_Air". Height at given level. Unit : metre.
line	Character*80	- Used in "read_amb_air" to read lines of data from "AMB.AIR" which are not used.

M	Real	- Used in "Differential_Equations". Momentum Unit : kg m / s.
Mdate	Character	- Used in "Initialize_Output". Date program run.
Mtim	Character	- Used in "Initialize_Output". Time program run.
Mass	Real	- Mass of fireball air.Unit:kg.
MFXZ	Real	- Calculated in "CLEAN_FIREBALL" and used in "ambient_enthalpy" and "GRAM_ATOM". Mole fraction of ambient XZ. XZ = Ar,H2O,CO2,N2,O2
momentum_stop	Logical	- Used in Main program, determines whether there is a time stop or a momentum stop criterion on the run.
MS	Integer	- Read from "D.FILe". MS is the number of species making up the fireball air.
mwa	Real	- Molecular weight of air.
n1	Real	- Used in "Matrix". Variables used in matrix inversion.
n2	"	- " " "
n3	"	- " " "
NEWDATA.DAT	Input File	- Used in "CHNOS3". Contains thermochemical data for species in fireball.
nXX	Real	- Used in "ambient_enthalpy". Mole number of XX per kg fireball air at 298 Kelvin. XX: Ar,C,H2O,H2,C6H6,C8H18,O2
nXZ	Real	- Dummy variable (same as anXX) used in "GRAM_ATOM". Mole number of XZ per kg of ambient air. XZ : H2O,N2,O2,CO2,Ar,O,N,C,H
nfXX	Real	- Used in "Mass_Balance_Update". Mole number of XX per Kg fireball air. XX : Ar,C,E,H,N,O,S
NgasS	Integer	- Read from "D.FILe". NgasS is the number of gaseous species in the fireball.
NNS	Integer	- Read from "D.FILe". NNS is the number of negatively charged species in the fireball.
NOM	Character Array	- Read from "D.FILe". NOM(J) contains the character discription for the "Output_Data" subroutine for species J.
NPS	Integer	- Read from "D.FILe". NPS is the number of possitively charged species in the fireball.

NS	Integer	- Read from "D.FILe". NS is the number of gaseous uncharged species in the fireball air.
NsolS	Integer	- Read from "D.FILe". NsolS is the number of condensed species in the fireball.
nTotalCharged	Real	- Used in "scramble" as the sum of all charged species.
Number_of_des	Integer	- Used in Main program and "Runge_Kutta". Number of Simultaneous Differential Equations.
OLD_MASS	Real	- Used in "Differential_Equations" and "Mass_Balance_Update". Mass from previous time step.
Op_fb	Real	- Used in "Loss_Radiant". Opacity of Fireball. Unit : cm**-1.
OpTr	Real	- Used in "Loss_Radiant". Opacity x Transmission coefficient.
P_ambient	Real	- Used in "Differential_Equations". Ambient Pressure. Unit : Pascal.
pgrm	Character	- Used in "Initialize_Output".
pii	Real Constant	- 3.14159d0
PMAC	Real	- Used in "PMAC". Planck Mean Absortion Coefficient of fireball for given temperature. Unit : cm**-1.
PMAC_Log	Real Array	- Used in "PMAC". Log to the base ten of Planck Mean Absortion Coefficients at differing densities and temperatures.
PMAC_log10	Real	- Used in "PMAC". Log to the base ten of Planck Mean Absortion Coefficients at required density and temperature.
pressure	Real	- Dummy variable (same as P_ambient) used in "ICAO_Air" representing ambient pressure at given level. Unit : Pascal.
PTOT	Real	- Used in "CHNOS3". Pressure Unit : Atmosphere.
R	Real Constant	- Used in "CHNOS3". Gas constant. Units : cal/Kelvin/mole. (=1.98717).
rad	Real	- Dummy variable (same as radius) used in "Loss_Radiant". Radius of fireball. Unit : metre.

rad_loss0	Real	- Used in "Initialize" and "Output_Data". Radiation lost to far field at time zero. Radiant Fireball flux at t=0.
Radiant_Loss	Real	- Used in "Loss_Radiant" and "Differential_Equations". The energy lost as radiation to the far field. Unit : Joule.
RadInCM	Real	- Used in "Loss_Radiant". Radius of fireball. Unit : cm.
radius	Real	- Used in "Differentail_Equations". Radius of Fireball. Unit : metre.
RATIO	Real	- Used in "MF_H2O". Used to interpolate for the mole fraction of H2O in ambient air given the height.
RMW	Real	- Used in "ambient_enthalpy" and "GRAM_ATOM". Relative molecular weight of ambient air. Unit : gram / mole.
Setting_up	Logical	- Used in function "PMAC". Allows the function to read the data, required for calculation, only the first time that the function is entered.
sigma	Real	- Read from "D.FILe" and used in "Loss_Radiant". Stefan-Boltzmann contant.
Species_num	Integer	- Used in "scramble". Read from "D.FILe" as number of species in fireball.
SR	Real Array	- Calculated in "read_in_CHNOS" and used in ENTR. Enthalpy of fireball species at 298.15 Kelvin.
SR	Real Array	- Used in "read_amb_air" to read from "AMB.AIR" the entropy at 298.15 Kelvin.
start_time	Real Constant	- Used in "Initailize" as time that program starts at. Unit : second.
SUMPS	Real	- Used in "scramble" as the sum of the possitively charged species.
SUMNS	Real	- Used in "scramble" as the sum of the negatively charged species.
t	Real	- Used in "Runge_Kutta" as time variable. Unit : second.
T	Real	- Used in "ENTH", "Loss_Radiant". Temperature. Unit : Kelvin.
T_ambient	Real	- Used in "Differential_Equations". Ambient Temperature. Unit : Kelvin.

T_fb	Real	- Used in "Differentail_Equations", "Loss_Radiant". Temperature of Fireball. Unit : Kelvin.
T1	Real	- Used in "Matrix". Variables used in matrix inversion.
T2	"	- " " "
T3	"	- " " "
TDMF	Real	- Used in "Clean_Fireball". Sum of dry mole fractions of molecules making up ambient air.
temp	Real	- Dummy variable (same as T_ambient) used in "ICAO_Air"and "ambient_enthalpy". Temperature at given level. Unit : Kelvin.
temper	Real Array	- Used in "Intialize" and "Find_T". Temperature corresponding to enthalpy per kg of fireball. Unit : Joule / mole.
time Time	Real	- Current time. Unit : second.
times_up	Real Constant	- Read from "D.FILe" and used in Main program to stop the program when the time reaches "times_up". Unit : second.
TL	Real	- Used in "ENTH","CENTH" and "ENTR". Temperature lower limit. Unit : Kelvin.
TL	Real Array	- Used in "PMAC". Array of temperatures corresponding to PMAC values given by Gilmore.
TMF	Real	- Used in "Clean_Fireball". Sum of mole fractions of molecules making up ambient air [including H2O].
Tnum	Integer	- Read from "D.FILe" as number of gases in ambient air.
TNT_equivalent	Real Constant	- Used in "Initalise" as conversion factor for energy in joule of given size bomb. Unit : Joule / Megaton (4.186e15 joule per megatom).
TotalGasMole	Real	- Used in "scramble" as the mole sum of all gaseous species per Kg fireball.
TotalMole	Real	- Used in "scramble" as the mole sum of all species per kg fireball.
TotalnfXX	Real	- Used in "Mass_Balance_Update" as the total fireball mole of XX. XX : Ar,C,E,H,N,O,S
TotalSolidMole	Real	- Used in "scramble" as the mole sum of all condensed species per kg fireball.
Tr	Real	- Used in "Loss_Radiant". Transmissivity.

Tratio	Real	- Used in "PMAC". Temperature ratio which will allow for interpolation of log to base ten of PMAC.
TU	Real	- Used in "ENTH","CENTH" and "ENTR". Temperature upper limit. Unit : Kelvin.
u	Real	- Used in "Differential_Equations". Speed of Fireball. Unit : metre / second.
V	Real	- Used in "Differential_Equations". Dimensionless mass of fireball.
vapour	Logical	- Dummy variable (same as water_include) used in "Clean_Fireball". Determines whether water vapour to be included in ambient air.
W	Real Constant	- Read from "D.FILe" and used in "Initialise". Yield of blast in Megaton. Unit : Megaton.
water	Character*3	- Used in "Initialise_Output". Allows the output to state whether water vapour included in the system.
WATERp	Real Array	- Used in "MF_H2O". Array of mole fractions of H2O in ambient air. Corresponds to array variable HEIGHT which contains the height above ground in metre.
water_include	Logical	- Read from "D.FILe" and determines whether water vapour is to be included in the calculations.
weight	Real	- Used in "Runge_Kutta".
Y	Real Array	- Used in "scramble" as the number of mole of the species per kg fireball.
z	Real	- Used in "ICAO_Air", "MF_H2O". Height above ground. Unit : Metre.
Z_fb	Real	- Used in "scramble" as the dissociation coefficient.
Zdsh	Real	- Used in "scramble" as the dissociation coefficient. Formula : TotalGasMole * mwa / 1000.0d0

```
  ccccccccccccccccccccc      D.FILE      ccccccccccccccccccccc

N                  : Time stop else Momentum stop
0.00               : time for run
0.44               : beta
9.80665            : 9.80665 Gravity ; Metre per second squared.
0.10               : entrain (noramally 0.1)
0.01               : Initial increment in time (normally 0.02)
8000.0             : Initial fireball temperature
0.00000            : Initial detonation height
0.5                : k
30000              : Last count
28.9607            : Molecular Weight Of Ambient Air
0.567d-07          : 0.567d-07 sigma : Stefan-Boltzmann constant
N2/O2/Ar/H2O/CO2   : Type of atmosphere used
FBOMBS81.dat       : file_name(1)
FBOMBS82.dat       : file_name(2)
FBOMBS83.dat       : file_name(3)
FBOMBS84.dat       : file_name(4)
FBOMBS85.dat       : file_name(5)
FBOMBS86.dat       : file_name(6)
FBOMBS87.dat       : file_name(7)
FBOMBS88.dat       : file_name(8)
FBOMBS89.dat       : file_name(9)
1.00               : Bomb size (Mt)
Y                  : Water vapour included in ambient air calculations [Y/N]
0.780836           : (0.78084) DMFN2  ; dry mole fraction of N2
0.209481           : (0.20948) DMFO2  ; dry mole fraction of O2
0.000340           : (0.00034) DMFCO2 ; dry mole fraction of CO2
0.009343           : (0.009343) DMFAr  ; dry mole fraction of Argon
49                 : Total number of species.
48                 : Total number of gaseous species.
34 1 ...nCH... 3 ..nC(G).. 6 ..nCH2... 10 ...nCN... 12 ..nCN2...
14 ...nCO... 16 ..nCO2... 19 ...nC2... 21 ..nC2H2.. 24 ..nC2N...
27 ...nC3... 29 ...nC4... 31 ...nC5... 32 ...nH.... 33 ...nHN...
34 ..nHNO... 35 .nHNO2g.. 36 .nHNO2gl. 38 ...nHO... 39 ..nHO2...
41 ...nH2... 42 ..nH2N... 44 ..nH2O... 45 ..nH2O2.. 50 ...nN....
51 ...nNO... 52 ..nNO2... 55 ...nN2... 56 ..nN2O... 60 ...nN3...
61 ...nO.... 64 ...nO2... 66 ...nO3... 122 ...nAr...
: Total number and description of uncharged species.
Y                  : Charged species included [Y/N]
10 85 ..nNO+... 97 ...nH+... 98 ..nAr+... 99 ...nC+...
104 ...nN+... 105 ..nN2+... 106 ...nO+... 107 ..nO2+... 111 ..nCN+...
114 ..nHO+...
: Total number and description of possitively charged species.
4 93 ...nE-... 95 ...nO-... 100 ...nN-... 101 ..nN2-...
: Total number and description of negatively charged species.
1 89 ..nC(s)..
: Total number and description of non gaseous species.
 C=1  H=2  N=3  O=4  S=7 Ar=5  E=6   : C=1,H=2,N=3,O=4,S=5,Ar=6,E=(+,-)=7
: Order of introduction of the different entities.
5 1 2 3 4 5                  : From "N2.det". Ambient air gaseous species.
0.000                        : Fractional part of H2 in H2O
```

```
ccccccccccccccccccccccccc  COMMON.PUT  ccccccccccccccccccccccccc

LOGICAL Charged
LOGICAL water_include,Setting_up,momentum_stop
INTEGER count,Number_of_des,TYPE,counter
INTEGER Species_num,NS,NPS,NNS,NsolS,NgasS,MS
CHARACTER file_name*22,NOM*9,Atmos*20
REAL*8 initial_dt,initial_fb_temp,initial_z,k
REAL*8 Initial_fb_enthalpy,MtC
COMMON /OneA/ file_name(9),times_up,TYPE,MtC
COMMON /One/count,df(5),dt,f(5),k,last_count,M,rad_loss0
COMMON /two/chan_z,cVol,Ddash,Density,Fb,u,G
COMMON /three/initial_dt,initial_fb_temp,initial_z
COMMON /four/beta,D_ambient,entrain,Initial_fb_enthalpy,pii,W,mwa
COMMON /five/enth_ambient,TotalMole,Number_of_des
COMMON /six/atime,Mass,P_ambient,radius,T_ambient,T_fb,z,Z_fb
COMMON /seven/enth_fb_per_Mass,Enthalpy_per_kg
COMMON /eight/Charged,Setting_up,momentum_stop
COMMON /uno/TotalnfN,TotalnfO,TotalnfH,TotalnfC
COMMON /due/TotalnfAr,TotalnfS,TotalnfE
COMMON /quat/nfN,nfO,nfAr,nfC,nfS,nfH,nfE
COMMON /B/Op_fb,Tr,Radiant_Loss,sigma
COMMON /h2o/ water_include,Atmos
COMMON /CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
COMMON /hol/ Holdn(50),TotalGasMole,TotalSolidMole,Zdsh
COMMON /ms1/ Species_num,NOM(50),NS,NPS,NNS,NsolS,NgasS,IN(50),MS
COMMON /out/ nN,nNO,nN2,nO,nO2,nTotalCharged
COMMON /output/OpTr
COMMON /USE/Enthal(2000),temper(2000),cala,calb,calc,counter
```

```
      PROGRAM BOMBS

CC*****************************************************************CC
CC    This program requires three input files, namely D.FILe,    CC
CC    NEWDATA.DAT and AMB.AIR.                                   CC
CC    The file D.FILE contains the following :                   CC
CC        - whether or not a stop time is set for the run and    CC
CC          the value of the stop time.                          CC
CC        - the values of the following constants :              CC
CC            - beta, the fraction of bomb yield in the          CC
CC              fireball at start time                           CC
CC            - k, the added-mass coefficient                    CC
CC            - entrain, the fireball entrainment coefficient    CC
CC            - G, gravity                                       CC
CC            - sigma, Stefan-boltzmann constant                 CC
CC            - mwa, the molecular weight of ambient air         CC
CC        - the maximum number of time steps to be carried out   CC
CC        - whether or not water vapour is to be included in     CC
CC          the ambient air calculations                         CC
CC        - the type of atmosphere used                          CC
CC        - the nine output file names                           CC
CC        - the bomb size, initial detonation height, initial    CC
CC          fireball temperature and initial time increment      CC
CC        - the dry mole fractions of ambient air gases          CC
CC        - the total number of species, the number of gaseous   CC
CC          and non-gaseous species                              CC
CC        - whether or not charged species are included          CC
CC        - the total number and description of uncharged,       CC
CC          possitively charged and negatively charged species   CC
CC        - the order of introduction of the different species   CC
CC        - the number of ambient gases and their numerical      CC
CC          order as present in the AMB.AiR data file.           CC
CC                                                               CC
CC    The files NEWDATA.DAT and AMB.AIR contain the thermo-      CC
CC    chemical data for all the species in the fireball and the  CC
CC    ambient air gases respectively.                            CC
CC                                                               CC
CC*****************************************************************CC

      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      Number_of_des           = 5

      Call read_in_data
      Call Initialize

      Call Initialize_Output
      count = 0
   20 if (mod(count,5) .EQ. 0 .OR. count .EQ. last_count) Call
     *Output_Data
      if      (atime .GT. 0.299) dt = 0.025d0
      if      (atime .GT. 2.99)  dt = 0.25d0
      if      (atime .GT. 49.99) dt = 0.5d0
      if      (atime .GT. 99.99) dt = 1.0d0
      Call Runge_Kutta(Number_of_des,atime)
      Call Find_T
      count = count + 1
```

```
      if (momentum_stop) then
        if (M .GE. 0) go to 20
      else
       if (atime .LT. times_up) go to 20
      endif
      Call Output_Data
      Do 25 I = 1,9
       close(I+1)
   25 continue
      Stop
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE Initialize

      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      Real*8 initial_momentum,initial_radius

      counter                = 0
      initial_momentum       = 0.0d0
      initial_radius         = 0.0d0
      TNT_equivalent         = 4.186d15
      start_time             = 0.0d0
      Setting_up             = .true.
      pii                    = 3.14159d0
C
C     Call to SUBROUTINE CLEAN_FIREBALL. This sets up the data
C     going into "CHNOS3" for a non-surface blast condition.
C     Also set conditions for composition of the atmosphere.
C
      Call CLEAN_FIREBALL(initial_z,water_include,MFH2O,MFN2,MFO2,MFCO2,
     *MFAr)
C
C     Call to SUBROUTINE GRAM_ATOMS. This returns the number of
C     mole of the of each element, N,O,H,C and Ar given the mole
C     fractions of the molecules per kilogram of ambient air.
C
      CALL GRAM_ATOM(MFH2O,MFN2,MFO2,MFCO2,MFAr,nfN,nfO,nfH,nfC,nfAr)
C
      nfS            = 0.0d0
      nfE            = 0.0d0
      enth_fb_per_Mass = 0.0d0
      z              = initial_z
      radius         = initial_radius
      dt             = initial_dt
      atime          = start_time
      CALL ICAO_Air(initial_z,T_ambient,P_ambient,D_ambient)
C
C     It is necessary to calculate the enthalpy of the species, which
C     will make up the fireball, at ambient enthalpy. This will be
C     subtracted from the enthalpy at initial fireball temperature.
C     Hence the Mass of the fireball will be calculated, given the
C     knowledge of the initial fireball energy.
C
      T_fb = initial_fb_temp
      write(0,4) nfN,nfO,nfC,nfH,nfAr,nfS,nfE
```

```
C
      Call read_in_CHNOS
      Call scramble(T_fb,P_ambient)
C
      Call read_amb_air
      Call ambient_enthalpy(initial_z,T_ambient)
C
      write(0,15) enth_ambient
      Initial_fb_enthalpy = beta * W * TNT_equivalent
      temper(1)      = initial_fb_temp
      Mass           = Initial_fb_enthalpy/(Enthalpy_per_kg-enth_ambient)
      Enthal(1)      = Initial_fb_enthalpy / Mass
      f(1)           = initial_z
      f(2)           = initial_momentum
      f(3)           = 3.0d0 * Mass / 4.0d0 / pii
      f(4)           = Enthalpy_per_kg
C
C     f(5) is energy radiated from fireball, assumed zero at start
C
      f(5)           = 0.0d0
      write(0,10) Mass,Initial_fb_enthalpy,Enthalpy_per_kg
      TotalnfN       = Mass * nfN
      write(0,12) TotalnfN
      TotalnfO       = Mass * nfO
      write(0,12) TotalnfO
      TotalnfC       = Mass * nfC
      write(0,12) TotalnfC
      TotalnfH       = Mass * nfH
      write(0,12) TotalnfH
      TotalnfAr      = Mass * nfAr
      write(0,12) TotalnfAr
      TotalnfS       = Mass * nfS
      write(0,12) TotalnfS
      TotalnfE       = Mass * nfE
      write(0,12) TotalnfE
C
      Call Differential_Equations
C
      rad_loss0 = df(5)
C
C------ALL THE FORMAT STATEMENTS
C
    4 Format(' nfN = ',d10.4,' nfO = ',d10.4,' nfC = ',d10.4,
     *' nfH = ',d10.4,' nfAr = ',d10.4,' nfS = ',d10.4,' nfE = ',d10.4)
   10 Format(' Mass = ',d12.6,' kg  FB_ENTH = ',D12.6,' J  enth/kg = ',
     *d12.6,' J/kg')
   12 Format(' IT = ',d20.14)
   15 Format(' enth_ambient = ',d14.8)
      RETURN
      END
```

```
      SUBROUTINE CLEAN_FIREBALL(hite,vapour,MFH2O,MFN2,MFO2,MFCO2,MFAr)
C
C.......Calculates the molar fractions of ambient air gases for a
C.......non-surface blast condition.
C.......DMF indicates molar fraction of species in dry air.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      Logical vapour
      Common /now/ DMFN2,DMFO2,DMFCO2,DMFAr

      IF (vapour) THEN
        MFH2O = MF_H2O(hite)
        ELSE
        MFH2O    = 0.000D0
      ENDIF
      MFN2           = DMFN2  * (1.0D0 - MFH2O)
      MFO2           = DMFO2  * (1.0D0 - MFH2O)
      MFCO2          = DMFCO2 * (1.0D0 - MFH2O)
      MFAr           = DMFAr  * (1.0D0 - MFH2O)
      TMF = MFH2O + MFN2 + MFO2 + MFCO2 + MFAr
      if (dabs(1.0d0 - TMF) .GE. 1.0d-5) then
        write(0,4) TMF,MFH2O,MFN2,MFO2,MFCO2,MFAr
        Stop
      endif
C
C-----ALL THE FORMAT STATEMENTS
C
    4 Format(' TMF should equal 1.00, but instead it equals ',D20.14,'.'
     *,/,' This program is now terminated. Check the data file "d.file".
     * ',/,' MFH2O = ',d14.8,' MFN2 = ',d14.8,' MFO2 = ',d14.8,/,
     *' MFCO2 = ',d14.8,' MFAr = ',d14.8)
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      DOUBLE PRECISION FUNCTION MF_H2O(z)
C
C.....Returns the volumetric mixing ratio (V/V) or molar fraction of
C.....H2O at a given height.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)
      DIMENSION HEIGHT(17),WATERp(17)
      INTEGER I
      DATA HEIGHT/0.001D3,2.0D3,4.0D3,6.0D3,8.0D3,10.0D3,12.0D3,14.0D3,
     *16.0D3,18.0D3,20.0D3,22.0D3,24.0D3,26.0D3,28.0D3,30.0D3,31.0D3/
      DATA WATERp/10129.0D-6,6110.0D-6,3055.0D-6,1447.0D-6,434.0D-6,
     *59.0D-6,37.0D-6,16.0D-6,14.0D-6,10.0D-6,10.0D-6,10.0D-6,10.0D-6,
     *10.0D-6,10.0D-6,10.0D-6,10.0D-6/

      IF (z .GT. 31.0d3) THEN
        MF_H2O = 10.0D-6
        RETURN
      ENDIF
      IF (z .LE. HEIGHT(1)) THEN
        MF_H2O = 10129.0D-6
        RETURN
      ENDIF
      I = 0
```

```
    1 I = I + 1
      IF (z .LT. HEIGHT(I + 1)) THEN
        RATIO = (z - HEIGHT(I)) / (HEIGHT(I + 1) - HEIGHT(I))
        MF_H2O = WATERp(I) + (RATIO * (WATERp(I + 1) - WATERp(I)))
        RETURN
      ELSE
       IF (I.LT.16) THEN
        GOTO 1
       ELSE
        WRITE(0,2)
    2   FORMAT(' ERROR ....... SOMTHING WRONG IN DB FUNCTION MF_H2O ')
        STOP
       ENDIF
      ENDIF
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE GRAM_ATOM(MFH2O,MFN2,MFO2,MFCO2,MFAr,nN,nO,nH,nC,nAr)
C
C.....Returns the number of moles of each of the elements N,O,H,C and Ar
C.....given the mole fractions of the molecules per kilogram of ambient
C.....air.It first calculates the relative molecular weight of the
C.....system.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      RMW  = (MFH2O * 18.015D0) + (MFN2 * 28.014D0) + (MFO2 * 31.998D0)
     1     + (MFCO2 * 44.009D0) + (MFAr * 39.950D0)

      nH2O = MFH2O * 1000.0D0 / RMW
      nN2  = MFN2  * 1000.0D0 / RMW
      nO2  = MFO2  * 1000.0D0 / RMW
      nCO2 = MFCO2 * 1000.0D0 / RMW
      nAr  = MFAr  * 1000.0D0 / RMW
      nO   = nH2O + (2.0d0 * nCO2) + (2.0d0 * nO2)
      nN   = 2.0d0 * nN2
      nC   = nCO2
      nH   = 2.0d0 * nH2O
      nAr  = nAr
      RETURN
      END
```

```
      SUBROUTINE ambient_enthalpy(z,temp)
C
C.....Calculates relative molecular weight of new ambient air
C.....composition and enthalpy of 1kg ambient air.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      Logical water_include
      integer Tnum,Number_of_des
      CHARACTER Atmos*20

      COMMON /five/enth_ambient,TotalMole,Number_of_des
      common /h2o/ water_include,Atmos
      common /amb/ Tnum,ISN(6),ENTHAL(6)

      Save

      Do 40 J = 1,Tnum
       Do 50 I = 1,6
        if (ISN(J) .EQ. I) then
         ENTHAL(J) = ENTH(J,temp) * 4.186D+00
        endif
   50  continue
   40 continue
C
      Call CLEAN_FIREBALL(z,water_include,MFH2O,MFN2,MFO2,MFCO2,MFAr)
C
      RMW  = (MFH2O * 18.015D0) + (MFN2 * 28.014D0) + (MFO2 * 31.998D0)
     1     + (MFCO2 * 44.009D0) + (MFAr * 39.950D0)
      nH2O = MFH2O * 1000.0D0 / RMW
      nN2  = MFN2  * 1000.0D0 / RMW
      nO2  = MFO2  * 1000.0D0 / RMW
      nCO2 = MFCO2 * 1000.0D0 / RMW
      nAr  = MFAr  * 1000.0D0 / RMW
      nH2  = 0.0d0
      enth_ambient = (ENTHAL(1) * nN2)  + (ENTHAL(2) * nO2) +
     1              (ENTHAL(3) * nAr) + (ENTHAL(4) * nH2O) +
     2               (ENTHAL(5) * nCO2)  + (ENTHAL(6) * nH2)
  120 RETURN
      END
```

```
      FUNCTION ENTH(I,T)
C
C.....Calculates the enthalpy of the ambient air species for temperature
C.....T, where T is up to 6000 Kelvin.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      Common /det/ ER(6),C(6,6)
C
      ENTH=ER(I)
      IF(T.LT.298.15) THEN
       TL = 298.15
       TU = T
       ENTH=ENTH + (C(I,1)
     1           +  C(I,2)*TL
     2           +  C(I,3)*TL*TL
     3           +  C(I,4)*(TL**(-2.0)))*(TU-TL)
       RETURN
      ENDIF
      TL=298.15
      TU=T
      ENTH=ENTH+C(I,1)*(TU-TL)
     1         +C(I,2)*(TU*TU-TL*TL)/2.
     2         +C(I,3)*(TU*TU*TU-TL*TL*TL)/3.
     3         -C(I,4)*(1./TU-1./TL)
      RETURN
      END
C
C-------------------------------------------------------------------------
C
      SUBROUTINE ICAO_Air(z,temp,pressure,density)
C
C.....Calculates the temperature, pressure and density of the ambient
C.....air from the height of the fireball.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      Real*8 level(6),lapse(6),l_temp(6),l_press(6),l_density(6)
      Data level/ 0.0000,11000.0,20000.0,32000.0,47000.0,51000.0/
      Data lapse/-0.0065,0.000,0.001,0.0028,0.000,-0.0028/
      Data l_temp /288.150,216.6,216.6,228.6,270.6,270.6/
      Data l_press/101325.,22630.3,5475.00,867.921,110.988,66.9270/
      Data l_density/1.225d+0,3.6394d-1,8.803d-2,1.3225d-2,1.4277d-3,
     *8.615d-4/

      GdivR                   = 0.034164d0
      J = 0
   10 J = J + 1
      if (z .GE. level(J+1)) go to 10
      temp = l_temp(J) + lapse(J) * (z - level(J))
      if (lapse(J) .eq. 0.0) then
       pressure = l_press(J) *  dexp(-GdivR*(z-level(J))/l_temp(J))
       density =  l_density(J) * pressure / l_press(J)
      else
       pressure = l_press(J) * ((temp/l_temp(J))**(-GdivR/lapse(J)))
       density = l_density(J) * ((temp/l_temp(J))**(-((GdivR/lapse(J))+1
     1 .0)))
      endif
      RETURN
      END
```

```
      SUBROUTINE scramble(temp,pressure)
C
C.....Stores all final molar amounts for species in fireball in Holdn
C.....array and calculates the sum of all gaseous species, all solid
C.....species, all species, all negatively charged species and all
C.....positively charged species.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

C
      Call CHNOS3(temp,pressure,nfN,nfO,nfH,nfC,nfAr,nfS,nfE,Enthalpy_pe
     *r_kg)
C
      do 14 I=1,Species_num
   14  Holdn(I) = Y(I)

      TotalGasMole = 0.0d0

      Do 16 I = 1,NgasS
   16 TotalGasMole = TotalGasMole + Y(I)

      Zdsh = TotalGasMole * mwa / 1000.0d0

      TotalSolidMole = 0.0d0

      Do 18 I = NgasS+1,Species_num
   18 TotalSolidMole = TotalSolidMole + Y(I)

      TotalMole = TotalGasMole + TotalSolidMole
      Z_fb = TotalGasMole * mwa / 1000.0d0

      SUMPS = 0.0d0
      SUMNS = 0.0d0

      Do 24 I = NS+1,NS+NPS
   24 SUMPS = SUMPS + Y(I)

      Do 26 I = NS+NPS+1,NS+NPS+NNS
   26 SUMNS = SUMNS + Y(I)

      nTotalCharged = SUMPS - SUMNS

      RETURN
      END
```

```
      SUBROUTINE CHNOS3(temp,pressure,aN,aO,aH,aC,aAr,aS,aE,Enthalpy_per
     *_kg)
C
C.....Calculates the enthalpy per kg of fireball air given the pressure,
C.....temperature and molar amounts of each element in the fireball air.
C
C     TEST PROGRAM FOR FREE ENERGY MINIMIZATION ROUTINE
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      DIMENSION enthalpy(50)
C
      COMMON/AAA/A(50,10),AKT(50),AKTF(50),B(10),BO(10),G(50)
      COMMON/BBB/IVAR,KH(20),L,L1,M,M1,MA,MB,MP,MS,MF(15),ML(15)
      COMMON/CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
      common /xaa/ MG,MR
      COMMON/NINE/J1,J2,J3,J4,J5,J6,J7
C
C      ENTER GM ATOMS OF ELEMENTS
C
      B(J1) = aC
      B(J2) = aH
      B(J3) = aN
      B(J4) = aO
      B(J5) = aS
      B(J6) = aAr
      B(J7) = aE
C
C      ENTER TEMPERATURE, PRESSURE
C
      T    = temp
      PTOT = pressure / 101325.0d0
C
C      CALCULATE GIBBS ENERGIES OF SPECIES
C
      R=1.98717
      DO 70 I=1,MS
       G(I)=(CENTH(I,T)-T*ENTR(I,T))/R/T
C
C     The value of CENTH(I,T) has units Kcal/mole.
C     However enthalpy(I) needs to have units of J/mole.
C     It is then necessary to multiply CENTH(I,T)
C     by 4.186D+00 to convert cal/mole to J/mole.
C
       enthalpy(I) = 4.186D+00 * CENTH(I,T)
   70 CONTINUE
C
C      ENTER INITIAL GUESS
C
      YTOT(1)=MG
      DO 75 I=1,MS
       Y(I)=1.
   75 CONTINUE
C
      CALL GAUSS
C
      IF(M.GT.MP)THEN
       WRITE(0,117) M,MP
       stop
      ENDIF
```

```
      Enthalpy_per_kg = 0.0d0
      DO 80 I=1,MS
       Enthalpy_per_kg = Enthalpy_per_kg + (enthalpy(I) * Y(I))
   80 CONTINUE
      RETURN
  117 FORMAT(' M = ',I3,'  MP = ',I3,/' EQUILIBRIUM NOT ATTAINED')
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE Find_T
C
C.....Finds the temperature that corresponds to the enthalpy.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      counter = counter + 1
      if (counter .LT. 4) then
       Enthal(counter) = enth_fb_per_Mass
   20  Call scramble(T_fb,P_ambient)
       Delta_Enth = (enth_fb_per_Mass-Enthalpy_per_kg)/
     * (enth_fb_per_Mass)
       if (dabs(Delta_Enth) .GT. 0.0001d0) then
        T_fb = T_fb * sqrt((Delta_Enth + 1.0D0))
        goto 20
       else
        temper(counter)=T_fb
       endif
      else
   40  Call matrix
       T_fb = cala * (enth_fb_per_Mass * enth_fb_per_Mass) + calb *
     *enth_fb_per_Mass + calc
       If (T_fb .LT. 000.0d0. OR. T_fb .GT. 10000.0d0) Call Show_Status
       Call scramble(T_fb,P_ambient)
       temper(counter)=T_fb
       Enthal(counter)=Enthalpy_per_kg
       Delta_Enth = (enth_fb_per_Mass - Enthalpy_per_kg)
     *              / (enth_fb_per_Mass)
       if (dabs(Delta_Enth) .GT. 0.001) then
        if (dabs(temper(counter)-temper(counter-1)) .LT. 1.0) then
         if ((temper(counter-2)-temper(counter-1)) .GE. 0.0) then
          temper(counter) = temper(counter-1) - 1.0
         else
          temper(counter) = temper(counter-1) + 1.0
         endif
         Call scramble(temper(counter),P_ambient)
         Enthal(counter) = Enthalpy_per_kg
        endif
        counter = counter + 1
        goto 40
       endif
      endif
      RETURN
      END
```

```
      SUBROUTINE matrix
C
C.....A 2nd order polynomial fit used by Find_T to approximate guess
C.....temperature corresponding to enthalpy.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      Integer counter
      Real*8 T1,T2,T3,E1,E2,E3,E1sqr,E2sqr,E3sqr,E1nul,E2nul,E3nul
      Real*8 cala,calb,calc,temper,Enthal
      Real*8 n1,n2,n3
      Common /USE/Enthal(2000),temper(2000),cala,calb,calc,counter

      T1    = temper(counter - 3)
      T2    = temper(counter - 2)
      T3    = temper(counter - 1)
      E1sqr = Enthal(counter - 3) ** 2.0d0
      E1    = Enthal(counter - 3) ** 1.0d0
      E1nul = Enthal(counter - 3) ** 0.0d0
      E2sqr = Enthal(counter - 2) ** 2.0d0
      E2    = Enthal(counter - 2) ** 1.0d0
      E2nul = Enthal(counter - 2) ** 0.0d0
      E3sqr = Enthal(counter - 1) ** 2.0d0
      E3    = Enthal(counter - 1) ** 1.0d0
      E3nul = Enthal(counter - 1) ** 0.0d0
      n1    = E2sqr / E1sqr
      n2    = E3sqr / E1sqr
      E2sqr = 0.0d0
      E2    = E2 - (n1 * E1)
      E2nul = E2nul - (n1 * E1nul)
      E3sqr = 0.0d0
      E3    = E3 - (n2 * E1)
      E3nul = E3nul - (n2 * E1nul)
      T2    = T2 - (n1 * T1)
      T3    = T3 - (n2 * T1)
      n3    = E3 / E2
      E3    = E3 - (n3 * E2)
      E3nul = E3nul - (n3 * E2nul)
      T3    = T3 - (n3 * T2)
      calc  = T3 / E3nul
      calb  = (T2 - (E2nul * calc)) / E2
      cala  = (T1 - (E1 * calb) - (E1nul * calc)) / E1sqr
      RETURN
      END
```

```
      FUNCTION CENTH(I,T)
C
C.....Calculates the enthalpy of any fireball species at any temperature
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/DAT/C1(50,5),C2(50,5),C3(50,5),C4(50,5)
      COMMON/DAT/HR(50),SR(50),NR(50),HH(50,5),TM(50,5)
C
      CENTH=HR(I)
      N=NR(I)
      IF (T.LT.298.15) THEN
       J = 1
       TL = 298.15
       TU = T
       CENTH=CENTH + (C1(I,J)
     1           +  C2(I,J)*TL
     2           +  C3(I,J)*TL*TL
     3           +  C4(I,J)*(TL**(-2.0)))*(TU-TL)
       CENTH=CENTH+HH(I,J)
      ELSE
       DO 30 J=1,N
        TL=298.15
        IF (J.GT.1) TL=TM(I,J-1)
        TU=T
        IF (T.GE.TM(I,J)) TU=TM(I,J)
        CENTH=CENTH+C1(I,J)*(TU-TL)
     1         +C2(I,J)*(TU*TU-TL*TL)/2.
     2         +C3(I,J)*(TU*TU*TU-TL*TL*TL)/3.
     3         -C4(I,J)*(1./TU-1./TL)
        IF (T.LT.TM(I,J)) GOTO 40
        CENTH=CENTH+HH(I,J)
   30   CONTINUE
        IF (T .GT. TM(I,N)) THEN
         TL = TM(I,N)
         TU = T
         CENTH = CENTH + (C1(I,N)
     1           +  C2(I,N)*TL
     2           +  C3(I,N)*TL*TL
     3           +  C4(I,N)*(TL**(-2.0)))*(TU-TL)
        ENDIF
       ENDIF
   40  RETURN
       END
```

```
      FUNCTION ENTR(I,T)
C
C.....Calculates the entropy of any fireball species at any temperature.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/DAT/C1(50,5),C2(50,5),C3(50,5),C4(50,5)
      COMMON/DAT/HR(50),SR(50),NR(50),HH(50,5),TM(50,5)
C
      ENTR=SR(I)
      N=NR(I)
      IF (T.LT.298.15) THEN
       J = 1
       TL = 298.15
       TU = T
       ENTR=ENTR + (C1(I,J)
     1          +  C2(I,J)*TL
     2          +  C3(I,J)*TL*TL
     3          +  C4(I,J)*(TL**(-2.0)))*DLOG(TU/TL)
       ENTR=ENTR+HH(I,J)/TM(I,J)
      ELSE
       DO 30 J=1,N
        TL=298.15
        IF (J.GT.1) TL=TM(I,J-1)
        TU=T
        IF (T.GE.TM(I,J)) TU=TM(I,J)
        ENTR=ENTR+C1(I,J)*LOG(TU/TL)
     1         +C2(I,J)*(TU-TL)
     2         +C3(I,J)*(TU*TU-TL*TL)/2.
     3         -C4(I,J)*(1./TU/TU-1./TL/TL)/2.
        IF (T.LT.TM(I,J)) GOTO 40
        ENTR=ENTR+HH(I,J)/TM(I,J)
   30  CONTINUE
       IF (T .GT. TM(I,N)) THEN
        TL = TM(I,N)
        TU = T
        ENTR=ENTR + (C1(I,N)
     1          +  C2(I,N)*TL
     2          +  C3(I,N)*TL*TL
     3          +  C4(I,N)*(TL**(-2.0)))*DLOG(TU/TL)
       ENDIF
      ENDIF
   40 RETURN
      END
```

```
      SUBROUTINE GAUSS
C
C.....Calculates the number of moles of each species in fireball air per
C.....kilogram of fireball air.
C
C       ERIKSSON FREE ENERGY MINIMIZATION ROUTINE
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/AAA/A(50,10),AKT(50),AKTF(50),B(10),BO(10),G(50)
      COMMON/BBB/IVAR,KH(20),L,L1,M,M1,MA,MB,MP,MS,MF(15),ML(15)
      COMMON/CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
C
      DIMENSION F(50),IFAS(10),ISOL(10),NSUM(200),OPI(20)
      DIMENSION R(20,21),YFTOT(20),YX(50)
C
      IS=-1
      MSUM=0
      NG=0
    1 ISUM=0
      MSA=0
      IF (MS.GE.M1) THEN
       DO 2 I=M1,MS
        IF (Y(I).NE.0.) THEN
         MSA=MSA+1
         ISOL(MSA)=I
         ISUM=ISUM+2**(I+MP-M1)
        ENDIF
    2  CONTINUE
      ENDIF
    3 MPA=0
      NG=NG+1
      IF (NG.EQ.501) NG=1
      NSUM(NG)=ISUM
      YSUM=0.
      DO 4 M=1,MP
       IF (YTOT(M).NE.0.) THEN
        MPA=MPA+1
        IFAS(MPA)=M
        NSUM(NG)=NSUM(NG)+2**(M-1)
        YSUM=YSUM+YTOT(M)
       ENDIF
    4 CONTINUE
      IF (NSUM(NG).GE.IS.AND.MPA+MSA.LE.L) GOTO 11
    5 IS=IS+2
      DO 6 N=1,MPA
       M=IFAS(N)
       YTOT(M)=0.
    6 CONTINUE
      ISUM=0
      MPA=1
      YTOT(1)=1.
      IF (MSA.NE.0) THEN
       DO 7 N=1,MSA
        I=ISOL(N)
        Y(I)=0.
    7  CONTINUE
       MSA=0
      ENDIF
```

```
      IF (IS.NE.1) THEN
       IT=IS
    8  M=0
    9  M=M+1
       IF (2**M.LE.IT) GOTO 9
       IF (M.LE.MP) THEN
        MPA=MPA+1
        IFAS(MPA)=M
        YTOT(M)=1.
        MA=MF(M)
        MB=ML(M)
        DO 10 I=MA,MB
         Y(I)=YSUM
   10    CONTINUE
       ELSE
        I=M+ML(MP)-MP
        IF(I.GT.MS)RETURN
        MSA=MSA+1
        ISOL(MSA)=I
        ISUM=ISUM+2**(M-1)
        Y(I)=YSUM
       ENDIF
       IT=IT-2**(M-1)
       IF (IT.EQ.1) GOTO 8
       IF (MPA+MSA.GT.L) GOTO 5
      ENDIF
      IFAS(1)=1
      IF (NSUM(NG).GT.IS) NG=NG+1
      NSUM(NG)=IS
   11 IF (NG.NE.1) THEN
       DO 12 K=2,NG
        IF (NSUM(NG).EQ.NSUM(K-1)) GOTO 5
   12  CONTINUE
       IF (NSUM(NG-1).EQ.MSUM) NSUM(NG-1)=-NSUM(NG-1)
       IF (NG.LT.3.OR.NSUM(NG-2).NE.-MSUM.OR.NSUM(NG).EQ.MSUM) GOTO 13
       NSUM(NG-2)=-NSUM(NG-2)
      ENDIF
   13 LX=0
   14 DO 15 J=1,L
       BO(J)=B(J)
   15 CONTINUE
      DO 18 M=1,MP
       MA=MF(M)
       MB=ML(M)
       IF (YTOT(M).LE.0.) THEN
        DO 16 I=MA,MB
         AKT(I)=0.
         AKTF(I)=1.
         Y(I)=0.
         YF(I)=0.
   16    CONTINUE
       ELSE
        DO 17 I=MA,MB
         IF (Y(I).LT.1.E-8) Y(I)=1.E-8
         AKTF(I)=1.
   17    CONTINUE
        CALL ABER
        IF (YTOT(M).EQ.0.) GOTO 3
       ENDIF
   18 CONTINUE
```

```
      DMIN=1.E-6
      IVAR=0
      IVARJ=ML(1)-MS
      LS1=L+MPA+MSA
      LS=LS1-1
      LS2=LS+2
   19 DO 20 J=1,LS1
       DO 20 K=J,LS2
        R(J,K)=0.
   20 CONTINUE
      DO 24 N=1,MPA
       L1=L+N
       M=IFAS(N)
       MA=MF(M)
       MB=ML(M)
       DO 23 I=MA,MB
        IF (Y(I).NE.0.) THEN
         F(I)=G(I)+LOG(AKT(I))
         R(L1,LS2)=R(L1,LS2)+F(I)*Y(I)
         DO 22 J=1,L
          IF (A(I,J).NE.0.) THEN
           AY=A(I,J)*Y(I)
           R(J,L1)=R(J,L1)+AY
           R(J,LS2)=R(J,LS2)+AY*F(I)
           DO 21 K=J,L
            R(J,K)=R(J,K)+AY*A(I,K)
   21      CONTINUE
          ENDIF
   22    CONTINUE
        ENDIF
   23  CONTINUE
       DO 24 J=1,L
        R(J,LS2)=R(J,LS2)-R(J,L1)
   24 CONTINUE
      IF (MSA.NE.0.) THEN
       DO 25 N=1,MSA
        I=ISOL(N)
        K=L+MPA+N
        R(K,LS2)=G(I)
        DO 25 J=1,L
         R(J,K)=A(I,J)
   25  CONTINUE
      ENDIF
      DO 26 J=2,LS1
       N=J-1
       DO 26 K=1,N
        R(J,K)=R(K,J)
   26 CONTINUE
      DO 28 K=1,L
       IF (KH(K).NE.0) THEN
        DO 27 J=1,LS1
         R(J,LS2)=R(J,LS2)-PI(K)*R(J,K)
   27   CONTINUE
       ENDIF
       R(K,LS2)=R(K,LS2)+B0(K)
   28 CONTINUE
      DO 32 K=1,LS
       IF (KH(K).EQ.0) THEN
        ELMAX=0.
```

```
      DO 29 J=K,LS1
       IF (ABS(R(J,K)).LE.ELMAX.OR.KH(J).NE.0) GOTO 29
       MROW=J
       ELMAX=ABS(R(J,K))
 29    CONTINUE
       IF (ELMAX.LE.0.) THEN
        IF(B0(K))5,32,5
       ENDIF
       IF (MROW.NE.K) THEN
        DO 30 N=K,LS2
         RADBYT=R(MROW,N)
         R(MROW,N)=R(K,N)
         R(K,N)=RADBYT
 30     CONTINUE
       ENDIF
       KA=K+1
       DO 31 J=KA,LS1
        RKVOT=R(J,K)/R(K,K)
        DO 31 N=KA,LS2
         R(J,N)=R(J,N)-RKVOT*R(K,N)
 31    CONTINUE
      ENDIF
 32 CONTINUE
    DO 34 N=1,LS1
     K=LS2-N
     IF (KH(K).EQ.0) THEN
      IF (R(K,K).EQ.0..AND.R(K,LS2).EQ.0.) THEN
       PI(K)=0.
       K=K-L-MPA
       IF (K.LE.0) GOTO 34
       I=ISOL(K)
       Y(I)=0.
       GOTO 1
      ENDIF
      PI(K)=R(K,LS2)/R(K,K)
      KA=K-1
      IF (KA.NE.0) THEN
       DO 33 J=1,KA
 33     R(J,LS2)=R(J,LS2)-PI(K)*R(J,K)
      ENDIF
     ENDIF
 34 CONTINUE
    IF (IVAR.EQ.0.OR.IVARJ.GE.0.OR.SLAM.LT.0.1) GOTO 46
    DO 35 J=1,L
     IF (ABS(PI(J)).GT.1.E-8.AND.ABS(OPI(J)/PI(J)-1.).GT.DMIN) GOTO 44
 35 CONTINUE
    NR=0
    IF (NG.NE.1) THEN
     DO 36 K=2,NG
      IF (NSUM(NG).EQ.-NSUM(K-1)) NR=NR+1
 36  CONTINUE
    ENDIF
    DO 38 M=1,MP
     IF (YTOT(M).LE.0.) THEN
      MA=MF(M)
      MB=ML(M)
      CALL XBER
      YFTOT(M)=0.
```

```
      DO 37 I=MA,MB
       YFTOT(M)=YFTOT(M)+YF(I)
       YX(I)=YF(I)
       AKT(I)=0.
       YF(I)=0.
37     CONTINUE
       IF (M.EQ.1) YFTOT(M)=YFTOT(M)/PTOT
      ENDIF
38  CONTINUE
39  DIFM=1.
    DO 40 M=1,MP
     IF (YTOT(M).GT.0..OR.YFTOT(M).LE.DIFM) GOTO 40
     KA=M
     DIFM=YFTOT(M)
40  CONTINUE
    IF (DIFM.NE.1.) THEN
     IF (NR.NE.0) THEN
      NR=NR-1
      YFTOT(KA)=1.
      GOTO 39
     ENDIF
     MSUM=NSUM(NG)
     YTOT(KA)=1.
     MA=MF(KA)
     MB=ML(KA)
     DO 41 I=MA,MB
      Y(I)=YSUM*YX(I)/YFTOT(KA)
41   CONTINUE
     GOTO 3
    ENDIF
    IF (MS.GE.M1) THEN
     DIFM=0.
     DO 43 I=M1,MS
      IF (Y(I).LE.0.) THEN
       PIA=-G(I)
       DO 42 J=1,L
        PIA=PIA+A(I,J)*PI(J)
42     CONTINUE
       IF (PIA.GE.DIFM) THEN
        KA=I
        DIFM=PIA
       ENDIF
      ENDIF
43   CONTINUE
     IF (DIFM.NE.0.) THEN
      Y(KA)=YSUM
      GOTO 1
     ENDIF
    ENDIF
    IVARJ=0
    GOTO 46
44  DO 45 J=1,L
     OPI(J)=PI(J)
45  CONTINUE
46  SLAM=1.
    DO 48 N=1,MPA
     L1=L+N
     M=IFAS(N)
     MA=MF(M)
     MB=ML(M)
```

```
      DO 48 I=MA,MB
       IF (Y(I).NE.0.) THEN
        PIA=F(I)-PI(L1)
        DO 47 J=1,L
         PIA=PIA-A(I,J)*PI(J)
47      CONTINUE
        YX(I)=PIA*Y(I)
        IF (PIA.GT.SLAM) SLAM=PIA
       ENDIF
48    CONTINUE
      IF (SLAM.GT.1.) SLAM=0.9/SLAM
      IF (MSA.NE.0) THEN
       DO 49 N=1,MSA
        I=ISOL(N)
        K=L+MPA+N
        IF (PI(K).LT.0.) L1=0
        IF (IVAR.EQ.0.OR.PI(K).GT.-Y(I)) GOTO 49
        IF (SLAM.LT.1..AND.-PI(K)/YSUM.LT.1.E8) GOTO 49
        Y(I)=0.
        GOTO 1
49     Y(I)=ABS(PI(K))
      ENDIF
      DO 50 J=1,L
       B0(J)=B(J)
50    CONTINUE
      YSUM=0.
      DO 52 N=1,MPA
       M=IFAS(N)
       MA=MF(M)
       MB=ML(M)
       DO 51 I=MA,MB
        IF (Y(I).NE.0.) THEN
         Y(I)=Y(I)-SLAM*YX(I)
         IF (Y(I).LT.1.E-10) Y(I)=0.
        ENDIF
51     CONTINUE
       CALL ABER
       IF (YTOT(M).EQ.0.) GOTO 3
       YSUM=YSUM+YTOT(M)
52    CONTINUE
      IVAR=IVAR+1
      IF (IVAR.EQ.25.OR.IVAR.EQ.50) DMIN=100.*DMIN
      IF (IVAR.EQ.75) GOTO 5
      IF (IVARJ.LT.0.OR.L1.EQ.0.OR.SLAM.LT.1.) GOTO 19
      IVARJ=IVARJ+1
      IF (IVARJ.NE.10) THEN
       DO 53 N=1,MPA
        M=IFAS(N)
        MA=MF(M)
        MB=ML(M)
        DO 53 I=MA,MB
         IF (Y(I).GT.0.) THEN
          IF (ABS(YX(I))/Y(I).GT.1.E-6) GOTO 19
         ENDIF
53     CONTINUE
      ENDIF
```

```
      DO 54 N=1,MPA
       M=IFAS(N)
       MA=MF(M)
       MB=ML(M)
       CALL XBER
       DO 54 I=MA,MB
        IF (Y(I).LE.0.) THEN
         Y(I)=YTOT(M)*YF(I)
         IF (Y(I).LT.1.E-10.OR.LX.EQ.1) GOTO 54
         LX=1
         GOTO 14
        ENDIF
   54 CONTINUE
      IF (IVARJ.EQ.10) M=0
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE ABER
C
C      CALCULATE ACTIVITIES OF SPECIES FROM MOLE FRACTIONS
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/AAA/A(50,10),AKT(50),AKTF(50),B(10),B0(10),G(50)
      COMMON/BBB/IVAR,KH(20),L,L1,M,M1,MA,MB,MP,MS,MF(15),ML(15)
      COMMON/CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
C
      YTOT(M)=0.
      DO 2 I=MA,MB
       YTOT(M)=YTOT(M)+Y(I)
    2 CONTINUE
      IF (YTOT(M).LT.1.E-8) THEN
       YTOT(M)=0.
      ELSE
    4  If (M .eq. 1) YTOT(1) = YTOT(1)/PTOT
       DO 6 I=MA,MB
        YF(I)=Y(I)/YTOT(M)
    6  CONTINUE
       DO 8 I=MA,MB
        AKT(I)=AKTF(I)*YF(I)
    8  CONTINUE
      ENDIF
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE XBER
C
C      CALCULATE MOLE FRACTIONS OF SPECIES FROM ACTIVITIES
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON/AAA/A(50,10),AKT(50),AKTF(50),B(10),B0(10),G(50)
      COMMON/BBB/IVAR,KH(20),L,L1,M,M1,MA,MB,MP,MS,MF(15),ML(15)
      COMMON/CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
      DIMENSION OYF(50)
C
```

```
      PLOG=LOG(PTOT)
      IF (PLOG.LT.10.) PLOG=10.
      DO  6 I=MA,MB
       IF (Y(I).LE.0.) THEN
        PIA=-G(I)
        DO  2 J=1,L
         IF (A(I,J).NE.0.) THEN
          IF (PI(J).EQ.0.) GOTO 4
          PIA=PIA+A(I,J)*PI(J)
         ENDIF
    2   CONTINUE
        IF (PIA.GT.PLOG) PIA=PLOG
        IF (PIA.LT.-200.) GOTO  4
        AKT(I)=EXP(PIA)
        YF(I)=AKT(I)
        GOTO  6
    4   AKT(I)=0.
        YF(I)=0.
       ENDIF
    6 CONTINUE
      IVAR=0
    8 IVAR=IVAR+1
      DO  10 I=MA,MB
       OYF(I)=YF(I)
       YF(I)=AKT(I)/AKTF(I)
   10 CONTINUE
      DO  12 I=MA,MB
       IF (YF(I).GT.0.) THEN
        IF (ABS(OYF(I)/YF(I)-1.).GT.1.E-4) GOTO   8
       ENDIF
   12 CONTINUE
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE Runge_Kutta(N,t)
C
C.....4th order Runge-Kutta for solving differential equations.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      Integer count
      Real*8 k,M
      Dimension f_new(10),f_old(10)
      Dimension c(0:3),b(0:3)
      Data c/0.0d0,0.5d0,0.5d0,1.0d0/
      Data b/1.0d0,2.0d0,2.0d0,1.0d0/
      Common /One/count,df(5),dt,f(5),k,last_count,M,rad_loss0

      Do 10 J = 1,N
       f_new(J) = f(J)
       f_old(J) = f(J)
   10 continue
      Do 40 I = 0,3
       del_t = c(I) * dt
       weight = b(I) * dt / 6.0d0
       Do 20 J = 1,N
        f(J) = f_old(J) + del_t * df(J)
   20  continue
```

```
      Call Differential_Equations
      Do 30 J = 1,N
       f_new(J) = f_new(J) + weight * df(J)
   30  continue
   40 continue
      Do 50 J = 1,N
       f(J) = f_new(J)
   50 continue
      t = t + dt
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE Differential_Equations

      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      chan_z     = f(1) - z
      z          = f(1)
      enth_fb_per_Mass = f(4)
      If (enth_fb_per_Mass .GT. Initial_fb_enthalpy) Call DO_enth_halt
      Call ICAO_air(z,T_ambient,P_ambient,D_ambient)
      OLD_MASS   = Mass
      Mass       = f(3) * 4.0d0 / 3.0d0 * pii
      Call Mass_Balance_Update(OLD_MASS,Mass,z)
      cVol       = Mass * TotalGasMole * 8.3143 * T_fb / P_ambient
      radius     = ((cVol * 3.0d0 / (4.0d0 * pii)) ** (1.0d0/3.0d0))
      V          = (radius ** 3.0d0) * D_ambient
      Ddash      = f(3) / V
      Density    = Ddash * D_ambient
      M          = f(2) / (Ddash + k)
      u          = M / V
      Fb         = G * V * (1.0d0 - Ddash)
      dm         = 3.0d0 * entrain*dabs(M/radius)*Ddash**(2.0d0 / 3.0d0)
      Call ambient_enthalpy(z,T_ambient)
      df(1)      = u
      df(2)      = Fb
      df(3)      = dm
      Call Loss_Radiant(T_fb,radius,T_ambient)
      df(5)      = Radiant_Loss
      df(4)      = - (f(4) - enth_ambient) * dm * 4.0d0 / 3.0d0*pii/Mass
     1             - df(5) / Mass
     2             - (G * u / Ddash)
      RETURN
      END
```

```
      SUBROUTINE Mass_Balance_Update(OLD_MASS,BMass,hite)
C
C.....Calculates the increase in mass of atoms in fireball and updates
C.....for present time.
C
      IMPLICIT DOUBLE PRECISION(A-H,M-Z)

      include 'common.put'
C
C     The "an" stands for the amount per kilogram of ambient air.
C     Thus "anN" means the amount of atomic nitrogen, in mole,
C     per kilogram of ambient air.
C     These values are returned from the GRAM_ATOM SUBROUTINE.
C
      Call CLEAN_FIREBALL(hite,water_include,MFH2O,MFN2,MFO2,MFCO2,MFAr)

      CALL GRAM_ATOM(MFH2O,MFN2,MFO2,MFCO2,MFAr,anN,anO,anH,anC,anAr)

      delta_mass   = BMass - OLD_MASS
      nfS          = 0.0d0
      nfE          = 0.0d0
      anS          = 0.0d0
      anE          = 0.0d0
      TotalnfN     = TotalnfN  + (delta_mass * anN)
      TotalnfO     = TotalnfO  + (delta_mass * anO)
      TotalnfH     = TotalnfH  + (delta_mass * anH)
      TotalnfC     = TotalnfC  + (delta_mass * anC)
      TotalnfAr    = TotalnfAr + (delta_mass * anAr)
      TotalnfS     = TotalnfS  + (delta_mass * anS)
      TotalnfE     = TotalnfE  + (delta_mass * anE)
      nfN  = TotalnfN / BMass
      nfO  = TotalnfO / BMass
      nfH  = TotalnfH / BMass
      nfAr = TotalnfAr / BMass
      nfC  = TotalnfC / BMass
      nfS  = TotalnfS / BMass
      nfE  = TotalnfE / BMass
      RETURN
      END
```

```
      SUBROUTINE Loss_Radiant(T,rad,T_amb)
C
C.....Calculates the energy lost to far field due to radiation.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      Real*8 Initial_fb_enthalpy,mwa
      Common /output/OpTr
      Common /four/beta,D_ambient,entrain,Initial_fb_enthalpy,pii,W,mwa
      Common /hol/ Holdn(50),TotalGasMole,TotalSolidMole,Zdsh
      Common /B/Op_fb,Tr,Radiant_Loss,sigma

      if(T .GT. 3035.0d0) then
       Tr = 0.954d0 - 1.0d-8 * ((T - 6000.0d0) ** 2.0d0)
      else
       if(T .GT. 704.0d0) then
        Tr = 0.535d0 * dlog(T) -3.5077d0
       else
        Tr = 0.0d0
       endif
      endif

      If(TotalSolidMole .GT. 3.0d-05) then
       Op_fb = 1.0d0
      else
       RadInCM = 100.0d0 * rad
       Op_fb = 1.0d0 - Dexp(-4.0d0*2.0d0*RadInCM*PMAC(T))
      endif
      OpTr          = Op_fb * Tr
      Radiant_Loss        = 4.0d0 * pii * (rad * rad)
     1                     * OpTr * sigma * ((T * T) + (T_amb * T_amb))
     2                     * (T + T_amb) * (T - T_amb)
      RETURN
      END
C
C-----------------------------------------------------------------------
C
      Function PMAC(T_fb)
C
C.....Returns the Planck Mean Absorption Coefficient (PMAC).
C
      IMPLICIT DOUBLE PRECISION(A-H,M-Z)

      Logical Charged,Setting_up,momentum_stop
      Common /two/chan_z,cVol,Ddash,Density,Fb,u,G
      Common /eight/Charged,Setting_up,momentum_stop
      Common /Dim/ DL(4),TL(7),PMAC_Log(8,4)

      If(Setting_up) then
        Setting_up = .false.
        D_ground = 1.225D+00
        TL(1) = 0.0d0
        TL(2) = 1000.0d0
        TL(3) = 2000.0d0
        TL(4) = 3000.0d0
        TL(5) = 4000.0d0
        TL(6) = 6000.0d0
        TL(7) = 8000.0d0
        DL(1) = 0.0d0
        DL(2) =-1.0d0
```

```
      DL(3) =-2.0d0
      DL(4) =-3.0d0
      PMAC_Log(1,1) = Dlog10(5.4d-08)
      PMAC_Log(2,1) = Dlog10(5.4d-08)
      PMAC_Log(3,1) = Dlog10(1.7d-06)
      PMAC_Log(4,1) = Dlog10(4.4d-05)
      PMAC_Log(5,1) = Dlog10(5.7d-04)
      PMAC_Log(6,1) = Dlog10(6.1d-03)
      PMAC_Log(7,1) = Dlog10(1.4d-02)
      PMAC_Log(8,1) = Dlog10(1.4d-02)
      PMAC_Log(1,2) = Dlog10(1.7d-09)
      PMAC_Log(2,2) = Dlog10(1.7d-09)
      PMAC_Log(3,2) = Dlog10(5.4d-08)
      PMAC_Log(4,2) = Dlog10(1.6d-06)
      PMAC_Log(5,2) = Dlog10(2.6d-05)
      PMAC_Log(6,2) = Dlog10(1.9d-04)
      PMAC_Log(7,2) = Dlog10(5.1d-04)
      PMAC_Log(8,2) = Dlog10(5.1d-04)
      PMAC_Log(1,3) = Dlog10(5.3d-11)
      PMAC_Log(2,3) = Dlog10(5.3d-11)
      PMAC_Log(3,3) = Dlog10(1.7d-09)
      PMAC_Log(4,3) = Dlog10(7.0d-08)
      PMAC_Log(5,3) = Dlog10(7.7d-07)
      PMAC_Log(6,3) = Dlog10(6.6d-06)
      PMAC_Log(7,3) = Dlog10(1.9d-05)
      PMAC_Log(8,3) = Dlog10(1.9d-05)
      PMAC_Log(1,4) = Dlog10(1.7d-12)
      PMAC_Log(2,4) = Dlog10(1.7d-12)
      PMAC_Log(3,4) = Dlog10(5.3d-11)
      PMAC_Log(4,4) = Dlog10(2.7d-09)
      PMAC_Log(5,4) = Dlog10(2.1d-08)
      PMAC_Log(6,4) = Dlog10(3.3d-07)
      PMAC_Log(7,4) = Dlog10(5.9d-07)
      PMAC_Log(8,4) = Dlog10(5.9d-07)
    endif

    Den = Density / D_ground
    D_log = Dlog10(Den)

    Do 1 I = 1,3
     If(D_log .LE. DL(I) .and. D_log .GE. DL(I+1)) K = I
1   Continue
    Do 2 I = 1,7
     If(T_fb .GE. TL(I)) J = I
2   Continue
    Dratio = (D_log - DL(K)) / (DL(K+1) - DL(K))
    D1 = PMAC_Log(J,K) + (Dratio * (PMAC_Log(J,K+1) - PMAC_Log(J,K)))
    D2 = PMAC_Log(J+1,K) + (Dratio * (PMAC_Log(J+1,K+1) -
   1     PMAC_Log(J+1,K)))
    Tratio = (T_fb - TL(J)) / (TL(J+1) - TL(J))
    PMAC_log10 =  D1 + (Tratio * (D2 - D1))
    PMAC = 10.0d0 ** PMAC_log10
    RETURN
    END
```

```
      SUBROUTINE Initialize_Output
C
C.....Sets initial information in headers for data output to screen
C.....and output files.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'
      Character water*3,pgrm*25
      Character*8 Mtim,Mdate
      Character*13 sub,sub2
      Character*50 Form,Form1

      pgrm                    = 'FB10.FOR  17/11/1986'
      if(water_include) then
        water = 'Yes'
      else
        water = 'No '
      endif

      Call Date(Mdate)
      Call Time(Mtim)

      Do 10 I = 2,10
       write(I,2) W,Mdate,file_name(I-1),Mtim,pgrm,Species_num,Atmos,
     * beta,Z_fb,T_fb,entrain,times_up,G,sigma,water
   10 continue
      write(3,4)
      write(4,6)
      Form = ' Time   T_fb  TGasMole  TSolMole  TotalMol  Z_fb '
      Form1 = '  s      K     mole/kg   mole/kg   mole/kg   --  '
      If(Charged) then
       If(Species_num .LE. 6) then
        write(5,*) Form,(NOM(I),' ',I=1,Species_Num),' nTotCharged'
        write(5,*) Form1,(' mole/kg  ',I=1,Species_num),' mole/kg'
       else
        write(5,*) Form,(NOM(I),' ',I=1,7)
        write(5,*) Form1,(' mole/kg  ',I=1,7)
       endif
      endif
      If( .not. Charged) then
       if(Species_num .LE. 7) then
        write(5,*) Form,(NOM(I),' ',I=1,Species_num)
        write(5,*) Form1,(' mole/kg  ',I=1,Species_num)
       else
        write(5,*) Form,(NOM(I),' ',I=1,7)
        write(5,*) Form1,(' mole/kg  ',I=1,7)
       endif
      endif
      write(6,7)
      sub  = '  Time   T_fb'
      sub2 = '   s      K  '
      Do 20 JK = 1,3
       J = 8 + ((JK - 1) * 11) - 1
       If(Species_num .GT. J-1 .and. Charged) then
        if(Species_num .LE. J+10) then
         write(JK+6,*) sub,(' ',NOM(I),I=J+1,Species_num),' nTotCharged'
         write(JK+6,*) sub2,('  mole/kg ',I=J+1,Species_num+1)
        else
         write(JK+6,*) sub,(' ',NOM(I),I=J+1,J+11)
```

```
          write(JK+6,*) sub2,('  mole/kg ',I=J+1,J+11)
         endif
        endif
        if(Species_num .GT. J .and. .not. Charged) then
         if(Species_num .LE. J+11) then
          write(JK+6,*) sub,(' ',NOM(I),I=J+1,Species_num)
          write(JK+6,*) sub2,('  mole/kg ',I=J+1,Species_num)
         else
          write(JK+6,*) sub,(' ',NOM(I),I=J+1,J+11)
          write(JK+6,*) sub2,('  mole/kg ',I=J+1,J+11)
         endif
        endif
   20 Continue
      If(Species_num .GT. 39 .and. Charged) then
       if(Species_num .LE. 50) then
        write(10,*) sub,(' ',NOM(I),I=41,Species_num),' nTotCharged'
        write(10,*) sub2,('  mole/kg ',I=41,Species_num+1)
       else
        write(0,1)
        Stop
       endif
      endif
      if(Species_num .GT. 40 .and. .not. Charged) then
       if(Species_num .LE. 51) then
        write(10,*) sub,(' ',NOM(I),I=41,Species_num)
        write(10,*) sub2,('  mole/kg ',I=41,Species_num)
       else
        write(0,1)
        Stop
       endif
      endif
      write(0,8) W,Mdate,Mtim,pgrm,beta,Z_fb,T_fb,entrain
      RETURN
C
C--------ALL THE FORMAT STATEMENTS
C
    1 Format(' There are too many species in the system [more than 51]
     * ',/,' See Initialize_Output SUBROUTINE. Program now Terminated. '
     * )
    2 Format(f7.4,' Megaton Bomb Yield.  Date data taken on ',A8,/,' Th
     *is data was stored in file : ',A12,' at time ',A8,/,' Data generat
     *ed from program : ',A25,/,' Input data read by program from file :
     * D.file  Includes ',I2,' Species.  Type of atmosphere : ',A20,/,
     *' Beta = ',f5.3,' Init Z = ',f6.4,' Init Tf = ',f6.1,' Entrain = '
     *,f4.2,/,' Time for run : ',f5.1, '
     *Gravity : ',f7.5,' Sigma : ',d9.3,' Water vapour included : ',A3)
    4 Format(47x,'f(1)  df(1),u',/,'  Time  T_fb   H_top  H_bot   rad
     *  cVol       z     speed   ch_z OpTr      R      TotR/W  R/Ro  Zdsh
     *',/,'   s      K      m       m       m      m**3       m      m/s
     *  m    --      J/s')
    6 Format(15x,'df(1),u',33X,'df(2),Fb  ',/,'  Time   T_fb  speed
     *cVol      Mass    Den  Ddsh  Buoyancy   P_amb  T_amb   fb_N   fb_O
     *   fb_H   fb_Ar  fb_C  TotalMole',/,
     *'    s      K     m/s      m**3       kg    kg/m3  --    kg.m/s2
     *  Pa      K   mol/kg mol/kg mol/kg mol/kg mol/kg  mole/kg')
```

```
    7 Format(15x,'f(1)',12X,'f(2)      f(3)',7X,'f(4)    df(1)  df(2),Fb
     *    df(3)',/,'  Time  T_fb     z    T_amb   M(Dd+k)   V.Ddash    e
     *nth_fb    u    Bouyancy     dm       df(4)     dt   enth_amb   Ent
     *hperkg',/,'   s     K       m       K     kg.m/s      kg        Jou
     *le    m/s    kg.m/s2    kg/s       J/s      s    Joule/kg   Joule/
     *kg')
    8 Format(X,f7.4,' Megaton Bomb Yield.  Date data taken on ',A8,/,' T
     *his data was stored in files :  d?.dat at time ',A8,/,' Data gener
     *ated from program : ',A25,/,' Beta = ',f5.3,' Init Z = ',f6.4,' In
     *it Tf = ',f6.1,/,'  entrain = ',f4.2,/,46x,'f(1)  df(1),u'
     *,/,'  Time   T_fb  H_top  H_bot  rad      cVol       z    speed  c
     *h_z OpTr    Ro  ',/,'   s      K      m      m      m       m**3
     *    m      m/s     m      --    J/s  ')
      END
C
C-------------------------------------------------------------------------
C
      SUBROUTINE Output_Data
C
C.....Outputs data to screen and output files.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      Character Form2*47,Form3*41,Form4*45,Form5*39

       write(0,1) nTotalCharged,count
       write(3,4) atime,T_fb,z+radius,z-radius,radius,cVol,z,u,chan_z,
     *OpTr,Radiant_Loss,f(5)/Initial_fb_enthalpy,df(5)/rad_loss0,Zdsh

       write(4,5) atime,T_fb,u,cVol,Mass,Density,Ddash,Fb,P_ambient,
     *T_ambient,nfN,nfO,nfH,nfAr,nfC,TotalMole

       Form2 = '(F6.2,F7.1,3D10.3,X,F5.3,'//char(Species_num +48)/
     */'D10.3,d10.3)'

       Form3 = '(F6.2,F7.1,3D10.3,X,F5.3,7D10.3)'

       If(Charged) then
        If(Species_num .LE. 6) then
         write(5,Form2) atime,T_fb,TotalGasMole,TotalSolidMole,TotalMole
     *,Z_fb,(Holdn(I),I=1,Species_num),nTotalCharged
        else
         write(5,Form3) atime,T_fb,TotalGasMole,TotalSolidMole,TotalMole
     *,Z_fb,(Holdn(I),I=1,7)
        endif
       endif

       If(.not. Charged) then
        if(Species_num .LE. 7) then
         write(5,6) atime,T_fb,TotalGasMole,TotalSolidMole,TotalMole,
     *Z_fb,(Holdn(I),I=1,Species_num)
        else
         write(5,Form3) atime,T_fb,TotalGasMole,TotalSolidMole,TotalMole
     *,Z_fb,(Holdn(I),I=1,7)
        endif
       endif
```

```
        write(6,7) atime,T_fb,z,T_ambient,f(2),f(3),f(4)*Mass,u,Fb,
     *df(3),df(4),dt,enth_ambient,f(4)

        Do 10 JK = 1,4
         J = 8 + ((JK - 1) * 11)
         If(Species_num .GT. J-2 .and. Charged .and. Species_num .LE. J+9
     * ) then
          L  = Species_num - J + 2
          LL = L - 9
          write(0,2) L,LL,J,JK
          if (LL .GE. 1) then
           Form4 = '(X,F6.2,F6.0,9d10.3,'//char(LL + 48)//'d10.3,D10.3)'
          else
           Form4 = '(X,F6.2,F6.0,'//char(L + 48)//'d10.3,D10.3)'
          endif
          write(JK+6,Form4) atime,T_fb,(Holdn(I),I=J,Species_num),nTotalC
     *harged
         endif

         IF(Species_num .GT. J-1 .and. .not. Charged .and. Species_num .
     * LE. J+10) then
          L  = Species_num - J + 1
          LL = L - 9
          if (LL .GE. 1) then
           Form5 = '(X,F6.2,F6.0,9d10.3,'//char(LL + 48)//'d10.3)'
          else
           Form5 = '(X,F6.2,F6.0,'//char(L + 48)//'d10.3)'
          endif
          write(JK+6,Form5) atime,T_fb,(Holdn(I),I=J,Species_num)
         endif
         If(Species_num .GT. J+9 .and. Charged)
     * then
          write(JK+6,3) atime,T_fb,(Holdn(I),I=J,J+10)
         endif
         IF(Species_num .GT. J+10 .and. .not. Charged ) then
          write(JK+6,3) atime,T_fb,(Holdn(I),I=J,J+10)
         endif
   10  Continue

        write(0,8) atime,T_fb,z+radius,z-radius,radius,cVol,z,u,chan_z,
     *OpTr,Radiant_Loss

        RETURN
C
C---------ALL THE FORMAT STATEMENTS
C
     1  Format(' O nTotalCharged = ',D10.4,'  count = ',I3)
     2    Format(' L = ',I3,'  LL = ',I3,' J = ',I3, '  JK = ',I3)
     3    Format(X,F6.2,F6.0,11d10.3)
     4  Format(X,f6.2,f6.0,f7.0,f7.0,f7.1,D11.4,f8.1,f7.2,f6.1,
     *f6.3,D10.3,X,F6.4,X,F6.4,F6.3)
     5  Format(F7.2,F7.1,F7.2,2D10.3,2F6.3,D10.3,F9.1,F6.1,5F7.3,D10.3)
     6    Format(F6.2,F7.1,3D10.3,X,F5.3,7D10.3)
     7  Format(X,F6.2,F6.0,F8.1,F6.1,' ',D10.3,D10.3,' ',D10.3,' ',
     *F5.1,' ',2D10.3,' ',D10.3,' ',F4.2,' ',D10.3,' ',D10.3)
     8  Format(X,F6.2,F7.1,F7.0,F7.0,F6.0,' ',D9.3,F8.1,F7.2,F6.1,
     *F6.2,' ',D9.3)
        END
```

```
      SUBROUTINE read_amb_air
C
C.....Reads and stores the thermochemical dat for the ambient air
C.....calculations of enthalpy from the file AMB.AIR.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      Logical water_include
      integer Tnum,Number_of_des
      CHARACTER Junk*6,line*80,Atmos*20

      Common /det/ ER(6),C(6,6)
      COMMON /five/enth_ambient,TotalMole,Number_of_des
      common /h2o/ water_include,Atmos
      common /amb/ Tnum,ISN(6),ENTHAL(6)
      DIMENSION SR(6)

       write(0,101)
       read(12,*) Tnum,(ISN(I),I=1,Tnum)
       write(0,*) Tnum,(ISN(I),I=1,Tnum)
       write(2,*) Tnum,(ISN(I),I=1,Tnum)
       Do 10 I = 1,Tnum
        Open(unit = 11,file = 'amb.air',status ='old')
        Do 20 J = 1,6
         if (ISN(I) .EQ. J) then
          read(11,102) Junk,ER(I),SR(I)
          write(0,103) Junk,ER(I),SR(I)
          read(11,104) line
          write(0,105)line
          read(11,104) line
          write(0,105)line
          read(11,106) (C(I,K),K = 1,6)
          write(0,107) (C(I,K),K = 1,6)
          read(11,104) line
          write(0,105)line
          ENTHAL(I) = ENTH(I,temp) * 4.186D+00
          write(0,108) I,ENTHAL(I)
         else
          do 30 K = 1,5
           read(11,104) line
  30       continue
         endif
  20    continue
        close(unit=11)
  10   continue
       RETURN
C
C-------ALL THE FORMAT STATEMENTS
C
  101 Format(' Enter the number of gaseous molecules to be entrained ',/
     *   ,' followed by the molecule number [separated by spaces] ... ')
  102 Format(A6,F11.2,F8.3)
  103 Format(X,A6,F11.2,F8.3)
  104 Format(A)
  105 Format(X,A)
  106 Format(4D12.6,2F10.2)
  107 Format(X,4D12.6,2F10.2)
  108 Format(' ENTHAL(',I1,') = ',d20.14)
      END
```

```
      SUBROUTINE read_in_CHNOS
C
C.....Reads and stores all the thermochemical data for the species
C.....in the fireball from the file NEWDATA.DAT.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      Logical Charged,Setting_up,momentum_stop
      CHARACTER ELA*2,ELF*2,ELN*2,ANF*80,NOM*9,ch
      Integer Species_num
C
      DIMENSION ELA(50,6),ATA(50,6),NEA(50),ELN(50)
      DIMENSION DA(5,6),ELF(50),ATF(50)
      DIMENSION IN(50)
C
      COMMON/AAA/A(50,10),AKT(50),AKTF(50),B(10),B0(10),G(50)
      COMMON/BBB/IVAR,KH(20),L,L1,M,M1,MA,MB,MP,MS,MF(15),ML(15)
      COMMON/CCC/PI(20),PTOT,T,Y(50),YO(50),YF(50),YTOT(20)
      COMMON/DAT/C1(50,5),C2(50,5),C3(50,5),C4(50,5)
      COMMON/DAT/HR(50),SR(50),NR(50),HH(50,5),TM(50,5)
      Common /ms1/ Species_num,NOM(50),NS,NPS,NNS,NsolS,NgasS
      common /xaa/ MG,MR
      common /eight/Charged,Setting_up,momentum_stop
      COMMON/NINE/J1,J2,J3,J4,J5,J6,J7

      OPEN(UNIT=13,FILE='NEWDATA.DAT',STATUS='OLD')
C
C     ENTER THE SPECIES TO BE USED
C
      write(0,101)
C
C     The input data is read on all species. MS is the total number of
C     species to be read. NS is the number of non charged species. NPS
C     is the number of possitively charged species. NNS is the number
C     of negatively charged species.
C
      READ(12,*)MS
      READ(12,*)NgasS
      READ(12,*)NS,(IN(J),NOM(J),J=1,NS)
      Read(12,102)ANF
      Read(12,*)ch
      WRITE(2,103)MS
      WRITE(2,104)NgasS
      WRITE(2,*)NS,(IN(J),NOM(J),J=1,NS)
      write(2,102)ANF
      write(2,105)ch
      Charged = .false.
      if(ch .eq. 'y' .or. ch .eq. 'Y') then
       Charged = .true.
       Read(12,*)NPS,(IN(J),NOM(J),J=NS+1,NS+NPS)
       WRITE(2,*)NPS,(IN(J),NOM(J),J=NS+1,NS+NPS)
       Read(12,102)ANF
       write(2,102)ANF
       Read(12,*)NNS,(IN(J),NOM(J),J=NS+NPS+1,NS+NPS+NNS)
       WRITE(2,*)NNS,(IN(J),NOM(J),J=NS+NPS+1,NS+NPS+NNS)
       Read(12,102)ANF
       write(2,102)ANF
      endif
      if(ch .eq. 'n' .or. ch .eq. 'N') then
       Read(12,106) NPS,NNS
```

```
      WRITE(2,106) NPS,NNS
      write(0,107) NPS,NNS
     endif

     Read(12,*)NsolS,(IN(J),NOM(J),J=NS+NPS+NNS+1,NS+NPS+NNS+NsolS)
     WRITE(2,*)NsolS,(IN(J),NOM(J),J=NS+NPS+NNS+1,NS+NPS+NNS+NsolS)
     Read(12,102)ANF
     WRITE(2,102)ANF

     IF(NS+NPS+NNS+NsolS .NE. MS) then
      write(0,108)
      stop
     endif

     READ(12,109) J1,J2,J3,J4,J5,J6,J7
     WRITE(2,110) J1,J2,J3,J4,J5,J6,J7
     READ(12,102)ANF
     WRITE(2,102)ANF
     write(0,*) MS,(IN(J),NOM(J),J=1,MS)
     Species_num = MS

     REWIND 13
     NREC=0
  10 READ(13,111,END=35)NE,NN,ND,DH,DS
     NREC=NREC+1
     READ(13,112)(ELF(J),ATF(J),J=1,NE)
     READ(13,113)ANF(1:NN)
     DO 15 J=1,ND
  15  READ(13,114)(DA(J,K),K=1,6)
C
C    STORE SPECIES DATA IN MEMORY
C
     DO 30 I=1,MS
      IF (NREC.EQ.IN(I)) THEN
       DO 20 J=1,NE
        ELA(I,J)=ELF(J)
        ATA(I,J)=ATF(J)
  20    CONTINUE
        NEA(I)=NE
        HR(I)=DH
        SR(I)=DS
        NR(I)=ND
        DO 25 J=1,ND
         C1(I,J)=DA(J,1)
         C2(I,J)=DA(J,2)
         C3(I,J)=DA(J,3)
         C4(I,J)=DA(J,4)
         TM(I,J)=DA(J,5)
         HH(I,J)=DA(J,6)
  25    CONTINUE
        GOTO 10
      ENDIF
  30  CONTINUE
      GOTO 10
```

```
C
C     COUNT ELEMENTS, GASES AND CONDENSED SPECIES
C
  35  L=0
      MG=0
      MR=0
      DO 50 I=1,MS
       IF (ELA(I,NEA(I)).EQ.' G') THEN
        MG=MG+1
       ELSE
        MR=MR+1
       ENDIF
       N=NEA(I)-1
       DO 45 J=1,N
        IF (L.EQ.0) THEN
         L=L+1
         ELN(L)=ELA(I,J)
         GOTO 45
        ELSE
         DO 40 K=1,L
          IF (ELA(I,J).EQ.ELN(K)) GOTO 45
  40     CONTINUE
         L=L+1
         ELN(L)=ELA(I,J)
        ENDIF
  45   CONTINUE
  50  CONTINUE
      L1=L+1
      M1=MG+1
      MP=1
      MF(1)=1
      ML(1)=MG
      WRITE(0,115)L,MG,MR
C
C     SET UP ATOM MATRIX
C
      DO 55 I=1,MS
       DO 55 K=1,L
  55  A(I,K)=0.
      DO 65 I=1,MS
       N=NEA(I)-1
       DO 60 J=1,N
        DO 60 K=1,L
         IF (ELA(I,J).EQ.ELN(K)) A(I,K)=ATA(I,J)
  60   CONTINUE
       WRITE(0,116) I,(A(I,K),K=1,L)
  65  CONTINUE
      close(13)
      RETURN
C
C--------ALL THE FORMAT STATEMENTS
C
  101 Format(' Enter the number of species to be processed, followed',/
     *,' by the species number according to the NEWDATA.DAT table, ',/,
     * ' all numbers in integer form and separated by a space. ',/,
     *' ... ')
  102 Format(x,A)
  103 Format(' MS = ',I2,'         : Total species number. ')
  104 Format(' NgasS = ',I2,'      : Total gaseous species number. ')
  105 Format(x,A1,'                : Charged species included [Y/N]. ')
```

```
  106 Format(I3,//,I3,/)
  107 Format(' NPS = ',I3,' NNS = ',I3)
  108 Format(' The charged and non_charged species do not sum to "MS",
     *',/,' the number of total species. Check D.file, this program ',/,
     *' is now terminated. ')
  109 Format(7(4x,I1))
  110 Format(' C=',I1,' H=',I1,' N=',I1,' O=',I1,' S=',I1,' Ar=',I1,' E
     *=',I1)
  111 FORMAT(3I2,F11.2,F8.3)
  112 FORMAT(10(A2,F6.3))
  113 FORMAT(A)
  114 FORMAT(4F12.0,2F10.0)
  115 FORMAT(/' ELEMENTS ',I2,' GASES ',I2,' CONDENSED ',I2)
  116 FORMAT(1X,I2,10(2X,F6.3))
      END
C
C-----------------------------------------------------------------------
C
      SUBROUTINE read_in_data
C
C.....Reads all the initial conditions from the input file D.FIL.
C
      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      include 'common.put'

      Character*8 Mtim,Mdate,ch*1

      Common /now/ DMFN2,DMFO2,DMFCO2,DMFAr

      open(12,file = 'd.file',Status = 'old')
      Rewind 12

      Read(12,1) ch,times_up,beta,G,entrain,initial_dt,initial_fb_temp
      write(0,2) ch,times_up,beta,G,entrain,initial_dt,initial_fb_temp
      momentum_stop = .true.
      if (ch.EQ.'Y'.OR.ch.EQ.'y') then
       momentum_stop = .false.
      else
       if (ch.NE.'N'.AND.ch.NE.'n') then
        write (0,3)
        stop
       endif
      endif

      Read(12,3) initial_z,k,last_count,mwa,sigma,Atmos
      write(0,4) initial_z,k,last_count,mwa,sigma,Atmos

      write(0,5) Number_of_des

      Do 100 I = 1 , 9
       Read(12,6) file_name(I)
       write(0,7) I,file_name(I)
       open(I+1,file = file_name(I),Status = 'new',RECL = 130)
  100 continue
```

```
      Call Date(Mdate)
      Call Time(Mtim)
      write(2,8) Mtim,Mdate
      write(2,2) ch,times_up,beta,G,entrain,initial_dt,initial_fb_temp
      write(2,4) initial_z,k,last_count,mwa,sigma,Atmos
      write(2,5) Number_of_des

      do 110 i=1,9
       write(2,7) I,file_name(I)
  110 continue
      read(12,9) W
      write(0,10) W
      write(2,10) W

      write(0,11) initial_fb_temp

      Read(12,6) ch
      write(2,12) ch
      write(0,12) ch
      water_include = .true.
      if (ch .eq. 'N' .or. ch .eq. 'n') then
       water_include = .false.
      else
       if (ch .ne. 'Y' .and. ch .ne. 'y') then
        write(0,13)
        stop
       endif
      endif
      write(0,16) initial_z
      Read(12,19) DMFN2,DMFO2,DMFCO2,DMFAr
      write(0,20) DMFN2,DMFO2,DMFCO2,DMFAr
      write(2,20) DMFN2,DMFO2,DMFCO2,DMFAr
      TDMF = DMFN2 + DMFO2 + DMFCO2 + DMFAr
      if (TDMF .NE. 1.0d0) then
        write(0,21) TDMF
        Stop
      endif
      RETURN
    1 Format(/x,A1,/x,f5.1,/x,f4.2,/x,f7.5,/x,f4.2,/x,f4.2,/x,f6.1)
    2 Format(' Time stop else momentum stop [Y/N] : ',A1,/,
     1' Computer stop time  : ',f5.1,/x,'beta',16x,': ',f4.2,
     2/x,'Gravity',13x,': ',f7.5,/x,'Entrain coefficient : ',f4.2,/x,
     3'Initail_dt',10x,': ',f4.2,/x,'Initial_fb_temp',5x,': ',f6.1)
    3 Format(x,f7.1,/x,f3.1,/x,I5,/x,F8.5,/x,d9.3,/x,A20)
    4 Format(' Initial_z',11x,': ',f7.1,/x,'k',19x,': ',f3.1,
     1/x,'last_count',10x,': ',I5,/x,'Mlr weight air',6x,': ',F8.5,
     2/x,'sigma',15x,': ',d9.3,/x'Atmosphere type',5x,': 'A20)
    5 Format(' Number_of_des = ',I1)
    6 Format(x,A)
    7 Format(x'Output File Name #',I1,' : ',A12)
    8 Format(20X,'Fbombs81.dat : Copy of input file "D.fil". Produced at
     1 ',A8,' on ',A8)
    9 Format(x,f6.3)
   10 Format(x,'Bomb size           : ',f6.3,' Mt. ')
   11 Format(' initial_fb_temp = ',d14.8)
   12 Format(X,A1,15x,': Water vapour included in ambient air calculatio
     *ns [Y/N] ')
   13 Format(' Check "d.file" as a "Y" or "N" does not appear correctly
     *')
   16 Format('  Bomb detonation height is ',d10.4,' metre. ')
```

```
   19 Format(x,f8.6,/x,f8.6,/x,f8.6,/x,f8.6)
   20 Format(' The mole fractions of the CLEAN_FIREBALL set of dry air
     * are : ',/,
     *       ' DMFN2  ..................................... ',D16.10  ,/,
     *       ' DMFO2  ..................................... ',D16.10  ,/,
     *       ' DMFCO2 ..................................... ',D16.10  ,/,
     *       ' DMFAr  ..................................... ',D16.10  )
   21 Format(' TDMF should equal 1.00, but instead it equals ',D16.10,'.
     * This program is now terminated. Check the data file "d.file". ')
      END
C
C-------------------------------------------------------------------------
C
      SUBROUTINE DO_enth_halt

      write(0,10)
   10 Format(    ' Enthalpy Error ')
      Stop
      RETURN
      END
C
C-------------------------------------------------------------------------
C
      SUBROUTINE Show_Status

      IMPLICIT DOUBLE PRECISION (A-H,M-Z)

      Integer counter
      Common /six/atime,Mass,P_ambient,radius,T_ambient,T_fb,z,Z_fb
      Common /seven/enth_fb_per_Mass,Enthalpy_per_kg
      Common /USE/Enthal(2000),temper(2000),cala,calb,calc,counter

      write(0,10) T_fb,enth_fb_per_Mass/Mass
      Do 30 I = (counter - 10),counter
       write(0,20) I,Enthal(I),I,temper(I)
   30 continue
      write(0,40) Enthalpy_per_kg
      Stop
      RETURN

   10 Format(' T_fb = ',D10.3,' enth = ',D10.3)
   20 Format(' Enthal(',I3,') = ',D10.3,' Temper(',I3,') = ',D10.3)
   40 Format(' Enthalpy_per_kg = ',D10.3)
      END
```