Appendix A

Stomatal conductance model

The derivation of the stomatal conductance model here is similar to that of Cowan (1977; see also Lloyd, 1991), but includes boundary layer conductances. Shown first is the transpiration rate in terms of the vapour pressure deficit of the air, accounting for heat loss through the boundary layer. The leaf flux equations for water vapour and heat are:

$$E = \frac{1}{r_{se} + r_{be}} (e_i - e_a) \tag{A1}$$

$$H = \frac{c_p}{r_{bH}} (\theta_l - \theta_a) \tag{A2}$$

If we assume that intercellular vapour concentration is the saturation value at leaf temperature, then using the approximation $e_{sat}(\theta_l) \approx e_{sat}(\theta_a) + s(\theta_l - \theta_a)$ (recall that s is the slope of saturation vapour concentration with temperature) we find:

$$E = \frac{1}{r_{se} + r_{be}} [D_a + s(\theta_l - \theta_a)]$$

which, combined with Equation (A2) leads to:

$$E = \frac{D_a + \frac{\varepsilon}{\lambda_E} r_{bH} H}{r_{se} + r_{be}}$$

where $\varepsilon = s\lambda_E / c_p$. The energy balance equation (Equation (2.19)) can then be used to find:

$$E = \frac{1}{r_{se} + r_{be}} \left[D_a + \frac{\frac{\varepsilon}{\lambda_E} r_{bH} R^*_{n,abs} - \varepsilon r_{bH} E}{1 + \frac{4\varepsilon_l \sigma \theta_a^3}{c_p} r_{bH}} \right]$$

which, on rearrangement, reveals:

$$E = \frac{D_0}{r_{se} + r_{be} + \varepsilon r^*_{bH}} \tag{A3}$$

where $r^*_{bH} = \frac{1}{g_{bH} + 4\varepsilon_l \sigma \theta_a^3 / c_p}$ and $D_0 = D_a + \varepsilon r^*_{bH} R^*_{n,abs} / \lambda_E$.

That D_0 is the leaf to air vapour pressure difference when the stomata are fully closed can be seen by equating Equations (A1) and (A3), so that:

$$e_i - e_a = \frac{D_0}{1 + \frac{\varepsilon r^*_{bH}}{r_{se} + r_{be}}}$$

which shows that $e_i - e_a \to D_0$ when $r_{se} \to \infty$.

Next we obtain an expression for assimilation rate with conductance being the only variable. First we note that for changes in c_i due to changes in stomatal conductance, assimilation is linear in c_i (Lloyd, 1991):

$$A = k(c_i - \Gamma)$$

Combining this with the leaf flux equation for CO_2,

$$A = g_c(c_a - c_i)$$

where $g_c = \frac{1}{r_{sc} + r_{bc}} = \frac{g_{sc}}{1 + r_{bc} g_{sc}}$ is the total conductance to CO_2 from the intercellular spaces to the canopy air space, we eliminate c_i to obtain:

$$A = \frac{k g_c (c_a - \Gamma)}{k + g_c} \tag{A4}$$

Now to relate stomatal conductance to the marginal water loss per unit carbon gain, we start with the premise that the stomatal opening varies in such a way as to maintain

a constant ratio between the incremental change in assimilation and transpiration rates due to an incremental change in conductance (Cowan and Farquhar, 1977):

$$\frac{\partial E}{\partial g_{sc}} \Big/ \frac{\partial A}{\partial g_{sc}} = \lambda \tag{A5}$$

The derivative of A with respect to g_c is found from Equation (A4) as:

$$\frac{\partial A}{\partial g_c} = \frac{k^2(c_a - \Gamma)}{(k + g_c)^2}$$

Substituting $k = \frac{A}{c_a - \Gamma - A/g_c}$ from Equation (A4) we find:

$$\frac{\partial A}{\partial g_c} = \frac{A^2}{g_c^2(c_a - \Gamma)}$$

and to relate this to stomatal conductance only, note that:

$$\frac{\partial g_c}{\partial g_{sc}} = \frac{1}{(1 + r_{bc} g_{sc})^2}$$

so that the expression is equally valid for g_{sc}:

$$\frac{\partial A}{\partial g_{sc}} = \frac{\partial A}{\partial g_c}\frac{\partial g_c}{\partial g_{sc}} = \frac{A^2}{g_{sc}^2(c_a - \Gamma)} \tag{A6}$$

The derivative of transpiration with respect to g_{sc} is found from Equation (A3), noting that $r_{se} = \frac{1}{1.6 g_{sc}}$, as:

$$\frac{\partial E}{\partial g_{sc}} = \frac{1.6 D_0}{\left[1 + 1.6 g_{sc}(r_{bc} + \varepsilon r_{bH}^*)\right]^2} \tag{A7}$$

From Equation (A5), we can then solve for g_{sc} to obtain:

$$g_{sc} = \frac{A}{\sqrt{\frac{1.6 D_0}{\lambda}(c_a - \Gamma)} - \frac{1.6A}{g_{be}^*}}$$

where $g_{be}^* = \frac{1}{r_{be} + \varepsilon r_{bH}^*}$, which is Equation (2.16) in the text.

Appendix B

Analytic assimilation solutions

Equation (2.4) expressed as the assimilation flux per unit leaf area can be written as:

$$A = \frac{1}{r_{sc} + r_{bc}} (c_a - c_i) \tag{B1}$$

where assimilation A is defined positive. From the biochemical equations for assimilation, intercellular CO_2 concentration can be expressed in terms of assimilation rate and maximum photosynthetic capacity or electron transport. From Equation (2.10) for the Rubisco-limited case we obtain:

$$c_i = \frac{-(V_l \Gamma_* + K' A_V)}{A_V - V_l} \tag{B2}$$

and from Equation (2.11) for the electron transport-limited case:

$$c_i = \frac{-(8A_J \Gamma_* + J\Gamma_*)}{4A_J - J} \tag{B3}$$

Quadratic solutions for the net assimilation rate are obtained by combining Equation (B1) with the expression for stomatal conductance (Equation (2.16)) and that for c_i in either the Rubisco-limited or electron transport-limited case (Equation (B2) or (B3), as:

$$A = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

where for the Rubisco-limited case we have:

$$
\begin{aligned}
a\,(\text{Rubisco-ltd}) &= r_{bc} - \tfrac{1.6}{g^*_{be}} \\
b\,(\text{Rubisco-ltd}) &= \sqrt{\tfrac{1.6D_0(c_a-\Gamma)}{\lambda}} - c_a - K' + (R_l - V_l)\left(r_{bc} - \tfrac{1.6}{g^*_{be}}\right) \\
c\,(\text{Rubisco-ltd}) &= (R_l - V_l)\left(\sqrt{\tfrac{1.6D_0(c_a-\Gamma)}{\lambda}} - c_a\right) - V_l\Gamma_* - K'R_l
\end{aligned}
$$

and for the electron transport-limited case:

$$
\begin{aligned}
a\,(\text{light-ltd}) &= 4\left(r_{bc} - \tfrac{1.6}{g^*_{be}}\right) \\
b\,(\text{light-ltd}) &= 4\left(\sqrt{\tfrac{1.6D_0(c_a-\Gamma)}{\lambda}} - c_a - 2\Gamma_*\right) + (4R_l - J)\left(r_{bc} - \tfrac{1.6}{g^*_{be}}\right) \\
c\,(\text{light-ltd}) &= (4R_l - J)\left(\sqrt{\tfrac{1.6D_0(c_a-\Gamma)}{\lambda}} - c_a\right) - \Gamma_*\,(J + 8R_l)
\end{aligned}
$$

These equations represent the exact solutions to the Farquhar biochemical photosynthesis model coupled to the Cowan/Farquhar stomatal conductance optimisation model. Analytic solutions are desirable where possible in modelling applications in order to avoid compounded round-off errors during iterations and potential degeneration into chaotic behaviour (Baldocchi, 1994).

Appendix C

Canopy long wave radiation

The long wave radiation penetration in a canopy is derived here for isothermal leaves (ie. leaves at air temperature), with the effect of leaf temperature being taken into account in the leaf energy balance (Equation (2.19)). Equations for long wave radiation from the sky and ground are given separately in §2.1. Considering first the downward long wave radiation I_{LW} in the canopy, an incremental increase in leaf area decreases the radiation penetration by an amount proportional to the leaf area increase and the radiation intensity. It also contributes an additional amount according to the Stefan-Boltzmann law, also in proportion to the additional leaf area:

$$\frac{dI_{LW,canopy}\downarrow}{dL} = -I_{LW,canopy}\downarrow + \varepsilon_l \sigma \theta_a^4$$

Integrating, with the initial condition that $I_{LW,canopy}\downarrow = 0$ when $L = 0$, we obtain:

$$I_{LW,canopy}\downarrow = \varepsilon\sigma\theta_a^4\left(1 - \mathrm{e}^{-L}\right)$$

This equation assumes that the radiation is in the downward direction only and that the leaves are all horizontally aligned. Since long wave radiation is emitted in all directions equally and we have assumed a spherical leaf angle distribution, the coefficient for diffuse radiation is inserted to account for the effect of leaf angle as for diffuse radiation:

$$I_{LW,canopy}\downarrow = \varepsilon\sigma\theta_a^4\left(1 - \mathrm{e}^{-k_d L}\right)$$

Similarly, radiation in the upward direction is found as:

$$I_{LW,canopy} \uparrow = \varepsilon\sigma\theta_a^4 \left(1 - \mathrm{e}^{-k_d(\Lambda - L)}\right)$$

Appendix D

Optimisation procedure

The source forms given by Equations (2.4) to (2.6), with fluxes obtained from Equation (2.8), are input into Equations (2.2) and (2.3) with parameters initialised to calculate the concentration profile. The parameters are then adjusted to improve the fit of the modelled to measured concentration until the best possible fit is obtained.

Equation (2.1) can be represented by the function $C(z_i; \mathbf{a})$, a function of the model parameters $\mathbf{a} = (a_1, a_2, ..., a_n)$ evaluated at height z_i. The goodness of fit of the modelled concentration to the measured concentration is measured by the "chi-square" function, given by:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{\hat{C}_i - C\left(z_i; \mathbf{a}\right)}{\sigma_i} \right)^2 \tag{D1}$$

where the sum is taken over the N levels at which the concentration is either measured or interpolated; $C(z_i; \mathbf{a})$ is the modelled concentration at level z_i; $\hat{C}_i$ is the measured concentration at level z_i; and σ_i is the standard deviation of the measured concentration.

The model seeks to minimise χ^2 with respect to the a_k parameters. To do this, the χ^2 function is expanded as a Taylor series to second order, so that:

$$\chi_n^2 = \chi_c^2 + \left(\mathbf{a}_n - \mathbf{a}_c\right) \nabla \chi_c^2 + \frac{1}{2} \left(\mathbf{a}_n - \mathbf{a}_c\right) \nabla^2 \chi_c^2 \left(\mathbf{a}_n - \mathbf{a}_c\right)$$

where χ_n^2 is chi-square calculated with the new set of parameters $\mathbf{a}_n$; χ_c^2 is that calculated

with the current set $\mathbf{a}_c$; and $\nabla\chi^2$ and $\nabla^2\chi^2$ are the first and second derivative matrices, respectively, of χ^2 with respect to $\mathbf{a}$. From this Taylor expansion it can be seen that the function is minimised (with zero derivative) when:

$$\nabla^2\chi_c^2\left(\mathbf{a}_n - \mathbf{a}_c\right) = -\nabla\chi_c^2$$

Hence if the Taylor approximation is a good one, the final model parameters can be obtained immediately from this equation. In reality, though, we must start with an initial estimate of $\mathbf{a}$ that is likely to be far from the optimised set, so that the initial χ^2 will also be far from the final one. It is then useful to use iterations of the "steepest descent" method, where a step is taken along the gradient towards the minimum by:

$$\left(\mathbf{a}_n - \mathbf{a}_c\right) = -\left(\text{constant}\right)\nabla\chi_c^2$$

The gradient of χ^2 with respect to the parameters $\mathbf{a}$ has components:

$$\frac{\partial\chi^2}{\partial a_j} = -2\sum_{i=1}^{N}\frac{\left[\hat{C}_i - C\left(z_i;\mathbf{a}\right)\right]}{\sigma_i^2}\frac{\partial C\left(z_i;\mathbf{a}\right)}{\partial a_j} \qquad j = 1,...,4 \tag{D2}$$

and the Hessian or second derivative matrix has components:

$$\frac{\partial^2\chi^2}{\partial a_j\partial a_k} = 2\sum_{i=1}^{N}\frac{1}{\sigma_i^2}\left[\frac{\partial C\left(z_i;\mathbf{a}\right)}{\partial a_j}\frac{\partial C\left(z_i;\mathbf{a}\right)}{\partial a_k} - \left[C_i - C\left(z_i;\mathbf{a}\right)\right]\frac{\partial^2 C\left(z_i;\mathbf{a}\right)}{\partial a_j\partial a_k}\right] \tag{D3}$$

The second derivative term in Equation (D3) can generally be ignored because the multiplying term, the difference between the measured and modelled concentrations, should be just the random error of measurement at each point. This can be of either sign and should be have no systematic relationship with the model, and so will tend to cancel out when summed over the concentration levels (Press et al., 1992).

The equation to be solved, then, is:

$$\sum_{k=1}^{M} \alpha_{jk} \delta a_k = \beta_j \tag{D4}$$

or, for the steepest descent method:

$$\delta a_j = (\text{constant})\, \beta_{\text{j}} \tag{D5}$$

where α_{jk} and β_j are defined as:

$$\alpha_{jk} \equiv \frac{1}{2} \frac{\partial^2 \chi^2}{\partial a_j \partial a_k} \qquad \beta_j \equiv -\frac{1}{2} \frac{\partial \chi^2}{\partial a_j}$$

The usual method for deciding which of the two equations above to attempt to solve is the Levenberg-Marquardt method, which gives varying weight to the two methods depending on the distance from the minimum. The constant in Equation (D5) is set to the reciprocal of the diagonal elements of the Hessian matrix α_{jk}, multiplied by a scaling factor λ_{LM}. A new matrix α' is defined by:

$$\alpha'_{jk} = \begin{cases} \alpha_{jk}\,(1 + \lambda_{LM})\,, & j = k \\ \alpha_{jk}, & j \neq k \end{cases} \tag{D6}$$

and this is substituted for α_{jk} in Equation (D4) to produce a single equation to be solved:

$$\sum_{k=1}^{M} \alpha'_{jk} \delta a_k = \beta_j \tag{D7}$$

The scaling factor λ_{LM} allows Equation (D7) to revert to Equation (D4) near the minimum as λ_{LM} approaches zero, or to Equation (D5) away from the minimum when λ_{LM} is large, causing the diagonal elements to dominate. The value of λ_{LM} chosen for a given iteration depends on the improvement or otherwise of the χ^2 merit function over the previous iteration.

From Equations (D2) and (D3), which are needed to solve Equation (D7) above, it can be seen that it is necessary to compute the derivative of the modelled concentration $C(z_i;\mathbf{a})$ with respect to the parameters $\mathbf{a}$. This is composed of near field and far field components:

$$\begin{aligned}\frac{\partial C}{\partial a_k} &= \frac{\partial C_n}{\partial a_k} + \frac{\partial C_f}{\partial a_k} \\ &= \int_0^\infty \frac{1}{\sigma_{w0}} \frac{\partial S(z_0)}{\partial a_k} \left[k_n \left(\frac{z-z_0}{\sigma_{w0} T_{L0}} \right) + k_n \left(\frac{z+z_0}{\sigma_{w0} T_{L0}} \right) \right] \\ &+ \frac{S(z_0)}{\sigma_{w0}} \frac{\partial T_{L0}}{\partial a_k} \left\{ \frac{\partial k_n[(z-z_0)/(\sigma_{w0}T_{L0})]}{\partial T_{L0}} + \frac{\partial k_n[(z+z_0)/(\sigma_{w0}T_{L0})]}{\partial T_{L0}} \right\} dz_0 \\ &- \frac{\partial C_n(z_R)}{\partial a_k} + \int_z^{z_R} \frac{1}{\sigma_w^2(z) T_L(z)} \left[\frac{\partial F(z)}{\partial a_k} - \frac{F(z)}{T_L} \frac{\partial T_L}{\partial a_k} \right] dz\end{aligned}$$

Since calculating the analytical derivatives of the source and flux profiles with respect to the adjustable parameters becomes very complicated, these are calculated numerically according to:

$$\frac{\partial S\left(z,\mathbf{a}\right)}{\partial a_k} \approx \frac{S\left(z;\mathbf{a}:a_k+\varepsilon_k\right) - S\left(z;\mathbf{a}:a_k-\varepsilon_k\right)}{2\varepsilon_k}$$

where ε_k is chosen as 0.001 times the magnitude of the parameter in question. The flux, near-field kernel and Lagrangian time scale derivatives were treated in the same way. The flux itself is calculated numerically from the source profile using the trapezoidal rule applied to Equation (2.8).

The inverse of the Hessian matrix is the covariance matrix C_{ij} of the standard errors in the adjustable parameters (Press *et al.*, 1992): $[C] \equiv [\alpha]^{-1}$. Hence the diagonals of $[C]$ give the standard errors in the parameters, as:

$$\sigma(a_i) = \sqrt{C_{ii}}$$

and the parameter correlation coefficients are given by:

$$r_{ij} = \frac{C_{ij}}{\sqrt{C_{ii}C_{jj}}}$$

If the measurement errors are normally distributed, then the confidence intervals in the parameters $\pm\delta a_i$ are related to the standard error in the usual way, with the 68% confidence interval being $\pm\sigma(a_i)$, the 95% interval $\pm 2\sigma(a_i)$, and so on.

Appendix E

Pressure-height relation

The hydrostatic equation states that:

$$\frac{dp}{dz} = -\rho g \tag{E1}$$

where p is the atmospheric pressure; z is the height above the ground; ρ is the air density and g is the acceleration due to gravity ($\sim$10 ms^{-2}). If we assume that temperature decreases with height at the adiabatic lapse rate γ, then temperature T can be written:

$$T = T_0 - \gamma z \tag{E2}$$

where T_0 is the temperature at the ground. Combining Equations (E1) and (E2) with the universal gas law, $p = \rho R_m T$ we obtain:

$$\frac{d\ln p}{dz} = -\frac{g}{R_m(T_0 - \gamma z)}$$

which can be integrated to show:

$$\frac{p}{p_0} = \left(1 - \frac{\gamma z}{T_0}\right)^{\frac{g}{R_m \gamma}} \tag{E3}$$

Equation (E3) can be rearranged to obtain height z in terms of pressure p, given by Equation (3.10). It can also be combined again with the universal gas law to find the variation in density with height:

$$\frac{\rho}{\rho_0} = \left(1 - \frac{\gamma z}{T_0}\right)^{\frac{g}{R_m \gamma} - 1}$$

Appendix F

Temperature correction

The rate of change of temperature measured by an instrument with a delayed response (T_{meas}) is related to the temperature difference between the sensor and ambient air at temperature T_{real} as (eg. Middleton and Spilhaus, 1953):

$$\frac{dT_{meas}}{dt} = -k(T_{meas} - T_{real})$$

where the proportionality constant k is the inverse time constant of the instrument. This equation can easily be integrated if the ambient temperature T_{real} is constant. In the application to measurement of temperature in the atmospheric boundary layer, however, ambient temperature is changing with height. If we consider that temperature decreases approximately linearly with height according to the adiabatic lapse rate, and if the rate of ascent of the aircraft is approximately constant, then ambient temperature will decrease linearly with time in an ascending flight. The reverse is true for a descending flight. The measured temperature is then related to the ambient temperature by:

$$\frac{dT_{meas}}{dt} = -k[T_{meas} - (T_0 + \beta t)] \tag{F1}$$

where β is the time rate of change of temperature and T_0 is the ground temperature. Equation (F1) can be integrated to obtain real temperature in terms of measured temperature by multiplying both sides by the factor e^{kt}:

$$e^{kt}\frac{dT_{meas}}{dt} + kT_{meas}e^{kt} = ke^{kt}(T_0 + \beta t) \tag{F2}$$

The left hand side is just the time derivative of the factor $T_{meas}\mathrm{e}^{kt}$. The first term on the right hand side is also easily integrated, while the second can be integrated by parts using the rule:

$$\int udv = uv - \int vdu$$

We make the substitution $u = x$ and $dv = k\beta\mathrm{e}^{kt}$, so that $du = dx$ and $v = \beta\mathrm{e}^{kt}$. The integration can then be carried out as:

$$\begin{aligned}
\int k\beta t\mathrm{e}^{kt}dt &= \int udv \\
&= uv - \int vdu \\
&= \beta t\mathrm{e}^{kt} - \int \beta\mathrm{e}^{kt}dt \\
&= \beta t\mathrm{e}^{kt} - \frac{\beta}{k}\mathrm{e}^{kt} + c
\end{aligned}$$

Assuming the sensor to be in equilibrium with the ambient air at the start of the flight, the constant c is evaluated as $\frac{\beta}{k}$ using the initial condition $T_{meas} = T_0$ at time $t = 0$ and the above result is substituted into Equation (F2) to obtain:

$$T_{meas}\mathrm{e}^{kt} = T_0\mathrm{e}^{kt} + \beta t\mathrm{e}^{kt} - \frac{\beta}{k}\mathrm{e}^{kt} + \frac{\beta}{k}$$

Multiplying both sides by e^{-kt} we find:

$$T_{meas} = T_0 + \beta t - \frac{\beta}{k}(1 - \mathrm{e}^{-kt})$$

With our initial assumption that $T_{real} = T_0 + \beta t$ we then have:

$$T_{meas} = T_{real} - \frac{\beta}{k}(1 - \mathrm{e}^{-kt})$$

Rearranging to solve for T_{real} and neglecting the exponential term at long times, we obtain Equation (3.11):

$$T_{real} = T_{meas} + \frac{\beta}{k}$$

Appendix G

Canopy model source code

G.1 Driver code

G.1.1 Full optimisation

The following Fortran 90 code carries out the optimisation of canopy and turbulence parameters plus variable ground fluxes for the ten periods with isotope data on 25 and 27 May 2000, by minimising the difference between modelled and measured concentrations.

```
$debug
      PROGRAM CANMOD
! PROGRAM FOR FINDING CANOPY SOURCE DENSITY PROFILE S FROM
! CONCENTRATION PROFILE C, ASSUMING STEADY, ONE-DIMENSIONAL TRANSFER.
! CONVENTIONS AND USING MULTILAYER CANOPY MODEL WITH UNKNOWN
! PARAMETERS. DEVELOPED FROM INVERSE LAGRANGIAN SOURCE CODE OF
! RAUPACH 1988-1998. DATA INPUT SECTION MODIFIED BY RAY LEUNING 1996.
! MANY REDUNDANT ARRAYS AND ROUTINES STILL PRESENT
!
! Subroutines used
!   comskp          (sousub15.for) skip comments in data input file
!   air             (sousub15.for) calculate latent ht vap, RHO, CP, VOLM
!   turb2           (sousub15.for) calculate sigw and TL
!   dispm2           (sousub15.for) calculate dispersion matrix
!   integr         (sousub15.for) integrate a function
!   shsor2           (sousub15.for) sort profiles into ascending order
!   phi             (sousub15.for) calculate phi for stability correction
!   interp          (sousub15.for) interpolate concentration profiles
!   intval          (sousub15.for) calculate interpolated value
!   calce           (sousub15.for) calculate flux profiles
!   calcs           (sousub15.for) calculate source profiles
!   assim           (sousub15.for) calculate leaf level fluxes
!
!   numfiles=615 for half hour, 308 for hour averages
!
      PARAMETER(MMAX=20,NMAX=40,numfiles=308,NUML=20,NUMP=9,IFMAX=1000, &
            ISMAX=1000, NUMK=36,NISO=16,NUMAK=54,IDMAX=400)

! NUML=NUMBER OF CONCENTRATION LEVELS TO BE USED FROM SPLINE INTERPOLATION
! NUMP=NUMBER OF SCALAR INPUT PROFILES
! SOURCE ARRAYS, DIMENSION 0:M WHERE M IS NUMBER OF SOURCE LEVELS,
! Z=HEIGHT OF TOP OF SOURCE LAYERS; ZS=HEIGHT OF MIDDLE OF SOURCE LAYERS;
! DZ=WIDTH OF SOURCE LAYERS
```

```
! SA,SH,SE,SC = OUTPUT SOURCES; FA,FH,FE,FC = OUTPUT FLUXES
! DIS,DISN,DISF = DISPERSION MATRICES
! T,Q,EV,CC,CA,CH,CE = PROFILES INPUT TO MODEL (AFTER INTERPOLATION)
! ZC NO LONGER USED

      REAL, DIMENSION(:),ALLOCATABLE :: Z,ZS,DZ,FA,SA, &
               FH,SH,FE,SE,FC,SC,HH,BBADD
    REAL, DIMENSION(:,:), ALLOCATABLE :: DIS,DISN,DISF
    REAL, DIMENSION(:), ALLOCATABLE :: T,Q,EV,CC,CA,CH,CE,BB
    REAL*8, DIMENSION(:), ALLOCATABLE :: AA,AAL
    integer, dimension(:), allocatable :: IP
    REAL, DIMENSION(:,:), ALLOCATABLE :: ZC
    REAL A1,AH,A0,CCH,CC0,ALFAA,ALFAB,ALFAC,HC,SWREF,USREF
      COMMON /T/ ITURB,A1,AH,A0,CCH,CC0,alfaa,alfab,ALFAC, &
                HC,SWREF,USREF,IST_COR

      character mon*3,day(numfiles)*2,timdat*32
      character*60 exptnam,filenam,path_IP,path_OP
      character*90 path_file_IP,outfila,outfils,outdmtx

!     arrays needed for profile and flux data for "numfiles" half-hrs
      character*8                                        &   !time & date
      Ttdate(numfiles),VPdate(numfiles),C2date(numfiles), &
      Fxdate(numfiles),SIGCDAT(NUMFILES),SIGHDAT(NUMFILES), &
      SIGTDAT(NUMFILES),D13DAT(NISO)

      integer Tthr(numfiles),Ttmin(numfiles),VPhr(numfiles), &
             VPmin(numfiles),C2hr(numfiles),C2min(numfiles), &
             Fxhr(numfiles),Fxmin(numfiles),SIGCHR(NUMFILES),SIGCMIN(NUMFILES), &
            SIGHHR(NUMFILES),SIGHMIN(NUMFILES),SIGTHR(NUMFILES),SIGTMIN(NUMFILES),&
            D13HR(NISO),D13MIN(NISO),NLI(NISO),nday(numfiles),ISN(10),ICN(10)
      common /timx/ Tthr,Ttmin,VPhr,VPmin,C2hr,C2min, Fxhr,Fxmin
      integer timm(numfiles,10)
      equivalence (timm,Tthr)

! Ttz,VPz,C2z = ORIGINAL HEIGHTS OF INPUT PROFILES
! ZZZ = INCORPORATES ORIGINAL HEIGHTS OF ALL SCALAR PROFILES INTO ONE MATRIX AFTER SORTING
! NLEVL = ORIGINAL NUMBER OF LEVELS FOR PROFILE DATA
! X,Y = HEIGHTS AND PROFILES AFTER INTERPOLATION
! XPROF=X
! Tt,VP,C2 = ORIGINAL INPUT PROFILES
! PROF = INCORPORATES ALL ORIGINAL PROFILES INTO ONE MATRIX AFTER SORTING
! PRAF = DUMMY MATRIX FOR SORTING
! SPROF = ALL PROFILES AFTER INTERPOLATION IN ONE MATRIX

      REAL, DIMENSION(:), ALLOCATABLE :: Ttz, VPz, C2z, za,B,C13Z,O18Z
    REAL, DIMENSION(:,:), ALLOCATABLE :: ZZZ,C13ZZ
    integer nlevl(11),nref(11)      !number of levels for each scalar
    REAL ZREF,Y(NUML),X(NUML),XPROF(NUML),NEWZ,NEWPROF

    REAL, DIMENSION(:,:), ALLOCATABLE :: Tt, VP, C2, PRAF,SPROF,F,SIGC,SIGH,SIGT,dcdt,DHDT,DTDT,&
                                         D13C,D18O
    REAL, DIMENSION(:,:,:), ALLOCATABLE :: PROF
    REAL*8 RL, DTIME(NUMFILES)

      real H(numfiles),E(numfiles),FC2(numfiles),zMonU(numfiles),us(numfiles),swr(numfiles), &
         LE(numfiles),RN(numfiles),Uh(numfiles), APRESS(NUMFILES),PAR(NUMFILES), fd(numfiles), &
         SWDR(NUMFILES),LWDR(NUMFILES),LZMON(10),LMON(NUMFILES)
      common /flxx/ H,E,FC2,sgw,us,ur
      real flxx(numfiles,7)
      equivalence (flxx,H)

! ARRAYS FOR ALTERNATIVE MODEL USING BETA FUNCTION TO CONSTRAIN SOURCE AND FLUX PROFILES
    REAL SK(6,0:ISMAX),FK(6,0:IFMAX),DSK(5,0:IDMAX+1,NUMAK),PARAM(5),&
        CD(NUML),SREF(6,10),THETL(0:ISMAX), ESATL(0:ISMAX),DUMMY(NUML),DEF(0:ISMAX), &
        DELTA(0:ISMAX),CICA(0:ISMAX),DEL18(0:ISMAX),RAIR(10), &
```

```
        SWDREF(10), LWDREF(10),TLPROF(0:ISMAX),ZOBUKHOV(0:ISMAX)
    REAL    AK(NUMAK),ake(NUMAK), AKO(NUMAK),GF(3,10),GFO(3,10),GFE(3,10), &
            TL1(10),TL1O(10),TL1E(10),TL2(10),TL2O(10),TL2E(10),STL1(10),STL2(10)
    REAL CK(6,NUML),CKO(6,NUML), CKK(6,NUML),CKKK(6,NUML), CKE(6,NUML), &
        SIGCO2(NUML), SIGH2O(NUML),SIGTEMP(NUML),SIG13C(NUML),SIG18O(NUML), &
        SIG(5,NUML),SIGO(5,NUML)
    REAL SPROFALL(NUML,9,10),APROFALL(NUML,4,10),DCKALL(5,NUML,NUMAK,10),CKALL(6,NUML,10), &
        DCKEALL(5,NUML,NUMAK,10),DCKOALL(5,NUML,NUMAK,10),CKEALL(6,NUML,10), &
        CKOALL(6,NUML,10), &
        CKKALL(6,NUML,10),DCKD(5,NUML,NUMAK,10),DFKTOP(3,10,NUMAK),FKTOP(3,10), &
        SPROFD(NUML),APROFC(NUML),APROFO(NUML), SPROFP(NUML), APROFCP(NUML), APROFOP(NUML)
    REAL DFK(5,0:IDMAX+1,NUMAK),DCK(5,NUML,NUMAK),FKS(6,0:ISMAX),esource(0:ISMAX), &
        ESOURCE2(0:ISMAX),&
        HSOURCE(0:ISMAX),HFLUX(0:IFMAX),HSOURCE2(0:ISMAX),eflux(0:IFMAX), CFLUX(0:IFMAX), &
        CFLUX1(NUMAK,0:IFMAX),CFLUX2(NUMAK,0:IFMAX),CSOURCE(0:ISMAX),CSOURCE2(0:ISMAX), &
        ZPR(0:IFMAX), EFLUX1(NUMAK,0:IFMAX),EFLUX2(NUMAK,0:IFMAX), &
        HFLUX1(NUMAK,0:IFMAX),HFLUX2(NUMAK,0:IFMAX), EFLUXD(0:IFMAX),CFLUXD(0:IFMAX), &
        HFLUXD(0:IFMAX), &
        THETLE(0:ISMAX),ZOUT(IFMAX),C13FLUX(0:IFMAX),C13FLUXD(0:IFMAX),C13SOURCE(0:ISMAX), &
        RLAI(0:ISMAX),RC(0:ISMAX),RH(0:ISMAX),RNN(0:ISMAX),C13FLUX1(NUMAK,0:IFMAX), &
        C13FLUX2(NUMAK,0:IFMAX),RSUM13(5), RSUM14(5), RSUM15(5),RSUM18(5), &
        C13SOURCE2(0:ISMAX),rsum4(5),rsum5(5),rsum6(5),rsum7(5),rsum8(5), RSUM12(5), &
        RSUM10(5), &
        DFKS(10,0:ISMAX), RSUM9(5),KBF,KDF, DCKO(5,NUML,NUMAK), DCKE(5,NUML,NUMAK), &
        O18SOURCE(0:ISMAX),O18FLUX(0:IFMAX),O18FLUXD(0:IFMAX),O18SOURCE2(0:ISMAX), &
        O18FLUX2(NUMAK,0:IFMAX),O18FLUX1(NUMAK,0:IFMAX), &
        H18SOURCE(0:ISMAX),H18FLUX(0:IFMAX),H18FLUXD(0:IFMAX),DEL_E(0:ISMAX),ZSOU(0:ISMAX)
    REAL*8 ALP(NUMK,NUMK), BET(NUMK),DAK(NUMK),AINV(NUMK,NUMK),DSUM
    REAL    LSIGC(2),LSIGH(2),LSIGT(2),BINT(3),BINT2(3),IPN(NUMK), DCT(6,2),DCFT(5,2),&
            DISC(NUML,0:ISMAX),DISCN(NUML,0:ISMAX),DISCF(NUML,0:ISMAX), &
            APROF(NUML,2,NISO), AZZ(NUML)
    REAL*8 BETA12(3),costfn(10,3,9),HESS(NUMK,NUMK),&
            check(2,2),xxxx(numk)
    EXTERNAL BETA, BETAI
    DATA RSTD /0.011894/, R18STD /0.0020672/

! PRESET PARAMETERS:
      rmair = 0.02897        ! molecular weights (kg/mol) for air
      rmh2o = 0.018016       ! for H2O
      rmco2 = 0.044022       ! for CO2
      pmb   = 1000.          ! atmospheric pressure
      PI    = 3.14159
    IWRITE=1               ! WRITE DISPERSION MATRIX
    ICOST=1

! open and read control information from sourcfil.dat
      open(1,file='sourcfil_15d.dat')
2     call comskp(1)                    !skip comments

      read(1,*,end=450) exptnam         !experiment name
    read(1,105,end=450) path_IP        !Input path name
    read(1,105,end=450) path_OP        !Output path name
105   format(a60)
      flen_IP=len_trim(path_IP)
      flen_OP=len_trim(path_OP)

      do 10000 iblk=1,300               !read up to 300 days' file names

        READ(1,*,end=450) (nlevl(I),I=1,3),(NREF(i),i=1,3),HC,PMB
        READ(1,*,err=450,END=450) ITURB,MO,M,A1,AH,AO,CCH,CCO,alfaa,alfab,ALFAC,ist_cor
        read(1,*,err=450,end=450) nfil

    MN=MAXVAL(NLEVL)
    ALLOCATE(Z(0:M),ZS(0:M),DZ(0:M),FA(0:M),SA(0:M),FH(0:M),SH(0:M), &
             FE(0:M),SE(0:M),FC(0:M),SC(0:M))
      ALLOCATE(AA((M+1)**2),BB(M+1),IP(M+1),AAL((M+1)**2),BBADD(M+1))
```

```
    ALLOCATE(T(NUML),Q(NUML),EV(NUML),CC(NUML), CA(NUML),CH(NUML),CE(NUML))
    ALLOCATE(ZC(NUMP,MN),HH((M+1)**2))
    ALLOCATE(Tt(MN,NUMFILES),C2(MN,NUMFILES),&
             VP(MN,NUMFILES),PROF(MN,NUMP,NUMFILES),SPROF(NUML,NUMP), &
            SIGC(MN,NUMFILES),SIGH(MN,NUMFILES),SIGT(MN,NUMFILES), &
            dcdt(mn,numfiles),DHDT(MN,NUMFILES),DTDT(MN,NUMFILES))
    ALLOCATE(PRAF(MN,NUMFILES),D13C(MN,NISO),D18O(MN,NISO))
    ALLOCATE(Ttz(MN),C2z(MN),VPz(MN),ZA(MN),ZZZ(MN,NUMP),B(MN),F(4,MN),C13Z(MN),O18Z(MN))
    ALLOCATE(DIS(NUML,0:M),DISN(NUML,0:M),DISF(NUML,0:M),C13ZZ(MN,NISO))

        do ifa=1,nfil                          !read data for profiles and fluxes
          read(1,105) filenam
        path_file_IP=path_IP(1:flen_IP)//filenam

          open(10,file=path_file_IP,mode='read')

          select case(filenam(1:1))
            case('C','c')
              select case(filenam(2:2))
              case('O','o')                     !CO2
                  read(10,*)
                read(10,200) (C2z(j),j=1,nlevl(1))        !read in CO2 heights
                do i=1,numfiles                             !read in CO2 data
                  read(10,210,end=100,err=460) &
                   C2date(i),C2hr(i),C2min(i),(C2(j,i),j=1,nlevl(1))
                end do
100             continue
                nC2=i-1                                  !# CO2 records

              end select

            case('T','t')
              read(10,*)                                 !skip one line
              read(10,200) (Ttz(j),j=1,nlevl(2))            !read in Temp heights
              do i=1,numfiles                               !read in Temp data
                read(10,210,end=120,err=480)  &
                  Ttdate(i),Tthr(i),Ttmin(i),(Tt(j,i),j=1,nlevl(2))
              end do
120           continue
              nTt=i-1                                    !# Temp records


            case('V','v')
              read(10,*)                                 !skip one line
              read(10,200) (VPz(j),j=1,nlevl(3))            !read in VP heights
              do i=1,numfiles                               !read in VP data
                read(10,210,end=130,err=490)  &
                  VPdate(i),VPhr(i),VPmin(i),(VP(j,i),j=1,nlevl(3))
              end do
130           continue
              nVP=i-1                                    !# VP records

            case('F','f')
              read(10,*,end=140,err=500)                 !skip 1 lines
              do i=1,numfiles
                  read(10,220,end=140) fxdate(i),fxhr(i),fxmin(i), &
                    H(i),LE(i),E(i),FC2(i),us(i),swr(i),zMonU(i),Uh(i),rn(i), &
                    PAR(I),fd(i),SWDR(I),LWDR(I)
              end do
140           continue
              nfx=i-1                                    !# flux records

        CASE('A','a')
            READ(10,*)
            DO I=1,NUMFILES
                READ(10,270,END=141) DTIME(I), apress(i)
```

```
          ENDDO
141       CONTINUE
!       apress=1000.

        CASE('S','s')
          SELECT CASE(FILENAM(7:7))
          CASE('C')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,230, END=150,ERR=510) nday(i),SIGCDAT(I),SIGCHR(I), &
               SIGCMIN(I),(SIGC(J,I),J=1,NLEVL(1))
            ENDDO
150         CONTINUE
          CASE('H')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,231, END=160,ERR=520) SIGHDAT(I),SIGHHR(I), &
               SIGHMIN(I),(SIGH(J,I),J=1,NLEVL(3))
            ENDDO
160         CONTINUE
          CASE('T')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,231, END=170,ERR=530) SIGTDAT(I),SIGTHR(I), &
               SIGTMIN(I),(SIGT(J,I),J=1,NLEVL(2))
            ENDDO
170         CONTINUE
          CASE('A')
            READ(10,*)
            READ(10,*)
            READ(10,232) (SIGCO2(J),J=1,NUML)
            READ(10,232) (SIGH2O(J),J=1,NUML)
            READ(10,232) (SIGTEMP(J),J=1,NUML)
            READ(10,232) (SIG13C(J),J=1,NUML)
            READ(10,232) (SIG18O(J),J=1,NUML)
          END SELECT

        CASE('D','d')
          SELECT CASE(FILENAM(2:2))
          CASE('C','c')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,240,END=180) (DCDT(J,I),J=1,NLEVL(1))
            ENDDO
180         CONTINUE
          CASE('H','h')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,240,END=190) (DHDT(J,I),J=1,NLEVL(1))
            ENDDO
190         CONTINUE
          CASE('T','t')
            READ(10,*)
            READ(10,*)
            DO I=1,NUMFILES
              READ(10,240,END=195) (DTDT(J,I),J=1,NLEVL(1))
            ENDDO
195         CONTINUE
          CASE('I','i')
            SELECT CASE(FILENAM(4:4))
            CASE('C','c')
```

```
                READ(10,*)
                READ(10,260,END=196) (C13z(j),j=1,6)        !read in C13 heights
                DO I=1,NISO
                  READ(10,250) D13DAT(I),D13HR(I),D13MIN(I),(D13C(J,I),J=1,6)
                ENDDO
196             CONTINUE
              CASE('O','o')
                READ(10,*)
                READ(10,200,END=197) (O18Z(J),J=1,6)
                DO I=1,NISO
                  READ(10,260) (D18O(J,I),J=1,6)
                ENDDO
197             CONTINUE
              END SELECT
            END SELECT

          case default
            stop 'Incorrect profile or flux file type'

          end select

200       format(18x,10f8.2)
210       format(a8,i5,1x,i2,10f8.2)
220     format(a8,i5,1x,i2,6f8.2,f9.2,6f8.2)
230     format(I5,a8,i5,1x,i2,10f8.3)
231     format(a8,i5,1x,i2,10f8.3)
232     FORMAT(8X,20F8.2)
240     FORMAT(16X,10ES11.3)
250     FORMAT(A8,I5,1X,I2,6F9.3)
260     FORMAT(16X,6F9.3)
270     FORMAT(F11.3,F8.2)

        end do                                          !end reading block
!                  of files

    IX=1
    RMSIGC=MAXVAL(SIGC)
    RMSIGH=MAXVAL(SIGH)
    RMSIGT=MAXVAL(SIGT)
    LSIGC=MAXLOC(SIGC)
    LSIGH=MAXLOC(SIGH)
    LSIGT=MAXLOC(SIGT)
    WRITE(*,*) RMSIGC, RMSIGH, RMSIGT
    WRITE(*,*) LSIGC, LSIGH, LSIGT
!     assume numfiles 1/2h records available each day
      nrec=numfiles
    PROF(:,1,:)=C2
    PROF(:,2,:)=TT
    PROF(:,3,:)=VP
    PROF(:,4,:)=SIGC
    PROF(:,5,:)=SIGT
    PROF(:,6,:)=SIGH
    PROF(:,7,:)=DCDT
    PROF(:,8,:)=DTDT
    PROF(:,9,:)=DHDT
    SIG(1,:)=SIGCO2; SIG(2,:)=SIGTEMP; SIG(3,:)=SIGH2O; SIG(4,:)=SIG13C; SIG(5,:)=SIG18O
    SIG(1,:)=0.5*SIG(1,:);SIG(2,:)=0.2*SIG(2,:);SIG(3,:)=0.2*SIG(3,:)
    SIG(4,:)=0.4*SIG(4,:);SIG(5,:)=0.6*SIG(5,:) !WEIGHT TO EACH SPECIES
ZZZ(:,1)=C2z
    ZZZ(:,2)=Ttz
    ZZZ(:,3)=VPz
    ZZZ(:,4)=C2z
    ZZZ(:,5)=Ttz
    ZZZ(:,6)=VPz
    ZZZ(:,7)=C2z
    ZZZ(:,8)=Ttz
```

```
    ZZZ(:,9)=VPz
    PROF(:,4,:)=0.3; PROF(:,5,:)=0.3    !CONSTANT SIGMA FOR EACH (10X ACTUAL)
!   PROF(7:9,6,:)=0.3
    prof(:,6,:)=0.05
    DO I=4,9
        NLEVL(I)=NLEVL(I-3)
    ENDDO

! INTERPOLATE D13C, D18O PROFILES
    TH=28.0
    DO I=1,NISO
    NLI(I)=6;C13ZZ(:,I)=C13Z
      DO J=1,6      !CHECK FOR BAD DATA
        IF(ABS(D13C(J,I)).GT.100.) THEN
          NLI(I)=NLI(I)-1
          DO K=J,5
            D13C(K,I)=D13C(K+1,I)
            D18O(K,I)=D18O(K+1,I)
            C13ZZ(K,I)=C13Z(K+1)
          ENDDO
        ENDIF
      ENDDO
! INTERPOLATE ISOTOPE PROFILES
      CALL INTERP(NLI(I),C13ZZ(:,I),D13C(:,I),X,Y,TH,NUML,NV,B,F)
      APROF(:,1,I)=Y; AZZ=X
      CALL INTERP(NLI(I),C13ZZ(:,I),D18O(:,I),X,Y,TH,NUML,NV,B,F)
      APROF(:,2,I)=Y
    ENDDO

!       sort profiles into ascending height order - assume order read in is sequential
!       but may be in descending order
!       utilise equivalence between zzz(nv,nf) & Ttz(nv),VPz(nv,nf)...
!       and between prof(nv,nf,nr) & Tt(nv,nr),VP(nv,nr)...
        do nf=1,NUMP                                !profile files
          if(zzz(1,nf).gt.zzz(nlevl(nf),nf)) then     !need to reverse order
            do nv=1,nlevl(nf)                           !n levels
              za(nv)=zzz(nlevl(nf)-nv+1,nf)             !write to scratch array
              do nr=1,nrec                          !nrec records common to
! all profile files
                praf(nv,nr)=prof(nlevl(nf)-nv+1,nf,nr)   !write to scratch array
              end do                                !end record loop
            end do                                  !end level loop
!           now overwrite arrays in ascending order
            do nv=1,nlevl(nf)
              zzz(nv,nf)=za(nv)
              do nr=1,nrec
                prof(nv,nf,nr)=praf(nv,nr)
              end do
            end do
          end if
        end do

      DO II=1,NUMP
       DO nv=1,MN
          zc(II,nv)=ZZZ(nv,II)
       ENDDO
      ENDDO

!       end data input section

!       ***********************************************************

!       construct output file names from date info
        mon=Ttdate(1)(6:8)
        day(1)=Ttdate(1)(3:4)
        if(day(1).eq.' ') day(1)='0'
```

```
!       ************************************************************
!
!   INTERPOLATE BETWEEN PROFILES
!
!   FOR TEMPERATURE ADD LINEARLY INTERPOLATED POINT BETWEEN TOP TWO
!   VALUES TO PREVENT LARGE DEVIATIONS OF THE SPLINE BETWEEN THESE TWO

        NEWZ=(ZZZ(NLEVL(2),2)+ZZZ(NLEVL(2)-1,2))/2
        ZZZ(NLEVL(2)+1,2)=ZZZ(NLEVL(2),2)
        ZZZ(NLEVL(2),2)=NEWZ
        NLEVL(2)=NLEVL(2)+1

        do 3000 nr=1,nrec

!   ADD LINEARLY INTERPOLATED POINT BETWEEN TOP TWO VALUES FOR T AND
!   SIGT TO PREVENT SPURIOUS MAXIMA/MINIMA IN THE INTERPOLATION

        NEWPROF=(PROF(NLEVL(2)-1,2,NR)+PROF(NLEVL(2)-2,2,NR))/2
        PROF(NLEVL(2),2,NR)=PROF(NLEVL(2)-1,2,NR)
        PROF(NLEVL(2)-1,2,NR)=NEWPROF
        NEWPROF=(PROF(NLEVL(2)-1,5,NR)+PROF(NLEVL(2)-2,5,NR))/2
        PROF(NLEVL(2),5,NR)=PROF(NLEVL(2)-1,5,NR)
        PROF(NLEVL(2)-1,5,NR)=NEWPROF

    TH=28.0
    DO II=1,NUMP
    CALL INTERP(NLEVL(II),ZZZ(:,II),PROF(:,II,NR),X,Y,TH,NUML,NV,B,F)
        SPROF(:,II)=Y
    ENDDO
        XPROF=X

!      ****************************************************************
!       Process all available records using MRR's program

!         check if turbulence data available (criterion: u* > 5 m/s)
          if(us(nr).gt.5.) goto 3000               !try next record

        day(nr)=Ttdate(nr)(3:4)

!         assign fluxes at reference height
          Href=H(nr)                                !heat
          Eref=LE(nr)                               !LE
          F2ref=FC2(nr)                             !CO2
          swref=swr(nr)                            !sig_w above canopy
!          usref=swref/a1                               !u* ref
USREF=US(NR)
          tc=SPROF(NUML,2)                     !ambient temp (deg C)
      Uhref=Uh(nr)/1.3

!         check if stability correction required
        if(ist_cor.eq.1) then
          IF(TTHR(NR).LT.5) ZMON=10.
          IF(TTHR(NR).GE.5.AND.TTHR(NR).LT.7) ZMON=50.
          IF(TTHR(NR).GE.7.AND.TTHR(NR).LT.8) ZMON=100.
          IF(TTHR(NR).GE.8.AND.TTHR(NR).LT.10) ZMON=500.
          IF(TTHR(NR).GE.10.AND.TTHR(NR).LT.17) ZMON=1000.
          IF(TTHR(NR).GE.17.AND.TTHR(NR).LT.18) ZMON=500.
          IF(TTHR(NR).GE.18.AND.TTHR(NR).LT.19) ZMON=100.
          IF(TTHR(NR).GE.19.AND.TTHR(NR).LT.22) ZMON=50.
          IF(TTHR(NR).GE.22) ZMON=10.
!         IF(TTHR(NR).LT.10.OR.TTHR(NR).GT.16) THEN
!         zMon=17./zMonU(nr)                          !Monin Obukhov length
         else
          zMon=-1000.                             !set to large -ve value
!                                                              -> neutral
!         endif
```

```
        end if

          call air(tc,pmb,rho,volm,capp,rlam,qsat,epsi)
          LMON(NR)=USREF**3*(TC+273.15)*RHO*CAPP/0.4/10./HREF   !MONIN-OBUKHOV LENGTH

3000    continue

!*************************************************************************
    I=1
    NUMISO=0
    DO 4004 NR=1,NREC
    NIS=3
!   IF((DAY(NR).EQ.'25'.OR.DAY(NR).EQ.'27').AND.(TTHR(NR).GE.6.AND.TTHR(NR).LT.21)) THEN
    ISOTOPE: DO NI=1,NISO
  IF(D13DAT(NI)(3:4).EQ.DAY(NR).AND.(D13HR(NI)+1).EQ.TTHR(NR).AND. &
    TTHR(NR).LT.21.AND.TTHR(NR).GE.6) THEN
        ISN(I)=NI; ICN(I)=NR; NUMISO=NUMISO+1
    DO II=1,3
    CALL INTERP(NLEVL(II),ZZZ(:,II),PROF(:,II,ICN(I)),X,Y,TH,NUML,NV,B,F)
        SPROFALL(:,II,I)=Y
    CALL INTERP(NLEVL(II),ZZZ(:,II),PROF(:,II+6,ICN(I)),X,Y,TH,NUML,NV,B,F)
        SPROFALL(:,II+6,I)=Y    ! STORAGE
    ENDDO
      CALL INTERP(NLI(NI),C13ZZ(:,NI),D13C(:,NI),X,Y,TH,NUML,NV,B,F)
      APROFALL(:,1,I)=Y; AZZ=X
      CALL INTERP(NLI(NI),C13ZZ(:,NI),D18O(:,NI),X,Y,TH,NUML,NV,B,F)
      APROFALL(:,2,I)=Y
    I=I+1
    ENDIF
    ENDDO ISOTOPE
4004 CONTINUE
    ISOTOPES=2  ! USE ISOTOPES IN OPTIMISATION: 0=NO, 1=NO BUT CALCULATE ISOTOPE PROFILES,
                !   2=YES FOR 13C, 3=YES FOR 13C AND 18O
202 IEDDY=0     ! USE EDDY FLUXES TO CONSTRAIN 1=YES, 0=NO
    IDIS=0      ! USE DISPM SUBROUTINE FOR DERIV AND CONC CALC 1=YES, 0=NO
    ICOSTFN=2   ! 1=calculate cost function (chisq) only, no optimisation,
                ! 2=output error covariance matrix
    IFLAGC=0    ! exit after one iteration, including deriv calc
    IOPT=1      ! 0=OPTIMISE ONLY GROUND FLUXES, 1=ALSO OTHER PARAMS
    IPM=1
    IN=1
    IF(ISOTOPES.GE.1) THEN
      DO IT=1,NUMISO
      IF(IT.EQ.5.OR.IT.EQ.10) THEN
        CALL INTERP(NLEVL(1),ZZZ(:,1),PROF(:,1,ICN(IT)+3),X,Y,TH,NUML,NV,B,F)
        SPROFD=Y
        CALL INTERP(NLI(ISN(IT)+1),C13ZZ(:,ISN(IT)+1),D13C(:,ISN(IT)+1),X,Y,TH,NUML,NV,B,F)
        APROFC=Y
        CALL INTERP(NLI(ISN(IT)+1),C13ZZ(:,ISN(IT)+1),D18O(:,ISN(IT)+1),X,Y,TH,NUML,NV,B,F)
        APROFO=Y
        SPROFP=SPROFALL(:,1,IT-1)
        APROFCP=APROFALL(:,1,IT-1)
        APROFOP=APROFALL(:,2,IT-1)
        APROFALL(:,3,IT)=(APROFALL(:,1,IT)/1000.+1.)*RSTD*(SPROFD-SPROFP)/6./3600.+ &
            (APROFC-APROFCP)/1000./6./3600.*RSTD*SPROFALL(:,1,IT)
        APROFALL(:,4,IT)=(APROFALL(:,2,IT)/1000.+1.)*R18STD*(SPROFD-SPROFP)/6./3600.+ &
            (APROFO-APROFOP)/1000./6./3600.*RSTD*SPROFALL(:,1,IT)
      ELSEIF(IT.EQ.1.OR.IT.EQ.6) THEN
        SPROFD=SPROFALL(:,1,IT+1)
        APROFC=APROFALL(:,1,IT+1)
        APROFO=APROFALL(:,2,IT+1)
        CALL INTERP(NLEVL(1),ZZZ(:,1),PROF(:,1,ICN(IT)-3),X,Y,TH,NUML,NV,B,F)
        SPROFP=Y
        APROFALL(:,3,IT)=(APROFALL(:,1,IT)/1000.+1.)*RSTD*(SPROFD-SPROFP)/6./3600.+ &
            (APROFC-APROFALL(:,1,IT))/1000./3./3600.*RSTD*SPROFALL(:,1,IT)
        APROFALL(:,4,IT)=(APROFALL(:,2,IT)/1000.+1.)*R18STD*(SPROFD-SPROFALL(:,1,IT)) &
```

```
          /3./3600.+ (APROFO-APROFALL(:,2,IT))/1000./3./3600.*RSTD*SPROFALL(:,1,IT)
      ELSE
        SPROFD=SPROFALL(:,1,IT+1); SPROFP=SPROFALL(:,1,IT-1)
        APROFC=APROFALL(:,1,IT+1); APROFCP=APROFALL(:,1,IT-1)
        APROFO=APROFALL(:,2,IT+1); APROFOP=APROFALL(:,2,IT-1)
        APROFALL(:,3,IT)=(APROFALL(:,1,IT)/1000.+1.)*RSTD*(SPROFD-SPROFP)/6./3600.+ &
            (APROFC-APROFCP)/1000./6./3600.*RSTD*SPROFALL(:,1,IT)
        APROFALL(:,4,IT)=(APROFALL(:,2,IT)/1000.+1.)*R18STD*(SPROFD-SPROFP)/6./3600.+ &
            (APROFO-APROFOP)/1000./6./3600.*RSTD*SPROFALL(:,1,IT)
      ENDIF
      ENDDO
    ENDIF
      NIS=NI
    open(unit=10, file='cprof')
    OPEN(UNIT=11, FILE='SPROFC')
    OPEN(UNIT=16, FILE='EPROF')
    OPEN(UNIT=13, FILE='HPROF')
    OPEN(UNIT=17, FILE='SPROFE')
    OPEN(UNIT=14, FILE='SPROFH')
    open(unit=12, file='FLUXC')
    OPEN(UNIT=15, FILE='FLUXH')
    OPEN(UNIT=18, FILE='FLUXE')
    OPEN(UNIT=19, FILE='FLUXLE')
    OPEN(UNIT=20, FILE='MOREPROFS')
    OPEN(UNIT=21, FILE='C13PROF')
    OPEN(UNIT=22, FILE='SPROFC13')
    OPEN(UNIT=23, FILE='GROUNDFLUX')
    OPEN(UNIT=24, FILE='PROFC')
    OPEN(UNIT=25, FILE='PROFE')
    OPEN(UNIT=26, FILE='PROFH')
    OPEN(UNIT=27, FILE='PROFC13')
    OPEN(UNIT=28, FILE='O18PROF')
    OPEN(UNIT=29, FILE='SPROFO18')
    OPEN(UNIT=30, FILE='H18PROF')
    OPEN(UNIT=31, FILE='CHISQ')
    IWRITE=0
    IFLAG=0
    DO IT=1,NUMISO
!   GTEMP(IT)=PROF(1,2,ICN(IT)); RNREF(IT)=RN(ICN(IT));
        SWDREF(IT)=SWDR(ICN(IT)); LWDREF(IT)=LWDR(ICN(IT))
    ENDDO

    NUMLEV=NUML             ! USE INTERPOLATED DATA
    SPROF(:,1)=SPROF(:,1)+9.8   !CORRECT FOR OFFSET OF IRGA DATA FROM FLASK DATA

    SPROFALL(:,1,:)=SPROFALL(:,1,:)+9.8
    IS2=50      !NUMBER OF SEGMENTS FOR DC/DA INTEGRALS
    IS3=100     !NUMBER OF POINTS WHERE SOURCE AND FLUX ARE EVALUATED
    IS4=50      !NUMBER OF POINTS WHERE FLUX IS CALCULATED INITIALLY

    FIS2=FLOAT(IS2)
    FIS3=FLOAT(IS3)
    FIS4=FLOAT(IS4)

    HCS=23.     ! CANOPY HEIGHT
    COUNT=0.
    DO IK=1,NUMLEV      ! NUMBER OF HEIGHTS BELOW HC
        IF(XPROF(IK).LE.HCS) THEN
            COUNT=COUNT+1
        ENDIF
    ENDDO

    RL=0.001                ! INITIALISE RL (LEV-MARQ LAMBDA)

    SLA1=5.; SLA2=4.    ! LEAF AREA PROFILE SHAPE
    GHR=0.003   ! GH MULTIPLIER
```

```
WIND=3.     ! WIND EXTINCTION COEFFICIENT
CICA=0.6    ! INITIAL CI/CA
D13RESP=18. ! RESP. DISCR.
D18RESP=15. ! RESP. DISCR.
SIGO=SIG    !RETAIN ORIGINAL SIGMAS FOR CONCENTRATIONS

VM=1.       ! VMAX PARAMETER
SVMi=10.        ! SIGMA OF VMAX PARAMETER
ALAM=250.   ! LAMBDA FOR GS
SALAMi=2500.    ! SIGMA FOR LAMBDA
TLP1=20.    ! TL EXP COEFF
STLP1I=200. ! SIGMA FOR TL EXP COEFF
TLP2=0.30   ! TL U*/H AT TOP OF CANOPY
STLP2I=3.   ! SIGMA FOR TL U*/H
KDF=0.78        ! DIFFUSE RADIATION EXTINCTION COEFFICIENT
SKDFI=8.        ! SIGMA FOR KDF
KBF=0.50        ! DIRECT RADIATION EXTINCTION COEFFICIENT
SKBFI=5.        ! SIGMA FOR KBF

AK(1:3)=(/0.0458,-0.0268,0.00566/)      ! CO2, HEAT AND H2O FLUX AT GROUND
DO I=1,NUMISO       ! GROUND FLUXES FOR EACH OF THE TEN PROFILE SETS
    GF(1,I)=0.05; GF(2,I)=-0.05; GF(3,I)=0.01
ENDDO
AK(4)=VM        ! VMAX PARAMETER
AK(5)=ALAM  ! LAMBDA FOR GS
AK(6)=TLP1  ! TL EXP COEFF
AK(7)=TLP2  ! TL U*/H AT TOP OF CANOPY
AK(8)=KDF   ! DIRECT RADIATION EXTINCTION COEFFICIENT
AK(9)=KBF   ! DIFFUSE RADIATION EXTINCTION COEFFICIENT
TL1=TLP1
TL2=TLP2
STL1=STLP1I
STL2=STLP2I

PARAM(1:2)=(/SLA1,SLA2/)    ! SHAPE OF LEAF AREA PROFILE
PARAM(3)=GHR        ! GH MULTIPLIER
PARAM(4)=4.3        ! TOTAL LEAF AREA INDEX
PARAM(5)=WIND       ! WIND EXTINCTION COEFFICIENT

TENLOOPS: DO I=1,NUMISO
DO NPR=1,3
SREF(NPR,I)=SPROFALL(NUMLEV,NPR,I)
  DO J=1,NUMLEV ! CHECK FOR BAD STD DEVIATION + BAD DC/DT
    IF(SPROF(J,NPR+3).LT.0.001) SPROF(J,NPR+3)=1.
    IF(ABS(SPROF(J,NPR+3)).GT.900.) SPROF(J,NPR+3)=1.
    IF(ABS(SPROF(J,NPR+6)).GT.900.) SPROF(J,NPR+6)=0.
  ENDDO
ENDDO
IF(ISOTOPES.GE.1) THEN
  SREF(4,I)=(APROFALL(NUMLEV,1,I)/1000.+1)*RSTD*SREF(1,I)
  RAIR(I)=SREF(4,I)/SREF(1,I)
  SREF(5,I)=(APROFALL(NUMLEV,2,I)/1000.+1)*R18STD*SREF(1,I)
ENDIF
ZREF=XPROF(NUMLEV)
SMOW=0.0020052; DEL_S=-18.; ESTAR=9.2
D18VAP=-26.
SREF(6,I)=(1.+D18VAP/1000.)*SMOW*SREF(3,I)

DO N=1,NUMLEV
  DO L=1,6
    CKOALL(L,N,I)=SREF(L,I)
  ENDDO
ENDDO
CKALL=CKOALL; CKKALL=CKOALL; AKO=AK; AKE=AK; IFLAG0=1; GFO=GF; GFE=GF
TL1O=TL1; TL2O=TL2; TL1E=TL1; TL2E=TL2
ENDDO TENLOOPS
```

```
    NC=0.;X2=0.
    do i=0,is3-1
        ZAH=FLOAT(I)/FIS3*COUNT
        J=NINT(ZAH)
        IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
        IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
        IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
        IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
        ZAH=NC/FIS3*COUNT
        ZAL=X1+(X2-X1)*ZAH
        ZAH=ZAL/HCS
        ZSOU(I)=ZAL
    ENDDO
    ZSOU(IS3)=HCS

1122    AK=AKO; GF=GFO; TL1=TL1O; TL2=TL2O
    IF(ICOSTFN.EQ.1) THEN
        IF(IPM.EQ.2) THEN
            AK(IN)=AK(IN)+0.1*AK(IN)
        ELSEIF(IPM.EQ.3) THEN
            AK(IN)=AK(IN)-0.1*AK(IN)
        ENDIF
    ENDIF

    ITER=200
3035    DO 3025 IA=1,ITER           ! ITERATE FOR AK PARAMETERS

    LIM=0
    CKOALL=CKALL
    DCKOALL=DCKALL
    IF(AK(4).LT.0.1) THEN; SVM=SVM/2.; LIM=1; ELSEIF(IFLAG0.EQ.1) THEN; SVM=SVMI; ENDIF
    IF(AK(5).LT.50.) THEN; SALAM=SALAM/2.; LIM=1
    ELSEIF(AK(5).GT.1500.) THEN; SALAM=SALAM/2.; LIM=1;
    ELSEIF(IFLAG0.EQ.1) THEN; SALAM=SALAMI; ENDIF
    IF(NUMK.EQ.54) THEN
      DO IT=1,10
        IF(TL1(IT).LT.8.)THEN; STL1(IT)=STL1(IT)/2.; LIM=1
        ELSEIF(TL1(IT).GT.200.) THEN; STL1(IT)=STL1(IT)/2.; LIM=1
        ELSEIF(IFLAG0.EQ.1) THEN; STL1(IT)=STLP1I; ENDIF
        IF(TL2(IT).LT.0.1)THEN; STL2(IT)=STL2(IT)/2.; LIM=1
        ELSEIF(TL2(IT).GT.0.6)THEN; STL2(IT)=STL2(IT)/2.; LIM=1
        ELSEIF(IFLAG0.EQ.1)THEN; STL2(IT)=STLP2I; ENDIF
      ENDDO
    ELSEIF(NUMK.NE.54) THEN
      IF(AK(6).LT.8.) THEN; STLP1=STLP1/2.; LIM=1
      ELSEIF(AK(6).GT.200.) THEN; STLP1=STLP1/2.; LIM=1;
      ELSEIF(IFLAG0.EQ.1)THEN; STLP1=STLP1I; ENDIF
      IF(AK(7).LT.0.1) THEN; STLP2=STLP2/2.; LIM=1
      ELSEIF(AK(7).GT.0.6) THEN; STLP2=STLP2/2.; LIM=1;
      ELSEIF(IFLAG0.EQ.1)THEN; STLP2=STLP2I; ENDIF
      IF(AK(7).LE.0.1.AND.AK(6).LE.0.2) THEN
        STLP2=STLP2/2.; STLP1=STLP1/2.; LIM=1
      ENDIF
    ENDIF
    IF(AK(8).LT.0.01) THEN; SKDF=SKDF/2.; LIM=1;
    ELSEIF(AK(8).GT.1.) THEN; SKDF=SKDF/2.; LIM=1; ELSEIF(IFLAG0.EQ.1)THEN; SKDF=SKDFI; ENDIF
    IF(AK(9).LT.0.01) THEN; SKBF=SKBF/2.; LIM=1
    ELSEIF(AK(9).GT.1.) THEN; SKBF=SKBF/2.; LIM=1; ELSEIF(IFLAG0.EQ.1)THEN; SKBF=SKBFI; ENDIF
    IF(LIM.EQ.1) THEN; AK=AKE; DCKALL=DCKEALL; CKALL=CKEALL; GF=GFE
        TL1=TL1E; TL2=TL2E; IFLAG0=0; GOTO 808; ENDIF

2121 CONTINUE

    TENLOOPS2: DO IT=1,NUMISO
!         check if stability correction required
```

```
        if(ist_cor.eq.1) then
          IF(TTHR(ICN(IT)).LT.5) ZMON=10.
          IF(TTHR(ICN(IT)).GE.5.AND.TTHR(ICN(IT)).LT.7) ZMON=50.
          IF(TTHR(ICN(IT)).GE.7.AND.TTHR(ICN(IT)).LT.8) ZMON=100.
          IF(TTHR(ICN(IT)).GE.8.AND.TTHR(ICN(IT)).LT.10) ZMON=500.
          IF(TTHR(ICN(IT)).GE.10.AND.TTHR(ICN(IT)).LT.17) ZMON=1000.
          IF(TTHR(ICN(IT)).GE.17.AND.TTHR(ICN(IT)).LT.18) ZMON=500.
          IF(TTHR(ICN(IT)).GE.18.AND.TTHR(ICN(IT)).LT.19) ZMON=100.
          IF(TTHR(ICN(IT)).GE.19.AND.TTHR(ICN(IT)).LT.22) ZMON=50.
          IF(TTHR(ICN(IT)).GE.22) ZMON=10.
         else
          zMon=-1000.                          !set to large -ve value
!                                                        -> neutral
        end if

    call air(SREF(2,IT),APRESS(ICN(IT)),rho,volm,capp,rlam,qsat,epsi)
    UHREF=UH(ICN(IT))/1.3
    USTARR=US(ICN(IT))
    SIGWR=SWR(ICN(IT))
    AK(1:3)=GF(:,IT)
    IFLAG0=1
      IF(IDIS.EQ.1) THEN
        call DISPM2(USTARR,SIGWR,AK(6),AK(7),timdat,NUMLev,NUMLEV,NUMLEV,0,IS3, &
        ZSOU,XPROF,zMon,DISC,DISCN,DISCF,IWRITE)
      ENDIF
    DO KK=1,3
! CALCULATE SOURCE AND FLUX PROFILES
    CALL CALCS(PARAM,FD(ICN(IT)),PAR(ICN(IT)),DEF,SWDREF(IT),LWDREF(IT), &
    DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT), &
    XPROF,SREF(1,IT),ESATL,thetl,RLAI,RC,RH,RNN,ak, &
    hcs,cica,CSOURCE,esource,is3,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
    UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
      SK(3,:)=ESOURCE   ! THESE ARE THE SOURCE PROFILES FROM Z=0 TO Z=H
      SK(2,:)=HSOURCE
      SK(1,:)=CSOURCE
      SK(4,:)=C13SOURCE
      SK(5,:)=O18SOURCE
      SK(6,:)=H18SOURCE

      CALL INTVAL(28., 28., TEMP, CO2, H2O,NLEVL,ZZZ,PROF,28.,20,ICN(IT))
      ESAT=6.1375*EXP(17.502*TEMP/(240.97+TEMP))
      SL=17.502*240.97/(240.97+TEMP)**2*esat
      sl=0.622*pmb/(pmb-0.378*esat)**2*sl
      ep=rlam/cAPp*sl
      eqg=ep*rnn(0)*.9/(ep+1.)
!     AK(3)=EQG*VOLM/RLAM/18.015/1.E-6
      CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM,FD(ICN(IT)), &
      PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
      DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF,SREF(1,IT),ak,hcs, &
      CFLUX,eflux,is4,0.,ZREF,HFLUX, C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
!WRITE TO DUMMY VARIABLES, FLUX PROFILES FROM Z=0 TO Z=H
      EFLUXD=EFLUX; HFLUXD=HFLUX; CFLUXD=CFLUX; C13FLUXD=C13FLUX; O18FLUXD=O18FLUX
      H18FLUXD=H18FLUX

        IF(IST_COR.EQ.1) THEN
          LZMON(IT)=USREF**3*(TC+273.15)*RHO*CAPP/0.4/10./HFLUX(IS4)    !MONIN-OBUKHOV LENGTH
          ZMON=LZMON(IT)
        ENDIF

    CKNR=0.;CKNR2=0.;CKNR3=0.;CKNR4=0.; CKNR5=0.;CKNR6=0.   !CALCULATE NEAR FIELD CONC AT ZREF
    NC=0.;X2=0.
    DO I=0,IS3-1
        ZAH=FLOAT(I)/FIS3*COUNT
        J=NINT(ZAH)
        IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
        IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
```

```
    IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
    IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
    ZAH=NC/FIS3*COUNT
    ZI=X1+(X2-X1)*ZAH
    ZID=ZI+(X2-X1)*COUNT/FIS3
    IF(NUMK.NE.54) THEN
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW0,TL0)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,zMon,SWD,TLD)
    ELSEIF(NUMK.EQ.54) THEN
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZI,zMon,SW0,TL0)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZID,zMon,SWD,TLD)
    ENDIF
    RKN1=RKN((ZREF-ZI)/SW0/TL0)+RKN((ZREF+ZI)/SW0/TL0)
    RKN2=RKN((ZREF-ZID)/SWD/TLD)+RKN((ZREF+ZID)/SWD/TLD)
        CKNR=CKNR+((SK(1,I))/SW0*RKN1+ (SK(1,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKNR2=CKNR2+((SK(2,I))/SW0*RKN1+ (SK(2,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKNR3=CKNR3+((SK(3,I))/SW0*RKN1+ (SK(3,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKNR4=CKNR4+((SK(4,I))/SW0*RKN1+ (SK(4,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKNR5=CKNR5+((SK(5,I))/SW0*RKN1+ (SK(5,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKNR6=CKNR6+(SK(6,I)/SW0*RKN1+ SK(6,I+1)/SWD*RKN2)/2./FIS3*HCS
ENDDO
DO N=1,NUMLEV       ! CALCULATE NEAR FIELD CONC AT EACH Zi (MEASUREMENT HEIGHTS)
CKF1=0.;CKF2=0.;CKF3=0.;CKF4=0.; CKF5=0.; CKF6=0.
CKN=0.;CKN2=0.;CKN3=0.;CKN4=0.; CKN5=0.; CKN6=0.
DCKF1E=0.;DCKF2E=0.;DCKF3E=0.;DCKF4E=0.; DCKF5E=0.
DCKNE=0.;DCKN2E=0.;DCKN3E=0.;DCKN4E=0.; DCKN5E=0.
DCKF1B=0.;DCKF2B=0.;DCKF3B=0.;DCKF4B=0.; DCKF5B=0.
DCKNB=0.;DCKN2B=0.;DCKN3B=0.;DCKN4B=0.; DCKF5B=0.
ZL=XPROF(N)
NC=0.;X2=0.
  DO I=0,IS3-1
    ZAH=FLOAT(I)/FIS3
    ZI=ZAH*HCS
    ZID=ZI+HCS/FIS3
    ZAH=FLOAT(I)/FIS3*COUNT
    J=NINT(ZAH)
    IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
    IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
    IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
    IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
    ZAH=NC/FIS3*COUNT
    ZI=X1+(X2-X1)*ZAH
    ZID=ZI+(X2-X1)*COUNT/FIS3
    IF(NUMK.NE.54) THEN
      EPS=0.001*AK(6)
      EPS2=0.001*AK(7)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW0,TL0)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,ZMON,SWD,TLD)
      CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZI,zMon,SW0,TL0E)
      CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZID,ZMON,SWD,TLDE)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZI,zMon,SW0,TL0B)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZID,ZMON,SWD,TLDB)
    ELSEIF(NUMK.EQ.54) THEN
      EPS=0.001*TL1(IT)
      EPS2=0.001*TL2(IT)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZI,zMon,SW0,TL0)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZID,ZMON,SWD,TLD)
      CALL TURB2(USTARR,SIGWR,TL1(IT)+EPS,TL2(IT),ZI,zMon,SW0,TL0E)
      CALL TURB2(USTARR,SIGWR,TL1(IT)+EPS,TL2(IT),ZID,ZMON,SWD,TLDE)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT)+EPS2,ZI,zMon,SW0,TL0B)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT)+EPS2,ZID,ZMON,SWD,TLDB)
    ENDIF
    IF((ZL-ZI).NE.0.0.AND.(ZL-ZID).NE.0.0) THEN
      RKN1=RKN((ZL-ZI)/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
      RKN1E=RKN((ZL-ZI)/SW0/TL0E)+RKN((ZL+ZI)/SW0/TL0E)
```

```
      RKN2E=RKN((ZL-ZID)/SWD/TLDE)+RKN((ZL+ZID)/SWD/TLDE)
      RKN1B=RKN((ZL-ZI)/SW0/TL0B)+RKN((ZL+ZI)/SW0/TL0B)
      RKN2B=RKN((ZL-ZID)/SWD/TLDB)+RKN((ZL+ZID)/SWD/TLDB)
    ELSEIF((ZL-ZI).EQ.0.0) THEN
      RKN1=RKN(HCS/FIS3/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
      RKN1E=RKN(HCS/FIS3/SW0/TL0E)+RKN((ZL+ZI)/SW0/TL0E)
      RKN2E=RKN((ZL-ZID)/SWD/TLDE)+RKN((ZL+ZID)/SWD/TLDE)
      RKN1B=RKN(HCS/FIS3/SW0/TL0B)+RKN((ZL+ZI)/SW0/TL0B)
      RKN2B=RKN((ZL-ZID)/SWD/TLDB)+RKN((ZL+ZID)/SWD/TLDB)
    ELSEIF((ZL-ZID).EQ.0.0) THEN
      RKN1=RKN((ZL-ZI)/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN(HCS/FIS3/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
      RKN1E=RKN((ZL-ZI)/SW0/TL0E)+RKN((ZL+ZI)/SW0/TL0E)
      RKN2E=RKN(HCS/FIS3/SWD/TLDE)+RKN((ZL+ZID)/SWD/TLDE)
      RKN1B=RKN((ZL-ZI)/SW0/TL0B)+RKN((ZL+ZI)/SW0/TL0B)
      RKN2B=RKN(HCS/FIS3/SWD/TLDB)+RKN((ZL+ZID)/SWD/TLDB)
    ENDIF
        DRKN1=(RKN1E-RKN1)/EPS
        DRKN2=(RKN2E-RKN2)/EPS
        DRKN1B=(RKN1B-RKN1)/EPS2
        DRKN2B=(RKN2B-RKN2)/EPS2
        CKN=CKN+((SK(1,I))/SW0*RKN1+ (SK(1,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN2=CKN2+((SK(2,I))/SW0*RKN1+ (SK(2,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN3=CKN3+((SK(3,I))/SW0*RKN1+ (SK(3,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN4=CKN4+((SK(4,I))/SW0*RKN1+ (SK(4,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN5=CKN5+((SK(5,I))/SW0*RKN1+ (SK(5,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN6=CKN6+(SK(6,I)/SW0*RKN1+ SK(6,I+1)/SWD*RKN2)/2./FIS3*HCS
        DCKNE=DCKNE+((SK(1,I))/SW0*DRKN1+ (SK(1,I+1))/SWD*DRKN2)/2./FIS3*HCS
        DCKN2E=DCKN2E+((SK(2,I))/SW0*DRKN1+ (SK(2,I+1))/SWD*DRKN2)/2./FIS3*HCS
        DCKN3E=DCKN3E+((SK(3,I))/SW0*DRKN1+ (SK(3,I+1))/SWD*DRKN2)/2./FIS3*HCS
        DCKN4E=DCKN4E+((SK(4,I))/SW0*DRKN1+ (SK(4,I+1))/SWD*DRKN2)/2./FIS3*HCS
        DCKN5E=DCKN5E+((SK(5,I))/SW0*DRKN1+ (SK(5,I+1))/SWD*DRKN2)/2./FIS3*HCS
        DCKNB=DCKNB+((SK(1,I))/SW0*DRKN1B+ (SK(1,I+1))/SWD*DRKN2B)/2./FIS3*HCS
        DCKN2B=DCKN2B+((SK(2,I))/SW0*DRKN1B+ (SK(2,I+1))/SWD*DRKN2B)/2./FIS3*HCS
        DCKN3B=DCKN3B+((SK(3,I))/SW0*DRKN1B+ (SK(3,I+1))/SWD*DRKN2B)/2./FIS3*HCS
        DCKN4B=DCKN4B+((SK(4,I))/SW0*DRKN1B+ (SK(4,I+1))/SWD*DRKN2B)/2./FIS3*HCS
        DCKN5B=DCKN5B+((SK(5,I))/SW0*DRKN1B+ (SK(5,I+1))/SWD*DRKN2B)/2./FIS3*HCS
  ENDDO
  DO I=0,IS3
    ZN=ZL+FLOAT(I)*(ZREF-ZL)/FIS3
    IF(NUMK.NE.54) THEN
      EPS=0.001*AK(6)
      EPS2=0.001*AK(7)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZN,zMon,SWZ,TLZ)
      CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZN,zMon,SWZ,TLZE)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZN,zMon,SWZ,TLZB)
    ELSEIF(NUMK.EQ.54) THEN
      EPS=0.001*TL1(IT)
      EPS2=0.001*TL2(IT)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZN,zMon,SWZ,TLZ)
      CALL TURB2(USTARR,SIGWR,TL1(IT)+EPS,TL2(IT),ZN,zMon,SWZ,TLZE)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT)+EPS2,ZN,zMon,SWZ,TLZB)
    ENDIF
    ! GET FLUX PROFILES FROM Z=Zi TO Z=ZREF FOR CONCENTRATION INTEGRAL, WHERE Zi IS HEIGHT
    ! OF CONCENTRATION MEASUREMENT, FOR EACH Zi
    IF(ZN.EQ.ZPR(IS4)) THEN
      FK(3,I)=EFLUX(IS4); FK(2,I)=HFLUX(IS4); FK(1,I)=CFLUX(IS4); FK(4,I)=C13FLUX(IS4)
      FK(5,I)=O18FLUX(IS4); FK(6,I)=H18FLUX(IS4)
    ELSEIF(ZN.GT.ZPR(IS4).OR.ZN.LT.ZPR(0)) THEN
      STOP 'ZN NOT IN RANGE FLUX CALC'
    ENDIF
    DO J=0,IS4-1
        IF(ZN.EQ.ZPR(J)) THEN   !IF Zi COINCIDES WITH A HEIGHT AT WHICH WE ALREADY HAVE
                                ! THE FLUX CALCULATION, TAKE THAT FLUX
          FK(3,I)=EFLUXD(J); FK(2,I)=HFLUXD(J); FK(1,I)=CFLUXD(J); FK(4,I)=C13FLUXD(J)
```

```
          FK(5,I)=O18FLUXD(J); FK(6,I)=H18FLUXD(J)
      ELSEIF(ZN.LT.ZPR(J+1).AND.ZN.GT.ZPR(J)) THEN    ! OTHERWISE CALCULATE ADDITIONAL
                              ! FLUX FROM HEIGHT WE HAVE TO HEIGHT WE WANT
          EADD=EFLUXD(J); HADD=HFLUXD(J); CADD=CFLUXD(J); C13ADD=C13FLUXD(J)
          O18ADD=O18FLUXD(J); H18ADD=H18FLUXD(J)
          ZR=ZPR(J)+(ZN-ZPR(J))*FIS4
          CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM,FD(ICN(IT)), &
          PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), DTIME(ICN(IT)),APRESS(ICN(IT)), &
          CKALL(:,:,IT),XPROF,SREF(1,IT),ak,hcs,CFLUX,eflux,is4,ZPR(J),ZR,HFLUX, &
              C13FLUX,O18FLUX,H18FLUX,UHREF,ZOUT,1)
          FK(3,I)=EFLUX(1)+EADD
          FK(2,I)=HFLUX(1)+HADD
          FK(1,I)=CFLUX(1)+CADD
          FK(4,I)=C13FLUX(1)+C13ADD
          FK(5,I)=O18FLUX(1)+O18ADD
          FK(6,I)=H18FLUX(1)+H18ADD
      ENDIF
  ENDDO
  FKS(3,I)=FK(3,I)/SWZ**2/TLZ     !DIVIDE BY SIGW^2 AND TL FOR INTEGRATION
  FKS(2,I)=FK(2,I)/SWZ**2/TLZ
  FKS(1,I)=FK(1,I)/SWZ**2/TLZ
  FKS(4,I)=FK(4,I)/SWZ**2/TLZ
  FKS(5,I)=FK(5,I)/SWZ**2/TLZ
  FKS(6,I)=FK(6,I)/SWZ**2/TLZ
  DTL=(TLZE-TLZ)/EPS
  DTLB=(TLZB-TLZ)/EPS2
  DFKS(3,I)=-FK(3,I)/SWZ**2/TLZ**2*DTL
  DFKS(2,I)=-FK(2,I)/SWZ**2/TLZ**2*DTL
  DFKS(1,I)=-FK(1,I)/SWZ**2/TLZ**2*DTL
  DFKS(4,I)=-FK(4,I)/SWZ**2/TLZ**2*DTL
  DFKS(5,I)=-FK(5,I)/SWZ**2/TLZ**2*DTL
  DFKS(6,I)=-FK(1,I)/SWZ**2/TLZ**2*DTLB
  DFKS(7,I)=-FK(2,I)/SWZ**2/TLZ**2*DTLB
  DFKS(8,I)=-FK(3,I)/SWZ**2/TLZ**2*DTLB
  DFKS(9,I)=-FK(4,I)/SWZ**2/TLZ**2*DTLB
  DFKS(10,I)=-FK(5,I)/SWZ**2/TLZ**2*DTLB
ENDDO
  DO I=0,IS3-1
    CKF1=CKF1+(FKS(1,I)+FKS(1,I+1))/2./FIS3*(ZREF-ZL)     ! INTEGRATE FROM Zi TO ZREF
    CKF2=CKF2+(FKS(2,I)+FKS(2,I+1))/2./FIS3*(ZREF-ZL)
    CKF3=CKF3+(FKS(3,I)+FKS(3,I+1))/2./FIS3*(ZREF-ZL)
    CKF4=CKF4+(FKS(4,I)+FKS(4,I+1))/2./FIS3*(ZREF-ZL)
    CKF5=CKF5+(FKS(5,I)+FKS(5,I+1))/2./FIS3*(ZREF-ZL)
    CKF6=CKF6+(FKS(6,I)+FKS(6,I+1))/2./FIS3*(ZREF-ZL)
    DCKF1E=DCKF1E+(DFKS(1,I)+DFKS(1,I+1))/2./FIS3*(ZREF-ZL)   ! INTEGRATE FROM Zi TO ZREF
    DCKF2E=DCKF2E+(DFKS(2,I)+DFKS(2,I+1))/2./FIS3*(ZREF-ZL)
    DCKF3E=DCKF3E+(DFKS(3,I)+DFKS(3,I+1))/2./FIS3*(ZREF-ZL)
    DCKF4E=DCKF4E+(DFKS(4,I)+DFKS(4,I+1))/2./FIS3*(ZREF-ZL)
    DCKF5E=DCKF5E+(DFKS(5,I)+DFKS(5,I+1))/2./FIS3*(ZREF-ZL)
    DCKF1B=DCKF1B+(DFKS(6,I)+DFKS(6,I+1))/2./FIS3*(ZREF-ZL)   ! INTEGRATE FROM Zi TO ZREF
    DCKF2B=DCKF2B+(DFKS(7,I)+DFKS(7,I+1))/2./FIS3*(ZREF-ZL)
    DCKF3B=DCKF3B+(DFKS(8,I)+DFKS(8,I+1))/2./FIS3*(ZREF-ZL)
    DCKF4B=DCKF4B+(DFKS(9,I)+DFKS(9,I+1))/2./FIS3*(ZREF-ZL)
    DCKF5B=DCKF5B+(DFKS(10,I)+DFKS(10,I+1))/2./FIS3*(ZREF-ZL)
  ENDDO
  CKALL(1,N,IT)=(CKN+CKF1-CKNR)+SREF(1,IT)    ! CONCENTRATION = NEAR-FIELD + FAR-FIELD -
  CKALL(2,N,IT)=(CKN2+CKF2-CKNR2)+SREF(2,IT)  ! NEAR-FIELD (ZREF) + CONC (ZREF)
  CKALL(3,N,IT)=(CKN3+CKF3-CKNR3)+SREF(3,IT)
  CKALL(4,N,IT)=(CKN4+CKF4-CKNR4)+SREF(4,IT)
  CKALL(5,N,IT)=(CKN5+CKF5-CKNR5)+SREF(5,IT)
  CKALL(6,N,IT)=(CKN6+CKF6-CKNR6)+SREF(6,IT)
  DO IK=1,6
      CKALL(IK,N,IT)=MIN(CKALL(IK,N,IT),1000.)
      CKALL(IK,N,IT)=MAX(CKALL(IK,N,IT),-1000.)
      IF(ABS(CKALL(IK,N,IT)).EQ.1000.) THEN
          CKALL(IK,N,IT)=SREF(IK,IT)
```

```
              INAN=1
              GOTO 2122
            ENDIF
        ENDDO
        DCKD(1,N,6,IT)=DCKNE+DCKF1E
        DCKD(2,N,6,IT)=DCKN2E+DCKF2E
        DCKD(3,N,6,IT)=DCKN3E+DCKF3E
        DCKD(4,N,6,IT)=DCKN4E+DCKF4E
        DCKD(5,N,6,IT)=DCKN5E+DCKF5E
        DCKD(1,N,7,IT)=DCKNB+DCKF1B
        DCKD(2,N,7,IT)=DCKN2B+DCKF2B
        DCKD(3,N,7,IT)=DCKN3B+DCKF3B
        DCKD(4,N,7,IT)=DCKN4B+DCKF4B
        DCKD(5,N,7,IT)=DCKN5B+DCKF5B
        FKTOP(1,IT)=CFLUXD(IS4)
        FKTOP(2,IT)=HFLUXD(IS4)
        FKTOP(3,IT)=EFLUXD(IS4)
        IF(IDIS.EQ.1) THEN
        DO NPR=1,5
        ASUM=0.
          DO I=1,IS3
            ASUM=ASUM+DISC(N,I)*SK(NPR,I)*(ZSOU(I)-ZSOU(I-1))
          ENDDO
          ASUM=ASUM+DISC(N,0)*FK(NPR,0)*ZSOU(1)
        CK(NPR,N)=ASUM+SREF(NPR,IT)
        ENDDO
        ENDIF
    ENDDO
    ENDDO
    ENDDO TENLOOPS2

2122    CSUM=0.
    TENLOOPS3: DO IT=1,NUMISO
        DO N=1,NUMLEV           ! CALCULATE CHI-SQUARE = SUM (C_MODEL - C_MEAS)^2 / SIGMA^2
         DO NPR=1,3
          CSUM=CSUM+(CKALL(NPR,N,IT)-SPROFALL(N,NPR,IT))**2/SIG(NPR,N)**2
         ENDDO
        ENDDO
        DO N=1,NUMLEV
          ZAN=FLOAT(N)/FLOAT(NUMLEV)
          IF(ISOTOPES.GT.1) THEN
!          C13K=(APROF(N,1,NIS)/1000.+1.)*RSTD*SPROF(N,1)
           C13K=(CKALL(4,N,IT)/CKALL(1,N,IT)/RSTD-1.)*1000.
           CSUM = CSUM + (C13K-APROFALL(N,1,IT))**2/SIG(4,N)**2
          ENDIF
          IF(ISOTOPES.GT.2) THEN
           O18K=(CKALL(5,N,IT)/CKALL(1,N,IT)/R18STD-1.)*1000.
           CSUM = CSUM + (O18K-APROFALL(N,2,IT))**2/SIG(5,N)**2
          ENDIF
        ENDDO
        IF(IEDDY.EQ.1) THEN
            SIGFC=ABS(0.05*FC2(ICN(IT))*VOLM)   ! SIGMA OF TOTAL C FLUX
            SIGFH=ABS(0.1*H(ICN(IT))/CAPP/RHO)  ! SIGMA OF TOTAL H FLUX
            SIGFE=ABS(0.1*E(ICN(IT))*VOLM)  ! SIGMA OF TOTAL E FLUX
           CSUM = CSUM + (FC2(ICN(IT))*VOLM-CFLUX(IS4))**2/SIGFC**2
           CSUM = CSUM + (H(ICN(IT))/CAPP/RHO-HFLUX(IS4))**2/SIGFH**2
           CSUM = CSUM + (E(ICN(IT))*VOLM-EFLUX(IS4))**2/SIGFE**2
        ENDIF
    ENDDO TENLOOPS3
!          CSUM = CSUM + (AK(4)-VM)**2/SVM**2
!          CSUM = CSUM + (AK(5)-ALAM)**2/SALAM**2
!          CSUM = CSUM + (AK(6)-TLP1)**2/STLP1**2
!          CSUM = CSUM + (AK(7)-TLP2)**2/STLP2**2
!          CSUM = CSUM + (AK(8)-KDF)**2/SKDF**2
!          CSUM = CSUM + (AK(9)-KBF)**2/SKBF**2
```

```
    CHISQN=CSUM

    IF(ABS(CHISQN-CHISQ).LT.0.1) GOTO 3045   !EXIT LOOP IF NO IMPROVEMENT IN CHISQ
    CHISQNN=0.
    IF(IA.EQ.1.OR.IAC.EQ.1) THEN    ! 1ST ITERATION
    CHISQ=CHISQN
    IAC=0
    CKEALL=CKALL
    ELSEIF(IA.EQ.ITER-1.AND.CHISQN.LE.CHISQ) THEN   ! 2ND LAST ITERATION, EXIT IF CHISQ
        AKO=DBLE(AK)                ! BETTER THAN PREVIOUS
        GOTO 3045
    ELSEIF(RL.GT.1.E+2) THEN    ! EXIT IF LAMBDA LARGE
      CKALL=CKOALL; AK=SNGL(AKO); GF=GFO; TL1=TL1O; TL2=TL2O
      GOTO 3045
    ELSEIF(INAN.EQ.1) THEN
      WRITE(*,*) 'CONC NAN, AK=',(AK(K),K=4,9)
      AK=SNGL(AKO)
      GF=GFO
      TL1=TL1O; TL2=TL2O
      CKALL=CKOALL
      DCKALL=DCKOALL
      RL=MAX(RL*100.,0.1)
      INAN=0
      GOTO 808
    ELSE IF (CHISQN.GE.CHISQ) THEN  ! INCREASE LAMBDA IF CHISQ WORSE AND USE PREVIOUS PARAMS
      AK=SNGL(AKO)
      GF=GFO
      TL1=TL1O; TL2=TL2O
      CKALL=CKOALL
      DCKALL=DCKOALL
      RL=MAX(RL*100.,0.1)
      GOTO 808
    ELSE IF (CHISQN.LT.CHISQ) THEN  ! DECREASE LAMBDA IF CHISQ BETTER, AND CONTINUE
    RL=RL/10.
    CHISQ=CHISQN
    AKE=AK; DCKEALL=DCKALL; CKEALL=CKALL; GFE=GF; TL1E=TL1; TL2E=TL2
    ELSE
      STOP 'PROBLEM SOMEWHERE, MAYBE PARAMS OUT OF RANGE'
      AK=SNGL(AKO)
      GF=GFO
      TL1=TL1O; TL2=TL2O
      CKALL=CKOALL
      DCKALL=DCKOALL
      RL=MAX(RL*100.,0.1)
      GOTO 808
    ENDIF
    write(*,*) 'Chisq = ',chisq, 'Iterations:',ia, '  RL =',RL
    WRITE(*,2233) 'AK =', (AK(K), K=4,9)
    IF(NUMK.EQ.54) THEN
        WRITE(*,2234) 'TL1 =', (TL1(K), K=1,10)
        WRITE(*,2234) 'TL2 =', (TL2(K), K=1,10)
    ENDIF
2233 FORMAT(A8,6F8.3)
2234 FORMAT(A8,10F7.2)
2235 FORMAT(A8,10F7.3)

1133    IF(ICOSTFN.EQ.1) THEN
        COSTFN(ICOST,IPM,IN)=CHISQ
      IF(IPM.EQ.1.OR.IPM.EQ.2) THEN
        IPM=IPM+1
        GOTO 1122
      ELSEIF(IPM.EQ.3.AND.IN.LT.9) THEN
        IN=IN+1
        IPM=2
        GOTO 1122
      ELSE
```

```
        ICOST=ICOST+1
        GOTO 3045
      ENDIF
    ENDIF

! CALCULATE DERIVATIVE OF SOURCE PROFILES WRT PARAMETERS
2022        ZL=0.

    SVM=SVMI;SALAM=SALAMI;STLP1=STLP1I;STLP2=STLP2I;SKDF=SKDFI;SKBF=SKBFI
    STL1=STLP1I;STL2=STLP2I
    TENLOOPS4: DO IT=1,NUMISO
!         check if stability correction required
        if(ist_cor.eq.1) then
          IF(TTHR(ICN(IT)).LT.5) ZMON=10.
          IF(TTHR(ICN(IT)).GE.5.AND.TTHR(ICN(IT)).LT.7) ZMON=50.
          IF(TTHR(ICN(IT)).GE.7.AND.TTHR(ICN(IT)).LT.8) ZMON=100.
          IF(TTHR(ICN(IT)).GE.8.AND.TTHR(ICN(IT)).LT.10) ZMON=500.
          IF(TTHR(ICN(IT)).GE.10.AND.TTHR(ICN(IT)).LT.17) ZMON=1000.
          IF(TTHR(ICN(IT)).GE.17.AND.TTHR(ICN(IT)).LT.18) ZMON=500.
          IF(TTHR(ICN(IT)).GE.18.AND.TTHR(ICN(IT)).LT.19) ZMON=100.
          IF(TTHR(ICN(IT)).GE.19.AND.TTHR(ICN(IT)).LT.22) ZMON=50.
          IF(TTHR(ICN(IT)).GE.22) ZMON=10.
          ZMON=LZMON(IT)
         else
          zMon=-1000.                           !set to large -ve value
!                                                       -> neutral
        end if

    call air(SREF(2,IT),APRESS(ICN(IT)),rho,volm,capp,rlam,qsat,epsi)
    UHREF=UH(ICN(IT))/1.3
    USTARR=US(ICN(IT))
    SIGWR=SWR(ICN(IT))
    AK(1:3)=GF(:,IT)
    OPTIMISE: IF(IOPT.EQ.1) THEN
      DO L=4,7
        K=L
        IF(L.EQ.6) K=8
        IF(L.EQ.7) K=9
        EPS=0.001*ABS(AK(K))        !DERIV SOURCE WRT AK(K)
        AKE=AK
        AKE(k)=AKE(k)+EPS
      CALL CALCS(PARAM,FD(ICN(IT)),PAR(ICN(IT)),DEF,SWDREF(IT),LWDREF(IT), &
        DTIME(ICN(IT)),APRESS(ICN(IT)), &
        CKALL(:,:,IT),XPROF,SREF(1,IT),ESATL,thetle,RLAI,RC,RH,RNN,akE, &
        hcs,cica,CSOURCE,esource,is2,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
        UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
        ESOURCE2=ESOURCE
        CSOURCE2=CSOURCE
        HSOURCE2=HSOURCE
        C13SOURCE2=C13SOURCE
        O18SOURCE2=O18SOURCE
        AKE=AK
        AKE(k)=AKE(k)-EPS
      CALL CALCS(PARAM,FD(ICN(IT)),PAR(ICN(IT)),DEF,SWDREF(IT),LWDREF(IT), &
      DTIME(ICN(IT)),APRESS(ICN(IT)), &
      CKALL(:,:,IT),XPROF,SREF(1,IT),ESATL,thetle,RLAI,RC,RH,RNN,akE, &
      hcs,cica,CSOURCE,esource,is2,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
      UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
        DO NK=0,is2
          EDIFF=(ESOURCE2(NK)-esource(NK))/2./EPS
          CDIFF=(CSOURCE2(NK)-CSOURCE(NK))/2./EPS
          HDIFF=(HSOURCE2(NK)-HSOURCE(NK))/2./EPS
          C13DIFF=(C13SOURCE2(NK)-C13SOURCE(NK))/2./EPS
          O18DIFF=(O18SOURCE2(NK)-O18SOURCE(NK))/2./EPS
          DSK(3,NK,K)=EDIFF
          DSK(1,NK,K)=CDIFF
```

```
      DSK(2,NK,K)=HDIFF
      DSK(4,NK,K)=C13DIFF
      DSK(5,NK,K)=O18DIFF
    ENDDO
  ENDDO

DO NPR=1,5      ! INTEGRATE SOURCE DERIVATIVES FROM Z=0 TO Z=H FOR DCN(ZREF)/DAK
RSUM1=0.; RSUM2=0.; RSUM3=0.
RSUM4(npr)=0.; RSUM5(npr)=0.; RSUM6(npr)=0.
RSUM7(npr)=0.; RSUM8(NPR)=0.; RSUM9(NPR)=0.
NC=0.;X2=0.
IF(IDIS.EQ.0) THEN
 DO NK=0,IS2-1
    ZI=FLOAT(NK)/FIS2*HCS
    ZID=ZI+HCS/FIS2
    ZAH=FLOAT(NK)/FIS2*COUNT
    J=NINT(ZAH)
    IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
    IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
    IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
    IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
    ZAH=NC/FIS2*COUNT
    ZI=X1+(X2-X1)*ZAH
    ZID=ZI+(X2-X1)*COUNT/FIS2
    IF(NUMK.NE.54) THEN
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW,TL)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,zMon,SWD,TLD)
    ELSEIF(NUMK.EQ.54) THEN
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZI,zMon,SW,TL)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZID,zMon,SWD,TLD)
    ENDIF
        RKNR=RKN((ZREF-ZI)/SWD/TLD)+RKN((ZREF+ZI)/SWD/TLD)
        RKNRD=RKN((ZREF-ZID)/SWD/TLD)+RKN((ZREF+ZID)/SWD/TLD)
      RSUM4(npr)=RSUM4(npr)+(DSK(NPR,NK,4)/SW*RKNR+ &
        DSK(NPR,NK+1,4)/SWD*RKNRD)/2./FIS2*HCS
      RSUM5(npr)=RSUM5(npr)+(DSK(NPR,NK,5)/SW*RKNR+ &
        DSK(NPR,NK+1,5)/SWD*RKNRD)/2./FIS2*HCS
      RSUM8(npr)=RSUM8(npr)+(DSK(NPR,NK,8)/SW*RKNR+ &
        DSK(NPR,NK+1,8)/SWD*RKNRD)/2./FIS2*HCS
      RSUM9(npr)=RSUM9(npr)+(DSK(NPR,NK,9)/SW*RKNR+ &
        DSK(NPR,NK+1,9)/SWD*RKNRD)/2./FIS2*HCS
 ENDDO
ENDIF
IF(NPR.EQ.1) THEN       ! CALCULATE FLUX DERIVATIVES WRT PARAMETERS,
  ZL=0.                 ! FIRST FROM Z=0 TO Z=ZREF
  DO L=4,7
    K=L
    IF(L.EQ.6) K=8
    IF(L.EQ.7) K=9
    EPS=0.001*ABS(AK(K))
    AKE=AK
    AKE(k)=AKE(k)+EPS   ! FIND FLUX WITH AK(K)+EPSILON, K=4,10
    CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM,FD(ICN(IT)), &
    PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
    DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF,SREF(1,IT),ake,hcs, &
    CFLUX,eflux,is4,ZL,ZREF,HFLUX, &
        C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
    EFLUX1(K,:)=EFLUX
    CFLUX1(K,:)=CFLUX
    HFLUX1(K,:)=HFLUX
    C13FLUX1(K,:)=C13FLUX
    O18FLUX1(K,:)=O18FLUX
    AKE=AK
    AKE(k)=AKE(k)-EPS   ! FIND FLUX WITH AK(K)-EPSILON, K=4,10
    CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM,FD(ICN(IT)), &
    PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
```

```
    DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF,SREF(1,IT),ake,hcs, &
    CFLUX,eflux,is4,ZL,ZREF,HFLUX, &
        C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
    EFLUX2(K,:)=EFLUX
    CFLUX2(K,:)=CFLUX
    HFLUX2(K,:)=HFLUX
    C13FLUX2(K,:)=C13FLUX
    O18FLUX2(K,:)=O18FLUX
  ENDDO
ENDIF
enddo
ENDIF OPTIMISE
DO N=1, NUMLEV
do npr=1,5
ZL=XPROF(N)
SSUM1=0.; SSUM2=0.; SSUM3=0.
SSUM4=0.; SSUM5=0.; SSUM6=0.
SSUM7=0.; SSUM8=0.; SSUM9=0.
NC=0.;X2=0.
OPTIMISE2: IF(IOPT.EQ.1) THEN
IF(IDIS.EQ.0) THEN
 DO NK=0,IS2-1
    ZI=FLOAT(NK)/FIS2*HCS
    ZID=ZI+HCS/FIS2
    ZAH=FLOAT(NK)/FIS2*COUNT
    J=NINT(ZAH)
    IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
    IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
    IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
    IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
    ZAH=NC/FIS2*COUNT
    ZI=X1+(X2-X1)*ZAH
    ZID=ZI+(X2-X1)*COUNT/FIS2
    IF(NUMK.NE.54) THEN
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW,TL)
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,zMon,SWD,TLD)
    ELSEIF(NUMK.EQ.54) THEN
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZI,zMon,SW,TL)
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZID,zMon,SWD,TLD)
    ENDIF
    IF((ZL-ZI).NE.0.0.AND.(ZL-ZID).NE.0.0) THEN !INTEGRATE OVER SOURCES
        RKN0=RKN((ZL-ZI)/SW/TL)+RKN((ZL+ZI)/SW/TL)
        RKND=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ELSEIF((ZL-ZI).EQ.0.0) THEN
      RKN0=RKN(HCS/FIS2/SW/TL)+RKN((ZL+ZI)/SW/TL)
      RKND=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ELSEIF((ZL-ZID).EQ.0.0) THEN
      RKN0=RKN((ZL-ZI)/SW/TL)+RKN((ZL+ZI)/SW/TL)
      RKND=RKN(HCS/FIS2/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ENDIF
      SSUM4=SSUM4+(DSK(NPR,NK,4)/SW*RKN0+ &
        DSK(NPR,NK+1,4)/SWD*RKND)/2./FIS2*HCS
      SSUM5=SSUM5+(DSK(NPR,NK,5)/SW*RKN0+ &
        DSK(NPR,NK+1,5)/SWD*RKND)/2./FIS2*HCS
      SSUM8=SSUM8+(DSK(NPR,NK,8)/SW*RKN0+ &
        DSK(NPR,NK+1,8)/SWD*RKND)/2./FIS2*HCS
      SSUM9=SSUM9+(DSK(NPR,NK,9)/SW*RKN0+ &
        DSK(NPR,NK+1,9)/SWD*RKND)/2./FIS2*HCS
 ENDDO
ENDIF
ENDIF OPTIMISE2
SUM1=0.; SUM2=0; SUM3=0.; SUM4=0.; SUM1A=0.; SUM5=0.; SUM8=0.; SUM9=0.; SUM1B=0.
  OPTIMISE3:IF(IOPT.EQ.1) THEN
  IF(NPR.EQ.1) THEN
  DO NK=0,IS2
    ZN=ZL+FLOAT(NK)*(ZREF-ZL)/FIS2
```

```
        IF(ZN.EQ.ZPR(IS4)) THEN ! TOP HEIGHT
            DO L=4,7
                K=L
                IF(L.EQ.6) K=8
                IF(L.EQ.7) K=9
                EPS=0.001*ABS(AK(K))
                DFK(3,NK,K)=(EFLUX1(K,IS4)-eflux2(K,IS4))/2./EPS
                DFK(2,NK,K)=(HFLUX1(K,IS4)-Hflux2(K,IS4))/2./EPS
                DFK(1,NK,K)=(CFLUX1(K,IS4)-Cflux2(K,IS4))/2./EPS
                DFK(4,NK,K)=(C13FLUX1(K,IS4)-C13flux2(K,IS4))/2./EPS
                DFK(5,NK,K)=(O18FLUX1(K,IS4)-O18flux2(K,IS4))/2./EPS
            ENDDO
        ELSEIF(ZN.GT.ZPR(IS4).OR.ZN.LT.ZPR(0)) THEN     ! ZN OUTSIDE RANGE OF FLUX CALCULATIONS
             STOP 'ZN NOT IN RANGE DERIV CALC'
        ENDIF
        DO J=0,IS4-1    ! FIND FLUX DERIVS FROM Z=Zi TO Z=ZREF
            IF(ZN.EQ.ZPR(J)) THEN   ! IF FLUX ALREADY CALCULATED FOR Zi, EASY
              DO L=4,7
                K=L
                IF(L.EQ.6) K=8
                IF(L.EQ.7) K=9
                EPS=0.001*ABS(AK(K))    ! FLUX DERIV WRT AK(K), K=4,10
                DFK(3,NK,K)=(EFLUX1(K,J)-eflux2(K,J))/2./EPS
                DFK(2,NK,K)=(HFLUX1(K,J)-Hflux2(K,J))/2./EPS
                DFK(1,NK,K)=(CFLUX1(K,J)-Cflux2(K,J))/2./EPS
                DFK(4,NK,K)=(C13FLUX1(K,J)-C13flux2(K,J))/2./EPS
                DFK(5,NK,K)=(O18FLUX1(K,J)-O18flux2(K,J))/2./EPS
              ENDDO
            ELSEIF(ZN.LT.ZPR(J+1).AND.ZN.GT.ZPR(J)) THEN    !OTHERWISE CALCULATE EXTRA FLUX FROM
                ZR=ZPR(J)+(ZN-ZPR(J))*FIS4      ! HEIGHT WE HAVE TO HEIGHT WE WANT
              DO L=4,7
                K=L
                IF(L.EQ.6) K=8
                IF(L.EQ.7) K=9
                EPS=0.001*ABS(AK(K))
                AKE=AK
                AKE(k)=AKE(k)+EPS
                CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM, &
FD(ICN(IT)),PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF, &
SREF(1,IT),ake,hcs,CFLUX,eflux,is4,ZPR(J),ZR,HFLUX, &
                    C13FLUX,O18FLUX,H18FLUX,UHREF,ZOUT,1)
                EFL1=EFLUX(1)+EFLUX1(K,J); CFL1=CFLUX(1)+CFLUX1(K,J); HFL1=HFLUX(1)+HFLUX1(K,J)
                C13FL1=C13FLUX(1)+C13FLUX1(K,J); O18FL1=O18FLUX(1)+O18FLUX1(K,J)
                AKE=AK
                    AKE(k)=AKE(k)-EPS
                CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM, &
FD(ICN(IT)),PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF, &
SREF(1,IT),ake,hcs,CFLUX,eflux,is4,ZPR(J),ZR,HFLUX, &
                    C13FLUX,O18FLUX,H18FLUX,UHREF,ZOUT,1)
                EFL2=EFLUX(1)+EFLUX2(K,J); CFL2=CFLUX(1)+CFLUX2(K,J); HFL2=HFLUX(1)+HFLUX2(K,J)
                C13FL2=C13FLUX(1)+C13FLUX2(K,J); O18FL2=O18FLUX(1)+O18FLUX2(K,J)
                DFK(3,NK,K)=(EFL1-efl2)/2./EPS
                DFK(2,NK,K)=(HFL1-Hfl2)/2./EPS
                DFK(1,NK,K)=(CFL1-Cfl2)/2./EPS
                DFK(4,NK,K)=(C13FL1-C13fl2)/2./EPS
                DFK(5,NK,K)=(O18FL1-O18fl2)/2./EPS
                AKE=AK
              ENDDO
            ENDIF
        ENDDO
      ENDDO
      ENDIF
      ENDIF OPTIMISE3
    IF(IDIS.EQ.0) THEN
```

```
 DO NK=0,IS2-1
      ZN=ZL+FLOAT(NK)*(ZREF-ZL)/FIS2
      ZND=ZN+(ZREF-ZL)/FIS2
      IF(NUMK.NE.54) THEN
        CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZN,ZMON,SWZ,TLZ)
        CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZND,ZMON,SWZD,TLZD)
      ELSEIF(NUMK.EQ.54) THEN
        CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZN,ZMON,SWZ,TLZ)
        CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZND,ZMON,SWZD,TLZD)
      ENDIF
      IF(NK.LT.IS2) THEN    ! INTEGRATE FLUX DERIVS FROM Z=Zi TO Z=ZREF
      SUM1=SUM1+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM1A=SUM1A+(D13RESP/1000./SWZ**2/TLZ+D13RESP/1000./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM1B=SUM1B+(D18RESP/1000./SWZ**2/TLZ+D18RESP/1000./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM2=SUM2+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM3=SUM3+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM4=SUM4+(DFK(NPR,NK,4)/SWZ**2/TLZ+DFK(NPR,NK+1,4)/SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM5=SUM5+(DFK(NPR,NK,5)/SWZ**2/TLZ+DFK(NPR,NK+1,5)/SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM8=SUM8+(DFK(NPR,NK,8)/SWZ**2/TLZ+DFK(NPR,NK+1,8)/SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      SUM9=SUM9+(DFK(NPR,NK,9)/SWZ**2/TLZ+DFK(NPR,NK+1,9)/SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
      ENDIF
 ENDDO
 ENDIF

DCKALL(1,N,1,IT)=SUM1   ! CONC DERIVS = INT(SOURCE DERIV) + INT (FLUX DERIV) - dC(ZREF)/dA
DCKALL(2,N,2,IT)=SUM2
DCKALL(3,N,3,IT)=SUM3
DCKALL(4,N,1,IT)=SUM1A
DCKALL(5,N,1,IT)=SUM1B
DCKALL(NPR,N,4,IT)=SSUM4+SUM4-RSUM4(npr)
DCKALL(NPR,N,5,IT)=SSUM5+SUM5-RSUM5(npr)
DCKALL(NPR,N,8,IT)=SSUM8+SUM8-RSUM8(npr)
DCKALL(NPR,N,9,IT)=SSUM9+SUM9-RSUM9(npr)
DFKTOP(1:3,IT,1:2)=DFK(1:3,NK,4:5)
DFKTOP(1:3,IT,5:6)=DFK(1:3,NK,8:9)
ENDDO
ENDDO

DO NPR=1,3
    DFK(NPR,:,NPR)=1.
ENDDO
    DFK(4,:,1)=D13RESP/1000.
    DFK(5,:,1)=D18RESP/1000.

IF(IDIS.EQ.1) THEN
DO N=1,NUMLEV
DO NPR=1,5
  DO L=1,7
    K=L
    IF(L.EQ.6) K=8
    IF(L.EQ.7) K=9
    ASUM=0.
      DO I=1,IS3
        ASUM=ASUM+DISC(N,I)*DSK(NPR,I,K)*(ZSOU(I)-ZSOU(I-1))
      ENDDO
      ASUM=ASUM+DISC(N,0)*DFK(NPR,0,K)*ZSOU(1)
    DCKALL(NPR,N,K,IT)=ASUM
  ENDDO
ENDDO
ENDDO
ENDIF

IF(ISOTOPES.GT.1) THEN
DO K=1,NUMAK
    DCKALL(4,:,K,IT)=DCKALL(4,:,K,IT)/CKALL(1,:,IT)/RSTD*1000. - &
    CKALL(4,:,IT)/CKALL(1,:,IT)**2/RSTD*1000.*DCKALL(1,:,K,IT)
```

```
        DCKALL(5,:,K,IT)=DCKALL(5,:,K,IT)/CKALL(1,:,IT)/R18STD*1000. -  &
CKALL(5,:,IT)/CKALL(1,:,IT)**2/R18STD*1000.*DCKALL(1,:,K,IT)
    ENDDO
    ENDIF

    ENDDO TENLOOPS4

    CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZREF,ZMON,SWTOP,TLTOP)
    DCKD(:,:,1:3,:)=DCKALL(:,:,1:3,:)
    DCKALL(:,:,1:6,:)=DCKALL(:,:,4:9,:)
    DCKALL(:,:,7:9,:)=0.
    DCKALL(:,:,3:4,:)=DCKD(:,:,6:7,:)
    DO IT=0,NUMISO-1
        DCKALL(:,:,7+IT*3:9+IT*3,IT+1)=DCKD(:,:,1:3,IT+1)
        DFKTOP(:,IT+1,7+IT*3:9+IT*3)=1.
    ENDDO
    IF(IOPT.EQ.0) THEN
        DCKALL(:,:,1:30,:)=DCKALL(:,:,7:36,:)
    ENDIF
    IF(NUMK.EQ.35) THEN     ! NO TL1
        DCKALL(:,:,3:35,:)=DCKALL(:,:,4:36,:)
        DFKTOP(:,:,3:35)=DFKTOP(:,:,4:36)
    ELSEIF(NUMK.EQ.33) THEN
        DCKALL(:,:,1,:)=DCKALL(:,:,2,:);DCKALL(:,:,2:3,:)=DCKALL(:,:,4:5,:)
        DCKALL(:,:,4:33,:)=DCKALL(:,:,7:36,:)
    ELSEIF(NUMK.EQ.34) THEN     ! NO TL1, NO VM
        DCKALL(:,:,3:35,:)=DCKALL(:,:,4:36,:)
        DCKALL(:,:,1:34,:)=DCKALL(:,:,2:35,:)
        DFKTOP(:,:,3:35)=DFKTOP(:,:,4:36)
        DFKTOP(:,:,1:34)=DFKTOP(:,:,2:35)
    ELSEIF(NUMK.EQ.54) THEN     ! OPTIMISE TL1 AND TL2 FOR EACH PROFILE SET SEPARATELY
        DCKALL(:,:,3:4,:)=DCKALL(:,:,5:6,:)
        DCKALL(:,:,25:54,:)=DCKALL(:,:,7:36,:)
        DCKALL(:,:,5:24,:)=0.
        DO IT=1,10
          DCKALL(:,:,3+IT*2:4+IT*2,IT)=DCKD(:,:,6:7,IT)
        ENDDO
    ENDIF


    IF(IA.EQ.1) DCKEALL=DCKALL

808 DO J=1,NUMK
        DO K=1,NUMK
            DSUM=0.
            TENLOOPS5: DO IT=1,NUMISO
            DO N=1,NUMLEV
              DO NPR=1,3
                  DSUM=DSUM+DCKALL(NPR,N,J,IT)*DCKALL(NPR,N,K,IT)/SIG(NPR,N)**2
              ENDDO
              IF(ISOTOPES.GT.1) THEN
                 DSUM=DSUM+DCKALL(4,N,J,IT)*DCKALL(4,N,K,IT)/SIG(4,N)**2
              ENDIF
              IF(ISOTOPES.GT.2) THEN
                 DSUM=DSUM+DCKALL(5,N,J,IT)*DCKALL(5,N,K,IT)/SIG(5,N)**2
              ENDIF
            ENDDO
            ENDDO TENLOOPS5
            HESS(J,K)=DSUM  ! RETAIN HESSIAN MATRIX
            DO N=1,NUMLEV
                IF(IOPT.EQ.1) THEN
                IF(NUMK.NE.54) THEN
                IF(J.EQ.1.AND.K.EQ.1) THEN
                  DSUM=DSUM+1./SVM**2
                ELSEIF(J.EQ.2.AND.K.EQ.2) THEN
                  DSUM=DSUM+1./SALAM**2
```

```
    ELSEIF(J.EQ.3.AND.K.EQ.3) THEN
      DSUM=DSUM+1./STLP1**2
    ELSEIF(J.EQ.4.AND.K.EQ.4) THEN
      DSUM=DSUM+1./STLP2**2
    ELSEIF(J.EQ.5.AND.K.EQ.5) THEN
      DSUM=DSUM+1./SKDF**2
    ELSEIF(J.EQ.6.AND.K.EQ.6) THEN
      DSUM=DSUM+1./SKBF**2
    ENDIF
    ELSEIF(NUMK.EQ.54) THEN
    IF(J.EQ.1.AND.K.EQ.1) THEN
      DSUM=DSUM+1./SVM**2
    ELSEIF(J.EQ.2.AND.K.EQ.2) THEN
      DSUM=DSUM+1./SALAM**2
    ELSEIF(J.EQ.3.AND.K.EQ.3) THEN
      DSUM=DSUM+1./SKDF**2
    ELSEIF(J.EQ.4.AND.K.EQ.4) THEN
      DSUM=DSUM+1./SKBF**2
    ELSEIF(J.EQ.5.AND.K.EQ.5) THEN
      DSUM=DSUM+1./STL1(1)**2
    ELSEIF(J.EQ.6.AND.K.EQ.6) THEN
      DSUM=DSUM+1./STL2(1)**2
    ELSEIF(J.EQ.7.AND.K.EQ.7) THEN
      DSUM=DSUM+1./STL1(2)**2
    ELSEIF(J.EQ.8.AND.K.EQ.8) THEN
      DSUM=DSUM+1./STL2(2)**2
    ELSEIF(J.EQ.9.AND.K.EQ.9) THEN
      DSUM=DSUM+1./STL1(3)**2
    ELSEIF(J.EQ.10.AND.K.EQ.10) THEN
      DSUM=DSUM+1./STL2(3)**2
    ELSEIF(J.EQ.11.AND.K.EQ.11) THEN
      DSUM=DSUM+1./STL1(4)**2
    ELSEIF(J.EQ.12.AND.K.EQ.12) THEN
      DSUM=DSUM+1./STL2(4)**2
    ELSEIF(J.EQ.13.AND.K.EQ.13) THEN
      DSUM=DSUM+1./STL1(5)**2
    ELSEIF(J.EQ.14.AND.K.EQ.14) THEN
      DSUM=DSUM+1./STL2(5)**2
    ELSEIF(J.EQ.15.AND.K.EQ.15) THEN
      DSUM=DSUM+1./STL1(6)**2
    ELSEIF(J.EQ.16.AND.K.EQ.16) THEN
      DSUM=DSUM+1./STL2(6)**2
    ELSEIF(J.EQ.17.AND.K.EQ.17) THEN
      DSUM=DSUM+1./STL1(7)**2
    ELSEIF(J.EQ.18.AND.K.EQ.18) THEN
      DSUM=DSUM+1./STL2(7)**2
    ELSEIF(J.EQ.19.AND.K.EQ.19) THEN
      DSUM=DSUM+1./STL1(8)**2
    ELSEIF(J.EQ.20.AND.K.EQ.20) THEN
      DSUM=DSUM+1./STL2(8)**2
    ELSEIF(J.EQ.21.AND.K.EQ.21) THEN
      DSUM=DSUM+1./STL1(9)**2
    ELSEIF(J.EQ.22.AND.K.EQ.22) THEN
      DSUM=DSUM+1./STL2(9)**2
    ELSEIF(J.EQ.23.AND.K.EQ.23) THEN
      DSUM=DSUM+1./STL1(10)**2
    ELSEIF(J.EQ.24.AND.K.EQ.24) THEN
      DSUM=DSUM+1./STL2(10)**2
    ENDIF
    ENDIF
    ENDIF
ENDDO
IF(IEDDY.EQ.1) THEN
  TENLOOPS6: DO IT=1,NUMISO
    SIGFC=ABS(0.05*FC2(ICN(IT))*VOLM)   ! SIGMA OF TOTAL C FLUX
    SIGFH=ABS(0.1*H(ICN(IT))/CAPP/RHO)  ! SIGMA OF TOTAL H FLUX
```

```
        SIGFE=ABS(0.1*E(ICN(IT))*VOLM)  ! SIGMA OF TOTAL E FLUX
        DSUM = DSUM + DFKTOP(1,IT,J)*DFKTOP(1,IT,K)/SIGFC**2
        DSUM = DSUM + DFKTOP(2,IT,J)*DFKTOP(2,IT,K)/SIGFH**2
        DSUM = DSUM + DFKTOP(3,IT,J)*DFKTOP(3,IT,K)/SIGFE**2
      ENDDO TENLOOPS6
    ENDIF
    IF(J.EQ.K) THEN             ! ALPHA(J,K) = SUM( (dC/dAj * dC/dAk)/SIGMA^2)
        ALP(J,K)=DSUM*(1.+RL)   ! *(1 + LAMBDA) ON DIAGONALS
    ELSE
        ALP(J,K)=DSUM
    ENDIF
ENDDO
DSUM=0.
TENLOOPS7: DO IT=1,NUMISO
DO NPR=1,3
  DO N=1,NUMLEV
    IF(DCKALL(NPR,N,J,IT).NE.0.) THEN
      DSUM=DSUM+(-CKALL(NPR,N,IT)+SPROFALL(N,NPR,IT))*DCKALL(NPR,N,J,IT)/SIG(NPR,N)**2
    ENDIF
  ENDDO
ENDDO
ENDDO TENLOOPS7
DO N=1,NUMLEV
    ZAH=FLOAT(N)/FLOAT(NUMLEV)
    TENLOOPS8: DO IT=1,NUMISO
    IF(ISOTOPES.GT.1) THEN
     C13K=(CKALL(4,N,IT)/CKALL(1,N,IT)/RSTD-1.)*1000.
     DSUM=DSUM+(-C13K+APROFALL(N,1,IT))*DCKALL(4,N,J,IT)/SIG(4,N)**2
    ENDIF
    IF(ISOTOPES.GT.2) THEN
     O18K=(CKALL(5,N,IT)/CKALL(1,N,IT)/R18STD-1.)*1000.
     DSUM=DSUM+(-O18K+APROFALL(N,2,IT))*DCKALL(5,N,J,IT)/SIG(5,N)**2
    ENDIF
    ENDDO TENLOOPS8
    IF(IOPT.EQ.1) THEN
    IF(NUMK.NE.54) THEN
    IF(J.EQ.1) THEN
      DSUM=DSUM+(-AK(4)+VM)/SVM**2
    ELSEIF(J.EQ.2) THEN
      DSUM=DSUM+(-AK(5)+ALAM)/SALAM**2
    ELSEIF(J.EQ.3) THEN
      DSUM=DSUM+(-AK(6)+TLP1)/STLP1**2
    ELSEIF(J.EQ.4) THEN
      DSUM=DSUM+(-AK(7)+TLP2)/STLP2**2
    ELSEIF(J.EQ.5) THEN
      DSUM=DSUM+(-AK(8)+KDF)/SKDF**2
    ELSEIF(J.EQ.6) THEN
      DSUM=DSUM+(-AK(9)+KBF)/SKBF**2
    ENDIF
    ELSEIF(NUMK.EQ.54) THEN
    IF(J.EQ.1) THEN
      DSUM=DSUM+(-AK(4)+VM)/SVM**2
    ELSEIF(J.EQ.2) THEN
      DSUM=DSUM+(-AK(5)+ALAM)/SALAM**2
    ELSEIF(J.EQ.3) THEN
      DSUM=DSUM+(-AK(8)+KDF)/SKDF**2
    ELSEIF(J.EQ.4) THEN
      DSUM=DSUM+(-AK(9)+KBF)/SKBF**2
    ELSEIF(J.EQ.5) THEN
      DSUM=DSUM+(-TL1(1)+TLP1)/STL1(1)**2
    ELSEIF(J.EQ.6) THEN
      DSUM=DSUM+(-TL2(1)+TLP2)/STL2(1)**2
    ELSEIF(J.EQ.7) THEN
      DSUM=DSUM+(-TL1(2)+TLP1)/STL1(2)**2
    ELSEIF(J.EQ.8) THEN
      DSUM=DSUM+(-TL2(2)+TLP2)/STL2(2)**2
```

```
            ELSEIF(J.EQ.9) THEN
              DSUM=DSUM+(-TL1(3)+TLP1)/STL1(3)**2
            ELSEIF(J.EQ.10) THEN
              DSUM=DSUM+(-TL2(3)+TLP2)/STL2(3)**2
            ELSEIF(J.EQ.11) THEN
              DSUM=DSUM+(-TL1(4)+TLP1)/STL1(4)**2
            ELSEIF(J.EQ.12) THEN
              DSUM=DSUM+(-TL2(4)+TLP2)/STL2(4)**2
            ELSEIF(J.EQ.13) THEN
              DSUM=DSUM+(-TL1(5)+TLP1)/STL1(5)**2
            ELSEIF(J.EQ.14) THEN
              DSUM=DSUM+(-TL2(5)+TLP2)/STL2(5)**2
            ELSEIF(J.EQ.15) THEN
              DSUM=DSUM+(-TL1(6)+TLP1)/STL1(6)**2
            ELSEIF(J.EQ.16) THEN
              DSUM=DSUM+(-TL2(6)+TLP2)/STL2(6)**2
            ELSEIF(J.EQ.17) THEN
              DSUM=DSUM+(-TL1(7)+TLP1)/STL1(7)**2
            ELSEIF(J.EQ.18) THEN
              DSUM=DSUM+(-TL2(7)+TLP2)/STL2(7)**2
            ELSEIF(J.EQ.19) THEN
              DSUM=DSUM+(-TL1(8)+TLP1)/STL1(8)**2
            ELSEIF(J.EQ.20) THEN
              DSUM=DSUM+(-TL2(8)+TLP2)/STL2(8)**2
            ELSEIF(J.EQ.21) THEN
              DSUM=DSUM+(-TL1(9)+TLP1)/STL1(9)**2
            ELSEIF(J.EQ.22) THEN
              DSUM=DSUM+(-TL2(9)+TLP2)/STL2(9)**2
            ELSEIF(J.EQ.23) THEN
              DSUM=DSUM+(-TL1(10)+TLP1)/STL1(10)**2
            ELSEIF(J.EQ.24) THEN
              DSUM=DSUM+(-TL2(10)+TLP2)/STL2(10)**2
            ENDIF
            ENDIF
            ENDIF
        ENDDO
        IF(IEDDY.EQ.1) THEN
          TENLOOPS8A: DO IT=1,NUMISO
            SIGFC=ABS(0.05*FC2(ICN(IT))*VOLM)   ! SIGMA OF TOTAL C FLUX
            SIGFH=ABS(0.1*H(ICN(IT))/CAPP/RHO)  ! SIGMA OF TOTAL H FLUX
            SIGFE=ABS(0.1*E(ICN(IT))*VOLM)  ! SIGMA OF TOTAL E FLUX
            DSUM = DSUM + (-FKTOP(1,IT)+FC2(ICN(IT))*VOLM)*DFKTOP(1,IT,J)/SIGFC**2
            DSUM = DSUM + (-FKTOP(2,IT)+H(ICN(IT))/CAPP/RHO)*DFKTOP(2,IT,J)/SIGFH**2
            DSUM = DSUM + (-FKTOP(3,IT)+E(ICN(IT))*VOLM)*DFKTOP(3,IT,J)/SIGFE**2
          ENDDO TENLOOPS8A
        ENDIF
        BET(J)=DSUM     ! BETA(J) = SUM (C_MOD - C_MEAS) * dC/dAj / SIGMA^2
    ENDDO

! SOLVE ALP*DAK=BET
    IF(IOPT.EQ.0) THEN
      DO J=1,NUMK
        DAK(J)=1./ALP(J,J)*BET(J)
      ENDDO
    ELSE

    CALL DLSGRR(NUMK,NUMK,ALP,NUMK,1.E-12,IRANK,AINV,NUMK)
        ! INVERSE OF ALP BY SINGULAR VALUE DECOMPOSITION
    DO K=1,NUMK
        DSUM=0.
        DO J=1,NUMK
            DSUM=DSUM+AINV(K,J)*BET(J)
        ENDDO
        DAK(K)=DSUM
    ENDDO
```

```
    ENDIF
    AKO=DBLE(AK)    ! KEEP OLD PARAMETERS, ADJUST NEW PARAMS
    GFO=GF
    TL1O=TL1; TL2O=TL2
    IF(IOPT.EQ.0) THEN
        DO I=1,NUMISO
            GF(:,I)=GF(:,I)+SNGL(DAK(I*3-2:I*3))
        ENDDO
    ELSE
        DO I=1,NUMISO
            IF(NUMK.EQ.33) THEN
                GF(:,I)=GF(:,I)+SNGL(DAK(1+I*3:3+I*3))
            ELSEIF(NUMK.EQ.34) THEN
                GF(:,I)=GF(:,I)+SNGL(DAK(2+I*3:4+I*3))
            ELSEIF(NUMK.EQ.35) THEN
                GF(:,I)=GF(:,I)+SNGL(DAK(3+I*3:5+I*3))
            ELSEIF(NUMK.EQ.36) THEN
                GF(:,I)=GF(:,I)+SNGL(DAK(4+I*3:6+I*3))
            ELSEIF(NUMK.EQ.54) THEN
                GF(:,I)=GF(:,I)+SNGL(DAK(22+I*3:24+I*3))
            ENDIF
        ENDDO
        IF(NUMK.NE.54) THEN
          AK(4:9)=AK(4:9)+SNGL(DAK(1:6))
        ELSEIF(NUMK.EQ.54) THEN
          AK(4:5)=AK(4:5)+SNGL(DAK(1:2)); AK(8:9)=AK(8:9)+SNGL(DAK(3:4))
          DO IT=1,10
            TL1(IT)=TL1(IT)+SNGL(DAK(3+IT*2))
            TL2(IT)=TL2(IT)+SNGL(DAK(4+IT*2))
          ENDDO
        ENDIF
!       AK(5)=AK(5)+SNGL(DAK(1)); AK(7:9)=AK(7:9)+SNGL(DAK(2:4))
    ENDIF
    EX=0.1
    IF (IFLAGC.EQ.1)    GOTO 3045

3025    CONTINUE

3045    CONTINUE

    AK=SNGL(AKO)
    TENLOOPS9: DO IT=1,NUMISO
    call air(SREF(2,IT),APRESS(ICN(IT)),rho,volm,capp,rlam,qsat,epsi)
    UHREF=UH(ICN(IT))/1.3
    USTARR=US(ICN(IT))
    SIGWR=SWR(ICN(IT))
    AK(1:3)=GF(:,IT)
        ZL=0.; ZREF=HCS
        CALL CALCS(PARAM,FD(ICN(IT)),PAR(ICN(IT)),DEF,SWDREF(IT), &
        LWDREF(IT),DTIME(ICN(IT)),APRESS(ICN(IT)), &
        CKALL(:,:,IT),XPROF,SREF(1,IT),ESATL,thetl,RLAI,RC,RH,RNN,ak, &
            hcs,cica,CSOURCE,esource,80,HSOURCE,C13SOURCE,O18SOURCE, &
            H18SOURCE,UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
    DO I=0,40
        SK(1,I)=CSOURCE(I*2)
        SK(2,I)=HSOURCE(I*2)
        SK(3,I)=ESOURCE(I*2)
        SK(4,I)=C13SOURCE(I*2)
        SK(5,I)=O18SOURCE(I*2)
        THETL(I)=THETL(I*2)
        RNN(I)=RNN(I*2)
        RLAI(I)=RLAI(I*2)
        RC(I)=RC(I*2)
        RH(I)=RH(I*2)
        DEF(I)=DEF(I*2)
        DELTA(I)=DELTA(I*2)
```

```
    CICA(I)=CICA(I*2)
    DEL18(I)=DEL18(I*2)
    DEL_E(I)=DEL_E(I*2)
    ZSOU(I)=ZSOU(I*2)
ENDDO

    CALL CALCE(APROFALL(:,3:4,IT),SPROFALL(:,7:9,IT),PARAM, &
    FD(ICN(IT)),PAR(ICN(IT)),SWDREF(IT),LWDREF(IT), &
    DTIME(ICN(IT)),APRESS(ICN(IT)),CKALL(:,:,IT),XPROF,SREF(1,IT), &
    ak,hcs,CFLUX,eflux,80,ZL,ZREF,HFLUX, &
        C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
DO I=0,40
    FK(3,I)=EFLUX(I*2)
    FK(2,I)=HFLUX(I*2)
    FK(1,I)=CFLUX(I*2)
    FK(4,I)=C13FLUX(I*2)
    FK(5,I)=O18FLUX(I*2)
    ZPR(I)=ZPR(I*2)
ENDDO

DO I=1,40
    ZI=ZSOU(I)
    CALL INTVAL2(ZI,TEMP,NUML,XPROF(1:NUML),CKALL(2,:,IT))
    CALL INTVAL2(ZI+0.1,TEMPD,NUML,XPROF(1:NUML),CKALL(2,:,IT))
    TK=TEMP+273.15;TKD=TEMPD+273.15;DELT=(TEMPD-TEMP)/(0.1)
    IF(NUMK.NE.54) THEN
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,ZMON,SWZ,TLZ)
    ELSEIF(NUMK.EQ.54) THEN
      CALL TURB2(USTARR,SIGWR,TL1(IT),TL2(IT),ZI,ZMON,SWZ,TLZ)
    ENDIF
    TLPROF(I)=TLZ
    ZOBUKHOV(I)=ZMON
ENDDO

    CSUM1=0.;CSUM2=0.;CSUM3=0.;CSUM4=0.;CSUM5=0.
    DO N=1,NUMLEV           ! CALCULATE CHI-SQUARE = SUM (C_MODEL - C_MEAS)^2 / SIGMA^2
      CSUM1=CSUM1+(CKALL(1,N,IT)-SPROFALL(N,1,IT))**2/SIG(1,N)**2
      CSUM2=CSUM2+(CKALL(2,N,IT)-SPROFALL(N,2,IT))**2/SIG(2,N)**2
      CSUM3=CSUM3+(CKALL(3,N,IT)-SPROFALL(N,3,IT))**2/SIG(3,N)**2
      IF(ISOTOPES.GT.1) THEN
       C13K=(CKALL(4,N,IT)/CKALL(1,N,IT)/RSTD-1.)*1000.
       CSUM4 = CSUM4 + (C13K-APROFALL(N,1,IT))**2/SIG(4,N)**2
      ENDIF
      IF(ISOTOPES.GT.2) THEN
       O18K=(CKALL(5,N,IT)/CKALL(1,N,IT)/R18STD-1.)*1000.
       CSUM5 = CSUM5 + (O18K-APROFALL(N,2,IT))**2/SIG(5,N)**2
      ENDIF
    ENDDO
    CHISQ1=CSUM1;CHISQ2=CSUM2;CHISQ3=CSUM3;CHISQ4=CSUM4;CHISQ5=CSUM5

WRITE(*,4036) CHISQ, (AK(K), K=1,9), dresp
do npr=1,3
IF(NPR.EQ.1) THEN
    ICP=10; ISP=11; IFP=12; FREF=FC2(ICN(IT)); FAC=1./VOLM
    PRINT *, 'CO2:'
ELSEIF(NPR.EQ.2) THEN
    ICP=13; ISP=14; IFP=15; FREF=H(ICN(IT)); FAC=RHO*CAPP
    WRITE(*,*) 'TEMPERATURE:'
ELSE
    ICP=16; ISP=17; IFP=18; FREF=E(ICN(IT)); FAC=1./VOLM
    WRITE(*,*) 'H2O:'
ENDIF
FKR=FK(NPR,40)*FAC
IF(IA.GE.ITER-1) THEN
 WRITE(*,*) 'ITERATIONS FINISHED'
ELSE
```

```
     WRITE(*,*) 'NO MORE IMPROVEMENT IN CHISQ AFTER ',IA,' ITERATIONS'
    ENDIF
    WRITE(*,*) 'FLUX= ',FKR,'EDDY FLUX= ',FREF
    WRITE(TIMDAT,116) day(ICN(IT)),mon,Tthr(ICN(IT)),Ttmin(ICN(IT))
116       format(4x,a2,'-',a3,5x,i2.2,':',i2.2)
    write(ICP,4021) timdat
4021    format(a32)
    WRITE(ICP,4032) (AK(K),K=1,9)
    WRITE(ICP,4034) CHISQ
    write(ICP,4031) (xprof(n),cKall(NPR,n,IT),sprofall(n,NPR,IT), N=1, NUMLEV)
    write(ICP,*)
4031    format(3f10.2)
4032    FORMAT(14F10.5)
    WRITE(ISP,4021) TIMDAT
    WRITE(ISP,4033)(ZSOU(I), SK(NPR,I)*FAC, FK(NPR,I)*FAC,THETL(I), I=1,40)
    WRITE(ISP,*)
4033    FORMAT(4F10.3)
4034    FORMAT(' CHISQ = ',F8.3)
    write(IFP,4035) timdat, FKR,fref,CHISQ
4035    format(a32,2f10.2,F8.2)
4036    FORMAT(' CHISQ = ',F8.3,' A = ',18F8.2)
4037 FORMAT(3F10.3,F10.5,2F10.7,F10.3,F10.2,F10.3,F10.2,F10.3,F10.3,ES10.3,6F10.3)
4038 format(a32)
4039 FORMAT(F10.2,2F10.3)
4040 FORMAT(3F10.3)
4041 FORMAT(A32, 5F10.2,7F8.3,9F10.2,F8.2,F8.3)
4042 FORMAT(20X,20F10.2)
4043 FORMAT(A20,20F10.2)
4044 FORMAT(A20,20F10.3)
4045 FORMAT(F10.2,2F10.5)
4046 FORMAT(2F10.3)
4047 FORMAT(9F10.3)
4048 FORMAT(36ES11.3)
4050 FORMAT(13A10)
    IF(NPR.EQ.3) THEN
        WRITE(19,4035) TIMDAT,FKR*RLAM*18.015*1.E-6,EREF,CHISQ*RLAM*18.015*1.E-6
    ENDIF

    ENDDO
    write(20,4038) timdat
555 WRITE(20,4050) '       Z',' LEAF TEMP','   NET RAD',' LEAF AREA', &
                '       GSC','       GBH', &
                '      DA','      DELTA','     CI/CA','  DELTA18O','  DELTA_E', &
                '       TL','    ZMON'
    WRITE(20,4037) (ZSOU(I), THETL(I), RNN(I), RLAI(I),1./RC(I)/volm,1./RH(I),DEF(I), &
                   DELTA(I), CICA(I), DEL18(I), DEL_E(I),TLPROF(I)*SIGWR/1.27/HCS, &
                   ZOBUKHOV(I),SK(1,I)/VOLM,FK(1,I)/VOLM, &
                   SK(2,I)/VOLM,FK(2,I)/VOLM, SK(3,I)*RHO*CAPP,FK(3,I)*RHO*CAPP, I=0,40)
    write(20,*)
     IF(ISOTOPES.GE.1) THEN
      WRITE(21,*)
      WRITE(21,*)
      WRITE(21,*)
      WRITE(21,4021) TIMDAT
      WRITE(21,4040) (XPROF(N),(CKALL(4,N,IT)/CKALL(1,N,IT)/RSTD-1.)*1000., &
                   APROFALL(N,1,IT),N=1,NUMLEV)
      WRITE(22,4021) TIMDAT
      WRITE(22,4039) (ZSOU(I), SK(4,I)/VOLM,FK(4,I)/VOLM, I=1,40)
      WRITE(22,*)
      WRITE(28,*)
      WRITE(28,*)
      WRITE(28,*)
      WRITE(28,4021) TIMDAT
      WRITE(28,4040) (XPROF(N),(CKALL(5,N,IT)/CKALL(1,N,IT)/R18STD-1.)*1000., &
                   APROFALL(N,2,IT),N=1,NUMLEV)
      WRITE(29,4021) TIMDAT
```

```
      WRITE(29,4045) (ZSOU(I), SK(5,I)/VOLM,FK(5,I)/VOLM, I=1,40)
      WRITE(29,*)
      WRITE(30,*)
      WRITE(30,*)
      WRITE(30,*)
      WRITE(30,4021) TIMDAT
      WRITE(30,4046) (XPROF(N),(CKALL(6,N,IT)/CKALL(3,N,IT)/SMOW-1.)*1000.,N=1,NUMLEV)
     ENDIF
    WRITE(*,*) 'ZMON= ', ZMON
      CALL INTVAL(28., 28., TEMP, CO2, H2O,NLEVL,ZZZ,PROF,28.,20,ICN(IT))
      ESAT=6.1375*EXP(17.502*TEMP/(240.97+TEMP))
      SL=17.502*240.97/(240.97+TEMP)**2*esat
      sl=0.622*pmb/(pmb-0.378*esat)**2*sl
      ep=rlam/cAPp*sl
      eqg=ep*rnn(0)*.9/(ep+1.)
    WRITE(23, 4041) TIMDAT, AK(1)/VOLM, AK(2)*CAPP*RHO, AK(3)/VOLM, &
         eqg,AK(3)/VOLM*RLAM*18.015*1.E-6,EP,ak(4),AK(5),AK(6),AK(7),ak(8),ak(9), &
         CHISQ,CHISQ1,CHISQ2,CHISQ3,CHISQ4,CHISQ5,dresp,TL1(IT),TL2(IT)
    IF(IX.EQ.1) THEN
        WRITE(24,4042) xprof
        WRITE(25,4042) XPROF
        WRITE(26,4042) XPROF
        WRITE(27,4042) XPROF
        IX=0
    ENDIF
    WRITE(24, 4043) TIMDAT, CK(1,1:NUMLEV)
    WRITE(24, 4043) TIMDAT, SPROF(1:NUMLEV,1)
    WRITE(25, 4043) TIMDAT, CK(2,1:NUMLEV)
    WRITE(25, 4043) TIMDAT, SPROF(1:NUMLEV,2)
    WRITE(26, 4043) TIMDAT, CK(3,1:NUMLEV)
    WRITE(26, 4043) TIMDAT, SPROF(1:NUMLEV,3)
    IF(ISOTOPES.GE.1) THEN
        WRITE(27, 4044) TIMDAT, ((CKALL(4,N,IT)/CKALL(1,N,IT)/RSTD-1.)*1000., N=1,NUMLEV)
        WRITE(27, 4044) TIMDAT, APROFALL(1:NUMLEV,1,IT)
    ENDIF
    ENDDO TENLOOPS9

    WRITE(*,*) (ak(n), n=1,9)
    IF(ICOSTFN.EQ.1) THEN
        WRITE(31, 4047) (COSTFN(ICOST-1,1,N),N=1,9)
        WRITE(31, 4047) (COSTFN(ICOST-1,2,N),N=1,9)
        WRITE(31, 4047) (COSTFN(ICOST-1,3,N),N=1,9)
    ELSEIF(ICOSTFN.EQ.2) THEN
! INVERT HESSIAN MATRIX
    CALL DLSGRR(NUMK,NUMK,HESS,NUMK,1.E-12,IRANK,AINV,NUMK)
        ! INVERSE OF HESS BY SINGULAR VALUE DECOMPOSITION
    DO K=1,numk
        WRITE(31, 4048) (AINV(J,K), J=1,numk)
    ENDDO
        WRITE(31, *) IRANK
    ENDIF

10000 end do                                !end reading control file

450   stop

460   stop 'error reading CO2 data'
480   stop 'error reading Temperature data'
490   stop 'error reading Vap press data'
500   stop 'error reading Flux data'
510   stop 'error reading SIGC data'
520   stop 'error reading SIGH data'
530   stop 'error reading SIGT data'

4     STOP
540   STOP 'CHISQ NaN'
```

```
      END
```

G.1.2 Ground flux optimisation

The following Fortran 90 code is modified from that given in §G.1.1 to optimise only ground fluxes for the 10 day period from 21-30 May 2000.

```
PROGRAM GROUNDFLUX

! Modified from program CANMOD to optimise only ground fluxes
! Subroutines used
!   comskp          (sousub15.for) skip comments in data input file
!   air             (sousub15.for) calculate latent ht vap, RHO, CP, VOLM
!   turb2           (sousub15.for) calculate sigw and TL
!   dispm2           (sousub15.for) calculate dispersion matrix
!   integr         (sousub15.for) integrate a function
!   shsor2           (sousub15.for) sort profiles into ascending order
!   phi             (sousub15.for) calculate phi for stability correction
!   interp          (sousub15.for) interpolate concentration profiles
!   intval          (sousub15.for) calculate interpolated value
!   calce           (sousub15.for) calculate flux profiles
!   calcs           (sousub15.for) calculate source profiles
!   assim           (sousub15.for) calculate leaf level fluxes

PARAMETER(MMAX=20,NMAX=40,numfiles=308,NUML=20,NUMP=9,IFMAX=1000, ISMAX=1000, &
                NUMK=3,NISO=16,NUMAK=36,IDMAX=400)

! NUML=NUMBER OF CONCENTRATION LEVELS TO BE USED FROM SPLINE INTERPOLATION
! NUMP=NUMBER OF SCALAR INPUT PROFILES
! SOURCE ARRAYS, DIMENSION 0:M WHERE M IS NUMBER OF SOURCE LEVELS,
! Z=HEIGHT OF TOP OF SOURCE LAYERS; ZS=HEIGHT OF MIDDLE OF SOURCE LAYERS;
! DZ=WIDTH OF SOURCE LAYERS
! SA,SH,SE,SC = OUTPUT SOURCES; FA,FH,FE,FC = OUTPUT FLUXES
! DIS,DISN,DISF = DISPERSION MATRICES
! T,Q,EV,CC,CA,CH,CE = PROFILES INPUT TO MODEL (AFTER INTERPOLATION)
! ZC NO LONGER USED

      REAL, DIMENSION(:),ALLOCATABLE :: Z,ZS,DZ,FA,SA, &
               FH,SH,FE,SE,FC,SC,HH,BBADD
    REAL, DIMENSION(:,:), ALLOCATABLE :: DIS,DISN,DISF
    REAL, DIMENSION(:), ALLOCATABLE :: T,Q,EV,CC,CA,CH,CE,BB
    REAL*8, DIMENSION(:), ALLOCATABLE :: AA,AAL
    integer, dimension(:), allocatable :: IP
    REAL, DIMENSION(:,:), ALLOCATABLE :: ZC
    REAL A1,AH,A0,CCH,CC0,ALFAA,ALFAB,ALFAC,HC,SWREF,USREF
      COMMON /T/ ITURB,A1,AH,A0,CCH,CC0,alfaa,alfab,ALFAC, &
                HC,SWREF,USREF,IST_COR

      character mon*3,day(numfiles)*2,timdat*32
      character*60 exptnam,filenam,path_IP,path_OP
      character*90 path_file_IP,outfila,outfils,outdmtx

!     arrays needed for profile and flux data for "numfiles" half-hrs
      character*8                                      &   !time & date
      Ttdate(numfiles),VPdate(numfiles),C2date(numfiles), &
      Fxdate(numfiles),SIGCDAT(NUMFILES),SIGHDAT(NUMFILES), &
      SIGTDAT(NUMFILES),D13DAT(NISO)

      integer Tthr(numfiles),Ttmin(numfiles),VPhr(numfiles), &
```

```
             VPmin(numfiles),C2hr(numfiles),C2min(numfiles), &
             Fxhr(numfiles),Fxmin(numfiles),SIGCHR(NUMFILES),SIGCMIN(NUMFILES), &
            SIGHHR(NUMFILES),SIGHMIN(NUMFILES),SIGTHR(NUMFILES),SIGTMIN(NUMFILES),&
            D13HR(NISO),D13MIN(NISO),NLI(NISO),nday(numfiles),ISN(10),ICN(10)
      common /timx/ Tthr,Ttmin,VPhr,VPmin,C2hr,C2min, Fxhr,Fxmin
      integer timm(numfiles,10)
      equivalence (timm,Tthr)

! Ttz,VPz,C2z = ORIGINAL HEIGHTS OF INPUT PROFILES
! ZZZ = INCORPORATES ORIGINAL HEIGHTS OF ALL SCALAR PROFILES INTO ONE MATRIX AFTER SORTING
! NLEVL = ORIGINAL NUMBER OF LEVELS FOR PROFILE DATA
! X,Y = HEIGHTS AND PROFILES AFTER INTERPOLATION
! XPROF=X
! Tt,VP,C2 = ORIGINAL INPUT PROFILES
! PROF = INCORPORATES ALL ORIGINAL PROFILES INTO ONE MATRIX AFTER SORTING
! PRAF = DUMMY MATRIX FOR SORTING
! SPROF = ALL PROFILES AFTER INTERPOLATION IN ONE MATRIX

      REAL, DIMENSION(:), ALLOCATABLE :: Ttz, VPz, C2z, za,B,C13Z,O18Z
    REAL, DIMENSION(:,:), ALLOCATABLE :: ZZZ,C13ZZ
    integer nlevl(11),nref(11)      !number of levels for each scalar
    REAL ZREF,Y(NUML),X(NUML),XPROF(NUML),NEWZ,NEWPROF

    REAL, DIMENSION(:,:), ALLOCATABLE :: Tt, VP, C2, PRAF,SPROF,F,SIGC,SIGH,SIGT,dcdt,&
                                         DHDT,DTDT,D13C,D18O
    REAL, DIMENSION(:,:,:), ALLOCATABLE :: PROF
    REAL*8 RL, DTIME(NUMFILES)

      real H(numfiles),E(numfiles),FC2(numfiles),zMonU(numfiles),us(numfiles),swr(numfiles), &
         LE(numfiles),RN(numfiles),Uh(numfiles), APRESS(NUMFILES),PAR(NUMFILES), fd(numfiles), &
         SWDR(NUMFILES),LWDR(NUMFILES),LZMON,LMON(NUMFILES)
      common /flxx/ H,E,FC2,sgw,us,ur
      real flxx(numfiles,7)
      equivalence (flxx,H)

! ARRAYS FOR ALTERNATIVE MODEL USING BETA FUNCTION TO CONSTRAIN SOURCE AND FLUX PROFILES
    REAL SK(6,0:ISMAX),FK(6,0:IFMAX),DSK(5,0:IDMAX+1,NUMAK),PARAM(5),&
        CD(NUML),SREF(6),THETL(0:ISMAX), ESATL(0:ISMAX),DUMMY(NUML),DEF(0:ISMAX), &
        DELTA(0:ISMAX),CICA(0:ISMAX),DEL18(0:ISMAX),RAIR(10), &
        SWDREF(10), LWDREF(10),TLPROF(0:ISMAX),ZOBUKHOV(0:ISMAX)
    REAL    AK(NUMAK),ake(NUMAK), AKO(NUMAK),GF(3,10),GFO(3,10),GFE(3,10)
    REAL CK(6,NUML),CKO(6,NUML), CKK(6,NUML),CKKK(6,NUML), CKE(6,NUML), &
        SIGCO2(NUML), SIGH2O(NUML),SIGTEMP(NUML),SIG13C(NUML),SIG18O(NUML), &
        SIG(5,NUML),SIGO(5,NUML)
    REAL SPROFALL(NUML,9,10),DCKALL(5,NUML,NUMAK,10),CKALL(6,NUML,10), &
        DCKEALL(5,NUML,NUMAK,10),DCKOALL(5,NUML,NUMAK,10),CKEALL(6,NUML,10), &
        CKOALL(6,NUML,10),CKKALL(6,NUML,10),DCKD(5,NUML,NUMAK),DFKTOP(3,NUMAK),FKTOP(3), &
        SPROFD(NUML),APROFC(NUML),APROFO(NUML), SPROFP(NUML), APROFCP(NUML), APROFOP(NUML)
    REAL DFK(5,0:IDMAX+1,NUMAK),DCK(5,NUML,NUMAK),FKS(6,0:ISMAX), &
    esource(0:ISMAX),ESOURCE2(0:ISMAX),&
        HSOURCE(0:ISMAX),HFLUX(0:IFMAX),HSOURCE2(0:ISMAX),eflux(0:IFMAX), CFLUX(0:IFMAX), &
        CFLUX1(NUMAK,0:IFMAX),CFLUX2(NUMAK,0:IFMAX),CSOURCE(0:ISMAX), &
        CSOURCE2(0:ISMAX),ZPR(0:IFMAX), &
        EFLUX1(NUMAK,0:IFMAX),EFLUX2(NUMAK,0:IFMAX), &
        HFLUX1(NUMAK,0:IFMAX),HFLUX2(NUMAK,0:IFMAX), EFLUXD(0:IFMAX), &
        CFLUXD(0:IFMAX),HFLUXD(0:IFMAX), &
        THETLE(0:ISMAX),ZOUT(IFMAX),C13FLUX(0:IFMAX),C13FLUXD(0:IFMAX),C13SOURCE(0:ISMAX), &
        RLAI(0:ISMAX),RC(0:ISMAX),RH(0:ISMAX),RNN(0:ISMAX),C13FLUX1(NUMAK,0:IFMAX), &
        C13FLUX2(NUMAK,0:IFMAX),RSUM13(5), RSUM14(5), RSUM15(5),RSUM18(5),RSUM10(5), &
        C13SOURCE2(0:ISMAX),rsum4(5),rsum5(5),rsum6(5),rsum7(5),rsum8(5), RSUM12(5), &
        DFKS(10,0:ISMAX), RSUM9(5),KBF,KDF, DCKO(5,NUML,NUMAK), DCKE(5,NUML,NUMAK), &
        O18SOURCE(0:ISMAX),O18FLUX(0:IFMAX),O18FLUXD(0:IFMAX),O18SOURCE2(0:ISMAX), &
        O18FLUX2(NUMAK,0:IFMAX),O18FLUX1(NUMAK,0:IFMAX), &
        H18SOURCE(0:ISMAX),H18FLUX(0:IFMAX),H18FLUXD(0:IFMAX),DEL_E(0:ISMAX),ZSOU(0:ISMAX)
    REAL*8 ALP(NUMK,NUMK), BET(NUMK),DAK(NUMK),AINV(NUMK,NUMK),DSUM
    REAL    LSIGC(2),LSIGH(2),LSIGT(2),BINT(3),BINT2(3),IPN(NUMK), DCT(6,2),DCFT(5,2),&
```

```
           DISC(NUML,0:ISMAX),DISCN(NUML,0:ISMAX),DISCF(NUML,0:ISMAX), &
           APROF(NUML,2), AZZ(NUML)
    REAL*8 BETA12(3),costfn(10,3,9),HESS(NUMK,NUMK),&
           check(2,2),xxxx(numk)
    EXTERNAL BETA, BETAI
    DATA RSTD /0.011894/, R18STD /0.0020672/

! PRESET PARAMETERS:
      rmair = 0.02897        ! molecular weights (kg/mol) for air
      rmh2o = 0.018016       ! for H2O
      rmco2 = 0.044022       ! for CO2
      pmb   = 1000.          ! atmospheric pressure
      PI    = 3.14159
    IWRITE=1               ! WRITE DISPERSION MATRIX
    ICOST=1

! open and read control information from sourcfil.dat
      open(1,file='sourcfil_15d.dat')
2     call comskp(1)                     !skip comments

      read(1,*,end=450) exptnam          !experiment name
    read(1,105,end=450) path_IP        !Input path name
    read(1,105,end=450) path_OP        !Output path name
105   format(a60)
      flen_IP=len_trim(path_IP)
      flen_OP=len_trim(path_OP)

      do 10000 iblk=1,300                !read up to 300 days' file names

        READ(1,*,end=450) (nlevl(I),I=1,3),(NREF(i),i=1,3),HC,PMB
        READ(1,*,err=450,END=450) ITURB,M0,M,A1,AH,A0,CCH,CC0,alfaa,alfab,ALFAC,ist_cor
        read(1,*,err=450,end=450) nfil

    MN=MAXVAL(NLEVL)
    ALLOCATE(Z(0:M),ZS(0:M),DZ(0:M),FA(0:M),SA(0:M),FH(0:M),SH(0:M), &
             FE(0:M),SE(0:M),FC(0:M),SC(0:M))
      ALLOCATE(AA((M+1)**2),BB(M+1),IP(M+1),AAL((M+1)**2),BBADD(M+1))
    ALLOCATE(T(NUML),Q(NUML),EV(NUML),CC(NUML), CA(NUML),CH(NUML),CE(NUML))
    ALLOCATE(ZC(NUMP,MN),HH((M+1)**2))
    ALLOCATE(Tt(MN,NUMFILES),C2(MN,NUMFILES),&
             VP(MN,NUMFILES),PROF(MN,NUMP,NUMFILES),SPROF(NUML,NUMP), &
            SIGC(MN,NUMFILES),SIGH(MN,NUMFILES),SIGT(MN,NUMFILES), &
            dcdt(mn,numfiles),DHDT(MN,NUMFILES),DTDT(MN,NUMFILES))
    ALLOCATE(PRAF(MN,NUMFILES),D13C(MN,NISO),D18O(MN,NISO))
    ALLOCATE(Ttz(MN),C2z(MN),VPz(MN),ZA(MN),ZZZ(MN,NUMP),B(MN),F(4,MN),C13Z(MN),O18Z(MN))
    ALLOCATE(DIS(NUML,0:M),DISN(NUML,0:M),DISF(NUML,0:M),C13ZZ(MN,NISO))

        do ifa=1,nfil                          !read data for profiles and fluxes
          read(1,105) filenam
        path_file_IP=path_IP(1:flen_IP)//filenam

          open(10,file=path_file_IP,mode='read')

          select case(filenam(1:1))
            case('C','c')
              select case(filenam(2:2))
              case('O','o')                    !CO2
                  read(10,*)
                read(10,200) (C2z(j),j=1,nlevl(1))        !read in CO2 heights
                do i=1,numfiles                              !read in CO2 data
                  read(10,210,end=100,err=460) &
                   C2date(i),C2hr(i),C2min(i),(C2(j,i),j=1,nlevl(1))
                end do
100             continue
                nC2=i-1                                 !# CO2 records
```

```
            end select

          case('T','t')
            read(10,*)                                  !skip one line
            read(10,200) (Ttz(j),j=1,nlevl(2))             !read in Temp heights
            do i=1,numfiles                                   !read in Temp data
              read(10,210,end=120,err=480)  &
                Ttdate(i),Tthr(i),Ttmin(i),(Tt(j,i),j=1,nlevl(2))
            end do
120         continue
            nTt=i-1                                     !# Temp records


          case('V','v')
            read(10,*)                                  !skip one line
            read(10,200) (VPz(j),j=1,nlevl(3))             !read in VP heights
            do i=1,numfiles                                   !read in VP data
              read(10,210,end=130,err=490)  &
                VPdate(i),VPhr(i),VPmin(i),(VP(j,i),j=1,nlevl(3))
            end do
130         continue
            nVP=i-1                                     !# VP records

          case('F','f')
            read(10,*,end=140,err=500)                  !skip 1 lines
            do i=1,numfiles
                read(10,220,end=140) fxdate(i),fxhr(i),fxmin(i), &
                  H(i),LE(i),E(i),FC2(i),us(i),swr(i),zMonU(i),Uh(i),rn(i), &
                  PAR(I),fd(i),SWDR(I),LWDR(I)
            end do
140         continue
            nfx=i-1                                     !# flux records

        CASE('A','a')
            READ(10,*)
            DO I=1,NUMFILES
                READ(10,270,END=141) DTIME(I), apress(i)
            ENDDO
141         CONTINUE
        apress=1000.

        CASE('S','s')
            SELECT CASE(FILENAM(7:7))
            CASE('C')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,230, END=150,ERR=510) nday(i),SIGCDAT(I),SIGCHR(I), &
                 SIGCMIN(I),(SIGC(J,I),J=1,NLEVL(1))
              ENDDO
150           CONTINUE
            CASE('H')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,231, END=160,ERR=520) SIGHDAT(I),SIGHHR(I), &
                 SIGHMIN(I),(SIGH(J,I),J=1,NLEVL(3))
              ENDDO
160           CONTINUE
            CASE('T')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,231, END=170,ERR=530) SIGTDAT(I),SIGTHR(I), &
                 SIGTMIN(I),(SIGT(J,I),J=1,NLEVL(2))
              ENDDO
```

```
170           CONTINUE
            CASE('A')
              READ(10,*)
              READ(10,*)
              READ(10,232) (SIGCO2(J),J=1,NUML)
              READ(10,232) (SIGH2O(J),J=1,NUML)
              READ(10,232) (SIGTEMP(J),J=1,NUML)
              READ(10,232) (SIG13C(J),J=1,NUML)
              READ(10,232) (SIG18O(J),J=1,NUML)
            END SELECT

        CASE('D','d')
            SELECT CASE(FILENAM(2:2))
            CASE('C','c')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,240,END=180) (DCDT(J,I),J=1,NLEVL(1))
              ENDDO
180           CONTINUE
            CASE('H','h')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,240,END=190) (DHDT(J,I),J=1,NLEVL(1))
              ENDDO
190           CONTINUE
            CASE('T','t')
              READ(10,*)
              READ(10,*)
              DO I=1,NUMFILES
                READ(10,240,END=195) (DTDT(J,I),J=1,NLEVL(1))
              ENDDO
195           CONTINUE
            CASE('I','i')
              SELECT CASE(FILENAM(4:4))
              CASE('C','c')
                READ(10,*)
                READ(10,260,END=196) (C13z(j),j=1,6)        !read in C13 heights
                DO I=1,NISO
                  READ(10,250) D13DAT(I),D13HR(I),D13MIN(I),(D13C(J,I),J=1,6)
                ENDDO
196             CONTINUE
              CASE('O','o')
                READ(10,*)
                READ(10,200,END=197) (O18Z(J),J=1,6)
                DO I=1,NISO
                  READ(10,260) (D18O(J,I),J=1,6)
                ENDDO
197             CONTINUE
              END SELECT
            END SELECT

          case default
            stop 'Incorrect profile or flux file type'

          end select

200       format(18x,10f8.2)
210       format(a8,i5,1x,i2,10f8.2)
220     format(a8,i5,1x,i2,6f8.2,f9.2,6f8.2)
230     format(I5,a8,i5,1x,i2,10f8.3)
231     format(a8,i5,1x,i2,10f8.3)
232     FORMAT(8X,20F8.2)
240     FORMAT(16X,10ES11.3)
250     FORMAT(A8,I5,1X,I2,6F9.3)
```

```
260     FORMAT(16X,6F9.3)
270     FORMAT(F11.3,F8.2)

        end do                                          !end reading block
!                                                      of files

    IX=1
    RMSIGC=MAXVAL(SIGC)
    RMSIGH=MAXVAL(SIGH)
    RMSIGT=MAXVAL(SIGT)
    LSIGC=MAXLOC(SIGC)
    LSIGH=MAXLOC(SIGH)
    LSIGT=MAXLOC(SIGT)
    WRITE(*,*) RMSIGC, RMSIGH, RMSIGT
    WRITE(*,*) LSIGC, LSIGH, LSIGT
!     assume numfiles 1/2h records available each day
      nrec=numfiles
    PROF(:,1,:)=C2
    PROF(:,2,:)=TT
    PROF(:,3,:)=VP
    PROF(:,4,:)=SIGC
    PROF(:,5,:)=SIGT
    PROF(:,6,:)=SIGH
    PROF(:,7,:)=DCDT
    PROF(:,8,:)=DTDT
    PROF(:,9,:)=DHDT
    SIG(1,:)=SIGCO2; SIG(2,:)=SIGTEMP; SIG(3,:)=SIGH2O; SIG(4,:)=SIG13C; SIG(5,:)=SIG18O
    SIG(1,:)=0.6*SIG(1,:);SIG(2,:)=0.2*SIG(2,:);SIG(3,:)=0.1*SIG(3,:)
        SIG(4,:)=0.3*SIG(4,:);SIG(5,:)=0.5*SIG(5,:) !WEIGHT TO EACH SPECIES
    ZZZ(:,1)=C2z
    ZZZ(:,2)=Ttz
    ZZZ(:,3)=VPz
    ZZZ(:,4)=C2z
    ZZZ(:,5)=Ttz
    ZZZ(:,6)=VPz
    ZZZ(:,7)=C2z
    ZZZ(:,8)=Ttz
    ZZZ(:,9)=VPz
    PROF(:,4,:)=0.3; PROF(:,5,:)=0.3    !CONSTANT SIGMA FOR EACH (10X ACTUAL)
!   PROF(7:9,6,:)=0.3
    prof(:,6,:)=0.05
    DO I=4,9
        NLEVL(I)=NLEVL(I-3)
    ENDDO

! INTERPOLATE D13C, D18O PROFILES
    TH=28.0
    DO I=1,NISO
    NLI(I)=6;C13ZZ(:,I)=C13Z
      DO J=1,6      !CHECK FOR BAD DATA
        IF(ABS(D13C(J,I)).GT.100.) THEN
          NLI(I)=NLI(I)-1
          DO K=J,5
            D13C(K,I)=D13C(K+1,I)
            D18O(K,I)=D18O(K+1,I)
            C13ZZ(K,I)=C13Z(K+1)
          ENDDO
        ENDIF
      ENDDO
      CALL INTERP(NLI(I),C13ZZ(:,I),D13C(:,I),X,Y,TH,NUML,NV,B,F)
      APROF(:,1)=Y; AZZ=X
      CALL INTERP(NLI(I),C13ZZ(:,I),D18O(:,I),X,Y,TH,NUML,NV,B,F)
      APROF(:,2)=Y
    ENDDO

!       sort profiles into ascending height order - assume order read in is sequential
```

```
!       but may be in descending order
!       utilise equivalence between zzz(nv,nf) & Ttz(nv),VPz(nv,nf)...
!       and between prof(nv,nf,nr) & Tt(nv,nr),VP(nv,nr)...
        do nf=1,NUMP                                    !profile files
          if(zzz(1,nf).gt.zzz(nlevl(nf),nf)) then        !need to reverse order
            do nv=1,nlevl(nf)                             !n levels
              za(nv)=zzz(nlevl(nf)-nv+1,nf)              !write to scratch array
              do nr=1,nrec                              !nrec records common to
!                                                        all profile files
                praf(nv,nr)=prof(nlevl(nf)-nv+1,nf,nr)   !write to scratch array
              end do                                    !end record loop
            end do                                      !end level loop
!           now overwrite arrays in ascending order
            do nv=1,nlevl(nf)
              zzz(nv,nf)=za(nv)
              do nr=1,nrec
                prof(nv,nf,nr)=praf(nv,nr)
              end do
            end do
          end if
        end do

      DO II=1,NUMP
       DO nv=1,MN
          zc(II,nv)=ZZZ(nv,II)
       ENDDO
      ENDDO

!       end data input section

!       ************************************************************

!       construct output file names from date info
        mon=Ttdate(1)(6:8)
        day(1)=Ttdate(1)(3:4)
        if(day(1).eq.' ') day(1)='0'
!       ************************************************************
!
!   INTERPOLATE BETWEEN PROFILES
!
!   FOR TEMPERATURE ADD LINEARLY INTERPOLATED POINT BETWEEN TOP TWO
!   VALUES TO PREVENT LARGE DEVIATIONS OF THE SPLINE BETWEEN THESE TWO

        NEWZ=(ZZZ(NLEVL(2),2)+ZZZ(NLEVL(2)-1,2))/2
        ZZZ(NLEVL(2)+1,2)=ZZZ(NLEVL(2),2)
        ZZZ(NLEVL(2),2)=NEWZ
        NLEVL(2)=NLEVL(2)+1

        do 3000 nr=1,nrec

!   ADD LINEARLY INTERPOLATED POINT BETWEEN TOP TWO VALUES FOR T AND SIGT

        NEWPROF=(PROF(NLEVL(2)-1,2,NR)+PROF(NLEVL(2)-2,2,NR))/2
        PROF(NLEVL(2),2,NR)=PROF(NLEVL(2)-1,2,NR)
        PROF(NLEVL(2)-1,2,NR)=NEWPROF
        NEWPROF=(PROF(NLEVL(2)-1,5,NR)+PROF(NLEVL(2)-2,5,NR))/2
        PROF(NLEVL(2),5,NR)=PROF(NLEVL(2)-1,5,NR)
        PROF(NLEVL(2)-1,5,NR)=NEWPROF


!      ************************************************************
!       Process all available records using MRR's program

!         check if turbulence data available (criterion: u* > 5 m/s)
          if(us(nr).gt.5.) goto 3000               !try next record
```

```
        day(nr)=Ttdate(nr)(3:4)

!         assign fluxes at reference height
          Href=H(nr)                              !heat
          Eref=LE(nr)                             !LE
          F2ref=FC2(nr)                           !CO2
          swref=swr(nr)                          !sig_w above canopy
!          usref=swref/a1                             !u* ref
        USREF=US(NR)
          tc=PROF(NLEVL(2),2,NR)                      !ambient temp (deg C)
          Uhref=Uh(nr)/1.3

!         check if stability correction required
        if(ist_cor.eq.1) then
          IF(TTHR(NR).LT.5) ZMON=10.
          IF(TTHR(NR).GE.5.AND.TTHR(NR).LT.7) ZMON=50.
          IF(TTHR(NR).GE.7.AND.TTHR(NR).LT.8) ZMON=100.
          IF(TTHR(NR).GE.8.AND.TTHR(NR).LT.10) ZMON=500.
          IF(TTHR(NR).GE.10.AND.TTHR(NR).LT.17) ZMON=1000.
          IF(TTHR(NR).GE.17.AND.TTHR(NR).LT.18) ZMON=500.
          IF(TTHR(NR).GE.18.AND.TTHR(NR).LT.19) ZMON=100.
          IF(TTHR(NR).GE.19.AND.TTHR(NR).LT.22) ZMON=50.
          IF(TTHR(NR).GE.22) ZMON=10.
!         IF(TTHR(NR).LT.10.OR.TTHR(NR).GT.16) THEN
!         zMon=17./zMonU(nr)                         !Monin Obukhov length
         else
          zMon=-1000.                            !set to large -ve value
!                                                            -> neutral
!         endif
        end if

          call air(tc,pmb,rho,volm,capp,rlam,qsat,epsi)
          LMON(NR)=USREF**3*(TC+273.15)*RHO*CAPP/0.4/10./HREF   !MONIN-OBUKHOV LENGTH

3000    continue

!*************************************************************************
    DO 4004 NR=1,NREC
    write(timdat,116) day(nr),mon,Tthr(nr),Ttmin(nr)
    WRITE(*,*) TIMDAT
    NIS=3
!   IF((DAY(NR).EQ.'25'.OR.DAY(NR).EQ.'27').AND.(TTHR(NR).GE.6.AND.TTHR(NR).LT.21)) THEN
    IF(NR.GT.60.AND.TTHR(NR).GE.6.AND.TTHR(NR).LT.21) THEN !.AND. &
        (DAY(NR).EQ.'28'.and.tthr(NR).EQ.6)) THEN
    TH=28.0
    DO II=1,3
    CALL INTERP(NLEVL(II),ZZZ(:,II),PROF(:,II,NR),X,Y,TH,NUML,NV,B,F)
        SPROF(:,II)=Y
    CALL INTERP(NLEVL(II),ZZZ(:,II),PROF(:,II+6,NR),X,Y,TH,NUML,NV,B,F)
        SPROF(:,II+6)=Y ! STORAGE
    ENDDO
        XPROF=X
202 IEDDY=0     ! USE EDDY FLUXES TO CONSTRAIN 1=YES, 0=NO
    IDIS=0      ! USE DISPM SUBROUTINE FOR DERIV AND CONC CALC 1=YES, 0=NO
    ICOSTFN=2   ! 1=calculate cost function (chisq) only, no optimisation,
                ! 2=output error covariance matrix
    IFLAGC=0    ! exit after one iteration, including deriv calc
    IOPT=0      ! 0=OPTIMISE ONLY GROUND FLUXES, 1=ALSO OTHER PARAMS
    IPM=1
    IN=1
    open(unit=10, file='cprof')
    OPEN(UNIT=11, FILE='SPROFC')
    OPEN(UNIT=16, FILE='EPROF')
    OPEN(UNIT=13, FILE='HPROF')
    OPEN(UNIT=17, FILE='SPROFE')
    OPEN(UNIT=14, FILE='SPROFH')
```

```
open(unit=12, file='FLUXC')
OPEN(UNIT=15, FILE='FLUXH')
OPEN(UNIT=18, FILE='FLUXE')
OPEN(UNIT=19, FILE='FLUXLE')
OPEN(UNIT=20, FILE='MOREPROFS')
OPEN(UNIT=21, FILE='C13PROF')
OPEN(UNIT=22, FILE='SPROFC13')
OPEN(UNIT=23, FILE='GROUNDFLUX')
OPEN(UNIT=24, FILE='PROFC')
OPEN(UNIT=25, FILE='PROFE')
OPEN(UNIT=26, FILE='PROFH')
OPEN(UNIT=27, FILE='PROFC13')
OPEN(UNIT=28, FILE='O18PROF')
OPEN(UNIT=29, FILE='SPROFO18')
OPEN(UNIT=30, FILE='H18PROF')
OPEN(UNIT=31, FILE='CHISQ')
IWRITE=0
IFLAG=0
SWDREF=SWDR(NR); LWDREF=LWDR(NR)

NUMLEV=NUML             ! USE INTERPOLATED DATA
SPROF(:,1)=SPROF(:,1)+9.8   !CORRECT FOR OFFSET OF IRGA DATA FROM FLASK DATA

IS2=50      !NUMBER OF SEGMENTS FOR DC/DA INTEGRALS
IS3=100     !NUMBER OF POINTS WHERE SOURCE AND FLUX ARE EVALUATED
IS4=50      !NUMBER OF POINTS WHERE FLUX IS CALCULATED INITIALLY

FIS2=FLOAT(IS2)
FIS3=FLOAT(IS3)
FIS4=FLOAT(IS4)

HCS=23.     ! CANOPY HEIGHT
COUNT=0.
DO IK=1,NUMLEV      ! NUMBER OF HEIGHTS BELOW HC
    IF(XPROF(IK).LE.HCS) THEN
        COUNT=COUNT+1
    ENDIF
ENDDO

RL=0.001                ! INITIALISE RL (LEV-MARQ LAMBDA)

SLA1=5.; SLA2=4.    ! LEAF AREA PROFILE SHAPE
GHR=0.003   ! GH MULTIPLIER
WIND=3.     ! WIND EXTINCTION COEFFICIENT
CICA=0.6    ! INITIAL CI/CA
D13RESP=17.6    ! RESP. DISCR.
D18RESP=15. ! RESP. DISCR.
SIGO=SIG    !RETAIN ORIGINAL SIGMAS FOR CONCENTRATIONS

VM=0.49     ! VMAX PARAMETER
SVMi=10.        ! SIGMA OF VMAX PARAMETER
ALAM=130.   ! LAMBDA FOR GS
SALAMi=2500.    ! SIGMA FOR LAMBDA
TLP1=23.    ! TL EXP COEFF
STLP1I=200. ! SIGMA FOR TL EXP COEFF
TLP2=0.25   ! TL U*/H AT TOP OF CANOPY
STLP2I=3.   ! SIGMA FOR TL U*/H
KDF=0.20        ! DIFFUSE RADIATION EXTINCTION COEFFICIENT
SKDFI=8.        ! SIGMA FOR KDF
KBF=0.37    ! DIRECT RADIATION EXTINCTION COEFFICIENT
SKBFI=5.        ! SIGMA FOR KBF

AK(1:3)=(/0.05,-0.02,0.01/)     ! CO2, HEAT AND H2O FLUX AT GROUND
AK(4)=VM        ! VMAX PARAMETER
AK(5)=ALAM  ! LAMBDA FOR GS
AK(6)=TLP1  ! TL EXP COEFF
```

```
    AK(7)=TLP2  ! TL U*/H AT TOP OF CANOPY
    AK(8)=KDF   ! DIRECT RADIATION EXTINCTION COEFFICIENT
    AK(9)=KBF   ! DIFFUSE RADIATION EXTINCTION COEFFICIENT

    PARAM(1:2)=(/SLA1,SLA2/)    ! SHAPE OF LEAF AREA PROFILE
    PARAM(3)=GHR        ! GH MULTIPLIER
    PARAM(4)=4.3        ! TOTAL LEAF AREA INDEX
    PARAM(5)=WIND       ! WIND EXTINCTION COEFFICIENT

    DO NPR=1,3
    SREF(NPR)=SPROF(NUMLEV,NPR)
      DO J=1,NUMLEV ! CHECK FOR BAD STD DEVIATION + BAD DC/DT
        IF(SPROF(J,NPR+3).LT.0.001) SPROF(J,NPR+3)=1.
        IF(ABS(SPROF(J,NPR+3)).GT.900.) SPROF(J,NPR+3)=1.
        IF(ABS(SPROF(J,NPR+6)).GT.900.) SPROF(J,NPR+6)=0.
      ENDDO
    ENDDO
    ZREF=XPROF(NUMLEV)
    SMOW=0.0020052; DEL_S=-18.; ESTAR=9.2
    D18VAP=-26.
    SREF(6)=(1.+D18VAP/1000.)*SMOW*SREF(3)
    SREF(4)=(-8./1000.+1)*RSTD*SREF(1)
    SREF(5)=(-1./1000.+1)*R18STD*SREF(1)

    DO N=1,NUMLEV
      DO L=1,6
        CKO(L,N)=SREF(L)
      ENDDO
    ENDDO
    CK=CKO; CKK=CKO; AKO=AK; AKE=AK; IFLAGO=1

    NC=0.;X2=0.
    do i=0,is3-1
        ZAH=FLOAT(I)/FIS3*COUNT
        J=NINT(ZAH)
        IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
        IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
        IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
        IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
        ZAH=NC/FIS3*COUNT
        ZAL=X1+(X2-X1)*ZAH
        ZAH=ZAL/HCS
        ZSOU(I)=ZAL
    ENDDO
    ZSOU(IS3)=HCS

1122    AK=AKO
    IF(ICOSTFN.EQ.1) THEN
        IF(IPM.EQ.2) THEN
            AK(IN)=AK(IN)+0.1*AK(IN)
        ELSEIF(IPM.EQ.3) THEN
            AK(IN)=AK(IN)-0.1*AK(IN)
        ENDIF
    ENDIF

    ITER=200
3035    DO 3025 IA=1,ITER           ! ITERATE FOR AK PARAMETERS

    LIM=0
    CKO=CK
    DCKO=DCK

2121 CONTINUE

!         check if stability correction required
        if(ist_cor.eq.1) then
```

```
          IF(TTHR(NR).LT.5) ZMON=10.
          IF(TTHR(NR).GE.5.AND.TTHR(NR).LT.7) ZMON=50.
          IF(TTHR(NR).GE.7.AND.TTHR(NR).LT.8) ZMON=100.
          IF(TTHR(NR).GE.8.AND.TTHR(NR).LT.10) ZMON=500.
          IF(TTHR(NR).GE.10.AND.TTHR(NR).LT.17) ZMON=1000.
          IF(TTHR(NR).GE.17.AND.TTHR(NR).LT.18) ZMON=500.
          IF(TTHR(NR).GE.18.AND.TTHR(NR).LT.19) ZMON=100.
          IF(TTHR(NR).GE.19.AND.TTHR(NR).LT.22) ZMON=50.
          IF(TTHR(NR).GE.22) ZMON=10.
         else
          zMon=-1000.                          !set to large -ve value
!                                                        -> neutral
        end if

    UHREF=UH(NR)/1.3
    USTARR=US(NR)
    SIGWR=SWR(NR)
    IFLAG0=1
      IF(IDIS.EQ.1) THEN
        call DISPM2(USTARR,SIGWR,AK(6),AK(7),timdat,NUMLev,NUMLEV,NUMLEV,0,IS3, &
                    ZSOU,XPROF,zMon,DISC,DISCN,DISCF,IWRITE)
      ENDIF
    DO KK=1,3
! CALCULATE SOURCE AND FLUX PROFILES
    CALL CALCS(PARAM,FD(NR),PAR(NR),DEF,SWDREF,LWDREF,DTIME(NR),APRESS(NR),CK, &
               XPROF,SREF,ESATL,thetl,RLAI,RC,RH,RNN,ak, &
               hcs,cica,CSOURCE,esource,is3,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
               UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
      SK(3,:)=ESOURCE   ! THESE ARE THE SOURCE PROFILES FROM Z=0 TO Z=H
      SK(2,:)=HSOURCE
      SK(1,:)=CSOURCE
      SK(4,:)=C13SOURCE
      SK(5,:)=O18SOURCE
      SK(6,:)=H18SOURCE

      CALL INTVAL(28., 28., TEMP, CO2, H2O,NLEVL,ZZZ,PROF,28.,20,NR)
      ESAT=6.1375*EXP(17.502*TEMP/(240.97+TEMP))
      SL=17.502*240.97/(240.97+TEMP)**2*esat
      sl=0.622*pmb/(pmb-0.378*esat)**2*sl
      ep=rlam/cAPp*sl
      eqg=ep*rnn(0)*.9/(ep+1.)
!     AK(3)=EQG*VOLM/RLAM/18.015/1.E-6
      CALL CALCE(SPROF(:,7:9),PARAM,FD(NR),PAR(NR),SWDREF,LWDREF,DTIME(NR),APRESS(NR), &
           CK,XPROF,SREF,ak,hcs,CFLUX,eflux,is4,0.,ZREF,HFLUX, &
           C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
!WRITE TO DUMMY VARIABLES, FLUX PROFILES FROM Z=0 TO Z=H
      EFLUXD=EFLUX; HFLUXD=HFLUX; CFLUXD=CFLUX; C13FLUXD=C13FLUX; O18FLUXD=O18FLUX; H18FLUXD=H18FLUX

!       IF(IST_COR.EQ.1) THEN
!         LZMON=USREF**3*(TC+273.15)*RHO*CAPP/0.4/10./HFLUX(IS4)    !MONIN-OBUKHOV LENGTH
!         ZMON=LZMON
!       ENDIF

    CKNR=0.;CKNR2=0.;CKNR3=0.;CKNR4=0.; CKNR5=0.;CKNR6=0.   !CALCULATE NEAR FIELD CONC AT ZREF
    NC=0.;X2=0.
    DO I=0,IS3-1
        ZAH=FLOAT(I)/FIS3*COUNT
        J=NINT(ZAH)
        IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
        IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
        IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
        IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
        ZAH=NC/FIS3*COUNT
        ZI=X1+(X2-X1)*ZAH
        ZID=ZI+(X2-X1)*COUNT/FIS3
        CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW0,TL0)
```

```
      CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,zMon,SWD,TLD)
      RKN1=RKN((ZREF-ZI)/SW0/TL0)+RKN((ZREF+ZI)/SW0/TL0)
      RKN2=RKN((ZREF-ZID)/SWD/TLD)+RKN((ZREF+ZID)/SWD/TLD)
          CKNR=CKNR+((SK(1,I))/SW0*RKN1+ (SK(1,I+1))/SWD*RKN2)/2./FIS3*HCS
          CKNR2=CKNR2+((SK(2,I))/SW0*RKN1+ (SK(2,I+1))/SWD*RKN2)/2./FIS3*HCS
          CKNR3=CKNR3+((SK(3,I))/SW0*RKN1+ (SK(3,I+1))/SWD*RKN2)/2./FIS3*HCS
          CKNR4=CKNR4+((SK(4,I))/SW0*RKN1+ (SK(4,I+1))/SWD*RKN2)/2./FIS3*HCS
          CKNR5=CKNR5+((SK(5,I))/SW0*RKN1+ (SK(5,I+1))/SWD*RKN2)/2./FIS3*HCS
          CKNR6=CKNR6+(SK(6,I)/SW0*RKN1+ SK(6,I+1)/SWD*RKN2)/2./FIS3*HCS
ENDDO
DO N=1,NUMLEV       ! CALCULATE NEAR FIELD CONC AT EACH Zi (MEASUREMENT HEIGHTS)
CKF1=0.;CKF2=0.;CKF3=0.;CKF4=0.; CKF5=0.; CKF6=0.
CKN=0.;CKN2=0.;CKN3=0.;CKN4=0.; CKN5=0.; CKN6=0.
ZL=XPROF(N)
NC=0.;X2=0.
  DO I=0,IS3-1
    ZAH=FLOAT(I)/FIS3
    ZI=ZAH*HCS
    ZID=ZI+HCS/FIS3
    ZAH=FLOAT(I)/FIS3*COUNT
    J=NINT(ZAH)
    IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
    IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
    IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
    IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
    ZAH=NC/FIS3*COUNT
    ZI=X1+(X2-X1)*ZAH
    ZID=ZI+(X2-X1)*COUNT/FIS3
    EPS=0.001*AK(6)
    EPS2=0.001*AK(7)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,zMon,SW0,TL0)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZID,ZMON,SWD,TLD)
   CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZI,zMon,SW0,TLOE)
    CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZID,ZMON,SWD,TLDE)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZI,zMon,SW0,TLOB)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZID,ZMON,SWD,TLDB)
    IF((ZL-ZI).NE.0.0.AND.(ZL-ZID).NE.0.0) THEN
      RKN1=RKN((ZL-ZI)/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ELSEIF((ZL-ZI).EQ.0.0) THEN
      RKN1=RKN(HCS/FIS3/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN((ZL-ZID)/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ELSEIF((ZL-ZID).EQ.0.0) THEN
      RKN1=RKN((ZL-ZI)/SW0/TL0)+RKN((ZL+ZI)/SW0/TL0)
      RKN2=RKN(HCS/FIS3/SWD/TLD)+RKN((ZL+ZID)/SWD/TLD)
    ENDIF
        CKN=CKN+((SK(1,I))/SW0*RKN1+ (SK(1,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN2=CKN2+((SK(2,I))/SW0*RKN1+ (SK(2,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN3=CKN3+((SK(3,I))/SW0*RKN1+ (SK(3,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN4=CKN4+((SK(4,I))/SW0*RKN1+ (SK(4,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN5=CKN5+((SK(5,I))/SW0*RKN1+ (SK(5,I+1))/SWD*RKN2)/2./FIS3*HCS
        CKN6=CKN6+(SK(6,I)/SW0*RKN1+ SK(6,I+1)/SWD*RKN2)/2./FIS3*HCS
  ENDDO
  DO I=0,IS3
    ZN=ZL+FLOAT(I)*(ZREF-ZL)/FIS3
    EPS=0.001*AK(6)
    EPS2=0.001*AK(7)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZN,zMon,SWZ,TLZ)
    CALL TURB2(USTARR,SIGWR,AK(6)+EPS,AK(7),ZN,zMon,SWZ,TLZE)
    CALL TURB2(USTARR,SIGWR,AK(6),AK(7)+EPS2,ZN,zMon,SWZ,TLZB)
    ! GET FLUX PROFILES FROM Z=Zi TO Z=ZREF FOR CONCENTRATION INTEGRAL, WHERE Zi IS HEIGHT
    ! OF CONCENTRATION MEASUREMENT, FOR EACH Zi
    IF(ZN.EQ.ZPR(IS4)) THEN
      FK(3,I)=EFLUX(IS4); FK(2,I)=HFLUX(IS4); FK(1,I)=CFLUX(IS4); FK(4,I)=C13FLUX(IS4)
      FK(5,I)=O18FLUX(IS4); FK(6,I)=H18FLUX(IS4)
    ELSEIF(ZN.GT.ZPR(IS4).OR.ZN.LT.ZPR(0)) THEN
```

```
      STOP 'ZN NOT IN RANGE FLUX CALC'
    ENDIF
    DO J=0,IS4-1
        IF(ZN.EQ.ZPR(J)) THEN   !IF Zi COINCIDES WITH A HEIGHT AT WHICH WE ALREADY HAVE
                                ! THE FLUX CALCULATION, TAKE THAT FLUX
            FK(3,I)=EFLUXD(J); FK(2,I)=HFLUXD(J); FK(1,I)=CFLUXD(J); FK(4,I)=C13FLUXD(J)
            FK(5,I)=O18FLUXD(J); FK(6,I)=H18FLUXD(J)
        ELSEIF(ZN.LT.ZPR(J+1).AND.ZN.GT.ZPR(J)) THEN    ! OTHERWISE CALCULATE ADDITIONAL
                                            ! FLUX FROM HEIGHT WE HAVE TO HEIGHT WE WANT
            EADD=EFLUXD(J); HADD=HFLUXD(J); CADD=CFLUXD(J); C13ADD=C13FLUXD(J)
            O18ADD=O18FLUXD(J); H18ADD=H18FLUXD(J)
            ZR=ZPR(J)+(ZN-ZPR(J))*FIS4
            CALL CALCE(SPROF(:,7:9),PARAM,FD(NR),PAR(NR),SWDREF,LWDREF,DTIME(NR),APRESS(NR), &
                CK,XPROF,SREF,ak,hcs,CFLUX,eflux,is4,ZPR(J),ZR,HFLUX, &
                C13FLUX,O18FLUX,H18FLUX,UHREF,ZOUT,1)
            FK(3,I)=EFLUX(1)+EADD
            FK(2,I)=HFLUX(1)+HADD
            FK(1,I)=CFLUX(1)+CADD
            FK(4,I)=C13FLUX(1)+C13ADD
            FK(5,I)=O18FLUX(1)+O18ADD
            FK(6,I)=H18FLUX(1)+H18ADD
        ENDIF
    ENDDO
    FKS(3,I)=FK(3,I)/SWZ**2/TLZ     !DIVIDE BY SIGW^2 AND TL FOR INTEGRATION
    FKS(2,I)=FK(2,I)/SWZ**2/TLZ
    FKS(1,I)=FK(1,I)/SWZ**2/TLZ
    FKS(4,I)=FK(4,I)/SWZ**2/TLZ
    FKS(5,I)=FK(5,I)/SWZ**2/TLZ
    FKS(6,I)=FK(6,I)/SWZ**2/TLZ
  ENDDO
    DO I=0,IS3-1
      CKF1=CKF1+(FKS(1,I)+FKS(1,I+1))/2./FIS3*(ZREF-ZL)     ! INTEGRATE FROM Zi TO ZREF
      CKF2=CKF2+(FKS(2,I)+FKS(2,I+1))/2./FIS3*(ZREF-ZL)
      CKF3=CKF3+(FKS(3,I)+FKS(3,I+1))/2./FIS3*(ZREF-ZL)
      CKF4=CKF4+(FKS(4,I)+FKS(4,I+1))/2./FIS3*(ZREF-ZL)
      CKF5=CKF5+(FKS(5,I)+FKS(5,I+1))/2./FIS3*(ZREF-ZL)
      CKF6=CKF6+(FKS(6,I)+FKS(6,I+1))/2./FIS3*(ZREF-ZL)
    ENDDO
    CK(1,N)=(CKN+CKF1-CKNR)+SREF(1)     ! CONCENTRATION = NEAR-FIELD + FAR-FIELD -
    CK(2,N)=(CKN2+CKF2-CKNR2)+SREF(2)   ! NEAR-FIELD (ZREF) + CONC (ZREF)
    CK(2,N)=MAX(CK(2,N),-2.); CK(2,N)=MIN(CK(2,N),25.)
    CK(3,N)=(CKN3+CKF3-CKNR3)+SREF(3)
    CK(3,N)=MAX(CK(3,N),0.); CK(3,N)=MIN(CK(3,N),15.)
    CK(4,N)=(CKN4+CKF4-CKNR4)+SREF(4)
    CK(5,N)=(CKN5+CKF5-CKNR5)+SREF(5)
    CK(6,N)=(CKN6+CKF6-CKNR6)+SREF(6)
    DO IK=1,6
        CK(IK,N)=MIN(CK(IK,N),1000.)
        CK(IK,N)=MAX(CK(IK,N),-1000.)
        IF(ABS(CK(IK,N)).EQ.1000.) THEN
            CK(IK,N)=SREF(IK)
            INAN=1
            GOTO 2122
        ENDIF
    ENDDO
    FKTOP(1)=CFLUXD(IS4)
    FKTOP(2)=HFLUXD(IS4)
    FKTOP(3)=EFLUXD(IS4)
    IF(IDIS.EQ.1) THEN
    DO NPR=1,5
    ASUM=0.
      DO I=1,IS3
        ASUM=ASUM+DISC(N,I)*SK(NPR,I)*(ZSOU(I)-ZSOU(I-1))
      ENDDO
      ASUM=ASUM+DISC(N,0)*FK(NPR,0)*ZSOU(1)
    CK(NPR,N)=ASUM+SREF(NPR)
```

```
        ENDDO
        ENDIF
    ENDDO
    ENDDO

2122    CSUM=0.
        DO N=1,NUMLEV           ! CALCULATE CHI-SQUARE = SUM (C_MODEL - C_MEAS)^2 / SIGMA^2
         DO NPR=1,3
          CSUM=CSUM+(CK(NPR,N)-SPROF(N,NPR))**2/SIG(NPR,N)**2
         ENDDO
        ENDDO
        IF(IEDDY.EQ.1) THEN
            SIGFC=ABS(0.05*FC2(NR)*VOLM)    ! SIGMA OF TOTAL C FLUX
            SIGFH=ABS(0.1*H(NR)/CAPP/RHO)   ! SIGMA OF TOTAL H FLUX
            SIGFE=ABS(0.1*E(NR)*VOLM)   ! SIGMA OF TOTAL E FLUX
           CSUM = CSUM + (FC2(NR)*VOLM-CFLUX(IS4))**2/SIGFC**2
           CSUM = CSUM + (H(NR)/CAPP/RHO-HFLUX(IS4))**2/SIGFH**2
           CSUM = CSUM + (E(NR)*VOLM-EFLUX(IS4))**2/SIGFE**2
        ENDIF

    CHISQN=CSUM

    IF(ABS(CHISQN-CHISQ).LT.0.1) GOTO 3045   !EXIT LOOP IF NO IMPROVEMENT IN CHISQ
    CHISQNN=0.
    IF(IA.EQ.1.OR.IAC.EQ.1) THEN    ! 1ST ITERATION
    CHISQ=CHISQN
    IAC=0
    CKE=CK
    ELSEIF(IA.EQ.ITER-1.AND.CHISQN.LE.CHISQ) THEN   ! 2ND LAST ITERATION, EXIT IF CHISQ
        AKO=DBLE(AK)                                ! BETTER THAN PREVIOUS
        GOTO 3045
    ELSEIF(RL.GT.1.E+2) THEN    ! EXIT IF LAMBDA LARGE
      CK=CKO; AK=SNGL(AKO)
      GOTO 3045
    ELSEIF(INAN.EQ.1) THEN
      WRITE(*,*) 'CONC NAN, AK=',(AK(K),K=4,9)
      AK=SNGL(AKO)
      CK=CKO
      DCK=DCKO
      RL=MAX(RL*100.,0.1)
      INAN=0
      GOTO 808
    ELSE IF (CHISQN.GE.CHISQ) THEN  ! INCREASE LAMBDA IF CHISQ WORSE AND USE PREVIOUS PARAMS
      AK=SNGL(AKO)
      CK=CKO
      DCK=DCKO
      RL=MAX(RL*100.,0.1)
      GOTO 808
    ELSE IF (CHISQN.LT.CHISQ) THEN  ! DECREASE LAMBDA IF CHISQ BETTER, AND CONTINUE
    RL=RL/10.
    CHISQ=CHISQN
    AKE=AK; DCKE=DCK; CKE=CK
    ELSE
      STOP 'PROBLEM SOMEWHERE, MAYBE PARAMS OUT OF RANGE'
      AK=SNGL(AKO)
      CK=CKO
      DCK=DCKO
      RL=MAX(RL*100.,0.1)
      GOTO 808
    ENDIF
    write(*,*) 'Chisq = ',chisq, 'Iterations:',ia, '  RL =',RL

1133    IF(ICOSTFN.EQ.1) THEN
        COSTFN(ICOST,IPM,IN)=CHISQ
      IF(IPM.EQ.1.OR.IPM.EQ.2) THEN
        IPM=IPM+1
```

```
        GOTO 1122
      ELSEIF(IPM.EQ.3.AND.IN.LT.9) THEN
        IN=IN+1
        IPM=2
        GOTO 1122
      ELSE
        ICOST=ICOST+1
        GOTO 3045
      ENDIF
    ENDIF

! CALCULATE DERIVATIVE OF SOURCE PROFILES WRT PARAMETERS
2022        ZL=0.

    SVM=SVMI;SALAM=SALAMI;STLP1=STLP1I;STLP2=STLP2I;SKDF=SKDFI;SKBF=SKBFI
!         check if stability correction required
        if(ist_cor.eq.1) then
          IF(TTHR(NR).LT.5) ZMON=10.
          IF(TTHR(NR).GE.5.AND.TTHR(NR).LT.7) ZMON=50.
          IF(TTHR(NR).GE.7.AND.TTHR(NR).LT.8) ZMON=100.
          IF(TTHR(NR).GE.8.AND.TTHR(NR).LT.10) ZMON=500.
          IF(TTHR(NR).GE.10.AND.TTHR(NR).LT.17) ZMON=1000.
          IF(TTHR(NR).GE.17.AND.TTHR(NR).LT.18) ZMON=500.
          IF(TTHR(NR).GE.18.AND.TTHR(NR).LT.19) ZMON=100.
          IF(TTHR(NR).GE.19.AND.TTHR(NR).LT.22) ZMON=50.
          IF(TTHR(NR).GE.22) ZMON=10.
!         ZMON=LZMON
         else
          zMon=-1000.                          !set to large -ve value
!                                                      -> neutral
        end if

    UHREF=UH(NR)/1.3
    USTARR=US(NR)
    SIGWR=SWR(NR)
    DO N=1, NUMLEV
    do npr=1,5
    ZL=XPROF(N)
    NC=0.;X2=0.
    SUM1=0.; SUM2=0; SUM3=0.; SUM4=0.; SUM1A=0.; SUM5=0.; SUM8=0.; SUM9=0.; SUM1B=0.
    IF(IDIS.EQ.0) THEN
     DO NK=0,IS2-1
          ZN=ZL+FLOAT(NK)*(ZREF-ZL)/FIS2
          ZND=ZN+(ZREF-ZL)/FIS2
          CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZN,ZMON,SWZ,TLZ)
          CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZND,ZMON,SWZD,TLZD)
          IF(NK.LT.IS2) THEN    ! INTEGRATE FLUX DERIVS FROM Z=Zi TO Z=ZREF
          SUM1=SUM1+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
          SUM1A=SUM1A+(D13RESP/1000./SWZ**2/TLZ+D13RESP/1000./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
          SUM1B=SUM1B+(D18RESP/1000./SWZ**2/TLZ+D18RESP/1000./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
          SUM2=SUM2+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
          SUM3=SUM3+(1./SWZ**2/TLZ+1./SWZD**2/TLZD)/2./FIS2*(ZREF-ZL)
          ENDIF
     ENDDO
     ENDIF

    DCK(1,N,1)=SUM1     ! CONC DERIVS = INT(SOURCE DERIV) + INT (FLUX DERIV) - dC(ZREF)/dA
    DCK(2,N,2)=SUM2
    DCK(3,N,3)=SUM3
    DCK(4,N,1)=SUM1A
    DCK(5,N,1)=SUM1B
    ENDDO
    ENDDO

    DO NPR=1,3
        DFK(NPR,:,NPR)=1.
```

```
    ENDDO
        DFK(4,:,1)=D13RESP/1000.
        DFK(5,:,1)=D18RESP/1000.

    IF(IDIS.EQ.1) THEN
    DO N=1,NUMLEV
    DO NPR=1,5
      DO L=1,7
        K=L
        IF(L.EQ.6) K=8
        IF(L.EQ.7) K=9
        ASUM=0.
          DO I=1,IS3
            ASUM=ASUM+DISC(N,I)*DSK(NPR,I,K)*(ZSOU(I)-ZSOU(I-1))
          ENDDO
          ASUM=ASUM+DISC(N,0)*DFK(NPR,0,K)*ZSOU(1)
        DCK(NPR,N,K)=ASUM
      ENDDO
    ENDDO
    ENDDO
    ENDIF

    CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZREF,ZMON,SWTOP,TLTOP)
    DCKD(:,:,1:3)=DCK(:,:,1:3)
    DCK(:,:,1:6)=DCK(:,:,4:9)
    DCK(:,:,7:9)=0.
    DCK(:,:,3:4)=DCKD(:,:,6:7)
        DCK(:,:,7:9)=DCKD(:,:,1:3)
        DFKTOP(:,7:9)=1.
    IF(IOPT.EQ.0) THEN
        DCK(:,:,1:3)=DCK(:,:,7:9)
    ENDIF

    IF(IA.EQ.1) DCKE=DCK

808 DO J=1,NUMK
        DO K=1,NUMK
            DSUM=0.
            DO N=1,NUMLEV
              DO NPR=1,3
                  DSUM=DSUM+DCK(NPR,N,J)*DCK(NPR,N,K)/SIG(NPR,N)**2
              ENDDO
            ENDDO
            HESS(J,K)=DSUM  ! RETAIN HESSIAN MATRIX
            IF(IEDDY.EQ.1) THEN
                SIGFC=ABS(0.05*FC2(NR)*VOLM)    ! SIGMA OF TOTAL C FLUX
                SIGFH=ABS(0.1*H(NR)/CAPP/RHO)   ! SIGMA OF TOTAL H FLUX
                SIGFE=ABS(0.1*E(NR)*VOLM)   ! SIGMA OF TOTAL E FLUX
                DSUM = DSUM + DFKTOP(1,J)*DFKTOP(1,K)/SIGFC**2
                DSUM = DSUM + DFKTOP(2,J)*DFKTOP(2,K)/SIGFH**2
                DSUM = DSUM + DFKTOP(3,J)*DFKTOP(3,K)/SIGFE**2
            ENDIF
            IF(J.EQ.K) THEN             ! ALPHA(J,K) = SUM( (dC/dAj * dC/dAk)/SIGMA^2)
                ALP(J,K)=DSUM*(1.+RL)   ! *(1 + LAMBDA) ON DIAGONALS
            ELSE
                ALP(J,K)=DSUM
            ENDIF
        ENDDO
        DSUM=0.
        DO NPR=1,3
          DO N=1,NUMLEV
            IF(DCK(NPR,N,J).NE.0.) THEN
              DSUM=DSUM+(-CK(NPR,N)+SPROF(N,NPR))*DCK(NPR,N,J)/SIG(NPR,N)**2
            ENDIF
          ENDDO
        ENDDO
```

```
        IF(IEDDY.EQ.1) THEN
            SIGFC=ABS(0.05*FC2(NR)*VOLM)    ! SIGMA OF TOTAL C FLUX
            SIGFH=ABS(0.1*H(NR)/CAPP/RHO)   ! SIGMA OF TOTAL H FLUX
            SIGFE=ABS(0.1*E(NR)*VOLM)   ! SIGMA OF TOTAL E FLUX
            DSUM = DSUM + (-FKTOP(1)+FC2(NR)*VOLM)*DFKTOP(1,J)/SIGFC**2
            DSUM = DSUM + (-FKTOP(2)+H(NR)/CAPP/RHO)*DFKTOP(2,J)/SIGFH**2
            DSUM = DSUM + (-FKTOP(3)+E(NR)*VOLM)*DFKTOP(3,J)/SIGFE**2
        ENDIF
        BET(J)=DSUM     ! BETA(J) = SUM (C_MOD - C_MEAS) * dC/dAj / SIGMA^2
    ENDDO

! SOLVE ALP*DAK=BET
    IF(IOPT.EQ.0) THEN
      DO J=1,NUMK
        DAK(J)=1./ALP(J,J)*BET(J)
      ENDDO
    ELSE

    CALL DLSGRR(NUMK,NUMK,ALP,NUMK,1.E-12,IRANK,AINV,NUMK)
    ! INVERSE OF ALP BY SINGULAR VALUE DECOMPOSITION
    DO K=1,NUMK
        DSUM=0.
        DO J=1,NUMK
            DSUM=DSUM+AINV(K,J)*BET(J)
        ENDDO
        DAK(K)=DSUM
    ENDDO

    ENDIF
    AKO=DBLE(AK)    ! KEEP OLD PARAMETERS, ADJUST NEW PARAMS
    IF(IOPT.EQ.0) THEN
        AK(1:3)=AK(1:3)+SNGL(DAK(1:3))
    ELSE
        AK(4:9)=AK(4:9)+SNGL(DAK(1:6))
    ENDIF
    EX=0.1
    IF (IFLAGC.EQ.1)    GOTO 3045

3025    CONTINUE

3045    CONTINUE

    AK=SNGL(AKO)
    UHREF=UH(NR)/1.3
    USTARR=US(NR)
    SIGWR=SWR(NR)
        ZL=0.; ZREF=HCS
        CALL CALCS(PARAM,FD(NR),PAR(NR),DEF,SWDREF,LWDREF,DTIME(NR),APRESS(NR),CK(:,:), &
            XPROF,SREF,ESATL,thetl,RLAI,RC,RH,RNN,ak, &
                hcs,cica,CSOURCE,esource,80,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
                UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
    DO I=0,40
        SK(1,I)=CSOURCE(I*2)
        SK(2,I)=HSOURCE(I*2)
        SK(3,I)=ESOURCE(I*2)
        SK(4,I)=C13SOURCE(I*2)
        SK(5,I)=O18SOURCE(I*2)
        THETL(I)=THETL(I*2)
        RNN(I)=RNN(I*2)
        RLAI(I)=RLAI(I*2)
        RC(I)=RC(I*2)
        RH(I)=RH(I*2)
        DEF(I)=DEF(I*2)
        DELTA(I)=DELTA(I*2)
        CICA(I)=CICA(I*2)
        DEL18(I)=DEL18(I*2)
```

```
        DEL_E(I)=DEL_E(I*2)
        ZSOU(I)=ZSOU(I*2)
    ENDDO

        CALL CALCE(SPROF(:,7:9),PARAM,FD(NR),PAR(NR),SWDREF,LWDREF,DTIME(NR),APRESS(NR), &
            CK(:,:),XPROF,SREF,ak,hcs,CFLUX,eflux,80,ZL,ZREF,HFLUX, &
            C13FLUX,O18FLUX,H18FLUX,UHREF,ZPR,0)
    DO I=0,40
        FK(3,I)=EFLUX(I*2)
        FK(2,I)=HFLUX(I*2)
        FK(1,I)=CFLUX(I*2)
        FK(4,I)=C13FLUX(I*2)
        FK(5,I)=O18FLUX(I*2)
        ZPR(I)=ZPR(I*2)
    ENDDO

    DO I=1,40
        ZI=ZSOU(I)
        CALL INTVAL2(ZI,TEMP,NUML,XPROF(1:NUML),CK(2,:))
        CALL INTVAL2(ZI+0.1,TEMPD,NUML,XPROF(1:NUML),CK(2,:))
        TK=TEMP+273.15;TKD=TEMPD+273.15;DELT=(TEMPD-TEMP)/(0.1)
        CALL TURB2(USTARR,SIGWR,AK(6),AK(7),ZI,ZMON,SWZ,TLZ)
        TLPROF(I)=TLZ
        ZOBUKHOV(I)=ZMON
    ENDDO

        CSUM1=0.;CSUM2=0.;CSUM3=0.;CSUM4=0.;CSUM5=0.
        DO N=1,NUMLEV           ! CALCULATE CHI-SQUARE = SUM (C_MODEL - C_MEAS)^2 / SIGMA^2
          CSUM1=CSUM1+(CK(1,N)-SPROF(N,1))**2/SIG(1,N)**2
          CSUM2=CSUM2+(CK(2,N)-SPROF(N,2))**2/SIG(2,N)**2
          CSUM3=CSUM3+(CK(3,N)-SPROF(N,3))**2/SIG(3,N)**2
        ENDDO
        CHISQ1=CSUM1;CHISQ2=CSUM2;CHISQ3=CSUM3;CHISQ4=CSUM4;CHISQ5=CSUM5

    WRITE(*,4036) CHISQ, (AK(K), K=1,9), dresp
    do npr=1,3
    IF(NPR.EQ.1) THEN
        ICP=10; ISP=11; IFP=12; FREF=FC2(NR); FAC=1./VOLM
        PRINT *, 'CO2:'
    ELSEIF(NPR.EQ.2) THEN
        ICP=13; ISP=14; IFP=15; FREF=H(NR); FAC=RHO*CAPP
        WRITE(*,*) 'TEMPERATURE:'
    ELSE
        ICP=16; ISP=17; IFP=18; FREF=E(NR); FAC=1./VOLM
        WRITE(*,*) 'H2O:'
    ENDIF
    FKR=FK(NPR,40)*FAC
    IF(IA.GE.ITER-1) THEN
     WRITE(*,*) 'ITERATIONS FINISHED'
    ELSE
     WRITE(*,*) 'NO MORE IMPROVEMENT IN CHISQ AFTER ',IA,' ITERATIONS'
    ENDIF
    WRITE(*,*) 'FLUX= ',FKR,'EDDY FLUX= ',FREF
    WRITE(TIMDAT,116) day(NR),mon,Tthr(NR),Ttmin(NR)
116       format(4x,a2,'-',a3,5x,i2.2,':',i2.2)
    write(ICP,4021) timdat
4021    format(a32)
    WRITE(ICP,4032) (AK(K),K=1,9)
    WRITE(ICP,4034) CHISQ
    write(ICP,4031) (xprof(n),cK(NPR,n),sprof(n,NPR), N=1, NUMLEV)
    write(ICP,*)
4031    format(3f10.2)
4032    FORMAT(14F10.5)
    WRITE(ISP,4021) TIMDAT
    WRITE(ISP,4033)(ZSOU(I), SK(NPR,I)*FAC, FK(NPR,I)*FAC,THETL(I), I=1,40)
    WRITE(ISP,*)
```

```
4033    FORMAT(4F10.3)
4034    FORMAT(' CHISQ = ',F8.3)
    write(IFP,4035) timdat, FKR,fref,CHISQ
4035    format(a32,2f10.2,F8.2)
4036    FORMAT(' CHISQ = ',F8.3,' A = ',18F8.2)
4037 FORMAT(3F10.3,F10.5,2F10.7,F10.3,F10.2,F10.3,F10.2,F10.3,F10.3,ES10.3,6F10.3)
4038 format(a32)
4039 FORMAT(F10.2,2F10.3)
4040 FORMAT(3F10.3)
4041 FORMAT(A32, 5F10.2,7F8.3,9F8.2)
4042 FORMAT(20X,20F10.2)
4043 FORMAT(A20,20F10.2)
4044 FORMAT(A20,20F10.3)
4045 FORMAT(F10.2,2F10.5)
4046 FORMAT(2F10.3)
4047 FORMAT(9F10.3)
4048 FORMAT(36ES11.3)
4050 FORMAT(13A10)
    IF(NPR.EQ.3) THEN
        WRITE(19,4035) TIMDAT,FKR*RLAM*18.015*1.E-6,EREF,CHISQ*RLAM*18.015*1.E-6
    ENDIF

    ENDDO
    write(20,4038) timdat
555 WRITE(20,4050) '       Z',' LEAF TEMP','   NET RAD',' LEAF AREA','       GSC', &
                '       GBH','      DA','      DELTA','     CI/CA','  DELTA18O','  DELTA_E', &
                '       TL','    ZMON'
    WRITE(20,4037) (ZSOU(I), THETL(I), RNN(I), RLAI(I),1./RC(I),1./RH(I),DEF(I), &
                    DELTA(I), CICA(I), DEL18(I), DEL_E(I),TLPROF(I),ZOBUKHOV(I), &
                    SK(1,I)/VOLM,FK(1,I)/VOLM, &
                    SK(2,I)/VOLM,FK(2,I)/VOLM, SK(3,I)*RHO*CAPP,FK(3,I)*RHO*CAPP, I=0,40)
    write(20,*)
    WRITE(*,*) 'ZMON= ', ZMON
      CALL INTVAL(28., 28., TEMP, CO2, H2O,NLEVL,ZZZ,PROF,28.,20,NR)
      ESAT=6.1375*EXP(17.502*TEMP/(240.97+TEMP))
      SL=17.502*240.97/(240.97+TEMP)**2*esat
      sl=0.622*pmb/(pmb-0.378*esat)**2*sl
      ep=rlam/cAPp*sl
      eqg=ep*rnn(0)*.9/(ep+1.)
    WRITE(23, 4041) TIMDAT, AK(1)/VOLM, AK(2)*CAPP*RHO, AK(3)/VOLM, &
         eqg,AK(3)/VOLM*RLAM*18.015*1.E-6,EP,ak(4),AK(5),AK(6),AK(7),ak(8),ak(9), &
         CHISQ,CHISQ1,CHISQ2,CHISQ3,CHISQ4,CHISQ5,dresp
    IF(IX.EQ.1) THEN
        WRITE(24,4042) xprof
        WRITE(25,4042) XPROF
        WRITE(26,4042) XPROF
        WRITE(27,4042) XPROF
        IX=0
    ENDIF
    WRITE(24, 4043) TIMDAT, CK(1,1:NUMLEV)
    WRITE(24, 4043) TIMDAT, SPROF(1:NUMLEV,1)
    WRITE(25, 4043) TIMDAT, CK(2,1:NUMLEV)
    WRITE(25, 4043) TIMDAT, SPROF(1:NUMLEV,2)
    WRITE(26, 4043) TIMDAT, CK(3,1:NUMLEV)
    WRITE(26, 4043) TIMDAT, SPROF(1:NUMLEV,3)

    WRITE(*,*) (ak(n), n=1,9)
    IF(ICOSTFN.EQ.1) THEN
        WRITE(31, 4047) (COSTFN(ICOST-1,1,N),N=1,9)
        WRITE(31, 4047) (COSTFN(ICOST-1,2,N),N=1,9)
        WRITE(31, 4047) (COSTFN(ICOST-1,3,N),N=1,9)
    ELSEIF(ICOSTFN.EQ.2) THEN
! INVERT HESSIAN MATRIX
    CALL DLSGRR(NUMK,NUMK,HESS,NUMK,1.E-12,IRANK,AINV,NUMK)
    ! INVERSE OF HESS BY SINGULAR VALUE DECOMPOSITION
    DO K=1,numk
```

```
        WRITE(31, 4048) (AINV(J,K), J=1,numk)
    ENDDO
        WRITE(31, *) IRANK
    ENDIF
    ENDIF
4004 CONTINUE

10000 end do                                !end reading control file

450   stop

460   stop 'error reading CO2 data'
480   stop 'error reading Temperature data'
490   stop 'error reading Vap press data'
500   stop 'error reading Flux data'
510   stop 'error reading SIGC data'
520   stop 'error reading SIGH data'
530   stop 'error reading SIGT data'

4     STOP
540   STOP 'CHISQ NaN'

      END
```

G.2 Subroutines

The following Fortran 90 code gives the subroutines and functions called in the programs CANMOD (§G.1.1) and GROUNDFLUX (§G.1.2) above.

```
$debug

!*******************************************************************************
      FUNCTION RKN(ZETA)
! NEAR-FIELD KERNEL FUNCTION IN LOCALIZED NEAR-FIELD THEORY
      DATA C1 /-0.39894/
      DATA C2 /-0.15623/
      IF (ZETA.EQ.0) STOP 'RKN: ZETA=0'
      EZ=EXP(-ABS(ZETA))
      RKN=C1*ALOG(1-EZ)+C2*EZ
      RETURN
      END

!*******************************************************************************
      FUNCTION RKNINT(ZETA)
! INTEGRAL OF NEAR-FIELD KERNEL FUNCTION: SMALL-ZETA APPROXIMATION
! VALID TO RELATIVE ERROR 0.001 AT ZETA=0.1
      DATA C3 /-0.39894/
      DATA C4 /-0.14127/
      DATA C5 /0.333333/
      ZA=ABS(ZETA)
      IF (ZA.EQ.0) THEN
        RKNINT=0
        RETURN
      END IF
      RKNINT=C3*ZA*(ALOG(ZA)-1)+C4*ZA+C5*(ZA**2)/2
      RETURN
      END

!*******************************************************************************
```

```
      SUBROUTINE INTEGR(X,Y,N,AREA)
! MRR, 26-NOV-80
! USES TRAPEZOIDAL RULE TO INTEGRATE FUNCTION Y(X), SPECIFIED AT N
! POINTS (X(I),Y(I),I=1,N). INTEGRAL GOES FROM X(1) TO X(N).
      DIMENSION X(N),Y(N)
      AREA=0
      IF (N.LE.0) STOP 'INTEGR: N<=0'
      IF (N.EQ.1) RETURN
      DO 1 I=2,N
      AREA=AREA+0.5*(Y(I)+Y(I-1))*(X(I)-X(I-1))
1     CONTINUE
      RETURN
      END

!***********************************************************************
      SUBROUTINE SHSOR2(X,N,NMAX,NS)
! MRR, 03-APR-85 (ADAPTED FROM SHSORT)
! SORT (X(I,1),I=1,N) INTO ASCENDING ORDER WITH SHELL SORT. FORCE
! OTHER SERIES ((X(I,K),I=1,N),K=2,NS) TO HAVE SAME ORDER AS X(I,1).
      DIMENSION X(NMAX,NS),Y(10)
      IF (NS.GT.10) STOP 'SHSOR2: NS TOO BIG'
! M=INCREMENT=2**IM-1 (IM=LIM,...,1)
      LIM=ALOG(REAL(N+1))/ALOG(2.0)
! LOOP THROUGH VALUES OF M (SORTING INCREMENT)
      DO 1 IM=LIM,1,-1
      M=2**IM-1
! DO M DIFFERENT M-SORTS, STARTING AT IS
      DO 2 IS=1,M
! DO 1 M-SORT BY INSERTION
      DO 3 I=IS+M,N,M
      DO 7 K=1,NS
7     Y(K)=X(I,K)
      DO 4 J=I-M,IS,-M
      IF (X(J,1).GT.Y(1)) THEN
        DO 5 K=1,NS
5       X(J+M,K)=X(J,K)
      ELSE
        DO 6 K=1,NS
6       X(J+M,K)=Y(K)
        GOTO 3
      END IF
4     CONTINUE
      DO 8 K=1,NS
8     X(IS,K)=Y(K)
3     CONTINUE
2     CONTINUE
1     CONTINUE
      DO 10 I=2,N
      IF (X(I,1).LT.X(I-1,1)) WRITE(*,100) I
100   FORMAT(' SHELL SORT: OUT OF ORDER AT I =',I6)
10    CONTINUE
      RETURN
      END

!***********************************************************************
      SUBROUTINE AIR(TC,PMB,RHO,VOLM,CAPP,RLAM,QSAT,EPSI)
! MRR, 09-NOV-88
! 23-OCT-91: MODIFIED TO USE BETTER GAS CONSTANT
!            MOLAR VOLUME VOLM INCLUDED IN ARGUMENT LIST
! AT TEMP T! (DEG C) AND PRESSURE PMB (MB), RETURN:
!   RHO  = AIR DENSITY (KG M-3)
!   VOLM = MOLAR VOLUME (M3 MOL-1)
!   CAPP = SPECIFI! HEAT OF AIR AT CONSTANT PRESSURE (J KG-1 K-1)
!   RLAM = LATENT HEAT FOR WATER (J KG-1)
!   QSAT = SATURATION SPECIFIC HUMIDITY (KG/KG) FROM TETEN FORMULA
```

```
!   EPSI = (RLAM/CAPP)*D(QSAT)/DT
! A,B,C ARE TETEN COEFFS
! RGAS = UNIVERSAL GAS CONSTANT IN J/MOL/K = 8.3143
! RMAIR, RMH2O = KG-MOLECULAR WEIGHTS OF DRY AIR, WATER
! NUMERICAL VALUES FROM WALLACE AND HOBBS
      DATA A /6.106/
      DATA B /17.27/
      DATA C /237.3/
      DATA RGAS /8.3143/
      DATA RMAIR /0.02897/
      DATA RMH2O /0.018016/
! CAPP IN (J/KG/K), RLAM IN (J/KG), RHO IN (KG/M3), VOLM IN (M3/MOL)
      CAPP = 1004
      RLAM = (2501-2.38*TC)*1000
      TK   = 273.16+TC
      PPA  = 100*PMB
      RHO  = RMAIR*PPA/(RGAS*TK)
      VOLM = RGAS*TK/PPA
! ES = SATURATION VAPOUR PRESSURE (MB), QSAT = SAT SPECIFIC HUM (KG/KG)
      ES=A*EXP(B*TC/(C+TC))
      QSAT=(RMH2O/RMAIR)*ES/PMB
      DESDT=ES*B*C/(C+TC)**2
      DQSDT=(RMH2O/RMAIR)*DESDT/PMB
      EPSI=(RLAM/CAPP)*DQSDT
      RETURN
      END

!***************************************************************************
      SUBROUTINE COMSKP(IUNIT)
! MRR, 5-AUG-83
! SKIPS COMMENT LINES IN CONTROL DATA FILE, STOPS IF EOF FOUND
      CHARACTER*1 COM
1     READ(IUNIT,100,END=2) COM
100   FORMAT(A1)
      IF(COM.EQ.'C'.or.com.eq.'c') GOTO 1
      BACKSPACE IUNIT
      RETURN
2     STOP 'CONTROL FILE EOF'
      END

!***************************************************************************
      subroutine phi(z,hc,zMon,phi_w,phi_h)
!     calculates phi_w=sig_w/u*
!     and        phi_h=(k.z/T*)(dT/dz)
!     assume z_ruff = 2.3*hc to obtain matching of T_L = 0.4 at z_ruff
!                  d = 0.75*hc
!     z       height above ground
!     hc      canopy height
!     zMon    Monin-Obukhov length

    if(abs(zMon).lt.0.001) then         !check for L=0
      phi_w=1.25
      phi_h=1.0
      return
    end if

    z_ruff=2.3*hc
    z_d=0.75*hc

    if(z.ge.z_ruff) then
      zeta=(z-z_d)/zMon
    else
      zeta=hc/zMon
!     zeta=(hc-z_d)/zMon
    end if
```

```
    if(zMon.le.0.0) then
      phi_w=1.25*(1.0+3.0*abs(zeta))**0.333
      phi_h=(1.0+16*abs(zeta))**(-0.5)
    else
      phi_w=1.25*(1.0+0.2*zeta)
      phi_h=1.0+5.0*zeta
    end if

    return
    end
!*******************************************************************************

    SUBROUTINE INTERP(NDATA,XDAT,FDATA,X,Y,TH,NUML,NINTV,BREAK,CSCOEF)

! TH = TOP HEIGHT, LEVEL TO WHICH INTERPOLATION SHOULD BE TAKEN
    INTEGER    NDATA

      INTEGER    NINTV, NOUT,N,L
      REAL       BREAK(NDATA), CSCOEF(4,NDATA), CSVAL, &
                 FDATA(NDATA), X(numl),XDATA(NDATA),XDAT(NDATA),Y(numl)
      EXTERNAL   CSAKM, CSVAL, UMACH

    N=NDATA
    XDATA=XDAT
!                                 Compute cubic spline interpolant
      CALL CSAKM (N, XDATA, FDATA, BREAK, CSCOEF)
!                                 Get output unit number
      CALL UMACH (2, NOUT)


      NINTV = NDATA - 1
! calculate spline values between data
!    xd=TH/NUML
     xd=TH/(NUML-1)
     X(1)=0.7
     Y(1)=CSVAL(X(1),NINTV,BREAK,CSCOEF)
!    X(2)=0.5
!    Y(2)=CSVAL(X(2),NINTV,BREAK,CSCOEF)
    do l=2,NUML
!   do l=1,NUML
     X(L)=XD*(L-1)
!    X(L)=XD*L
     Y(L)=CSVAL(X(L),NINTV,BREAK,CSCOEF)
    ENDDO

    RETURN
      END

!*******************************************************************************

    SUBROUTINE INTVAL(Z, DZ, TEMP, CO2, H2O,NLEVL,ZZZ,PROF,TH,NUML,NR)
! EVALUATE AVERAGE INTERPOLATED DATA FOR EACH SOURCE LAYER
    REAL B(9),FS(4,9),X(NUML),Y(NUML),ZZZ(9,3),PROF(9,9,308)
    INTEGER NLEVL(3),KK,I
    EXTERNAL   CSVAL

    DO KK=1,3
    CALL INTERP(NLEVL(KK),ZZZ(:,KK),PROF(:,KK,NR),X,Y,TH,NUML,NV,B,FS)
    SUMV=0
        DO I=1,40
        XV=(Z-DZ)+I*DZ/40
        SUMV=SUMV+CSVAL(XV,NV,B,FS)
        ENDDO
    IF (KK.EQ.1) THEN
    CO2=SUMV/40
    ELSEIF (KK.EQ.2) THEN
```

```
    TEMP=SUMV/40
    ELSEIF (KK.EQ.3) THEN
    H2O=SUMV/40
    ENDIF
    ENDDO

    RETURN
    END

!***************************************************************************

    subroutine CALCE(DER2,DER,PARAM,FD,PAR,SWD,LWDR,DTIME,PA,CK,XPROF,SREF,ak, &
                hcs,CFLUX,eflux,is3,ZZL,ZREF,HFLUX,C13FLUX,O18FLUX,H18FLUX,UHREF, &
                ZPR,IFLAG)
    PARAMETER ISMAX=1000, NUML=20
    REAL    PARAM(5), SREF(6),XPROF(100),DER(NUML,3),DER2(NUML,2),AK(18)
    REAL    eflux(0:ISMAX), &
            HFLUX(0:ISMAX),CFLUX(0:ISMAX),C13FLUX(0:ISMAX), &
             ZPR(0:ISMAX),O18FLUX(0:ISMAX), H18FLUX(0:ISMAX)
    REAL CK(6,NUML),KBF,KDF,LWDR
    REAL*8 DTIME
    integer istop
    external betai,BETA
    data A /6.1375/
    data B /17.502/
    data C /240.97/
    DATA RSTD /0.011894/, R18STD /0.0020672/, AB /7.4/, SMOW /0.0020052/

    DEL_S=-18.
    ISOTOPES=1
    HA=PARAM(1);HB=PARAM(2); DL=0.001
    BETLAI=BETA(HA,HB)
    istop=IS3
    fis3=float(istop)
    is4=10.
    fis4=float(is4)
    IL=20
    dz=(zref-zzl)/fis3/fis4
    ST=1.   ! STOMATA: 1=AMPHISTOMATOUS, 2=HYPOSTOMATOUS
    ALAM=AK(5) !*100.
    VFAC=AK(4)
    KDF=AK(8)
    KBF=AK(9)

    EFLUX(0)=AK(3) !/10.
    HFLUX(0)=AK(2) !/10.
    CFLUX(0)=AK(1) !/10.
    C13FLUX(0)=0.
      DRESP=17.6; DAIR=-8.2
      RAIR=(DAIR/1000.+1.)*RSTD
      C13FLUX(0)=CFLUX(0)*RAIR/(DRESP/1000.+1.)
      D18RESP=15.; D18AIR=-1.
      R18AIR=(D18AIR/1000.+1.)*R18STD
      O18FLUX(0)=CFLUX(0)*R18AIR/(D18RESP/1000.+1.)
      H18FLUX(0)=EFLUX(0)*((DEL_S-ESTAR)/1000.+1.)*SMOW
    ZPR(0)=0.
    AV13C=0.; ASUM=0.
    DO I=1,ISTOP
      ESUM=0.
      HSUM=0.
      CSUM=0.
      C13SUM=0.
      O18SUM=0.
      H18SUM=0.
      DO N=1,is4
        ZAH=ZZL/HCS+(FLOAT(N)/fis4*1./FIS3+FLOAT(I-1)/FIS3)*(ZREF-ZZL)/HCS
```

```
  ZAN=ZAH+(1./FIS4*1./FIS3)*(ZREF-ZZL)/HCS
  ZAL=ZAN*HCS
  IF(ZAL.LE.HCS) THEN
    WI=EXP(PARAM(5)*(ZAH-1))*UHREF
    WIN=EXP(PARAM(5)*(ZAN-1))*UHREF
    RH=1./PARAM(3)*(DL/WI)**0.5
    RHN=1./PARAM(3)*(DL/WIN)**0.5
    RLAI=(ZAH**(HA-1.) *(1-ZAH)**(HB-1.))/BETLAI*PARAM(4)
    RLAIN=(ZAN**(HA-1.) *(1-ZAN)**(HB-1.))/BETLAI*PARAM(4)
    CLAI=BETAI(1.-ZAH,HB,HA)*PARAM(4)
    CLAIN=BETAI(1.-ZAN,HB,HA)*PARAM(4)

CALL INTVAL2(zah*HCS,co,NUML,XPROF(1:NUML),CK(1,:))
CALL INTVAL2(zah*HCS,TA,NUML,XPROF(1:NUML),CK(2,:))
CALL INTVAL2(zah*HCS,WAT,NUML,XPROF(1:NUML),CK(3,:))
CALL INTVAL2(ZAH*HCS,D13C,NUML,XPROF(1:NUML),CK(4,:))
CALL INTVAL2(ZAH*HCS,D18O,NUML,XPROF(1:NUML),CK(5,:))
CALL INTVAL2(ZAH*HCS,D18V,NUML,XPROF(1:NUML),CK(6,:))
  RAIR=D13C/CO
  R18AIR=D18O/CO
  R18VAP=D18V/WAT
  TK=TA+273.15
  ESTAR=2.644-3206./TK+1.5342E6/TK**2
CALL INTVAL2(ZAN*HCS,coN,NUML,XPROF(1:NUML),CK(1,:))
CALL INTVAL2(ZAN*HCS,TAN,NUML,XPROF(1:NUML),CK(2,:))
CALL INTVAL2(ZAN*HCS,WATN,NUML,XPROF(1:NUML),CK(3,:))
CALL INTVAL2(ZAH*HCS,D13CN,NUML,XPROF(1:NUML),CK(4,:))
CALL INTVAL2(ZAH*HCS,D18ON,NUML,XPROF(1:NUML),CK(5,:))
CALL INTVAL2(ZAH*HCS,D18VN,NUML,XPROF(1:NUML),CK(6,:))
  RAIRN=D13CN/CON
  R18AIRN=D18ON/CON
  R18VAPN=D18VN/WATN
  TKN=TAN+273.15
  ESTARN=2.644-3206./TKN+1.5342E6/TKN**2
  CALL INTVAL2(ZAH*HCS,DC,NUML,XPROF(1:NUML),DER(:,1))
  CALL INTVAL2(ZAH*HCS,DT,NUML,XPROF(1:NUML),DER(:,2))
  CALL INTVAL2(ZAH*HCS,DE,NUML,XPROF(1:NUML),DER(:,3))
  CALL INTVAL2(ZAN*HCS,DCD,NUML,XPROF(1:NUML),DER(:,1))
  CALL INTVAL2(ZAN*HCS,DTD,NUML,XPROF(1:NUML),DER(:,2))
  CALL INTVAL2(ZAN*HCS,DED,NUML,XPROF(1:NUML),DER(:,3))
  CALL INTVAL2(ZAH*HCS,DCI,NUML,XPROF(1:NUML),DER2(:,1))
  CALL INTVAL2(ZAN*HCS,DCID,NUML,XPROF(1:NUML),DER2(:,1))
  CALL INTVAL2(ZAH*HCS,DOI,NUML,XPROF(1:NUML),DER2(:,2))
  CALL INTVAL2(ZAN*HCS,DOID,NUML,XPROF(1:NUML),DER2(:,2))

    CALL ASSIM(KBF,KDF,AG,ALAM,VFAC,FD,PAR,DTIME,SWD,LWDR,TA,PA,CO,WAT,CLAI, &
      RH,FC,FH,FE,TL,GSA,CI,DA,RNZ)
    CC=CI-0.1*CO
    HSOU=FH*RLAI/HCS-DT
    CSOU=FC*RLAI/HCS-DC
    ESOU=FE*RLAI/HCS-DE
    DELTA=4.4+22.6*CI/CO
    C13SOU=RAIR/(DELTA/1000.+1.)*CSOU-DCI
    DEL_V=(R18VAP/SMOW-1.)*1000.
    EK=(28./GSA/1.6+19.*RH*0.93*2.)/(1./GSA/1.6+RH*0.93*2.)
    DEL_E = DEL_S + (EK+ESTAR) + (DEL_V-DEL_S-EK)*WAT/(DA+WAT)
    DEL18 = AB + (DEL_E - (R18AIR/R18STD-1.)*1000.)*CC/(CO-CC)
    O18SOU=R18AIR/(DEL18/1000.+1.)*CSOU-DOI
    H18SOU=((DEL_E-ESTAR)/1000.+1.)*SMOW*ESOU
    CALL ASSIM(KBF,KDF,AG,ALAM,VFAC,FD,PAR,DTIME,SWD,LWDR,TAN,PA,CON,WATN,CLAIN, &
      RHN,FC,FH,FE,TL,GSA,CIN,DA,RNZ)
    CCN=CIN-0.1*CON
    HSOUN=FH*RLAIN/HCS-DTD
    CSOUN=FC*RLAIN/HCS-DCD
    ESOUN=FE*RLAIN/HCS-DED
    DELTAN=4.4+22.6*CIN/CON
```

```
          C13SOUN=RAIRN/(DELTAN/1000.+1.)*CSOUN-DCID
          DEL_VN=(R18VAPN/SMOW-1.)*1000.
          EKN=(28./GSA/1.6+19.*RHN*0.93*2.)/(1./GSA/1.6+RHN*0.93*2.)
          DEL_EN = DEL_S + (EKN+ESTARN) + (DEL_VN-DEL_S-EKN)*WATN/(DA+WATN)
          DEL18N = AB + (DEL_EN - (R18AIRN/R18STD-1.)*1000.)*CCN/(CON-CCN)
          O18SOUN=R18AIRN/(DEL18N/1000.+1.)*CSOUN-DOID
          H18SOUN=((DEL_EN-ESTARN)/1000.+1.)*SMOW*ESOUN

          HSUM=HSUM+(HSOU+HSOUN)/2. *DZ
          CSUM=CSUM+(CSOU+CSOUN)/2. *DZ
          ESUM=ESUM+(ESOU+ESOUN)/2. *DZ
          C13SUM=C13SUM+(C13SOU+C13SOUN)/2. *DZ
          O18SUM=O18SUM+(O18SOU+O18SOUN)/2. *DZ
          H18SUM=H18SUM+(H18SOU+H18SOUN)/2. *DZ

          AV13C=AV13C+RAIR/(DELTA/1000.+1.)*CSOU
          ASUM=ASUM+CSOU

        ENDIF
      ENDDO
      ZPR(I)=ZZL+FLOAT(I)/FIS3*(ZREF-ZZL)
      IF(IFLAG.EQ.1) THEN
        EFLUX(I)=ESUM; HFLUX(I)=HSUM; CFLUX(I)=CSUM; C13FLUX(I)=C13SUM
        O18FLUX(I)=O18SUM; H18FLUX(I)=H18SUM; GOTO 100
      ENDIF
      EFLUX(I)=ESUM+EFLUX(I-1)
      HFLUX(I)=HSUM+HFLUX(I-1)
      CFLUX(I)=CSUM+CFLUX(I-1)
      C13FLUX(I)=C13SUM+C13FLUX(I-1)
      O18FLUX(I)=O18SUM+O18FLUX(I-1)
      H18FLUX(I)=H18SUM+H18FLUX(I-1)
    ENDDO
      AV13C=AV13C/ASUM

100 return
    end

!***********************************************************************

    subroutine CALCS(PARAM,FD,PAR,DEF,SWD,LWDR,DTIME,PA,CK,XPROF,SREF,ESATL,thetl,RLAI,RC, &
                RH,RN,ak,hcs,cica,csource,esource,is3,HSOURCE,C13SOURCE,O18SOURCE,H18SOURCE, &
                UHREF,DRESP,DELTA,DEL18,DEL_E,ZSOU)
    PARAMETER ISMAX=1000, NUML=20
    real    PARAM(5),SREF(5),AK(18)
    REAL    esource(0:ISMAX),esatl(0:ISMAX),thetl(0:ISMAX), &
            csource(0:ISMAX),XPROF(100),C13SOURCE(0:ISMAX), &
            HSOURCE(0:ISMAX), RLAI(0:ISMAX),&
            RC(0:ISMAX),RN(0:ISMAX),RH(0:ISMAX), DEF(0:ISMAX),DELTA(0:ISMAX),&
            CICA(0:ISMAX),O18SOURCE(0:ISMAX),DEL18(0:ISMAX),H18SOURCE(0:ISMAX),DEL_E(0:ISMAX)
    REAL CK(6,NUML),KBF,KDF,ZSOU(0:ISMAX),LWDR
    REAL*8 DTIME
    integer istop
    external betai,BETA
    data A /6.1375/
    data B /17.502/
    data C /240.97/
    DATA RSTD /0.011894/, R18STD /0.0020672/, AB /7.4/, SMOW /0.0020052/

    COUNT=0.
    DO IK=1,NUML
        IF(XPROF(IK).LE.HCS) THEN
            COUNT=COUNT+1
        ENDIF
    ENDDO

    DEL_S=-18.
```

```
      ISOTOPES=1
      ST=1.       ! STOMATA: 1=AMPHISTOMATOUS, 2=HYPOSTOMATOUS
      HA=PARAM(1);HB=PARAM(2); DL=0.001
      BETLAI=BETA(HA,HB)
      ALAM=AK(5) !*100.
      VFAC=AK(4)
      KDF=AK(8)
      KBF=AK(9)

      istop=is3
      fis3=float(istop); DZ=1./FIS3
      IL=20
          AVDEL=0.
          CSUM=0.
          NC=0.;X2=0.
      do i=0,istop-1
          ZAH=FLOAT(I)/FIS3
          ZAL=ZAH*HCS
          ZAH=FLOAT(I)/FIS3*COUNT
          J=NINT(ZAH)
          IF(J.EQ.0)THEN; X1=0.;ELSE; X1=XPROF(J); ENDIF
          IF(X1.EQ.X2) THEN; NC=0.; ELSE; NC=NC+1; ENDIF
          IF(J.LT.20)THEN;IF(XPROF(J+1).GT.HCS) THEN; X2=HCS; ELSE; X2=XPROF(J+1); ENDIF;ENDIF
          IF(J.GT.0)THEN;IF(XPROF(J).GT.HCS) THEN; X1=HCS; ENDIF;ENDIF
          ZAH=NC/FIS3*COUNT
          ZAL=X1+(X2-X1)*ZAH
          ZAH=ZAL/HCS
          WI=EXP(PARAM(5)*(ZAH-1))*UHREF
          RH(I)=1./PARAM(3)*(DL/WI)**0.5
          RLAI(I)=(ZAH**(HA-1.) *(1-ZAH)**(HB-1.))/BETLAI*PARAM(4)
          CLAI=BETAI(1.-ZAH,HB,HA)*PARAM(4)

        CALL INTVAL2(ZAL,CO,NUML,XPROF(1:NUML),CK(1,:))
        CALL INTVAL2(ZAL,TA,NUML,XPROF(1:NUML),CK(2,:))
        CALL INTVAL2(ZAL,WAT,NUML,XPROF(1:NUML),CK(3,:))
        CALL INTVAL2(ZAL,D13C,NUML,XPROF(1:NUML),CK(4,:))
        CALL INTVAL2(ZAL,D18O,NUML,XPROF(1:NUML),CK(5,:))
        CALL INTVAL2(ZAL,D18V,NUML,XPROF(1:NUML),CK(6,:))
        RAIR=D13C/CO
        R18AIR=D18O/CO
        R18VAP=D18V/WAT
        TK=TA+273.15
        ESTAR=2.644-3206./TK+1.5342E6/TK**2

          CALL ASSIM(KBF,KDF,AG,ALAM,VFAC,FD,PAR,DTIME,SWD,LWDR,TA,PA,CO,WAT,CLAI, &
              RH(I),FC,FH,FE,TL,GSA,CI,DA,RNZ)
          CC=CI-0.1*CO
          HSOURCE(I)=FH*RLAI(I)/HCS
          CSOURCE(I)=FC*RLAI(I)/HCS
          ESOURCE(I)=FE*RLAI(I)/HCS
          DELTA(I)=4.4+22.6*CI/CO
          CICA(I)=CI/CO
          C13SOURCE(I)=RAIR/(DELTA(I)/1000.+1.)*CSOURCE(I)
          DEL_V=(R18VAP/SMOW-1.)*1000.
          EK=(28./GSA/1.6+19.*RH(I)*0.93*2.)/(1./GSA/1.6+RH(I)*0.93*2.)
          DEL_E(I) = DEL_S + (EK+ESTAR) + (DEL_V-DEL_S-EK)*WAT/(DA+WAT)
          DEL18(I) = AB + (DEL_E(I) - (R18AIR/R18STD-1.)*1000.)*CC/(CO-CC)
          O18SOURCE(I)=R18AIR/(DEL18(I)/1000.+1.)*CSOURCE(I)
          H18SOURCE(I)=((DEL_E(I)-ESTAR)/1000.+1.)*SMOW*ESOURCE(I)

          DEF(I)=DA
          THETL(I)=TL
          RC(I)=1./GSA
          AVDEL=AVDEL+DELTA(I)*CSOURCE(I)
          CSUM=CSUM+CSOURCE(I)
!         RN(0)=AG
```

```
        RN(I)=RNZ
        ZSOU(I)=ZAL
        CONTINUE
    enddo
    AVDEL=AVDEL/CSUM
    DRESP=AVDEL
    WI=UHREF;
    RH(ISTOP)=1./PARAM(3)*(DL/WI)**0.5; RLAI(ISTOP)=0.;
      CALL INTVAL2(HCS,TE,NUML,XPROF(1:NUML),CK(2,:))
      CALL INTVAL2(HCS,WAT,NUML,XPROF(1:NUML),CK(3,:))
      CALL INTVAL2(HCS,CO,NUML,XPROF(1:NUML),CK(1,:))
      CALL INTVAL2(HCS,D18O,NUML,XPROF(1:NUML),CK(5,:))
      CALL INTVAL2(HCS,D18V,NUML,XPROF(1:NUML),CK(6,:))
      R18AIR=D18O/CO
      R18VAP=D18V/WAT
      TK=TA+273.15
      ESTAR=2.644-3206./TK+1.5342E6/TK**2
    CALL ASSIM(KBF,KDF,AG,ALAM,vfac,FD,PAR,DTIME,SWD,LWDR,TA,PA,CO,WAT,CLAI, &
        RH(I),FC,FH,FE,TL,GSA,CI,DA,RNZ)
    CC=CI-0.1*CO
    RN(ISTOP)=RNZ
    DELTA(ISTOP)=4.4+22.6*CI/CO
    CICA(ISTOP)=CI/CO
    DEL_V=(R18VAP/SMOW-1.)*1000.
    EK=(28./GSA/1.6+19.*RH(ISTOP)*0.93*2.)/(1./GSA/1.6+RH(ISTOP)*0.93*2.)
    DEL_E(ISTOP) = DEL_S + (EK+ESTAR) + (DEL_V-DEL_S-EK)*WAT/(DA+WAT)
    DEL18(ISTOP) = AB + (DEL_E(ISTOP) - (R18AIR/R18STD-1.)*1000.)*CC/(CO-CC)
    THETL(ISTOP)=TL
    DEF(ISTOP)=DA
    RC(ISTOP)=1./GSA
    ESATL(ISTOP)=A*EXP(B*(THETL(ISTOP))/(C+THETL(ISTOP)))
    ESOURCE(istop)=0.
    HSOURCE(ISTOP)=0.
    CSOURCE(ISTOP)=0.
    C13SOURCE(ISTOP)=0.
    O18SOURCE(ISTOP)=0.
    H18SOURCE(ISTOP)=0.
    ZSOU(ISTOP)=HCS
    ESOURCE(0)=0.
    HSOURCE(0)=0.
    CSOURCE(0)=0.
    C13SOURCE(0)=0.
    O18SOURCE(0)=0.
    H18SOURCE(0)=0.
    ZSOU(0)=0.

222 RETURN
    END

!***********************************************************************


    SUBROUTINE INTVAL2(Z, INTV,NDATA,XDAT,FDATA)
! EVALUATE AVERAGE INTERPOLATED DATA FOR EACH SOURCE LAYER
    INTEGER    NINTV, N, NDATA
    REAL       BREAK(NDATA), CSCOEF(4,NDATA), CSVAL, &
                XDATA(NDATA),XDAT(NDATA),INTV
    REAL        FDATA(NDATA)
    EXTERNAL   CSAKM, CSVAL

    N=NDATA
    XDATA=XDAT
!                               Compute cubic spline interpolant
      CALL CSAKM (N, XDATA, FDATA, BREAK, CSCOEF)

      NINTV = NDATA - 1
```

```
      INTV=CSVAL(Z,NINTV,BREAK,CSCOEF)  ! GET INTERPOLATED VALUE AT Z

    RETURN
    END

!***********************************************************************


    SUBROUTINE ASSIM(KBF,KDF,AG,ALAM,vfac,FRACD,PAR,DTIME,SWD,LWDR,TEMP,PA,CA,WAT,CLAI,RBH, &
                FC,FH,FE,TL,GS,CI,DA,RNZ)
    IMPLICIT REAL(A-Z)
    PARAMETER (NALPHA=9)
    INTEGER I, ITER
    REAL*8 TIME, DTIME
    REAL ILSUN(NALPHA),ILSUNR(NALPHA),COSALPHA(NALPHA),ILSUNE(NALPHA),JLSUN25(NALPHA), &
        JLSUNT(NALPHA),CISU(NALPHA),AVLSU25(NALPHA),AJLSU25(NALPHA),AVLSUT(NALPHA), &
        AJLSUT(NALPHA),ALSUN(NALPHA),FALPHA(NALPHA),ALPHA(0:NALPHA), &
        ALSU25(NALPHA),GSCSU(NALPHA),GCSU(NALPHA),RSCSU(NALPHA),RSESU(NALPHA),TLSUN(NALPHA), &
        ESATLSU(NALPHA),FHSUN(NALPHA),FESUN(NALPHA),DOSUN(NALPHA)

    LAT=60.5
    PI = 3.1415926
    KC25=40.4*10.;KO25=24800. ! MICHAELIS-MENTON CONSTANTS OF RUBISCO FOR CO2 AND O2
    EAKC=59400.;EAKO=36000.;EAVCMX=64800.
    PCD=0.036   !CANOPY REFLECTION COEFFICIENT FOR DIFFUSE PAR
    PCDR=0.2    ! CANOPY REFLECTION COEFFICIENT FOR DIFFUSE NIR
    KD1=0.719   !diffuse and scattered diffuse PAR extinction coefficient (radians)
    KD=KDF
!   KD=KBF*1.56
!   KDF=KBF*1.56
    FA=0.426    ! FORWARD SCATTERING COEFFICIENT OF PAR IN THE ATMOSPHERE
    NO=150.     ! NITROGEN CONC AT CANOPY TOP (148 FROM WALCROFT ET AL. 1997, 250 from measured
                !   N conc and SLA=2.5 from Bergh et al. 1998, 280 from Kellomaki+Wang)
    KN=0.7      ! NITROGEN EXTINCTION COEFF (DE P + F)
    NB=0.       ! NITROGEN NOT USED IN PHOTOSYNTH (25 FROM DE P + F, -8 from Kellomaki+Wang)
    O2=20900.   ! OXYGEN CONCENTRATION
    SJ=625.9;HJ=256000.;EJ=48000. !parameter for electron transport(SJ,JK-1mol-1;HJ,Jmol-1;EJ is
!*  activation energy,Jmol-1)
    THETA=0.7   ! CURVATURE OF ELECTRON TRANSPORT WITH IRRADIANCE, DAVID 0.7, WALCROFT 0.9
    GSTR25=3.69*10. ! GAMMA STAR, CO2 COMPENSATION POINT, DAVID GIVES 3.69, WALCROFT 4.27
    ERKC=66400. ! EFFECT OF TEMP ON RESP
    PO=1013     ! ATM PRESSURE AT SEA LEVEL
    SIGMA=0.15  ! LEAF SCATTERING COEFFICIENT OF PAR
    ALPHA_L=PI/3.   !AVERAGE LEAF ANGLE
    AA=6.1375   ! CONSTANTS FOR SATURATION VAPOUR PRESSURE
    BB=17.502
    CC=240.97
    PAS=PA*100.     ! PRESSURE IN PASCALS
    TK=TEMP+273.15  ! TEMPERATURE IN K
    CP=1005.+(TEMP+23.15)**2/3364.  ! SPECIFIC HEAT CAPACITY OF AIR AT CONSTANT PRESSURE
    RLAM=(2501.-2.38*TEMP)*1000.    ! LATENT HEAT OF VAPOURISATION
    RMH2O=0.018     ! MOLECULAR WEIGHT OF WATER
    RMAIR=0.028965  ! AVERAGE MOLECULAR WEIGHT OF AIR
    RGAS=8.3143     ! IDEAL GAS CONSTANT
    RHO=RMAIR*PAS/RGAS/TK   ! AIR DENSITY
    VOLM=RGAS*TK/PAS        ! MOLAR VOLUME OF AIR
    EMISS=0.98      ! LEAF EMISSIVITY
    STEFBOL=5.67E-8 ! STEFAN-BOLTZMANN CONSTANT
    TIME=DTIME-36891.   ! TIME IN DAY OF YEAR PLUS FRACTIONAL HOUR
    CA=CA       ! CO2 IN UMOL/MOL
    CICA=0.6    ! INITIAL CI/CA
    CI=CICA*CA      ! INTERCELLULAR CO2
    CISUV=CI; CISUJ=CI; CISDJ=CI
    LAI=CLAI    ! CUMULATIVE LEAF AREA INDEX AT THIS LEVEL
    LC=4.3      ! TOTAL CANOPY LEAF AREA INDEX
    A2=0.425        ! FRACTION OF PAR ABSORBED BY PSII, TANIA USES 0.3, WALCROFT 0.2
```

```
    G0=0.01     ! INTERCEPT OF GC RELATION (MOL/M2/S) FROM LEUNING 1995
    A1=4.4      ! SLOPE OF GC RELATION LEUNING 1995
    D0=15.      ! VAPOUR PRESSURE DEFICIT PARAMETER FOR GC RELATION LEUNING 1995
    LAMBDA=ALAM ! GS MODEL LAMBDA
    FAC=vfac        ! FACTOR TO TWEAK VCMAX
    SWABS=0.2       ! ABSORPTION COEFF FOR NEAR IR
    KCOR=SWABS**0.5 ! K CORRECTION FOR SWABS
    KLW=KDF     ! EXT. COEFF. FOR DOWNWARD LW
    KL=KDF
    LWG=EXP(-KL*(LC-LAI))*EMISS*STEFBOL*273.15**4
    LWGA=KL*LWG
    LWD=-EMISS*STEFBOL*TK**4*(EXP(-KL*LAI)-1.)
    LWDA=-KL*EMISS*STEFBOL*TK**4*EXP(-KL*LAI)
    LWU=-EMISS*STEFBOL*TK**4*(EXP(-KL*(LC-LAI))-1.)
    LWUA=-KL*EMISS*STEFBOL*TK**4*EXP(-KL*(LC-LAI))
    LWL=EMISS*STEFBOL*TK**4*0.5 !FACTOR 0.5 FOR PROJECTED LA
!   LWDR=EMISS*STEFBOL*(TK-20.)**4

! LEAF ANGLES
    ALPHA(0)=0.
    DO I=1,NALPHA
        ALPHA(I)=PI/2./NALPHA*FLOAT(I)
        FALPHA(I)=-COS(ALPHA(I))+COS(ALPHA(I-1))
        COSALPHA(I)=0.5*(SIN(ALPHA(I))**2-SIN(ALPHA(I-1))**2)/FALPHA(I)
    ENDDO

!*RAD converts degrees into radians:
      RAD    = PI/180.
!*Sine and cosine of latitude:
      SINLAT = SIN(RAD*LAT)
      COSLAT = COS(RAD*LAT)
!*Maximal sine of declination:
      SINDCM = SIN(RAD*23.4)
!*Sine and cosine of declination
      SINDEC = -SINDCM*COS(2.*PI*(TIME+10.)/365.)
      COSDEC = SQRT(1.-SINDEC*SINDEC)
!*The terms A and B
      A      = SINLAT*SINDEC
      B      = COSLAT*COSDEC
!*Daylength
      DAYL   = 12.*(1.+(2./PI)*ASIN(A/B) )
!*Generating hour from day
      HOUR  = (TIME-IDINT(TIME))*24.
!*Determining solar noon (Iqbal, 1983) (TOO) and
!*    hour angle of sun (H) for specified location
!      PARAMETER LS=150.;LES=147.34
      GAD=2.*PI*(TIME-1.)/365.
      ET=0.017+0.4281*COS(GAD)-7.351*SIN(GAD)- &
         3.349*COS(2.*GAD)-9.371*SIN(2.*GAD)
!      TOO=12.+(4.*(LS-LES)-ET)/60.
      TOO=12.97
      H=PI*(HOUR-TOO)/12.
!*Sine of solar height:
      SINB=A+B*COS(H)
!*Solar constant fluctuations (W/m2):
      SC     = 1367. *(1.+0.033*COS(2.*PI*(TIME-10.)/365.))
!! INTEGRAL OF SINE OF SOLAR HEIGHT OVER THE HOUR OF THIS PROFILE
      SININT = A + B*12./PI*(SIN(PI/12.*(HOUR-TOO))-SIN(PI/12.*(HOUR-1.-TOO)))
!! RADIATION IN KJ/M2/HR FROM THAT IN W/M2
      RDD=max(SWD*3.6,0.)
!! HOURLY AVERAGE ATMOS. TRANS.
       ATMTR  = RDD/(SININT*3.6*SC)
!*diurnal radiation on top of canopy(micromol m-2s-1)
       S      = MAX(0.,ATMTR*SC*SINB*0.5*4.55)
! RADIATION AT TOP OF CANOPY FROM MEASURED PAR (MICROMOL/M2/S)
      S = MAX(0.,PAR)
```

```
      SR=MAX(0.,SWD)
      SLW=MAX(0.,LWDR)
!*beam radiation extinction coefficient of canopy(radians)
      KB=0.5/SINB
      KB=KBF/SINB
    PSWH=(1.-SWABS**0.5)/(1.+SWABS**0.5)
    PSWB=1.-EXP(-2.*PSWH*KB/(1.+KB))
    PSWD=0.38
!*beam and scattered beam PAR extinction coefficient (radians)
      KB1=0.46/SINB
      KB1=KBf*(1.-SIGMA)**0.5/SINB
      KD1=KDF*(1.-SIGMA)**0.5
!*canopy reflection coefficient for beam PAR
      PCB=1.-EXP(-2.*0.041*KB/(1.+KB))
      PCBR=1.-EXP(-2.*0.382*KB/(1.+KB))
      PCDR=1.-EXP(-2.*0.382*KD/(1.+KD))
!*the optical air mass
      M=(PA/PO)/SINB
!*fraction of diffuse radiation
      FD=(1.-ATMTR**M)/(1.+ATMTR**M*(1./FA-1.))
      FD=MIN(FRACD,1.)
!*Diffuse PAR per unit ground area (umolm-2s-1)
      ID=FD*S
      IDR=FD*(SR-S/4.55)
!*Beam PAR per unit ground area (umolm-2s-1)
      IB=(1.-FD)*S
      IBR=(1.-FD)*(SR-S/4.55)
! DIFFUSE PAR AT LEAF LEVEL
      IDL=(1.-pcd)*KD1*ID*EXP(-KD1*LAI)
      IDLR=(1.-pcdr)*KD*KCOR*IDR*EXP(-KD*KCOR*LAI)
      IDLW=KLW*SLW*EXP(-KLW*LAI)
! REFLECTED PAR AND NIR
      IDPAR=0.5*KL*(1.-sigma)**0.5*IB*EXP(-KB1*LC)*EXP(-KL*(1.-sigma)**0.5*(LC-LAI))
      IDNIR=0.5*KL*(swabs)**0.5*IBR*EXP(-KB*KCOR*LC)*EXP(-KL*(swabs)**0.5*(LC-LAI))
! SCATTERED PAR AT LEAF LEVEL
      IBS=IB*((1.-pcd)*KB1*EXP(-KB1*LAI)-(1.-sigma)*KB*EXP(-KB*LAI))
      IBSR=IBR*(KB*KCOR*EXP(-(1.-pcdr)*KB*KCOR*LAI)-swabs*KB*EXP(-KB*LAI))
!*Total canopy absorption
      IC=(1.-PCB)*IB*(1.-EXP(-KB1*LAI))+ &
         (1.-PCD)*ID*(1.-EXP(-KD1*LAI))
!*PAR absorbed by the sunlit leaves FOR WHOLE CANOPY
      ICSUN=IB*(1.-0.15)*(1.-EXP(-KB*LAI))+ &
            ID*(1.-PCD)*(1.-EXP(-(KD1+KB)*LAI))*KD1/(KD1+KB) &
            +IB*((1.-PCB)*(1.-EXP(-(KB1+KB)*LAI))*KB1/(KB1+KB) &
            -(1.-0.15)*(1.-EXP(-2.*KB*LAI))/2.)
! PAR ABSORBED BY SUNLIT LEAVES FOR INDIVIDUAL LEVELS
      DO I=1,NALPHA
        ILSUN(I)=(1.-SIGMA)*IB*COSALPHA(I)/SINB+IDL+IBS+IDPAR
        ILSUNR(I)=SWABS*IBR*COSALPHA(I)/SINB+IDLR+IBSR
      ENDDO
!*PAR absorbed by the shaded leaves
      ICSHD=IC-ICSUN
! PAR ABSORBED BY SHADED LEAVES FOR INDIVIDUAL LEAVES
      ILSHD=(IDL+IBS+IDPAR)
      ILSHDR=(IDLR+IBSR)
!*PAR effectively absorbed by the sunlit and shaded leaves
      ICSUNE=ICSUN*A2
      ICSHDE=ICSHD*A2
      ILSUNE=ILSUN*A2
      ILSHDE=ILSHD*A2
!*Total PAR effectively absorbed by the canopy
      ICTOTE=ICSUNE+ICSHDE
!*CANOPY NITROGEN DISTRIBUTION AND PHOTOSYNTHETIC CAPACITY
!*leaf nitrogen concentration per unit leaf area (mmolm-2)
!*for soybean
      NL= 148.
```

```
!*CANOPY PHOTOSYNTHETIC CAPACITY (at 25 oC)
!*photosynthetic rubisco capacity per unit leaf area (umolm-2s-1) 116 FOR SOYBEAN
! value of 32.7 from Walcroft et al. 1997 for Pinus radiata PER TOTAL LEAF AREA
! (20 AT 20 DEG C CONVERTS TO 32.7 AT 25 DEG C ACCORDING TO THEIR EQUATION)
! but this is 98 on a projected leaf area basis OR 65 PER ONE-SIDED LA
! WHICH IS ~ THE SAME AS PROJECTED LA FOR FIR AND SPRUCE COMPARED TO PINE
! KELLOMAKI+WANG 2000 GET XN25=0.18 AT 20 DEG, FOR N IN MMOL/PROJECTED LA
! REGRESSION IN WALCROFT HAS SLOPE 0.212
! factor 0.5 for post-winter recovery (rough guess from Bergh et al. 1998)
      VL25=65*0.5 ! this value not used - see below
!*ratio of measured Rubisco capacity to leaf N, 0.21 FROM WALCROFT ET AL. 1997
      XN25=VL25/(NL-NB)
      XN25=0.2*1.64*FAC ! 1.64 FACTOR TO CONVERT TO 25 DEG
                        ! FACTOR FAC FOR POST-WINTER RECOVERY
!*photosynthetic capacity of sunlit and shaded leaves
      EK=MAX(0.,(1.-EXP(-KN-KB*LAI)))
      VC25=LAI*XN25*(NO-NB)*(1.-EXP(-KN))/KN
      VCSU25=LAI*XN25*(NO-NB)*EK/(KB*LAI+KN)
      VCSD25=VC25-VCSU25
      NL=(NO-NB)*EXP(-KN*LAI/LC)+NB
!     NL=NO*EXP(-KN*LAI/LC)
      NL=NO
      VL25=XN25*(NL-NB)
!*maximum rate of electron transport per unit leaf area (umolm-2s-1)
      JMSU25=2.1*VCSU25
      JMSD25=2.1*VCSD25
! MAX ELECTRON TRANSPORT AT LEAF LEVEL AT 25 DEG ratio from Walcroft et al. 1997
      JML25=2.1*VL25
!*CANOPY PHOTOSYNTHESIS
!*The Farquhar et al.(1980.1982) equations
!*Effect of temperature on VL(leaf) AND VC(canopy)
!      VCT=VC25*EXP((1./298.-1./(TEMP+273.))*EAVCMX/8.314)
      VLT=VL25*EXP((1./298.-1./(TEMP+273.))*EAVCMX/8.314)
      VLT=VL25*EXP(26.35-65330/8.314/(TEMP+273.15)) !NEW PHOTOSYNTH PARAMS BERNACCHI ET AL. 2001
!      VCSUNT=VCSU25*EXP((1./298.-1./(TEMP+273.))*EAVCMX/8.314)
!      VCSHDT=VCSD25*EXP((1./298.-1./(TEMP+273.))*EAVCMX/8.314)
!*Effect of temperature on Michaelis-Menten constant of rubisco
!*     for O2 and CO2 (Pa)
      KO=KO25*EXP((1./298.-1./(TEMP+273.))*EAKO/8.314)
      KC=KC25*EXP((1./298.-1./(TEMP+273.))*EAKC/8.314)
      KO=100.*EXP(20.3-36380/8.314/(TEMP+273.15))   !NEW PHOTOSYNTH PARAMS BERNACCHI ET AL. 2001
      KC=EXP(38.05-79430/8.314/(TEMP+273.15))   !NEW PHOTOSYNTH PARAMS BERNACCHI ET AL. 2001
!*Effect of temperature on gammastar
      GMSTAR=GSTR25+(0.188*(TEMP-25.)+0.0036*(TEMP-25.)**2.)*10.
      GMSTAR=EXP(19.02-37830/8.314/(TEMP+273.15))   !NEW PHOTOSYNTH PARAMS BERNACCHI ET AL. 2001
! MAX ELECTRON TRANSPORT AT LEAF LEVEL AND TEMP
      JMLT=JML25*EXP((((TEMP+273.2)/298.2-1.)*EJ)/(8.3144*( &
           TEMP+273.2)))*(1.+EXP((298.2*SJ-HJ)/(298.2*8.3144)))/ &
           (1.+EXP((SJ*(TEMP+273.2)-HJ)/(8.3144*(TEMP+273.2))))
! ELECTRON TRANSPORT AT LEAF LEVEL AND TEMP
    DO I=1,NALPHA
      JLSUNT(I)=(ILSUNE(I)+JMLT-SQRT(((ILSUNE(I)+JMLT)**2.)-(4.*THETA* &
            ILSUNE(I)*JMLT)))/(2.*THETA)
    ENDDO
      JLSHDT=(ILSHDE+JMLT-SQRT(((ILSHDE+JMLT)**2.)-(4.*THETA* &
            ILSHDE*JMLT)))/(2.*THETA)
!*Assimilation rate limited by rubisco (Av) and
!*limited by electron transport (Aj) at 25 oC
      K25=40.4*(1.+O2/24800.)
      CISU(1:NALPHA)=CI
      CISD=CI
! FRACTION OF LEAVES IN SUN
      FSUN=EXP(-KB*LAI)
!*RESPIRATION (at 25 oC)
!*Leaf and canopy respiration (at 25 oC),umolm-2s-1
!*for soybean
```

```
      RL25=VL25*0.0089
!      RC25=(VC25*RL25/VL25)
!*Effect of temperature on respiration
!      PARAM ERKC=66400.
      RLT=RL25*EXP((1./298.-1./(TEMP+273.))*ERKC/8.314)
      RLT=RL25*EXP(18.73-46390/8.314/(TEMP+273.15)) !NEW PHOTOSYNTH PARAMS BERNACCHI ET AL. 2001
!      RCT=RC25*EXP((1./298.-1./(TEMP+273.))*ERKC/8.314)
      K=KC*(1.+O2/KO)
! FIND VAPOUR PRESSURE DEFICIT
        ESAT=AA*EXP(BB*TEMP/(CC+TEMP))
        DA=ESAT-WAT
! SLOPE OF VP WITH TEMP
        SL=BB*CC/(CC+TEMP)**2*ESAT
! CHANGE OF LATENT HEAT WITH SENSIBLE HEAT
        EPSILON=SL*RLAM/CP/PA
! GAMMA AT TEMP
        GMT=(RLT/VLT*K+GMSTAR)/(1.-RLT/VLT)
! EFFECTIVE BOUNDARY LAYER RESISTANCE FOR HEAT
        RBHSTAR=1./(1./(RBH*VOLM)+4.*STEFBOL*EMISS*TK**3/(CP*RHO*VOLM))
        H=1./(RBH*0.93*2.*VOLM+EPSILON*RBHSTAR)
! LEAF TO AIR VPD WHEN STOMATA ARE FULLY CLOSED
    DO I=1,NALPHA
        DOSUN(I)=DA+EPSILON*RBHSTAR/RLAM*(ILSUN(I)/4.55+ILSUNR(I)+ &
                IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR)
        DOSUN(I)=DOSUN(I)*1000.
    ENDDO
        DOSHD=DA+EPSILON*RBHSTAR/RLAM*(ILSHD/4.55+ILSHDR+ &
                IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR)
        DOSHD=DOSHD*1000.
!  IF(ITER.GT.0) THEN
    IF(DA.LT.0.) THEN
      DA=1. !; WAT=ESAT
    ENDIF

!ANALYTICAL SOLUTION FOR ASSIMILATION
    RBC=RBH*1.3*2.*VOLM
    DO I=1,NALPHA
      DSQRT=SQRT(1.6*DOSUN(I)*(CA-GMT)/LAMBDA)
      AAV=RBC-1.6/H
      BBV=DSQRT-CA-K+(RLT-VLT)*(RBC-1.6/H)
      CCV=(RLT-VLT)*(DSQRT-CA)-VLT*GMSTAR-K*RLT
      SQV=BBV**2-4.*AAV*CCV
      IF(SQV.LT.0) THEN
        AJLSUT(I)=0.
      ELSE
        AVLSUT(I)=(-BBV-SQRT(BBV**2-4.*AAV*CCV))/2./AAV
      ENDIF
      AAJ=4.*(RBC-1.6/H)
      BBJ=4.*(DSQRT-CA-2.*GMSTAR)+(4.*RLT-JLSUNT(I))*(RBC-1.6/H)
      CCJ=(4.*RLT-JLSUNT(I))*(DSQRT-CA)-8.*RLT*GMSTAR-JLSUNT(I)*GMSTAR
      SQJ=BBJ**2-4.*AAJ*CCJ
      IF(SQJ.LT.0) THEN
        AJLSUT(I)=0.
      ELSE
        AJLSUT(I)=(-BBJ-SQRT(BBJ**2-4.*AAJ*CCJ))/2./AAJ
      ENDIF
      ALSUN(I)=MIN(AVLSUT(I),AJLSUT(I))
    ENDDO
      DSQRT=SQRT(1.6*DOSHD*(CA-GMT)/LAMBDA)
      AAV=RBC-1.6/H
      BBV=DSQRT-CA-K+(RLT-VLT)*(RBC-1.6/H)
      CCV=(RLT-VLT)*(DSQRT-CA)-VLT*GMSTAR-K*RLT
      SQV=BBV**2-4.*AAV*CCV
      IF(SQV.LT.0.) THEN
        AVLSDT=0.
      ELSE
```

```
        AVLSDT=(-BBV-SQRT(BBV**2-4.*AAV*CCV))/2./AAV
      ENDIF
      AAJ=4.*(RBC-1.6/H)
      BBJ=4.*(DSQRT-CA-2.*GMSTAR)+(4.*RLT-JLSHDT)*(RBC-1.6/H)
      CCJ=(4.*RLT-JLSHDT)*(DSQRT-CA)-8.*RLT*GMSTAR-JLSHDT*GMSTAR
      SQJ=BBJ**2-4.*AAJ*CCJ
      IF(SQJ.LT.0) THEN
        AJLSDT=0.
      ELSE
        AJLSDT=(-BBJ-SQRT(BBJ**2-4.*AAJ*CCJ))/2./AAJ
      ENDIF
      ALSHD=MIN(AVLSDT,AJLSDT)

    DO I=1,NALPHA
        GSCSU(I)=(ALSUN(I))/(SQRT(MAX(1.6/LAMBDA*DOSUN(I)*(CA-GMT),0.))-1.6*(ALSUN(I))/H)
        GCSU(I)=1./(1./(GSCSU(I))+RBC)
        IF(GCSU(I).GT.0.) THEN
          CISU(I)=CA-ALSUN(I)/GCSU(I)
        ELSE
          CISU(I)=CA
        ENDIF
    ENDDO
        GSCSD=(ALSHD)/(SQRT(MAX(1.6/LAMBDA*DOSHD*(CA-GMT),0.))-1.6*(ALSHD)/H)
        GCSD=1./(1./(GSCSD)+RBC)
        IF(GCSD.GT.0.) THEN
          CISD=CA-ALSHD/GCSD
        ELSE
          CISD=CA
        ENDIF

! NET ASSIMILATION
      SUM=0.
    DO I=1,NALPHA
      SUM=SUM+ALSUN(I)*FALPHA(I)
    ENDDO
      ALTNET=SUM*FSUN+ALSHD*(1.-FSUN)

! BOUNDARY LAYER AND STOMATAL RESISTANCE FOR CO2 IN S/M
    GSUM=0.; CISUM=0.
    DO I=1,NALPHA
        GSUM=GSUM+GSCSU(I)*FALPHA(I)*VOLM
        RSCSU(I)=1./(GSCSU(I)*VOLM)
        CISUM=CISUM+CISU(I)*FALPHA(I)
        RSESU(I)=RSCSU(I)/1.6
    ENDDO
        AVGSCSU=GSUM
      RBC=RBH*1.3*2.; RSCSD=1./(GSCSD*VOLM)
      GS=AVGSCSU*FSUN+1./RSCSD*(1.-FSUN)
      CI=CISUM*FSUN+CISD*(1.-FSUN)
    IF(DA.EQ.0.) THEN
        RSCSU=0.; RSCSD=0.
    ENDIF
! BOUNDARY LAYER AND STOMATAL RESISTANCE FOR H2O
      RBE=RBH*0.93*2.; RSESD=RSCSD/1.6
! SATURATION VAPOUR DENSITY AT AIR TEMP + SLOPE WITH TEMP
      ESAT=AA*EXP(BB*TEMP/(CC+TEMP))*rmh2o/VOLM/1000.
      SL=BB*CC/(CC+TEMP)**2*esat
! LEAF TEMPERATURE FOR SHADE LEAVES
    LWL=EMISS*STEFBOL*(TLSHD+273.15)**4*0.5 !FACTOR 0.5 FOR PROJECTED LA
      TLSHD=TEMP + (ILSHD/4.55+ILSHDR+IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR - &
             RLAM*(ESAT-WAT*rmh2o/volm/1000.)/(RBE+RSESD))/ &
            (RLAM*SL/(RBE+RSESD) + RHO*CP/RBH + 4.*EMISS*STEFBOL*TK**3)
      TLSHD=MAX(TLSHD,0.);TLSHD=MIN(TLSHD,30.)
! SAT VAPOUR DENS AT LEAF TEMP FOR SHADE LEAVES
      ESATLSD=AA*EXP(BB*TLSHD/(CC+TLSHD))*rmh2o/VOLM/1000.
      TLSHD=TEMP + (ILSHD/4.55+ILSHDR+IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR - &
```

```
            RLAM*(ESATLSD-WAT*rmh2o/volm/1000.)/(RBE+RSESD))/ &
            (RHO*CP/RBH + 4.*EMISS*STEFBOL*TK**3)
      TLSHD=MAX(TLSHD,0.);TLSHD=MIN(TLSHD,30.)
! LEAF TEMPERATURE FOR SUN LEAVES
    DO I=1,NALPHA
    LWL=EMISS*STEFBOL*(TLSUN(I)+273.15)**4*0.5  !FACTOR 0.5 FOR PROJECTED LA
      TLSUN(I)=TEMP + (ILSUN(I)/4.55+ILSUNR(I)+IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR - &
            RLAM*(ESAT-WAT*rmh2o/volm/1000.)/(RBE+RSESU(I)))/ &
            (RLAM*SL/(RBE+RSESU(I)) + RHO*CP/RBH + 4.*EMISS*STEFBOL*TK**3)
      TLSUN(I)=MAX(TLSUN(I),0.); TLSUN(I)=MIN(TLSUN(I),30.)
! SAT VAPOUR DENS AT LEAF TEMP FOR SUN LEAVES
      ESATLSU(I)=AA*EXP(BB*TLSUN(I)/(CC+TLSUN(I)))*rmh2o/VOLM/1000.
      TLSUN(I)=TEMP + (ILSUN(I)/4.55+ILSUNR(I)+IDLW+LWGA+(LWDA+LWUA)+IDPAR/4.55+IDNIR - &
            RLAM*(ESATLSU(I)-WAT*rmh2o/volm/1000.)/(RBE+RSESU(I)))/ &
            (RHO*CP/RBH + 4.*EMISS*STEFBOL*TK**3)
      TLSUN(I)=MAX(TLSUN(I),0.); TLSUN(I)=MIN(TLSUN(I),30.)
    ENDDO
! AVERAGE LEAF TEMPERATURE
    TLSUM=0.
    DO I=1,NALPHA
        TLSUM=TLSUM+TLSUN(I)*FALPHA(I)
    ENDDO
      TL=TLSUM*FSUN+TLSHD*(1.-FSUN)
! HEAT FLUX (K M/S) AND WATER FLUX (MMOL/MOL M/S) FOR SUN LEAVES
    DO I=1,NALPHA
      FHSUN(I)=1./RBH*(TLSUN(I)-TEMP)
      FESUN(I)=1./(RBE+RSESU(I))*(ESATLSU(I)/RMH2O*VOLM*1000.-WAT)
    ENDDO
! HEAT FLUX (K M/S) AND WATER FLUX (MMOL/MOL M/S) FOR SHADE LEAVES
      FHSHD=1./RBH*(TLSHD-TEMP)
      FESHD=1./(RBE+RSESD)*(ESATLSD/RMH2O*VOLM*1000.-WAT)
! TOTAL HEAT (K M/S), WATER (MMOL/MOL M/S) AND CO2 (MICROMOL/MOL M/S) FLUX AT LEAF LEVEL
    HSUM=0.;ESUM=0.
    DO I=1,NALPHA
      HSUM=HSUM+FHSUN(I)*FALPHA(I)
      ESUM=ESUM+FESUN(I)*FALPHA(I)
    ENDDO
      FH=HSUM*FSUN+FHSHD*(1.-FSUN)
      FE=ESUM*FSUN+FESHD*(1.-FSUN)
      FC=-ALTNET*VOLM
! AVAIL ENERGY AT GROUND ASSUMING ALL PAR INTERCEPTED AND ALL ELSE NOT
      AG=SWD-PAR/4.55
! AVAIL ENERGY AT LEVEL Z
      ID=FD*S
      IB=(1.-FD)*S
      IDL=(1.-pcd)*ID*EXP(-KD1*LAI)
      IBS=IB*((1.-pcd)*EXP(-KB1*LAI)-(1.-sigma)*EXP(-KB*LAI))
      IDPAR=0.5*IB*EXP(-KB1*LC)*EXP(-KL*(1.-sigma)**0.5*(LC-LAI))
      ISUN=((1.-PCB)*IB+IDL+IBS-IDPAR)
      ISHD=(IDL+IBS-IDPAR)
      FSUN=EXP(-KB*LAI)
      IDR=FD*(SR-S/4.55)
      IBR=(1.-FD)*(SR-S/4.55)
      IDLR=(1.-pcdr)*IDR*EXP(-KD*KCOR*LAI)
      IBSR=IBR*((1.-pcdr)*EXP(-KB*KCOR*LAI)-swabs*EXP(-KB*LAI))
      IDLW=SLW*EXP(-KLW*LAI)
      IDNIR=0.5*IBR*EXP(-KB*KCOR*LC)*EXP(-KL*(swabs)**0.5*(LC-LAI))
      ISUNR=((1.-PCDR)*IBR+IDLR+IBSR+IDLW-IDNIR)
      ISHDR=(IDLR+IBSR+IDLW-IDNIR)
      FSUNR=EXP(-KB*LAI)
      RNZ=FSUNR*(ISUNR-LWU+LWD-LWG)+(1.-FSUNR)*(ISHDR-LWU+LWD-LWG)+ &
            (FSUN*ISUN+(1.-FSUN)*ISHD)/4.55

      RETURN
      END
```

```
!*************************************************************************

      subroutine TURB2 (UST,SWR,A6,B,z,ZMON, sw,TL)
!-------------------------------------------------------------------------
! Find turbulence statistics sw and TL at height z
! For sw,  use aa = sw/ustar:
!   when zhc > zhc1:     aa = aa1
!   when 1 < zhc < zhc1: aa varies linearly between aah and aa1
!   when 0 < zhc < 1:    aa varies exponentially with coefficient csw
! For TL, use cc = TL*ustar/hc
!   when zhc > 1:        cc = max (vonk*(z-disp)/(hc*aa1**2), cch)
!   when zhc0 < zhc < 1: cc = cch
!   when 0 < zhc < zhc0: cc varies linearly from 0 to cch
! 22-oct-99 (MRR): rewritten from version in SOURCE10
!-------------------------------------------------------------------------
      REAL AA1,AAH,A0,CCH,CC0,CSW,ALFAB,ALFAC,HC,SWREF,USTAR
      common /t/ ITURB,AA1,AAH,A0,CCH,CC0,CSW,ALFAB,ALFAC, &
            HC,SWREF,USTAR,IST_COR
      vonk  = 0.4                                ! von Karman constant
    DISP  = 0.75
    ZHC1  = 1.0
    ZHC0  = 0.15
    ZHC2  = -0.1
    ZHC3  = 0.1
      zhc   = z/hc
    ustar = swr/aa1
    c1    = vonk/aa1
    c0    = 0.3
    C00   = 0.0
    A=A6 !*100.
      if (zhc.lt.0) stop 'TURB: zhc < 0'
      call phi(z,hc,zMon,phi_w,phi_h)           !stability function
! Find sw:
    if(iturb.eq.1) then     ! exponential profile
        if (zhc.gt.zhc1) then
          aa  = aa1                                ! well above hc
        else if (zhc.ge.DISP) then
          aa  = aah + (zhc-DISP)*(aa1-aah)/(zhc1-DISP)   ! just above hc
        else
        aa  = aah * exp(csw*(zhc-DISP))             ! below hc
        end if
      sw    = aa * ustar
    elseif (iturb.eq.2) then    !linear profile
      if (zhc.gt.zhc1) then
        aa = aa1
      elseIF (ZHC.GT.ZHC2) THEN
        aa = a0 + (aa1-a0)*(zhc/zhc1)
      ELSEIF (ZHC.GT.ZHC3) THEN
        AA = 0.1 + (ZHC2*AA1+(1.-ZHC2)*A0-0.1)*(ZHC-ZHC3)/ZHC2
      ELSE
        AA = 0.00001 + (0.1 - 0.00001)*ZHC/ZHC3
!       AA = 0.01
      endif
    sw = aa*ustar
    elseif (iturb.eq.3) then    ! polynomial profile
      if(zhc.lt.1.05) then
        aa = -1.2344*zhc**3 + 1.698*zhc**2 + 0.6938*zhc + 0.099
      else
        aa = aa1
      endif
    sw = aa*ustar
    elseif(iturb.eq.4) then     ! cosine profile
      if(zhc.gt.1) then
        aa = aa1
      else
        aa = (aa1+a0)/2 + (aa1-a0)/2*cos(3.14159*(1-zhc))
```

```
      endif
    sw = aa*ustar
    else
     stop 'incorrect iturb'
    endif
    if(ist_cor.eq.1)    SW = ustar*aa*phi_w/1.25
! Find TL:
      if (zhc.gt.2) then
        cc1 = vonk*(zHC-disp)/(aa1**2)          ! above z=hc
        cc  = max(cc1,cch)
      else
        cc  = cch * min(1.0,zhc/zhc0)            ! below z=hc
      end if
    if(iturb.eq.5) then
      cc = c1*(zhc-disp) + (c0+c1*disp)*exp(-c1*zhc/(c0+c1*disp))
    elseif(iturb.eq.0) then
        cc = cc0 + (cch-cc0)*(zhc-zhc0)/(1.-zhc0)
      if(zhc.lt.zhc0) then
        cc = C00+(cc0-C00)*zhc/zhc0
      endif
    ELSEIF(ITURB.EQ.3) THEN
      IF(A.GT.100.) THEN
        IF(ZHC.EQ.0) THEN
          CC=0.
        ELSE
          CC=B
        ENDIF
      ELSE
        CC = B*(1.-EXP(-A*ZHC))/(1.-EXP(-A))
      ENDIF
    endif
    if(ist_cor.eq.1)    cc=cc*1.25**2/(phi_w**2 * phi_h)
      TL    = cc * (hc/ustar)

      return
      end

!*******************************************************************************

      SUBROUTINE DISPM2(US,SW,A,B,timdat,NMAX,nlevl,NREF,M0,M,Z,ZC,zMon, &
                      DIS,DISN,DISF,IWRITE)

! CALCULATE DISPERSION MATRIX DIS(I,J), SUCH THAT:
!   C(I)-C(NREF) = SUM (J=M0,M) (DIS(I,J)*S(J)*DZ(J))
! NEAR-FIELD AND FAR-FIELD PARTS OF DIS RETURNED SEPARATELY IN ARRAYS
! DISN AND DISF. PARAMETER NMAX SETS FIRST DIMENSION OF DIS ARRAYS.
! ARRAY ZC(N):  HOLDS HEIGHTS AT WHICH CONCENTRATIONS C(I) ARE GIVEN.
!               REFERENCE LEVEL IS I=NREF, SO ZREF=ZC(NREF).
! ARRAY Z(0:M): HOLDS HEIGHT LIMITS OF SOURCE LAYERS, WITH Z(M)=HC.
! CALCULATIONS EXTEND ONLY OVER SOURCES (J=M0,M), WHERE M0 IS 0 OR 1.
! ELEMENT J=0 CORRESPONDS TO GROUND-LEVEL SOURCE. FOR J>=1, UNIT SOURCE
! UNIFORMLY FILLS LAYER J, BETWEEN Z(J-1 AND Z(J)

    integer nint
      PARAMETER(NINT=50)
      REAL DIS(NMAX,0:M),DISN(NMAX,0:M),DISF(NMAX,0:M), &
               Z(0:M),ZC(nlevl),RINT(nint),ZI(nint), &
               zs(-1:nint),dz(-1:nint)
      character timdat*32
      HC=Z(M)
      ZREF=ZC(NREF)

! LOOP THROUGH INDICES OF ARRAY DIS(I,J)
      DO 1 J=M0,M                      ! LEVEL OF UNIT SOURCE
    IF(J.EQ.0) THEN
       ZS(J)=0
```

```
        DZ(J)=0
    ELSE
        zs(j)=(Z(J)+Z(J-1))/2
        dz(j)=Z(J)-Z(J-1)
    ENDIF
    call turb2(US,SW,A,B,zs(j),Zmon,swj,tlj)
      RLJ=SWJ*TLJ
      DO 2 I=1,nlevl                        ! LEVEL OF CONCENTRATION
! FIRST GET FAR-FIELD PART DISF, ASSUMING ZREF>HC
      IF (ZC(I).LT.ZREF) THEN          ! ZC(I) BELOW ZREF
        DO K=1,nint
          ZI(K) = ZC(I) + (ZREF-ZC(I))*FLOAT(K-1)/FLOAT(NINT-1)
          CALL TURB2(US,SW,A,B,ZI(K),zMon,SW,TL)
          RKF=SW**2*TL                 ! FAR-FIELD DIFFUSIVITY
      IF(J.GT.0) THEN
          IF (ZI(K).LE.Z(J-1)) THEN    ! GET FLUX FROM UNIT SOURCE
            FLUX=0                     ! FLUX BELOW SOURCE LAYER
          ELSE IF (ZI(K).LE.Z(J)) THEN
            FLUX=(ZI(K)-Z(J-1))/dz(j)    ! FLUX WITHIN SOURCE LAYER
          ELSE
            FLUX=1                     ! FLUX ABOVE SOURCE LAYER
          END IF
      ELSE
          FLUX=1
      ENDIF
          RINT(K)=FLUX/RKF
        END DO
        CALL INTEGR(ZI,RINT,NINT,AREAF)
      ELSE IF (ZC(I).EQ.ZREF) THEN     ! ZC(I) AT ZREF
        AREAF=0
      ELSE                             ! ZC(I) ABOVE ZREF
        DO K=1,NINT
          ZI(K) = ZREF + (ZC(I)-ZREF)*FLOAT(K-1)/FLOAT(NINT-1)
          CALL TURB2(US,SW,A,B,ZI(K),zMon,SW,TL)
          RKF=SW**2*TL                 ! FAR-FIELD DIFFUSIVITY
          FLUX=1                       ! ALWAYS TRUE SINCE ZREF>=HC
          RINT(K)=FLUX/RKF
        END DO
        CALL INTEGR(ZI,RINT,NINT,AREAF)
        AREAF=-AREAF                   ! SIGN CHANGE BECAUSE ZREF BELOW
      END IF
      DISF(I,J)=AREAF
! NOW GET NEAR-FIELD PART DISN
      IF (J.EQ.0) THEN                 ! GROUND SOURCE: NO NEAR FIELD
        DISN(I,J)=0
      ELSE                             ! ELEVATED SOURCE
        T2=RKN((ZC(I)+zs(j))/RLJ)/SWJ    ! C AT IMAGE Z
        T3=RKN((ZREF-zs(j))/RLJ)/SWJ     ! C AT ZREF
        T4=RKN((ZREF+zs(j))/RLJ)/SWJ     ! C AT IMAGE ZREF
        IF (ZC(I).GT.Z(J-1).AND.ZC(I).LE.Z(J)) THEN
!                                        ! C AT Z: ZC(I) IN SOURCE LAYER
          ZJI=(Z(J-1)+ZC(I))/2           ! JI IS MEAN OF Z(J-1),ZC(I)
          ZIJ=(ZC(I)+Z(J))/2             ! IJ IS MEAN OF ZC(I),Z(J)
          CALL TURB2(US,SW,A,B,ZJI,zMon,SWJI,TLJI)
          CALL TURB2(US,SW,A,B,ZIJ,zMon,SWIJ,TLIJ)
          T1JI=TLJI*RKNINT((ZC(I)-Z(J-1))/(SWJI*TLJI))
          T1IJ=TLIJ*RKNINT((Z(J)-ZC(I))/(SWIJ*TLIJ))
          T1=(T1JI+T1IJ)/dz(j)
        ELSE                             ! C AT Z: ZC(I) OUTSIDE SOURCE
          T1=RKN((ZC(I)-zs(j))/RLJ)/SWJ
        END IF
        DISN(I,J)=T1+T2-T3-T4
      END IF

! ADD DISF AND DISN TO GET DIS(I,J)
      DIS(I,J)=DISF(I,J)+DISN(I,J)
```

```
2     CONTINUE
1     CONTINUE

! WRITE DISPERSION MATRIX
    IF(IWRITE.EQ.1) THEN
      WRITE(4,300) timdat
300   FORMAT(1x,a32,' ARRAY ((DIS(I,J),J=M0,M),I=1,nlevl-1),  zc')
      write(4,310) nlevl,m
310   FORMAT(2i5)
      DO I=1,nlevl
        WRITE(4,330) (DIS(I,J),J=M0,M), zc(i)
      END DO
      write(4,320) m-m0+1
320   format(' M levels for source mid-heights with thickness:-'/i4)
      do j=1,m
        write(4,330) zs(j), dz(j)
      end do
      write(4,*)
330   FORMAT(1X,100F8.3)
    ENDIF

      RETURN
      END

!************************************************************************
```

Appendix H

Regional model source code

H.1 Correction of CO_2 profiles

The following Fortran 77 code applies a correction to the CO_2 concentration data according to the calibration gases (interpolated between heights at which calibration occurs).

```
c  Program to correct aircraft Licor CO2 data for drift in calibration
c  gas, using both 340ppm and 380ppm calibrations.
c  Written by Jon Lloyd early 1999, altered by Julie Styles for use
c  with uneven numbers of 340 and 380ppm calibrations 25/2/99.
c  First the csv file from the flight needs to be set up with time in
c  general format, and 34 in the time field before each set of five
c  340ppm calibration values, and 38 before each set of 380ppm
c  calibration values, and 99 before the sample values. Rows should be
c  deleted to leave only the last five valid calibration values, and to
c  allow time for the sample values to be valid (about twenty rows)
c  The file should be saved as csv, and the name should be altered
c  below to call the correct file.
    program flights
      implicit doubleprecision (a-h,o-z)
        real*8 co2cal(20000), co2raw,flow,press(20000),pirga(20000)
      real*8 praw, rh(20000),tair(20000), tirga(20000)
      real*8  cal(2,100), digtime(20000), ctime(2,100), cvalue(2)
      real*8 time,pr,dens,te,pt,virpt,wat,elev
      open (unit=11,file=
     x  'c:\siberia\flight data\Corrected data\z240798f6.intf')
      open (unit=16,file=
     x  'c:\siberia\flight data\corrected data\z240798f6.corr')
      open (unit=12, file=
     x  'c:\siberia\flight data\corrected data\temperature
     x\z240798f6.tempout')
        j = 1
      ical1 = 0
      ical2 = 0
      read (12,*,end=45) time,pr,dens,te,pt,virpt,wat,elev
      close (unit=12)
45      continue
130   do 30 i = 1,20000
      read (11,*,end=40) digtime(j), co2cal(j), co2raw, flow,
     x   press(j), pirga(j), praw, rh(j), tair(j), tirga(j),
     x   traw

1234    format (3x,6(i2,x),7x,f7.3,2x,f7.2,2x,f9.6,2x,f7.2,2x,f6.2,2x,
     x   f7.2,2x,3(f7.4,2x),f7.2)
```

```
      goto 51
131   continue
51      intday = int(digtime(j))
        if (intday.lt.100.and.intday.gt.30) goto 50
        digtime(j) = (digtime(j)-float(intday))*24.
        if (digtime(j).lt.1) then
            digtime(j)=digtime(j)+24
        endif
      j = j+ 1
30      continue
      do i=1,20000
        tair(i)=tair(i+60)
      enddo
50      continue
      if (intday.eq.38) then
      ical2 = ical2 + 1
      do 60 k = 1,4
        read (11,*) digtim , co2ca, co2ra, flow,
     x   pres, pirg, pra, rhum, ta, tirg,
     x   traw
      cal(2,ical2) = co2ca/4. + cal(2,ical2)
      if (k.eq.3) then
      intday = int(digtim)
        ctime(2,ical2) = (digtim-float(intday))*24.
        endif
60      continue
      write (6,*)  ctime(2,ical2), cal(2,ical2)
      goto 130
      else if (intday.eq.34) then
       ical1 = ical1 + 1
      do 70 k = 1,4
        read (11,*) digtim , co2ca, co2ra, flow,
     x   pres, pirg, pra, rhum, ta, tirg,
     x   traw
      cal(1,ical1) = co2ca/4. + cal(1,ical1)
      if (k.eq.3) then
      intday = int(digtim)
        ctime(1,ical1) = (digtim-float(intday))*24.
        endif
70      continue
      write (6,*) ctime(1,ical1), cal(1,ical1)
      goto 130
      else if (intday.eq.99) then
        goto 130
      else
        close (unit = 11)
        pause
      endif
40      continue
        nobs = j
       numcals: if (ical1.eq.ical2) then
      ncal = ical1
        if (ctime(1,ical1).lt.1) then
            ctime(1,ical1)=ctime(1,ical1)+24
        endif
        if (ctime(2,ical2).lt.1) then
            ctime(2,ical2)=ctime(2,ical2)+24
        endif
      do 10 i = 1,nobs
      do 10 iseg = 2,ncal
        timesegments: if (digtime(i).lt.ctime(2,iseg).and.digtime(i).gt.
     x     ctime(2,iseg-1)) then
      do ispan = 1,2
         cvalue(ispan) = cal(ispan,iseg-1) + (cal(ispan,iseg) -
     x    cal(ispan,iseg-1))/(ctime(ispan,iseg) - ctime(ispan,iseg-1))*
     x      (digtime(i) - ctime(ispan,iseg-1))
```

```
      enddo
      call calculate
      else if (digtime(i).gt.ctime(2,ical2)) then timesegments
        if (iseg.eq.2) then
            cvalue(1) = cal(1,ical1)
            cvalue(2) = cal(2,ical2)
      call calculate
        endif
      endif timesegments
10      continue
c the above only works if there are an equal number of 340ppm and 380ppm
c  calibrations - the next sequence does it if there are more 340ppm
c  calibrations than 380ppm ones
           else if (ical1.gt.ical2) then numcals
          ncal = ical1
      do 20 i = 1,nobs
      do 20 iseg = 2,ncal
        timesegs: if (digtime(i).lt.ctime(2,iseg).and.digtime(i).gt.
     x     ctime(2,iseg-1).and.iseg.le.ical2) then
      do ispan = 1,2
         cvalue(ispan) = cal(ispan,iseg-1) + (cal(ispan,iseg) -
     x    cal(ispan,iseg-1))/(ctime(ispan,iseg) - ctime(ispan,iseg-1))*
     x      (digtime(i) - ctime(ispan,iseg-1))
      enddo
      call calculate
      else if (digtime(i).gt.ctime(2,ical2).and.digtime(i).lt.
     x            ctime(1,ical1).and.digtime(i).lt.ctime(1,iseg)
     x          .and.digtime(i).gt.ctime(1,iseg-1)) then timesegs
         cvalue(1) = cal(1,iseg-1) + (cal(1,iseg) -
     x    cal(1,iseg-1))/(ctime(1,iseg) - ctime(1,iseg-1))*
     x      (digtime(i) - ctime(1,iseg-1))
            cvalue(2) = cal(2,ical2)
      call calculate
      else if (digtime(i).gt.ctime(1,ical1)) then timesegs
        if (iseg.eq.ncal) then
            cvalue(1) = cal(1,ical1)
            cvalue(2) = cal(2,ical2)
      call calculate
        endif
      endif timesegs
20  continue
     write (6,*) ical1, ical2
        endif numcals
        stop
    contains
    subroutine calculate
            co2new1 = co2cal(i) + 340. - cvalue(1)
          co2new2 = co2cal(i) + 380 - cvalue(2)
            co2new = (co2new1 + co2new2)/2.
          height = (1 - (press(i)/(pr*10))**(1/5.2568))*44308
           tk = tair(i) + 273.15
             abshum = 2.17/tk*rh(i)/100*613.75*exp(17.502*tair(i)/
     x         (240.97 + tair(i)))
           rho = press(i)/10/0.287/tk
           pottemp = tk*(1000./press(i))**0.286
           spechum = abshum/(rho*1000 + abshum)
           vpt = pottemp*(1 + 0.61*spechum)
           airmol = press(i)*100./ (8.314*tk)
             watervap = abshum/18./airmol*1e3
c        write (6,1345) digtime(i), cvalue(1),cvalue(2),co2cal(i)
c     x    , co2new,tair(i),pottemp-273.15,vpt-273.15,watervap, height
        write (16,1345) digtime(i), cvalue(1),cvalue(2),co2cal(i)
     x    , co2new, tair(i),pottemp-273.15,vpt-273.15,watervap, height
1345    format (f9.5, 4f8.3,3f8.2,f8.3, f9.2)
c        write (6,*) i, iseg,nobs, ical1, ical2
      end subroutine calculate
```

```
      end program flights
```

H.2 Interpolation and smoothing of CO_2 profiles

The following Fortran 77 code linearly interpolates CO_2 concentrations at heights where data are missing, and average concentrations are calculated for each 25-metre interval up to the highest measured height.

```
      program co2
      implicit none
      real(8) hourtime(20000), cal1(20000),cal2(20000)
      real(8) halftime,maxheight,height(20000),
     x    co2uncorr(20000),co2corr(20000)
      real(8) temp(20000),ptemp(20000),vpt(20000)
      real(8) h2o(20000),co2down(120,9000)
      real(8) co2up(120,9000),sumco2up(120),avco2up(120)
      real(8) avco2down(120),sumco2down(120)
      integer i,j,k,l,ni,nj,m,nk,nup,mi,num(120),numd(120),
     x     n,nm,ij,ik,il
     open (unit=10, file=
      x'c:\siberia98\flight data\corrected data\
      xz240798f6.corr')
     open (unit=20, file=
      x'c:\siberia98\flight data\corrected data\co2\
      xz240798f6.co2')

      sumco2up=0
      sumco2down=0
      num=1
      numd=1
      do i=1,20000
      read (10,*,end=30) hourtime(i),cal1(i),cal2(i),co2uncorr(i)
     x   ,co2corr(i),temp(i),ptemp(i),vpt(i),h2o(i),height(i)
      enddo
30    continue

c  ditch bad data before finding maximum height
      do mi=1,20000
      if (height(mi).gt.3500) then
        height(mi)=0
      endif
      enddo
      maxheight=maxval(height)

      do m=1,20000
      if (height(m).eq.maxheight) then
        halftime=hourtime(m)
    write(6,*) hourtime(m),maxheight
      endif
      enddo

      do nk=1,20000
    upflight:  if (hourtime(nk).lt.halftime) then
      do nup=1,120
      if (height(nk).lt.nup*25.and.height(nk).ge.(nup-1)*25.and.
     x    height(nk).ne.0.and.co2corr(nk).ne.0) then
      co2up(nup,num(nup))=co2corr(nk)
      num(nup)=num(nup)+1
```

```
      endif
      enddo
    endif upflight
    downflight: if (hourtime(nk).ge.halftime) then
      do ni=1,120
      if (height(nk).lt.ni*25.and.height(nk).ge.(ni-1)*25.and.
     x    height(nk).ne.0) then
      co2down(ni,numd(ni))=co2corr(nk)
      numd(ni)=numd(ni)+1
      endif
      enddo
    endif downflight
      enddo

      do j=1,9000
      do k=1,120
      sumco2up(k)=sumco2up(k)+co2up(k,j)
      sumco2down(k)=sumco2down(k)+co2down(k,j)
      enddo
      enddo
      do nj=1,120
      avco2up(nj)=sumco2up(nj)/(num(nj)-1)
      avco2down(nj)=sumco2down(nj)/(numd(nj)-1)
      if (num(nj).eq.1) then
      avco2up(nj)=0
      endif
      if (numd(nj).eq.1) then
      avco2down(nj)=0
      endif
      enddo
      call interpolate

      do l =1,120
      write(6,*) (l-1)*25, '  to' ,l*25, avco2up(l)
     x    ,avco2down(l),num(l),numd(l)
      write(20,1111) (l-1)*25,l*25,avco2up(l),
     x   avco2down(l)
      enddo
1111      format (2i10,2f10.4)

      contains
      subroutine interpolate
c Interpolate linearly between heights for missing data
      do n=2,120
      nm=1
      ij=1
    upzeros: if (avco2up(n).eq.0.and.avco2up(n-1).ne.0) then
       do ik=1,120-n
       if (avco2up(n+nm).eq.0) then
        nm=nm+1
       endif
       if ((n+nm+1).eq.120) then
       goto 24
       else if (avco2up(n+nm).ne.0) then
       avco2up(n)=(avco2up(n+nm)-avco2up(n-1))/(nm+1)
     x     +avco2up(n-1)
24     endif
       enddo
      endif upzeros
    downzeros: if (avco2down(n).eq.0.and.avco2down(n-1).ne.0) then
       do il=1,120-n
       if (avco2down(n+ij).eq.0) then
        ij=ij+1
       endif
       if ((n+ij+1).eq.120) then
       goto 26
```

```
       else if (avco2down(n+ij).ne.0) then
       avco2down(n)=(avco2down(n+ij)-avco2down(n-1))/(ij+1)
     x    + avco2down(n-1)
26     endif
       enddo
      endif downzeros
      enddo
      end subroutine interpolate
      end program co2
```

H.3 Virtual potential temperature and height calculation

The following Fortran 77 code applies a correction to the temperature and humidity data due to sensor response lag. It calculates virtual potential temperature using humidity and pressure data and height using pressure data as described in Appendix E.

```
    program TEMPERATURE

! program to calculate potential temperature, vpt, h2o conc and
elevation ! for the full flight profile - the CO2 calibration
program "flights" ! cuts out times when calibration was occuring.

      implicit doubleprecision (a-h,o-z)
      real*8 co2cal(20000), co2raw,flow,press(20000),pirga(20000)
      real*8 praw, rh(20000),tair(20000), tirga(20000),height(20000),difftemp(20000)
      real*8  cal(2,100), digtime(20001), ctime(2,100), cvalue(2),maxtime(1)
      open (unit=11,file='Z230798f2.TEMPIN')
      open (unit=16,file='z230798f2.tempout')
130   do i = 1,20000
      read (11,*,end=40) digtime(i), co2cal(i), co2raw, flow, &
        press(i), pirga(i), praw, rh(i), tair(i), tirga(i), &
        traw
      enddo
40    continue
      do i=1,20000
            digtime(i)=(digtime(i)-int(digtime(i)))*24
          height(i) = (1 - (press(i)/(press(1)))**(1/5.2568))*44308
      enddo
    do 50 i=1,20000
    if(digtime(i).eq.0) goto 50
      if (tair(i+64).ne.0.and.i.gt.64) then
           sumt=0.
           sumrh=0.
         do k=56,64
           sumt=tair(i+k)+sumt
           sumrh=rh(i+k)+sumrh
         enddo
           avt=sumt/9
           avrh=sumrh/9
           sumt=0
           sumrh=0
         do k=56,64
           sumt=tair(i-k)+sumt
           sumrh=rh(i-k)+sumrh
         enddo
           tav=sumt/9
           rhav=sumrh/9
             difftemp(i)=avt-tav
```

```
          diffrh=avrh-rhav
          difftime=(digtime(i+60)-digtime(i-60))*3600
          tav2=(tair(i-1)+tair(i)+tair(i+1))/3
          rhav2=(rh(i-1)+rh(i)+rh(i+1))/3
          tempcorr = tav2+273.15 + difftemp(i)/difftime*60
          rhcorr = rhav2 + diffrh/difftime*90
      elseif(i.le.64) then
        tempcorr=tair(i+60)+273.15
      elseif(tair(i+64).eq.0) then
        tempcorr=tair(i)+273.15-0.6
      endif
            if (tempcorr.ne.0) then
          tk = tair(i) + 273.15
          abshum = 2.17/tk*rh(i)/100*613.75*exp(17.502*tair(i)/ &
             (240.97 + tair(i)))
            abshum2 = 2.17/tempcorr*rh(i)/100*613.75* &
             exp(17.502*(tempcorr-273.15)/ &
             (240.97 + tempcorr-273.15))
          rho = press(i)/10./0.287/tk
          rho2 = press(i)/10./0.287/tempcorr
          pottemp = tk*(1000./press(i))**0.286
          pottemp2 = tempcorr*(1000./press(i))**0.286
          spechum = abshum/(rho*1000 + abshum)
          spechum2 = abshum2/(rho2*1000 + abshum2)
          virttemp=tk*(1.+0.61*spechum)
          virttemp2=tempcorr*(1.+0.61*spechum)
          vpt = pottemp*(1 + 0.61*spechum)
          vpt2 = pottemp2*(1 + 0.61*spechum)
          airmol = press(i)*100./ (8.314*tk)
          airmol2=press(i)*100./(8.314*tempcorr)
            watervap = abshum/18./airmol*1e3
            watervap2 = abshum2/18./airmol2*1e3
          cp = 1005+((virttemp2-250.)**2)/3364
          height2 = cp*vpt2/9.81* &
                     (1-(press(i)/press(1))**(287.05/cp))
    if(press(i).gt.0) then
          height3 = -287.05*virttemp2/9.81*log(press(i)/press(1))
    endif
          write (16,1345) digtime(i),press(i)/10.,rho,tair(i), &
           pottemp-273.15, &
           vpt-273.15,watervap, &
            tempcorr-273.15, pottemp2-273.15, vpt2-273.15, &
            height(i),height2,height3,rho2
          endif
50 continue
      rmaxdifftemp=maxval(difftemp)
      write(*,*) rmaxdifftemp
1345      format (f9.5, f9.3,f10.5,4f8.3,1x,f8.3,1x,f8.3,1x,f8.3,
&
            3f8.2,f10.5)

    end program TEMPERATURE
```

H.4 Density profiles

The following Fortran 77 code calculates average density for each 25-metre interval up to the highest measurement height, using density output from the code in §H.3 above.

```
      program density
```

```
      implicit none
      real(8) hourtime(20000), press(20000),rho(20000)
      real(8) halftime,maxheight,height(20000)
      real(8) temp(20000),ptemp(20000),vpt(20000),
     x    ctemp(20000),cptemp(20000),cvpt(20000),
     x    height2(20000),height3(20000),ch2o(20000),rho2(20000)
      real(8) h2o(20000),densdown(120,9000)
      real(8) densup(120,9000),sumdensup(120),avdensup(120),num(120)
      real(8) avdensdown(120),sumdensdown(120),numd(120)
      integer i,j,k,l,ni,nj,m,nk,nup,mi
    open (unit=10, file=
     x'c:\siberia98\flight data\corrected data\temperature\
     xz240798f6.tempout')
    open (unit=20, file=
     x'c:\siberia98\flight data\corrected data\temperature\
     xz240798f6.rho')

      sumdensup=0
      sumdensdown=0
      num=1
      numd=1
      do i=1,20000
      read (10,*,end=30) hourtime(i),press(i),rho(i),
     x   temp(i),ptemp(i),vpt(i),h2o(i),ctemp(i),cptemp(i),
     x   cvpt(i),height(i),height2(i),height3(i),rho2(i),ch2o(i)
      enddo
30    continue

c  ditch bad data before finding maximum height
      do mi=1,20000
      if (height(mi).gt.3500) then
        height(mi)=0
      endif
      enddo
      maxheight=maxval(height)

      do m=1,20000
      if (height(m).eq.maxheight) then
        halftime=hourtime(m)
    write(6,*) hourtime(m),maxheight
      endif
      enddo

      do nk=1,20000
    upflight:  if (hourtime(nk).lt.halftime) then
      do nup=1,120
      if (height(nk).lt.nup*25.and.height(nk).ge.(nup-1)*25.and.
     x    height(nk).ne.0) then
      densup(nup,num(nup))=rho2(nk)
      num(nup)=num(nup)+1
      endif
      enddo
    endif upflight
    downflight: if (hourtime(nk).ge.halftime) then
      do ni=1,120
      if (height(nk).lt.ni*25.and.height(nk).ge.(ni-1)*25.and.
     x    height(nk).ne.0) then
      densdown(ni,numd(ni))=rho2(nk)
      numd(ni)=numd(ni)+1
      endif
      enddo
    endif downflight
      enddo

      do j=1,9000
      do k=1,120
```

```
      sumdensup(k)=sumdensup(k)+densup(k,j)
      sumdensdown(k)=sumdensdown(k)+densdown(k,j)
      enddo
      enddo
      do nj=1,120
      avdensup(nj)=sumdensup(nj)/(num(nj)-1)
      avdensdown(nj)=sumdensdown(nj)/(numd(nj)-1)
      if (num(nj).eq.1) then
      avdensup(nj)=0
      endif
      if (numd(nj).eq.1) then
      avdensdown(nj)=0
      endif
      enddo

      do l =1,120
      write(6,*) (l-1)*25, '  to' ,l*25, avdensup(l)
     x    ,avdensdown(l),num(l),numd(l)
      write(20,1111) (l-1)*25,l*25,avdensup(l),
     x   avdensdown(l)
      enddo
1111      format (2i10,2f8.4)
      end program density
```

H.5 Virtual potential temperature profiles

The following Fortran 77 code calculates average virtual potential temperature for each 50-metre interval up to the highest measurement height using the virtual potential temperature output from §H.3 above.

```
$debug
      program avtemp
      implicit none
      real(8) hourtime(20000), press(20000),rho(20000)
      real(8) halftime,maxheight,height(20000)
      real(8) temp(20000),ptemp(20000),vpt(20000),
     x     ctemp(20000),cptemp(20000),cvpt(20000),height2(20000),
     x     height3(20000)
      real(8) h2o(20000),tempdown(2,60,9000)
      real(8) tempup(2,60,9000),sumtempup(2,60),avtempup(2,60),
     x          num(60)
      real(8) avtempdown(2,60),sumtempdown(2,60),numd(60)
      integer i,j,k,l,ni,nj,m,nk,nup,mi
    open (unit=10, file=
     x'c:\siberia98\flight data\corrected data\temperature\
     xz220798f1.tempout')
    open (unit=20, file=
     x'c:\siberia98\flight data\corrected data\temperature\
     xz220798f1.pt')

      sumtempup=0
      sumtempdown=0
      num=1
      numd=1
      do i=1,20000
      read (10,*,end=30) hourtime(i),press(i),rho(i),
     x   temp(i),ptemp(i),vpt(i),h2o(i),ctemp(i),cptemp(i),
     x   cvpt(i),height(i),height2(i),height3(i)
```

```
      enddo
30    continue

c  ditch bad data before finding maximum height
      do mi=1,20000
      if (height(mi).gt.3500) then
        height(mi)=0
      endif
      enddo
      maxheight=maxval(height)

      do m=1,20000
      if (height(m).eq.maxheight) then
        halftime=hourtime(m)
    write(6,*) hourtime(m),maxheight
      endif
      enddo

      do nk=1,20000
        if(cptemp(nk).gt.-50) then    !ditch bad data
    upflight:  if (hourtime(nk).lt.halftime) then
      do nup=1,60
      if (height(nk).lt.nup*50.and.height(nk).ge.(nup-1)*50.and.
     x    height(nk).ne.0) then
          tempup(1,nup,num(nup))=cptemp(nk)
          tempup(2,nup,num(nup))=cvpt(nk)
          num(nup)=num(nup)+1
      endif
      enddo
    endif upflight
    downflight: if (hourtime(nk).ge.halftime) then
      do ni=1,60
      if (height(nk).lt.ni*50.and.height(nk).ge.(ni-1)*50.and.
     x    height(nk).ne.0) then
          tempdown(1,ni,numd(ni))=cptemp(nk)
          tempdown(2,ni,numd(ni))=cvpt(nk)
          numd(ni)=numd(ni)+1
      endif
      enddo
    endif downflight
        endif
      enddo

      do j=1,9000
      do k=1,60
      do l=1,2
      sumtempup(l,k)=sumtempup(l,k)+tempup(l,k,j)
      sumtempdown(l,k)=sumtempdown(l,k)+tempdown(l,k,j)
      enddo
      enddo
      enddo
      do nj=1,60
      do l=1,2
      avtempup(l,nj)=sumtempup(l,nj)/(num(nj)-1)
      avtempdown(l,nj)=sumtempdown(l,nj)/(numd(nj)-1)
      if (num(nj).eq.1) then
      avtempup(l,nj)=0
      endif
      if (numd(nj).eq.1) then
      avtempdown(l,nj)=0
      endif
      enddo
      enddo

      do l =1,60
      write(6,*) (l-1)*50, '  to' ,l*50, avtempup(1,l)
```

```
     x    ,avtempdown(1,l),num(l),numd(l)
      write(20,1111) (l-1)*50,l*50,avtempup(1,l),
     x   avtempdown(1,l),avtempup(2,l),avtempdown(2,l)
      enddo
1111      format (2i10,4f8.4)
      end program avtemp
```