Technical Report No. 10

The Global Atmospheric Tracer Model TM2

Martin Heimann
Max-Planck-Institut für Meteorologie
Bundesstrasse 55, D-20146 Hamburg, Germany

Edited by:
Deutsches Klimarechenzentrum,
Modellbetreuungsgruppe,
Hamburg, October 1996

ISSN 0940-9327

Contents

1. SUMMARY PAGE

1.1 SHORT DESCRIPTION AND MAIN AUTHORS

TM2 is a three-dimensional atmospheric transport model which solves the continuity equation for an arbitrary number of atmospheric tracers on an Eulerian grid spanning the entire globe. It is driven by stored meteorological fields from analyses of a weather forecast model or from output of an atmospheric general circulation model. Tracer advection is calculated using the “slopes scheme” of Russell and Lerner [1981]. Vertical transport due to convective clouds is computed using a simplified version of the cloud mass flux scheme of Tiedke [1989]. Turbulent vertical transport is calculated by stability dependent vertical diffusion according to the scheme by Louis [1979].

1.1.1 Numerical Solution

The spatial structure of the model is a regular latitude-longitude grid and a sigma coordinate system in the vertical dimension. The base “coarse grid” version of the model uses a horizontal resolution of approximately 8^o latitude by 10^o longitude and 9 layers in the vertical dimension. The base timestep of the “coarse grid” version is four hours. In each base time step the model calculates the source and sink processes affecting each tracer, followed by the calculation of the transport processes.

1.1.2 Model History

The TM2 tracer model originated from the tracer model code developed at the Goddard Institute for Space Studies [Russell and Lerner, 1981]. This original model was based on the meteorology and monthly vertical convection statistics of the GISS general circulation climate model. A revised model version, TM1, included the preprocessing packages for arbitrary meteorological fields [Heimann and Keeling, 1989]. The present version, TM2, represents a largely rewritten code with additional new packages for the calculation of the subgridscale vertical transport by convective clouds and turbulent diffusion.

2. MODEL DESCRIPTION

2.1 MODEL PHYSICS AND DYNAMICS

2.1.1 Continuity equation

The atmospheric tracer model TM2 solves the continuity equation for a tracer in flux form

$$\frac{\partial}{\partial t}\rho\chi + \nabla \cdot \rho\vec{u}\chi = Q \tag{2.1.1.1}$$

on an Eulerian three-dimensional grid spanning the entire globe. χ denotes the tracer mixing ratio (in kg tracer mass per kg air mass), ρ the air density, $\vec{u}$ the wind velocity and Q the volume source/sink of the tracer. Denoting time-space averages over the model gridelements and model time step by overbars and deviations from these averages with primes, we obtain the equation for the averaged quantities:

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} + \nabla \cdot \bar{\rho}\bar{\vec{u}}\bar{\chi} + \nabla \cdot \bar{\rho}\overline{\vec{u}'\chi'} = \bar{Q} \tag{2.1.1.2}$$

where we have neglected density variations within the averaging volumes. Separating the horizontal and vertical directions we get

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} + \nabla_h \cdot \bar{\rho}\overline{\vec{u}_h}\bar{\chi} + \frac{\partial}{\partial z}\bar{\rho}\bar{w}\bar{\chi} + \nabla_h \cdot \bar{\rho}\overline{\vec{u}'_h\chi'} + \frac{\partial}{\partial z}\bar{\rho}\overline{w'\chi'} = \bar{Q} \tag{2.1.1.3}$$

where the subscript *h* refers to horizontal vectors.

The second and third term on the left hand side of equation (2.1.1.3) correspond to transport scales that are resolved on the model grid and are termed “advection” (see Section 2.1.4). The fourth and fifth term represent the effect of the unresolved scales on the volume averaged tracer mixing ratio. Assuming that no cross correlation exists between vertical and horizontal deviations from the volume averaged quantities we treat these terms independently. In the troposphere, for which the model is primarily designed, this assumption is justified, but it may not be appropriate in the stratosphere. The fourth term, called “horizontal diffusion”, represents changes induced by correlations between the horizontal deviations of the wind and the tracer concentration from their grid averages (see Section 2.1.6). The last term on the left hand side of equation (2.1.1.3) represents subgridscale transport in the vertical direction and is termed “vertical convection”. It is parameterized in the model by transport induced by cumulus clouds and by turbulent, stability dependent vertical diffusion (see Section 2.1.5).

Numerically, the model solves the continuity equation (2.1.1.3) in flux form and by splitting the transport operator in time. Formally, we may view the tracer masses in all the gridboxes as the components of a large vector, $\boldsymbol{n}(t)$. The discrete version of equation (2.1.1.3) is then written as

$$\boldsymbol{n}(t + \Delta t) = \boldsymbol{L}_{vconvec}\boldsymbol{L}_{hdiff}\boldsymbol{L}_{advec}(\boldsymbol{n}(t) + \boldsymbol{Q}(t)\Delta t) \tag{2.1.1.4}$$

where the discrete operators $\boldsymbol{L}_{advec}$, $\boldsymbol{L}_{hdiff}$ and $\boldsymbol{L}_{vconvec}$, corresponding to the transport processes as described above, operate sequentially on $\boldsymbol{n}$. Because the advection operator is represented in an explicit form the size of the base time step, Δt, is limited by the CFL criterion[1]. $\boldsymbol{Q}(t)\,\Delta t$ is the vector of tracer mass increments resulting from source-sink processes added to the tracer in the gridboxes during Δt .

2.1.2 Meteorological input data

TM2 solves the continuity equation in discrete form (2.1.1.4) for an arbitrary number of tracers based on given time-dependent three-dimensional meteorological fields of the geopotential, windvelocity, air temperature and relative humidity. Additionally, surface fluxes of latent heat are needed in order to calculate the subgridscale transport by cumulus clouds (see Section 2.1.5 and Appendix B.2).

These forcing fields may be obtained either from meteorological analyses or from the output of an atmospheric general circulation model. For a realistic representation of synoptic time scale transport processes the meteorological fields have to be available with a time step of 12 hours or shorter.

2.1.3 Coordinate system and time steps

The coordinate system used in the model consists of a regular latitude-longitude grid in the horizontal and sigma coordinates in the vertical dimension. The vertical σ coordinate is defined as:

$$\sigma = \frac{p - p_{top}}{p_{surf} - p_{top}} = \frac{p - p_{top}}{p_s} \qquad (2.1.3.1)$$

where $p_{surf} = p_{surf}(x, y, t)$ represents the surface pressure and p_{top} the pressure at the upper boundary of the atmosphere. In the standard model configuration $p_{top} = 10$ hPa. For convenience, we introduce $p_s = p_{surf} - p_{top}$. Throughout the model code and documentation p_s is denoted as “surface pressure”.

Transport is calculated using a basic time step Δt as defined in Section 2.1.1. The availablility of the meteorological data defines a second time step, Δt_{met}, which must be an integer multiple of the basic time step Δt .

2.1.4 Advection

Advection, i.e. tracer transport by the three-dimensional air mass fluxes resolved on the model grid is calculated using the “slopes scheme” developed by Russell and Lerner [1981]. In this scheme each tracer is represented within each gridboxe by a three-dimensional linear tracer mixing ratio distribution. This distribution is defined by the tracer mixing ratio at the center of the gridbox, χ, and and the components of the spatial gradient of the tracer mixing ratio within the gridbox, $\left(\frac{\partial \chi}{\partial x}, \frac{\partial \chi}{\partial y}, \frac{\partial \chi}{\partial z}\right)$. Notice that the tracer

1. Courant-Friedrichs-Lewy (CFL) criterion: The total amount of air moved out of any gridbox during Δt must not exceed the air mass present in the gridbox at the beginning of the time step; hence the size of the air mass fluxes determines the maximum permissible value of Δt .

mixing ratio thus defined is discontinuous at the borders of the gridboxes.

For efficiency of the program code the primary variables in the model are the tracer masses, $\boldsymbol{n}$, and the "slopes" of the tracer mass ($\boldsymbol{n}_x$,$\boldsymbol{n}_y$,$\boldsymbol{n}_z$) in the gridboxes. Assuming no variations of the air density within a particular gridbox the tracer mass is given by

$$n = \chi m = \chi \rho \Delta x \Delta y \Delta z \qquad (2.1.4.1)$$

where m denotes the air mass in the gridbox. For efficiency of the program code the tracer mass "slope" in the x-direction is defined as

$$n_x = m \frac{\partial \chi}{\partial x} \frac{\Delta x}{2} \qquad (2.1.4.2)$$

and analogously in the y- and z-directions.

With the additional information of the slopes the scheme exhibits much less numerical diffusion than e.g. a simple upstream formulation. This advantage is, however, obtained at the expense of storage and computational resources since four arrays are required for the representation of each simulated tracer.

For certain tracers negative concentrations are unacceptable. In regions of steep concentration gradients the calculated tracer mass slopes may become so large that negative tracer masses arise. If this is deemed unacceptable, this behaviour may be inhibited by limiting the absolute size of the tracer mass slopes. This, however, effectively implies a larger numerical diffusion of the advection scheme.

The slopes scheme requires as input data the air masses in each gridbox together with the air mass fluxes, i.e. the amount of air crossing the borders of the gridboxes during each time step. The fields of air masses and air mass fluxes on the tracer model grid based on a particular meteorological dataset (meteorological analyses or climate model output) are computed and saved during a preprocessing stage (see Appendix B.1). In a tracer model simulation run these fields are then read from disk.

For simplicity the three-dimensional advection process is further subdivided into one-dimensional transport in longitudinal, meridional and vertical direction. Because of the different typical sizes of the mass fluxes in the three directions these one-dimensional advection steps are further subdivided and performed sequentially in a spatial leap-frog pattern:

zonal advection during $\frac{1}{4}\Delta t$,

meridional advection during $\frac{1}{2}\Delta t$,

zonal advection during $\frac{1}{4}\Delta t$,

vertical advection during Δt,

zonal advection during $\frac{1}{4}\Delta t$,

meridional advection during $\frac{1}{2}\Delta t$,

zonal advection during $\frac{1}{4}\Delta t$

With this particular setup the effective advective time step in the longitudinal direction is $\frac{1}{4}$ and in the meridional direction $\frac{1}{2}$ of the base time step Δt. Close to the polar regions, where the zonal extent of the gridboxes becomes small, the zonal advection step is further subdivided in order to meet the CFL criterion.

During a one-dimensional advection time step the slopes scheme performs the following operations for a particular grid box (the mathematical formulae are given in Russell and Lerner [1981]):

1. Calculate the new air mass in the gridbox based on the old air mass and the air mass fluxes crossing the borders.
2. Calculate the amount of tracer that crosses the borders of the gridbox, based on the masses and slopes of the tracer in the gridbox and its adjacent neighbours.
3. Calculate the new tracer mass in the gridbox.
4. Calculate the new slopes of the tracer in the gridbox. The new slope in the transport direction under consideration is calculated by a “hardwired” least squares fit to the displaced tracer mass distribution. The new slopes in the two other spatial directions are calculated using an upstream formula.

2.1.5 Vertical convection

Subgridscale vertical transport is parameterized in the model by two processes: vertical diffusion and cumulus cloud transport. The vertical diffusion coefficients are calculated based on the stability of the air using the formulae given by Louis [1979] (listed in Appendix B.2). Tracer transport by subgridscale cumulus clouds is calculated using the mass flux scheme of Tiedke [1989]. Formally this scheme defines in each vertical grid column a statistical stationary cloud containing an updraft and a downdraft. The magnitude, entrainment and detrainment rates of these up- and downdrafts are computed based on the horizontal below-cloud divergence of moisture and on the buoyancy of the in-cloud air relative to the air outside of the cloud. Tracer is entrained into the up- and downdraft, mixed with the air inside the up- and downdraft and detrained into the environmental air. Furthermore, the net vertical air mass flux of the up- and downdraft induces a subgridscale subsidence flux of environmental air outside the cloud.

Figure 1 shows a schematic sketch of the subgridscale transport processes.

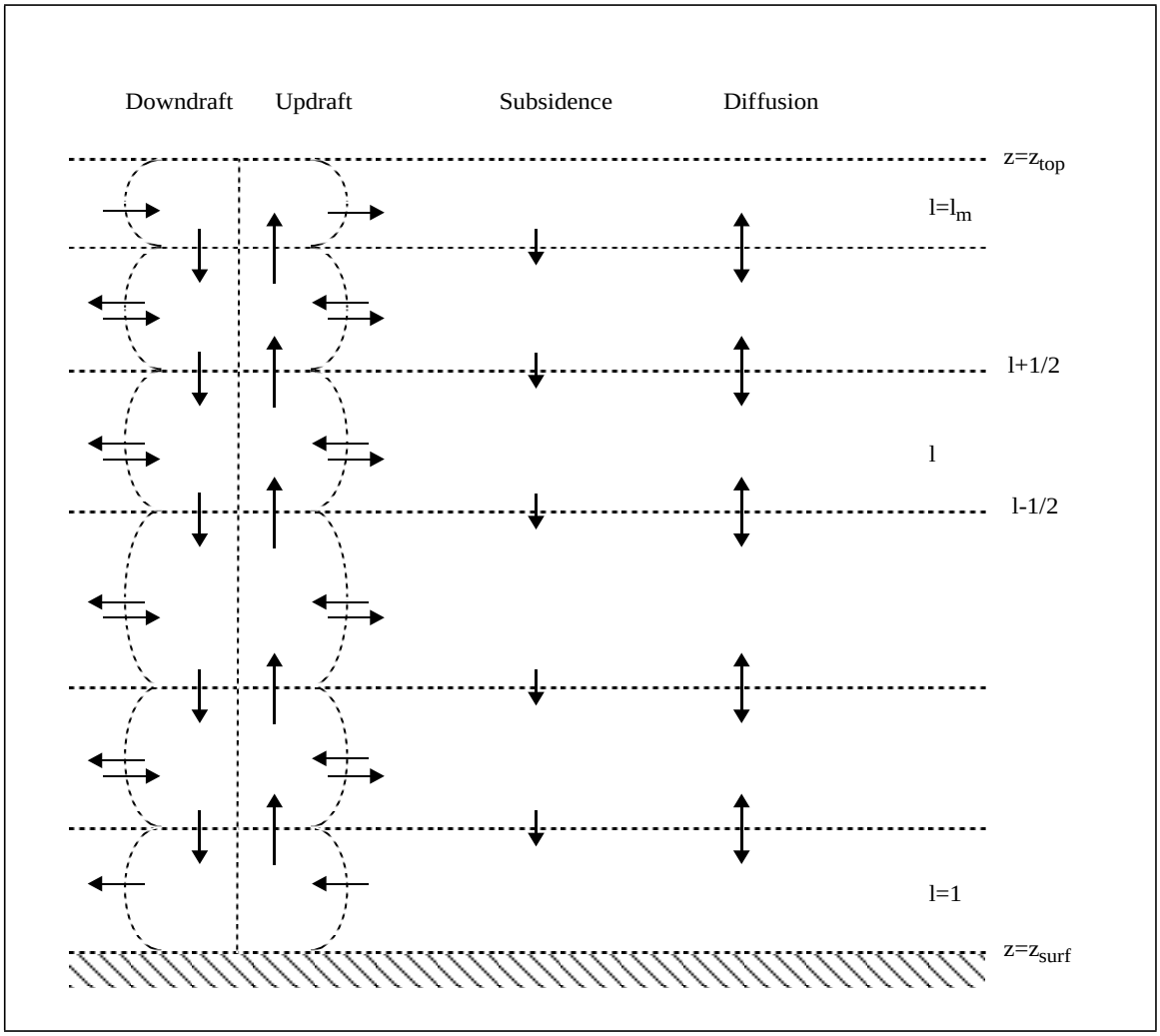

Figure 1 Schematic representation of the vertical subgridscale processes (termed “vertical convection”) computed in the tracer model

In a preprocessing stage, both the vertical diffusion coefficients and the parameters for the cumulus cloud transport are computed from the background meteorological fields (see Appendix B.2). In a tracer model simulation this information is read from disk at each meteorological time step Δt_{met}. One consequence of this setup is that the cloud transport processes operate during the entire meteorological time step Δt_{met}. Formally, the statistical cloud extends through all grid levels of the model. In practice the cloud extends only through the levels as computed in cloud model. Technically this is achieved by setting the appropriate entrainment and detrainment fluxes to zero.

Discretization

Since the tracer transport is linear, the effect of the subgridscale transport processes on the tracer masses on the different model levels during one base time step Δt may be represented simply by the multiplication of the vector of tracer masses in a vertical grid column by a “convection matrix” $\boldsymbol{C}$. The element

$[\boldsymbol{C}]_{lk}$ of this matrix thereby represents the fraction of the tracer mass of the gridbox at level *k* that ends up in the gridbox at level *l* after the "convection process". The elements of the convection matrix are determined from the discretized representation of the subgridscale transport processes. They are recomputed each time the diffusion coefficients and cumulus cloud transport parameters (entrainment and detrainment rates into and out of the up- and downdraft) are read from disk.

Considering only the vertical convection process alone, we have to discretize and solve the following equation:

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} = -\frac{\partial}{\partial z}\bar{\rho}\,\overline{\chi' w'} \tag{2.1.5.1}$$

during the time step Δt. The right hand side of equation (2.1.5.1) is represented by the divergence of the subgridscale tracer fluxes F_{vsub} (defined positive in the upward direction):

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} = -\frac{\partial}{\partial z}F_{vsub} = -\frac{\partial}{\partial z}(F_u + F_d + F_s + F_{vdiff}) \tag{2.1.5.2}$$

where F_u and F_d denote the tracer mass flux in the up- and downdraft, respectively, F_s the tracer mass flux by subsidence and F_{vdiff} the tracer mass fluxes by turbulent vertical diffusion.

Integrating equation (2.1.5.2) over a gridbox at height level *l* gives

$$\frac{d}{dt}n_l = \left(F_{vsub,\, l-\frac{1}{2}} - F_{vsub,\, l+\frac{1}{2}}\right) \tag{2.1.5.3}$$

where n_l denotes the tracer mass in the box. The subgridscale fluxes $F_{vsub,\, l\pm\frac{1}{2}}$ at the lower, $\left(l-\frac{1}{2}\right)$, and upper, $\left(l+\frac{1}{2}\right)$, border of the box can be represented as a linear function of the gridaverage tracer mixing ratios throughout the vertical column and hence of the tracer masses in each of the boxes in the vertical grid column:

$$F_{vsub,\, l+\frac{1}{2}} = \sum_{k=1}^{l_m} f_{l+\frac{1}{2},\, k}\, n_k \tag{2.1.5.4}$$

The coefficient $f_{l+\frac{1}{2},\, k}$ thus represents the fraction of the tracer mass of layer *k* that, per unit of time, crosses the layer boundary $l+\frac{1}{2}$ by means of subgridscale vertical transport processes. It is further split into components representing the tracer mass fractions crossing the layer boundary due to each individual subgridscale vertical transport process:

$$f_{l+\frac{1}{2},\, k} = f^{u}_{l+\frac{1}{2},\, k} + f^{d}_{l+\frac{1}{2},\, k} + f^{s}_{l+\frac{1}{2},\, k} + f^{vdiff}_{l+\frac{1}{2},\, k} \tag{2.1.5.5}$$

Explicit forms of $f^{x}_{l+\frac{1}{2},\, k}$ for each of the different vertical subgridscale processes are derived in the fol-

lowing subsections.

Equation (2.1.5.3) thus can be written for the entire column in matrix form[1]

$$\frac{d}{dt}\boldsymbol{n} = \boldsymbol{M}\ \boldsymbol{n} \tag{2.1.5.6}$$

where the elements of matrix $\boldsymbol{M}$ are given as

$$[\boldsymbol{M}]_{l,k} = f_{l-\frac{1}{2},k} - f_{l+\frac{1}{2},k} \tag{2.1.5.7}$$

For reasons of stability, equation (2.1.5.6) is integrated in implicit form:

$$\boldsymbol{n}(t+\Delta t) = \boldsymbol{n}(t) + \Delta t\boldsymbol{M}\cdot\boldsymbol{n}(t+\Delta t) \tag{2.1.5.8}$$

therefore the tracer masses after the convection step are related to the tracer masses before the convection step by

$$\boldsymbol{n}(t+\Delta t) = (\boldsymbol{I}-\Delta t\boldsymbol{M})^{-1}\cdot\boldsymbol{n}(t) = \boldsymbol{C}\cdot\boldsymbol{n}(t) \tag{2.1.5.9}$$

where $\boldsymbol{I}$ denotes the identity matrix and $\boldsymbol{C}$ is the convection matrix.

Subgridscale transport also affects the slopes of the tracer mass within the gridboxes. The horizontal tracer mass slopes $\boldsymbol{n}_x$ and $\boldsymbol{n}_y$ are transported similarly as the tracer mass, i.e. they are updated by

$$\boldsymbol{n}_x(t+\Delta t) = \boldsymbol{C}\cdot\boldsymbol{n}_x(t) \tag{2.1.5.10}$$

and

$$\boldsymbol{n}_y(t+\Delta t) = \boldsymbol{C}\cdot\boldsymbol{n}_y(t) \tag{2.1.5.11}$$

The vertical slopes are treated differently. It is assumed that the subgridscale processes tend to reduce the tracer mass slopes in the vertical direction. In the present scheme the vertical slopes are reduced to the fraction of tracer mass that remains at a particular level. These fractions are the diagonal elements of the convection matrix; hence for level *l* we have

$$\boldsymbol{n}_{z,l}(t+\Delta t) = [\boldsymbol{C}]_{ll}\boldsymbol{n}_{z,l}(t) \tag{2.1.5.12}$$

Transport by cumulus clouds

Subgridscale transport induced by cumulus clouds is represented by the three components

$$F_u + F_d + F_s = M_u\chi_u + M_d\chi_d - (M_u + M_d)\bar{\chi} \tag{2.1.5.13}$$

corresponding to the tracer flux in the updraft (subscript *u*), in the downdraft (subscript *d*) and induced by subsidence. M_u denotes the updraft air mass flux, M_d the downdraft and -(M_u+M_d) the mass flux by

1. Note that in this section the symbol $\boldsymbol{n}$ refers to the vector of tracer masses of a particular grid column only.

subsidence, all expressed in kg m^{-2} s^{-1}.

Entrainment and detrainment rates (expressed in kg m^{-3} s^{-1}) into the updraft ($E_u(z)$, $D_u(z)$) and into the downdraft ($E_d(z)$, $D_d(z)$) are given as functions of height, z, from the physical cloud model (see Appendix B.2). Updraft and downdraft mass fluxes and their in-cloud tracer mixing ratios are obtained by solving the discrete version of the following set of equations for the updraft:

$$\frac{\partial}{\partial z} M_u = E_u - D_u \tag{2.1.5.14}$$

$$\frac{\partial}{\partial z} F_u = E_u \bar{\chi} - D_u \chi_u \tag{2.1.5.15}$$

and for the downdraft:

$$\frac{\partial}{\partial z} M_d = E_d - D_d \tag{2.1.5.16}$$

$$\frac{\partial}{\partial z} F_d = E_d \bar{\chi} - D_d \chi_d \tag{2.1.5.17}$$

where we have denoted the fluxes of tracer in the up- and downdraft by

$$F_u = M_u \chi_u \tag{2.1.5.18}$$

$$F_d = M_d \chi_d \tag{2.1.5.19}$$

The appropriate boundary conditions are

$$\begin{aligned} M_u = F_u = 0 \text{ at } z = z_{surf} \\ M_d = F_d = 0 \text{ at } z = z_{top} \end{aligned} \tag{2.1.5.20}$$

where z_{surf} and z_{top} are the geopotential height of the bottom and the top of the grid column, respectively.

The continuity of mass also requires that

$$\int_{z_{surf}}^{z_{top}} (E_u - D_u) dz = 0 \tag{2.1.5.21}$$

$$\int_{z_{surf}}^{z_{top}} (E_d - D_d) dz = 0 \tag{2.1.5.22}$$

Using (2.1.5.18) the equations for the updraft and downdraft tracer fluxes (2.1.5.15) and (2.1.5.17) can be reexpressed as

$$\frac{\partial}{\partial z} F_u = E_u \bar{\chi} - D_u \frac{F_u}{M_u} \tag{2.1.5.23}$$

$$\frac{\partial}{\partial z} F_d = E_d \bar{\chi} - D_d \frac{F_d}{M_d} \tag{2.1.5.24}$$

Updraft

The discretized form of the equation for the updraft air mass flux, (2.1.5.14), is obtained by integrating over a gridbox at height level l

$$M_{u,\,l+\frac{1}{2}} - M_{u,\,l-\frac{1}{2}} = E_{u,\,l} - D_{u,\,l} \tag{2.1.5.25}$$

where $E_{u,l}$ and $D_{u,l}$ denote the air mass entrainment into and detrainment out of the updraft, respectively (in units of kg s^{-1}). $M_{u,\,l+\frac{1}{2}}$ denotes the updraft air mass flux at the model layer boundary $z_{l+\frac{1}{2}}$ (in units of kg s^{-1}). Similarly the discrete version of the equation for the updraft tracer mass flux, (2.1.5.23), becomes

$$F_{u,\,l+\frac{1}{2}} - F_{u,\,l-\frac{1}{2}} = E_{u,\,l}\bar{\chi}_l - D_{u,\,l}\frac{F_{u,\,l+\frac{1}{2}}}{M_{u,\,l+\frac{1}{2}}} \tag{2.1.5.26}$$

where, for stability reasons, we have used an implicit formulation on the right hand side. Solving for $F_{u,\,l+\frac{1}{2}}$ and using (2.1.5.25) we obtain

$$F_{u,\,l+\frac{1}{2}} = \left(F_{u,\,l-\frac{1}{2}} + E_{u,\,l}\frac{n_l}{m_l}\right)\left(1 - \frac{D_{u,\,l}}{M_{u,\,l-\frac{1}{2}} + E_{u,\,l}}\right) \tag{2.1.5.27}$$

where we have substituted the gridaveraged tracer mixing ratio $\bar{\chi}$ by the ratio of tracer mass n_l divided by the air mass m_l in the gridbox.

Equations (2.1.5.25) and (2.1.5.27) represent recursive relations that may be solved by starting from the surface using the boundary conditions (2.1.5.20) and working upwards. It is easily seen, that, using this procedure, $F_{u,\,l+\frac{1}{2}}$ at any layer boundary results in the form of equation (2.1.5.4), i.e. as a linear function of the tracer masses n_l, $l = 1,...,l_m$ of all the gridboxes in the grid column under consideration.

Explicitly, the coefficients $f^u_{l+\frac{1}{2},\,k}$ are determined by the recursion formula:

$$f^u_{\frac{1}{2},\,k} = 0, \qquad k = 1, \ldots, l_m$$

$$f^u_{l+\frac{1}{2},\,k} = \left(f^u_{l-\frac{1}{2},\,k} + \delta_{k,\,l}\frac{E_{u,\,l}}{m_l}\right)\left(1 - \frac{D_{u,\,l}}{M_{u,\,l-\frac{1}{2}} + E_{u,\,l}}\right) \tag{2.1.5.28}$$

where

$$\delta_{k,l} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{otherwise} \end{cases} \tag{2.1.5.29}$$

Downdraft

The derivation of the coefficients $f^{d}_{l+\frac{1}{2},k}$ for the downdraft tracer fluxes are derived analogous to the updraft as given in the previous section. We obtain the recursion formula

$$f^{d}_{l_m+\frac{1}{2},k} = 0, k = 1, \ldots, l_m \tag{2.1.5.30}$$

$$f^{d}_{l-\frac{1}{2},k} = \left(f^{d}_{l+\frac{1}{2},k} - \delta_{k,l}\frac{E_{d,l}}{m_l}\right)\left(1 + \frac{D_{d,l}}{M_{d,l+\frac{1}{2}} - E_{d,l}}\right) \tag{2.1.5.31}$$

Subsidence

The subgridscale flux induced by the subsidence air mass flux -(M_u+M_d) outside of the cloud is given by

$$F_s = -(M_u + M_d)\bar{\chi} \tag{2.1.5.32}$$

Using an upstream formulation this flux is written at the model layer boudaries as

$$F_{s,l+\frac{1}{2}} = -\left(M_{u,l+\frac{1}{2}} + M_{d,l+\frac{1}{2}}\right)\frac{n_{l+1}}{m_{l+1}} \tag{2.1.5.33}$$

Hence the coefficients $f^{s}_{l+\frac{1}{2},k}$ result in

$$f^{s}_{l+\frac{1}{2},k} = -\frac{M_{u,l+\frac{1}{2}} + M_{d,l+\frac{1}{2}}}{m_{l+1}}\delta_{l+1,k} \tag{2.1.5.34}$$

Vertical diffusion

The diffusive flux through the upper boundary of a gridbox at height level *l* is

$$F_{vdiff} = -A\rho K_v \frac{\partial\bar{\chi}}{\partial z} \tag{2.1.5.35}$$

where *A* denotes the horizontal area of the gridbox and ρ the air density. Rewritten in discretized form as functions of the tracer and air masses in the adjacent boxes this becomes

$$F_{vdiff,l+\frac{1}{2}} = -A\frac{\rho_{l+\frac{1}{2}}}{z_{l+1} - z_l}K_{l+\frac{1}{2}}\left(\frac{n_{l+1}}{m_{l+1}} - \frac{n_l}{m_l}\right) \tag{2.1.5.36}$$

and the coefficients $f^{vdiff}_{l+\frac{1}{2},k}$ are thus given by

$$f^{vdiff}_{l+\frac{1}{2},k} = -A\frac{\rho_{l+\frac{1}{2}}}{z_{l+1}-z_l}K_{l+\frac{1}{2}}\left(\frac{\delta_{l+1,k}}{m_{l+1}}-\frac{\delta_{l,k}}{m_l}\right) \qquad (2.1.5.37)$$

Global tuning of convection parametrization

For testing purposes the parametrization of vertical convection can be “tuned” by scaling the entrainment and detrainment fluxes with a global scaling parameter ζ_{cu} and the vertical diffusion coefficients with a global scaling parameter ζ_K.

2.1.6 Horizontal diffusion

Simulation experiments with earlier versions of the transport model (TM1, see Heimann and Keeling [1989], Keeling et al. [1989]) indicated, depending on the meteorological fields used to drive the model, that the interhemispheric transport on annual average was too weak, compared to that inferred from other tracer studies (notably ^{85}Kr and F-11). Based on this observation a horizontal diffusion term was included in the model in a way similar to the parametrization developed by Prather et al. [1987]. The strength of this additional term was controlled by a global parameter (a scale length) which provided a tool to “fine tune” the model’s interhemispheric exchange.

With the use of meteorological datasets from the analyses after 1985 the interhemispheric transport became stronger (most probably because of more realistic analyzed fields in the tropical regions). In the present version of TM2 (version 8.5) the code for the horizontal diffusion has been removed.

3. SYSTEM DESCRIPTION

The TM2 model system consists of three relatively independent components: The preprocessing program systems which generate the air mass fluxes (`mflux`) and subgridscale transport information (`subscal`) on the tracer model grid from stored meteorological fields, and TM2, the actual tracer model code itself. In practice the steps (`mflux`) and (`subscal`) are performed only once for a particular meteorological dataset and transport model geometry configuration. Figure 2 shows the information flow.

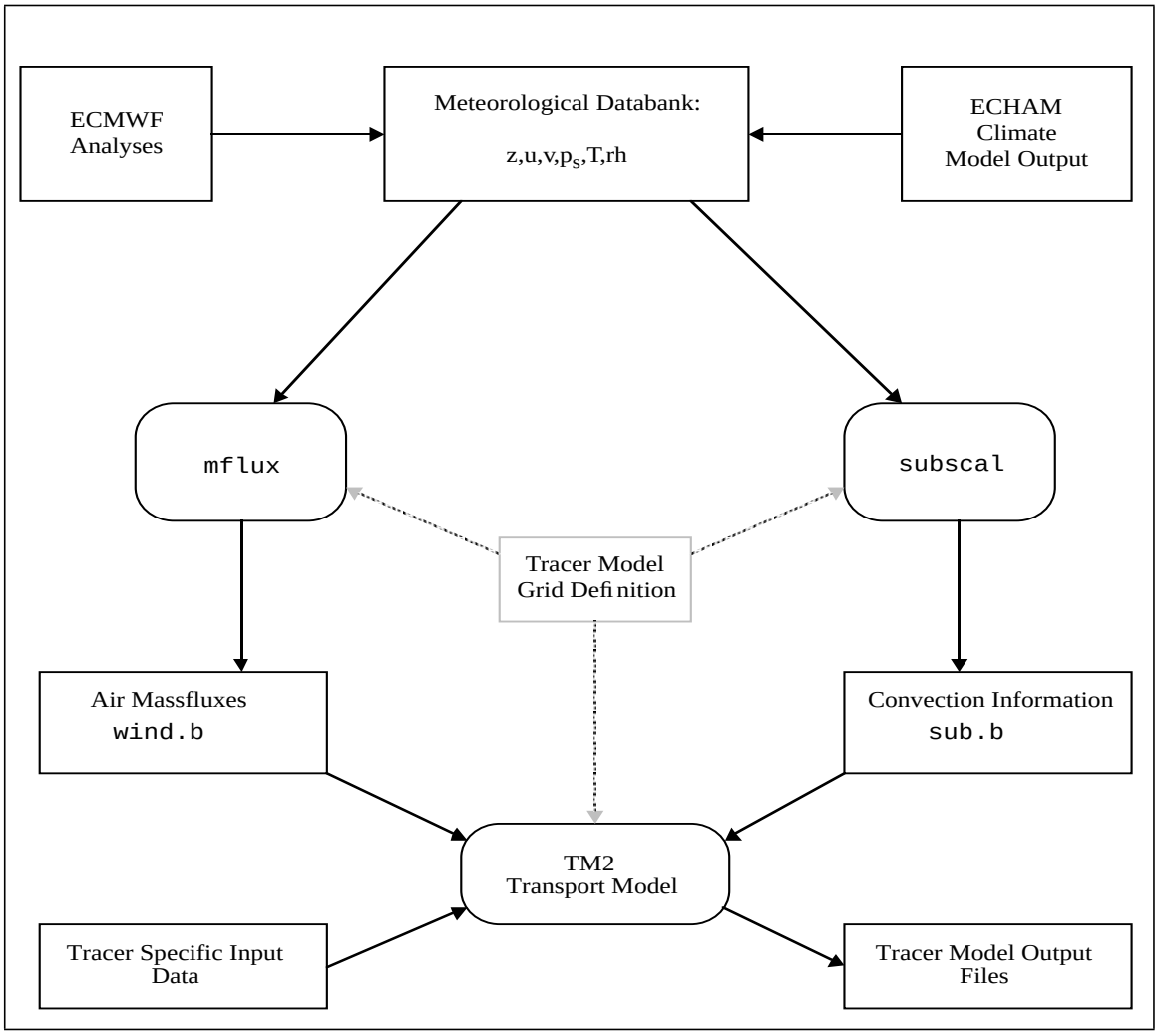

Figure 2 TM2 model information flow

The preprocessing modules `mflux` and `subscal` are described in Appendix B.1 and Appendix B.2. The following description focuses on the code of the tracer transport model itself.

3.1 SPATIAL GRID STRUCTURE

TM2 uses a regular latitude-longitude grid, with *im* boxes in the zonal, *jm* boxes in the meridional and *lm* layers in the vertical dimension. The standard “coarse grid” version of the model uses *im*=36, *jm*=24,

and *lm*=9.

In the zonal direction the longitudes (in degrees) ϕ_i of the centers of the gridboxes are located at

$$\phi_i = -180 + (i-1)dx\ ,\ \text{i=1,..., im} \qquad (3.1.1)$$

with a grid spacing of $dx = \frac{360}{im}$ and where west longitudes are designated to be negative.

In the meridional direction the *jm* grid boxes have an extent (in degrees) of $dy = \frac{180}{jm-1}$. They are spaced such that the latitudes (in degrees) Θ_j of the grid box centers are located at

$$\Theta_j = -90 + (j-1)dy\ ,\ \text{j=1,..., jm} \qquad (3.1.2)$$

(Southern latitudes are designated to be negative). The first and last (indices *j*=1 and *j*=*jm*) grid box are centered on the poles itself and have a meridional extent of $\frac{dy}{2}$. These polar boxes are not divided in the zonal direction and are referenced with zonal index *i*=1 (for indices *j*=1 and *j*=*jm* the elements with zonal indices *i*>1 are undefined.)

The horizontal layout of the standard “coarse grid” version of the model grid is shown in Figure 3.

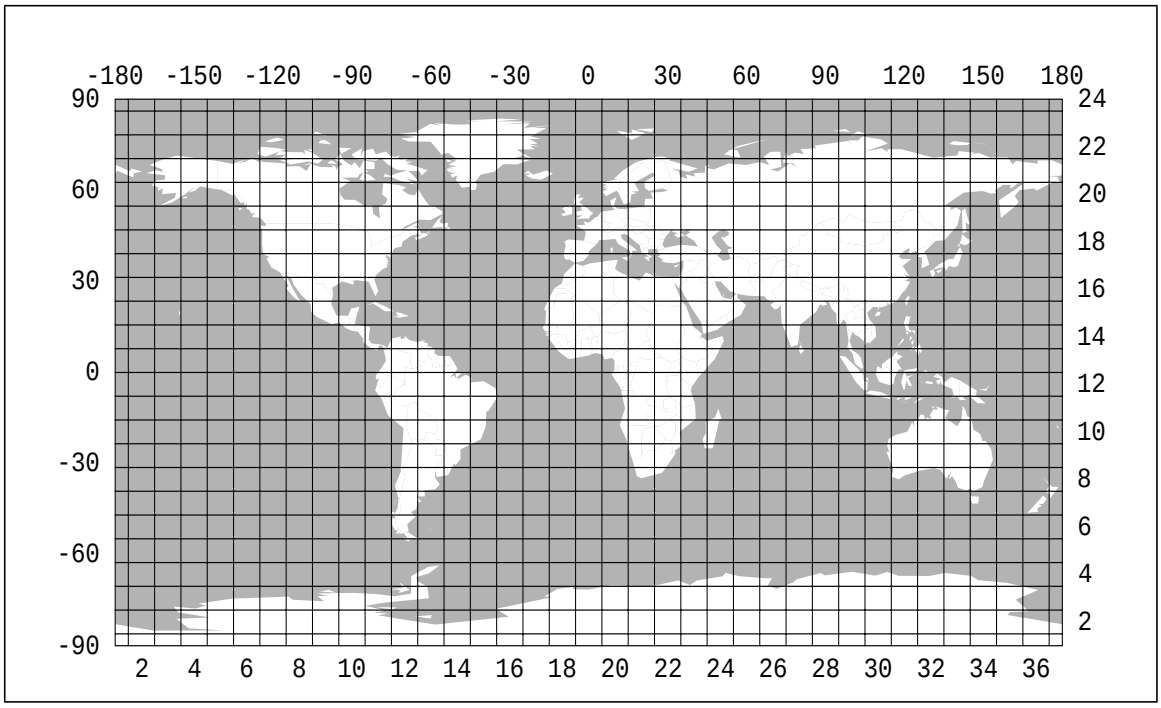

Figure 3 Horizontal layout of the “coarse grid” version of the model grid. The numbers on the top and left border denote longitude and latitude in degrees, respectively. The numbers on the bottom and right border denote the longitudinal (*i*) and latitudinal (*j*) indices of the gridboxes.

In the vertical dimension TM2 uses a sigma coordinate system with a non-uniform spacing. In the standard “coarse grid” version of the model the vertical spacing is specified as listed in Table 1.

Table 1 Vertical grid centers and boundaries in the standard “coarse grid” version of TM2

Model Level	σ coordinate Boundary	Delta	Global mean pressure coordinate (hPa) Boundary	Delta	Mean Level	Geopotential Height above Ground (km)[a]
	0.000000		10			30.7
9		0.0616		60	40	21.7
	0.061602		70			18.3
8		0.0821		80	110	15.4
	0.143737		150			13.4
7		0.1078		105	202	11.6
	0.251540		255			10.1
6		0.1386		135	323	8.53
	0.390144		390			7.21
5		0.1643		160	470	5.86
	0.554415		550			4.67
4		0.1745		170	635	3.56
	0.728953		720			2.57
3		0.1376		134	787	1.85
	0.866530		854			1.18
2		0.0821		89	894	0.80
	0.948665		943			0.36
1		0.0513		50	959	0.22
	1.000000		984			0.00

a. Approximate heights above ground based on a standard atmosphere at 40 N (annual average) [see Houghton, 1979].

3.2 Time and calendar calculations

The basic time unit of the model is seconds. The basic time step is defined by variable `ndyn` which describes the length of an advection time step. The counter `itau` contains the current simulation time. Functions are provided to convert simulation time instants expressed in seconds to a more readable representation as a 6-element integer vector: (year,month,day,hour,minute and seconds) or to a character string.

Four different calendar options are available (which can be selected with control variable `icalendo`):

1. Permanent 360 day years with 12 months of 30 days each.
2. Calendar with 365/366 day years (including correct leap years).
3. Permanent 365 day year.
4. Permanent 366 day leap year.

The choice of the calendar depends on the meteorological data to be used: (1) is appropriate with climate model output, (2) for simulations using multiyear meteorological analyses, (3) and (4) for simulations over several years but cycling repeatedly through the meteorology of a particular year.

3.3 MAIN MODEL LOOP

The main module (file `tracer.f`) contains the principal model loop. The sequence of tasks performed is listed in Table 2. Each task is controlled by a particular control variable contained in common block `m1` (see Section 3.5.2). These control variables specify the time steps for the particular tasks. Since the model simulation time is stepped in increments of the base advection time step `ndyn`, the values of each of these control variables must be set to an integer multiple of `ndyn`.

3.4 PROGRAM MODULES AND SOURCE CODE FILES

The program modules and their purposes are listed in Table 3. They are contained in eight source code files.

Table 2 Main loop of the tracer model

Subroutine	**Control Variable**	**Task**
start	-	initialization
begin of main loop		
bisave	nwrite	save model status
exitus	itaue	end of model run
rwind	nread	input of air mass fluxes
rconv	nread	input of convection information
chem1	nchem	update tracer fields by chemistry process 1
chem2	nchem	update tracer fields by chemistry process 2
source1	nsrce	update tracer fields by source process 1
source2	nsrce	update tracer fields by source process 2
advect	ndyn	calculate horizontal advection

Table 2 Main loop of the tracer model

convec	nconv	calculate vertical mixing by convection and vertical diffusion
diaga	ndiag	accumulate diagnostics
diagout	ndiagp1, ndiagp2	output of diagnostic and time averaged quantities
inctime	-	increment time by basic time step ndyn and update calendar
writc	ncheck	output of tracer mixing ratios at check locations
writx	ninst	output of instantaneous tracer mixing ratio fields
end of main loop		

Table 3 Modules grouped according to source code files and function

Main program (File tracer.f**)**	
`tracer`	main loop
Constants (File constants**)**	
	contains global constants, dimensions etc. (see section 3.5.1)
Common blocks (File common**)**	
`m1`	contains control parameters
`m2`	contains main tracer fields
`m3`	contains text string variables
`m4`	contains accumulator arrays
Tracer specific module (File trace0.f**)**	
`trace0`	tracer specific sources and sinks (contains entry points trace0, trace1, source1, source2, chem1, chem2, see section 4.1.1)
Subroutines related to advection process (File advecg.f**)**	
`advect`	performs one advection step
`dynam0`	calculates vertical mass fluxes
`dynamu`	east-west tracer transport
`dynamv`	south-north tracer transport
`dynamw`	vertical tracer transport
`rwind`	input of mass flux fields
Subroutines related to subgridscale transport processes (File convecg.f**)**	
`convec`	mix tracers by convection
`convma`	calculate convection matrix

Table 3 Modules grouped according to source code files and function

`fastminv`	inversion of convection matrices (vectorizing version)
`rconv`	read convection info
Subroutines related to diagnostic tasks (File diagnosg.f**)**	
`cnsrva`	keep track of conservation properties
`cnsrvp`	print conservation statistics
`diag0`	reset mean field accumulators to 0
`diaga`	mean field accumulator
`diagf`	output of mean mixing ratio fields
`diagj`	ouptut of zonally-vertically av. quantities
`diagjl`	output of zonally averaged quantities
`diagout`	control output of diagnostics
`jlmap`	output lat-height field as a table of numbers
`scale`	scale array of floating point values
`writd`	print mean field statistics
Control and utility modules (File controlg.f**)**	
`bisave`	call savemod alternatingly on different output units
`caldat`	calculate date from given julian date
`chardate`	format 6-number date to character form
`date2tau`	convert 6-number date to simulation time (in seconds)
`default`	set default values for control parameters
`exitus`	terminate simulation
`geomtry`	calculate and define model grid geometry
`inctime`	increment simulation time
`julday`	calculate julian date
`savemod`	save model state on disk
`start`	startup module
`tau2date`	convert simulation clock (in seconds) to 6 number date format
`tstamp`	print simulation time stamp with message
`writc`	write mixing ratios at checklocations
`writx`	write instantaneous mixing ratio fields

3.5 Constants, common blocks and control variables

Constants and the main common blocks are contained in separate source files which are “included” into most of the main subroutines.

3.5.1 Constants

A set of constants is defined in source file `constants`. The defined constants include the number of tracers and the dimensions of the model grids. This file also contains a series of switches, which allow the selection of particular computer environments (for machine dependent subroutine calls) and formats of the mass flux and convection data files.

```
c-----------------------------------------------------------------------
c dimensions, constants and i/o unit numbers
c this version selects the coarse 8d x 10d x 9l grid
c
c one tracer
c
c       changed to v 8.5         mh      mpi HH          14-feb-1994
c-----------------------------------------------------------------------
c grid dimensions and number of tracers
      integer im,jm,lm,ntrace
      parameter (im=36,jm=24,lm=9,ntrace=1)
c geometrical and physical constants
      real twopi,radius,grav,rgas,kapa
      parameter (twopi=6.283185,radius=6375000.,grav=9.81,
     .           rgas=287.,kapa=.286)
c auxilary constants
      real dlon,dlat
      integer np,imm1,jmm1,jmm2,lmm1,lmp1
      parameter (dlon=twopi/im,
     .           dlat=.5*twopi/(jm-1),
     .           np=im*jm,
     .           imm1=im-1,
     .           jmm1=jm-1,
     .           jmm2=jm-2,
     .           lmm1=lm-1,
     .           lmp1=lm+1)
c switch constants
c ! number of high latitude circles in each
c ! hemisphere with double u-advection
      integer lat2
      parameter (lat2=1)
c ! fextra=.true.: mass flux fields in extra-format
c !       =.false.: mass flux fields in standard
c !                 binary format
      logical fextra
      parameter (fextra=.false.)

c unit numbers (model uses the range from 1-24)
c standard control input
      integer kinput0
      parameter (kinput0=5)
c secondary control input
      integer kinput1
      parameter (kinput1=7)

c main control output
```

```
      integer kmain
      parameter (kmain=6)
c debug output
      integer kdebug
      parameter (kdebug=9)

c windmass flux fields
      integer kwind
      parameter (kwind=11)
c convection info
      integer kconv
      parameter (kconv=12)
c final model status
      integer ksave
      parameter (ksave=3)
c model status for restart
      integer krestart
      parameter (krestart=4)

c output of mix-ratios at check locations
      integer kcheck
      parameter (kcheck=16)
c output of diagnostic tables 1
c                                     (tracer info)
      integer kdiag1
      parameter (kdiag1=17)
c output of diagnostic tables 2
c                                     (air mass info)
      integer kdiag2
      parameter (kdiag2=18)
c output of instantaneous mixing
c                                      ratio fields
      integer kmix
      parameter (kmix=19)
c output of averaged mixing ratio fields
      integer kmeanx
      parameter (kmeanx=20)
c                                      output of averaged tracer and air
c                                      mass fields
      integer kmeanrmm
      parameter (kmeanrmm=23)
c temporary scratch files
      integer ktemp1
      parameter (ktemp1=14)
      integer ktemp2
      parameter (ktemp2=15)
c-----------------------------------------------------------------------
```

3.5.2 Common block `m1`

All control variables are contained in common block `m1`. A subset of these variables can be modified through the namelist `inputz` (see subroutine `start`). The name, default value and purpose of each of the control variables are described in the listing below. Constant `m1size` contains the length of common block `m1` in machine words.

```
c-----------------------------------------------------------------------
c***    common  m1      main control variables
c
c       version 8.5                                     mh, 23-feb-1994
      integer istart
```

```
      integer ndyn,nconv,ndiag,nchem,nsrce,nread,nwrite,ninst,ncheck,
     .        ndiff
      integer itau,    itaui,    itaue,    itaut,    itau0
      integer idate,idatei,idatee,idatet,idate0
      integer icalendo,iyear0,julday0
      integer noindc,indc
      integer idacc
      logical newyr,newmonth,newday,newsrun,cdebug,limits
      integer ndiagp1,ndiagp2
      integer nstep,nstep0
      real xi,cpu0,cpu1
      real fscale
      real ptop,psf,areag
      real czeta,czetak,dscale
      real dummyv

      integer m1first,m1size
      parameter (m1size=1+1+10+5+5*6+3+1+3*10+8+6+2+7+ntrace+3+3+64)
      common /m1/ m1first,
     .  istart,
     .  ndyn,nconv,ndiag,nchem,nsrce,nread,nwrite,ninst,ncheck,ndiff,
     .  itau,    itaui,    itaue,    itaut,    itau0,
     .  idate(6),idatei(6),idatee(6),idatet(6),idate0(6),
     .  icalendo,iyear0,julday0,
     .  noindc,indc(3,10),
     .  idacc(8),
     .  newyr,newmonth,newday,newsrun,cdebug,limits,
     .  ndiagp1,ndiagp2,
     .  nstep,nstep0,xi(3),cpu0,cpu1,
     .  fscale(ntrace),
     .  ptop,psf,areag,
     .  czeta,czetak,dscale,
     .  dummyv(64)
c-------------------------------------------------------------------------
c
c       main control variables, accessible through namelist 'inputz'
c       all times (unless noted) are given in seconds.
c       internally model time is kept in seconds (variables itau...) since
c       1st-jan-iyear0, 00:00:00
c-------------------------------------------------------------------------c
c       name            type    default                 purpose
c       ----            ----    -------                 -------
c
c       ndyn            integer 1*3600          length of full advection step
c                                               (seconds)
c       nconv           integer 1*3600          interval for convection
c                                               calculation
c       ndiff           integer 0               interval for horizontal
c                                               diffusion calculation
c       nchem           integer 0               interval for chemistry
c                                               calculations (sr 'chem1,chem2')
c       nsrce           integer 24*3600         interval for source calculation
c                                               (sr 'source1','source2')
c       czeta           real    1.              scaling factor for convection
c       czetak          real    1.              scaling factor for vertical
c                                               diffusion
c       limits          logical .false.         if set to .true. then
c                                               the slopes are limited
c                                               such that no negative tracer
c                                               masses should occur
c       dscale          real    250e3           length scale for horizontal
c                                               diffusion (m)
```

```
c       fscale(ntrace)  real    ntrace*1.       scaling factor for conversion
c                                               of mixingratios
c                                               in kg tracer per kg air to
c                                               practical mixing ratio units
c                                               (e.g. ppm)
c
c       istart          integer 10              start/restart options:
c                                               1       coldstart with initial
c                                                       fields set to 0
c                                               2       coldstart with initial
c                                                       fields computed
c                                                       in sr trace0.f,
c                                                       entry trace1
c                                               3       coldstart with initial
c                                                       fields read from model
c                                                       save file saveold.b
c                                               10      warmstart from file
c                                                       restart.b
c       nread           integer 12*3600         interval for input of
c                                               mass fluxes and convection info
c                                               (from files wind.b, subscal.b)
c       nwrite          integer 0               interval for alternate output
c                                               of restart status on
c                                               files save1.b and save2.b
c       ninst           integer 0               interval for output of instan-
c                                               taneous tracer mix ratio fields
c       ncheck          integer 0               interval for output of
c                                               tracer mix ratio
c                                               at checkpoints
c       noindc          integer 0               no of checkpoints
c       indc(3,10)      integer 30*0            indices of checkpoints
c                                               (i,j,l etc)
c       ndiag           integer 12*3600         interval for computing mean
c                                               quantities
c       ndiagp1         integer -2              interval for output of
c                                               conservation statistics
c                                               on file tables1.d, tables2.d.
c                                               -1      daily
c                                               -2      monthly
c                                               -3      yearly
c                                               >=0     interval in seconds
c       ndiagp2         integer -2              interval for output of time
c                                               averaged fields
c                                               -1      daily
c                                               -2      monthly
c                                               -3      yearly
c                                               >=0     interval in seconds
c       icalendo        integer 2               calendar type
c                                               1 permanent 360 day year
c                                                 calendar
c                                               2 real calendar
c                                               3 permanent 365 day year
c                                                 calendar
c                                               4 permanent 366 day year
c                                                 calendar
c       iyear0          integer 1980            base year for calendar
c                                               calculations (because of
c                                               overflow problems this should
c                                               deviate on a 32 bit machine
c                                               not more than +-65 years from
c                                               any year actually used in the
c                                               model runs)
```

```
c
c  date/times are expressed as yr,month,day,hour,min,sec
c
c       idatei(6)       integer 1980 1 1 0 0 0  date/time for start of
c                                               model run
c       idatee(6)       integer 1980 1 1 0 0 0  date/time for end of model run
c       idatet(6)       integer 1980 1 1 0 0 0  date/time after which instan-
c                                               taneous output (controlled
c                                               by 'ninst') is written
c
c       cdebug          logical false           if true then output of debug
c                                               info is written on file
c                                               debug.d
c
c-------------------------------------------------------------------------
c       other variables in common block m1
c
c       name            type    purpose
c       ----            ----    -------
c       m1first         integer dummy variable - indicates start of common
c                               block m1
c
c       itau            integer current model time
c       idate(6)        integer date corresponding to itau
c       itaui           integer start time (corresponds to idatei)
c       itaue           integer end time (corresponds to idatee)
c       itaut           integer time after which instantaneous output is
c                               written (corresponds to idatet)
c       itau0,idate0(6) integer time/date when diagnostic arrays were last
c                               reset
c       julday0         integer julian day of base time 1st-jan-iyear0,00:00:00
c                               Needed only when icalendo.eq.2
c       idacc(8)        integer counters:
c                               idacc(1)   no of times averaged tracer
c                                          mix ratio is calculated
c                               others are not used at present
c       newyr           logical .true.  if at beginning of a new year
c       newmonth        logical .true.  if at beginning of a new month
c       newday          logical .true.  if at beginning of a new day
c                               (i.e. at 00Z)
c       newsrun         logical .true.  if at beginning of a new run or at
c                               beginning of a continuation run
c       nstep           integer advection step counter for current run/or
c                               continuation run
c       nstep0          integer not needed
c       xi(3)           real    maximum courant numbers for the three
c                               directions (u-, v-, w-)
c       cpu0            real    process time at beginning of run (in seconds)
c       cpu1            real    process time at last reset time instant
c                               (in seconds)
c       ptop            real    pressure at top of atmosphere (1000Pa)
c       psf             real    mean average surface pressure (98400Pa)
c       areag           real    surface of globe
c       dummyv          real    dummy space for future additions to m1
c-------------------------------------------------------------------------
```

3.5.3 Common block m2

Common block m2 contains the main model fields, i.e. the arrays defining the vertical and horizontal geometry, the air masses of each gridbox and the masses and slopes of each tracer.

```
c-------------------------------------------------------------------------------
c***    common  m2      main model fields
c
      integer m2first
      real pland,m,rm,rxm,rym,rzm
      real sig,sige,dsig,dsigo,dxyp,dx,dy
      common /m2/ m2first,
     .  sig(lm),sige(lmp1),dsig(lm),dsigo(lmm1),dxyp(jm),
     .  dx(jmm1),dy,
     .  pland(im,jm),m(im,jm,lm),
     .  rm(im,jm,lm,ntrace),rxm(im,jm,lm,ntrace),
     .  rym(im,jm,lm,ntrace),rzm(im,jm,lm,ntrace)
c       variable        type    purpose
c       --------        ----    -------
c       m2first         integer dummy variable - indicates start of common
c                               block m2
c
c       sig(lm)         real    sigma levels of model layer centers
c       sige(lm+1)      real    sigma levels of model layer boundaries
c       dsig(lm)        real    thickness of model layers in sigma units
c                               (positive numbers!)
c       dsigo(lm-1)     real    distance between model layer centers in
c                               sigma units (positive numbers!)
c       dxyp(jm)        real    areas of grid cells (in m**2)
c       dx(jmm1)        real    length of grid boxes in zonal direction (m)
c       dy              real    length of grid boxes in meridional
c                               direction (m)
c       pland(im,jm)    real    land fraction in each grid cell
c       m(im,jm,lm)     real    air mass (kg)
c       rm(im,jm,lm)    real    tracer mass (kg)
c       rxm(im,jm,lm)   real    x-slope of tracer: m*(dchi/dx)*delta_x/2 (kg)
c       rym(im,jm,lm)   real    y-slope of tracer: m*(dchi/dy)*delta_y/2 (kg)
c       rzm(im,jm,lm)   real    z-slope of tracer: m*(dchi/dz)*delta_z/2 (kg)
c-------------------------------------------------------------------------------
```

3.5.4 Common Block `m3`

Common block `m3` contains the character variable `xlabel` of length 160 characters which identifies the particular model run. Constant `m3size` contains the length of common block `m3` in characters.

```
c-------------------------------------------------------------------------------
c***    common          m3      character strings
      integer m3size
      parameter (m3size=168)

      character*160 xlabel
      character*8 m3first
      common /m3/ m3first,
     .  xlabel
c
c       variable      type      purpose
c       --------      ----      -------
c       m3first       char      dummy variable - indicates beginning of common
c                               block m3
c
c       xlabel        char*160  run text label. last 8 characters contain
c                               model version info
c-------------------------------------------------------------------------------
```

3.5.5 Common Block m4

Common block m4 contains the accumulator arrays which hold the sums of various diagnostic quantities for subsequent averaging. Constant m4size contains the length of common block m4 in machine words.

```
c-------------------------------------------------------------------------
c***    common          m4      accumulator arrays
c
      integer m4size,m4first
      parameter
     .  ( m4size=1+jm*lm*4+jm*20+jm*lm*ntrace*10+jm*(ntrace+1)*10+
     .           im*jm*lm*ntrace)
      real amjl(jm,lm,4),aj(jm,20),ajln(jm,lm,ntrace,10),
     .     consrv(jm,0:ntrace,10),concm(im,jm,lm,ntrace)
      common /m4/ m4first,
     .  amjl,aj,ajln,consrv,concm
c       variable                type    purpose
c       --------                ----    -------
c       m4size                  integer dummy variable - indicates beginning
c                                       of common block m4
c
c       amjl(jm,lm,4)           real    not used
c       aj(jm,20)               real    not used
c       ajln(jm,lm,ntrace,10)   real    not used
c       consrv(jm,0:ntrace,10)  real    tracer/air masses in latitude bands
c                                       for conservation checking
c       concm(im,jm,lm,ntrace)  real    accumulator for averaged mixing
c                                       ratio fields
c-------------------------------------------------------------------------
```

4. USER'S MANUAL

4.1 RUNNING THE MODEL

Running the model involves four principal stages:

1. Programming of a tracer specific source module (subroutine **trace0**);
2. Preparation of job script file with settings of control parameters, definition of input and output files;
3. Submission of the job;
4. Postprocessing of the output data.

4.1.1 The tracer specific source module "trace0"

All tracer specific calculations are performed within module **`trace0`**. This module must be modified depending on the modeled tracer(s) under consideration. The module contains six entry points (which can be replaced by individual subroutines) :

`trace0`	Main entry point, called at model start up. Used to open and read tracer specific source/sink data files.
`trace1`	Called at the beginning of a model run only if `istart=2` is selected. Used to set the tracer fields to some initial, non-zero values.
`source1`	Called every `nsource` seconds for calculations assigned to "source" process 1.
`source2`	Called every `nsource` seconds for calculations assigned to "source" process 2.
`chem1`	Called every `nchem` seconds for calculations assigned to "chemistry" process 1.
`chem2`	Called every `nchem` seconds for calculations assigned to "chemistry" process 2.

Note that the calculations assigned to "source" and "chemistry" processes 1 and 2 can be chosen arbitrarily. The diagnostic routines simply keep track and print out separately the tracer conservation statistics for each of the four generic source/sink processes.

Within `trace0` the tracer masses and their slopes must be modified according to the type of source/sink process considered. The modification of the tracer mass by a source/sink process operating during the source timestep Δt_{source} (`nsource` or `nchem`) is straightforward. The modification of the slopes requires additional assumptions about the exact location within each grid box where the tracer is input by the source process. Based on this information the three-dimensional change in mixing ratio within the grid boxes must be determined, which subsequently has to be fit by a three-dimensional linear function. The modification of the slopes is then determined using the definition of the slopes (eq. 2.1.4.2). Clearly, if no additional information is available or if it is assumed that the source process is uniformly distributed within a particular grid box then the slopes do not need any modification.

A few standard cases are described below.

A volume source/sink

If we assume that the strength, Q_v, of the volume source/sink within a particular grid box is given in units of (kg tracer)(kg air)$^{-1}$s^{-1} then the change of the tracer mass n in the grid box during the source time step Δt_{source} is given by

$$\Delta n = \Delta t_{source} Q_v m \tag{4.1.1.1}$$

where m is the air mass of the grid box. Assuming further that the volume source is distributed uniformly within the grid box, then the tracer mass slopes need no modification.

A source/sink proportional to the mixing ratio of the tracer

In the case of a first order growth (or decay) process we are given a growth (or decay) rate constant λ (expressed in s^{-1}). The change in tracer mass during the source time step Δt_{source} is then:

$$\Delta n = \pm \Delta t_{source} \lambda n \tag{4.1.1.2}$$

In this case the slopes of the tracer are also affected by the source:

$$\begin{aligned} \Delta n_x &= \pm \Delta t_{source} \lambda n_x \\ \Delta n_y &= \pm \Delta t_{source} \lambda n_y \\ \Delta n_z &= \pm \Delta t_{source} \lambda n_z \end{aligned} \tag{4.1.1.3}$$

In the case of a large decay rate constant i. e. if $\lambda \Delta t \geq 0.5$, an implicit formulation might be required in order to prevent instabilities and/or negative concentrations. For the change in tracer mass the implicit form is

$$\Delta n = -\frac{\Delta t_{source} \lambda}{1 + \Delta t_{source} \lambda} n \tag{4.1.1.4}$$

and similar equations hold for the slopes.

A surface source/sink

In the case of a surface source/sink, Q_s, expressed in (kg tracer) m^{-2} s^{-1}, we assume that the tracer is inserted into the lowest model grid box at the lower boundary. The tracer mass in the gridbox is modified during the source time step by

$$\Delta n = \Delta t_{source} Q_s \Delta x \Delta y \tag{4.1.1.5}$$

where $\Delta x \Delta y$ denote the surface area of the grid box.

In this case the slopes also have to be modified. If the surface source/sink is assumed to be uniform within the grid box area then only the vertical slope is modified. The change in the vertical slope is found by fitting a least squares straight line to the change in the vertical tracer profile as generated by the surface

source. Assuming that the tracer is inserted as a spike at the lower box boundary it is easy to show that the least squares change in the vertical slope is given by

$$\Delta n_z = -3\Delta t_{source} Q_s \Delta x \Delta y \qquad (4.1.1.6)$$

This slope change is very large and may induce negative tracer concentrations. A more conservative approach limits the change in the vertical slope such that negative concentrations cannot occur

$$\Delta n_z = -\Delta t_{source} Q_s \Delta x \Delta y \qquad (4.1.1.7)$$

A simple example

The simple case listed below illustrates the code for reading at startup (entry `trace0`) a surface source field `tsource.d`, which contains the tracer input (in kg tracer per gridcell and hour). Subsequently, whenever entry `source1` is called, the appropriate amount of the tracer is input into the lowest model layer and, since this is a surface source, the vertical slope is modified.

```
      subroutine trace0
c-----------------------------------------------------------------------
c  this subroutine performes all calculations associated with tracer
c  specific sources and sinks. It also allows a specific initialization
c  of the tracer fields.
c
c  entry trace0  is called always from module 'start.f', and provides
c                a way to input tracer specific values.
c  entry trace1  is called from 'start.f' when istart = 2, and provides a
c                way to do some initialization, that have to be done only
c                at the beginning of a run.
c  entry source1 is called every nsrce seconds for calculation of source1
c                calculation.
c  entry source2 is called every nsrce seconds for calculation of source2
c                calculation.
c  entry chem1   is called every nchem seconds for calculation of chemistry1
c  entry chem2   is called every nchem seconds for calculation of chemistry2
c
c-----------------------------------------------------------------------
c This particular version of trace0 reads at the beginning of a run (or a
c continuation run) a constant source field "tmass" from file "tsource.d" which
c supposedly contains the tracer source (in units of kg tracer per hour
c and gridbox). In entry "source1" this source is input into the lowest
c model layer of the model. This example assumes only one tracer.
c
c                                       mh, 30-oct-1993
c-----------------------------------------------------------------------
      implicit none
c save ensures that all local variables keep their contents between
c subsequent entries into this module

      save

      include 'constants'
      include 'common'

      integer i,j,l

      real scf,sx,x
      real tmass(im,jm)
```

```
c----
c     trace0 is called at startup of the model
c----
      open(unit=40,file='tsource.d',status='old')
      read(40,*) tmass
      close(unit=40)

      return

      entry trace1
c----
c     this entry determines the initial values for tracer mass
c     and its slopes. It is called from 'start.f' when istart = 2.
c----
c
c
      return
c----
c----
      entry source1
c----
c     this entry changes the tracer mass and its slopes by
c     source process 1
c----

c       array tmass contains the source in kg tracer per gridbox and
c       per hour. Since this module is called every nsrce seconds
c       a scaling factor is needed:
        scf=nsrce/3600.

c                               this is a surface source, therefore
c                               add tracer to first layer and change
c                               the vertical tracer slope
        do 20 j=1,jm
          sx=0.
          do 19 i=1,im
              x=scf*tmass(i,j)
              rm(i,j,1,1)=rm(i,j,1,1)+x
              rzm(i,j,1,1)=rzm(i,j,1,1)-x
              sx=sx+x
19        continue
c                               array aj accumulates the total source
c                               input into the model in each latitude
c                               band. This information is however not
c                               used in model version 8.4
          aj(j,1)=aj(j,1)+sx
20      continue
      idacc(6)=idacc(6) + 1
      return
c----
      entry source2
c----
c     this entry changes the tracer mass and its slopes by
c     source process 2
c----

      return
c----
      entry chem1
c----
c     this entry changes the tracer mass and its slopes by chemistry
c     process 1
```

```
c----

      return
      entry chem2
c----
c     this entry changes the tracer mass and its slopes by chemistry
c     process 2
c----

      return
      end
```

4.1.2 Compilation of model code

All source code files including the files `constants`, `common` and `trace0.f` are best copied into a separate working directory. On the Cray™ systems the following command will produce an executable version (`tmodel`) of the tracer model:

```
cf77 *.f -o tmodel
```

4.1.3 Job script and control input parameters

TM2 reads from standard input the following information::

Lines 1-2: header information for the model run.

Line 3: pathname of file with air mass fluxes.

Line 4: pathname of convection information file.

Line 5: pathname of landfraction file

Subsequent lines: Namelist `inputz` which allows the setting of control variables in common block `m1` (see Section 3.5.3)

4.1.4 An example of a job script

The following is a simple job to run the model for one year.

```
# QSUB -eo
# QSUB -x
# QSUB -r demojob.job
# QSUB -o demojob.log
# QSUB -lT 800
# QSUB
#
#-------------------------------------------------------------------
#                                       Define run-code
setenv run "demo"
#
#                                       Directory of model code
setenv modeldir "regen:/pf/m/m211037/m/tm8l4"
#
#                          Directory of source modules/files
#                          This must be changed to the actual
#                          directory where i.e. trace0.f is located
setenv sourcedir "regen:/pf/m/m211037/m/co2b"
#
echo "TM-Run:  $run    ‘date‘"
#
```

```
#                       change to working directory
cd $TMPDIR
#
#                       get model and compile it
rcp $modeldir/tracer.f .
rcp $modeldir/advectg.f .
rcp $modeldir/subscalg.f .
rcp $modeldir/controlg.f .
rcp $modeldir/common .
rcp $modeldir/constants_cg_1t .
#
rcp $sourcedir/trace0d.f trace0.f
#
cf77 *.f -o tmodel
#
# copy source files to working directory if needed
# rcp $sourcedir/tsource.d .
#
# get script to retrieve meteodata and plandf file from unitree
rcp $modeldir/getmeteo .
getmeteo 87 cg
#
# run model
tmodel <<endinput
TM2 run ${run}  Demo Run
     performed jan-1-1999
wind.b
sub.b
plandf.d
 &inputz
 icalendo=3
 istart=1
 idatei=1987,1,1,0,0,0
 idatee=1988,1,1,0,0,0
 noindc=2
 indc(1,1)=1,indc(2,1)=1,indc(3,1)=1
 indc(1,2)=5,indc(2,2)=12,indc(3,2)=4
 ncheck=86400
 fscale(1)=2.4167e6
 ndiagp1=-2
 ndiagp2=-2
 nwrite=31536000
 ndyn=14400
 nconv=14400
 nchem=14400
 nsrce=14400
 czeta=1.
 czetak=1.
 &end
endinput
#
echo "TM model run completed --- ‘date‘"
#
# save output into $outputdir:
#
setenv outputdir $tmp
#
# convert fortran carriage control to standard formfeeds
#
asa <tables1.d >$TMPDIR/tables1x.d
cp $TMPDIR/tables1x.d $outputdir/tables1_$run.d
#
```

```
asa <tables2.d >$TMPDIR/tables2x.d
cp $TMPDIR/tables2x.d $outputdir/tables2_$run.d
#
cp check.d    $outputdir/check_$run.d
cp mmix.b     $outputdir/mmix_$run.b
cp mix.b      $outputdir/mix_$run.b
cp save.b     $outputdir/save_$run.b
#
echo "End of Job --- ‘date‘"
```

4.2 INPUT FILES

The model requires the following standard input data files (in addition to any data files that might be needed for the specific tracer(s)). Indicated sizes are given in Megawords (MW), where each word represents a real number. On a 64bit machine (e.g. Cray) one word thus corresponds to 8 bytes, on a 32bit machine (e.g. Sun workstation) one word corresponds to 4 bytes.

`wind.b` Air mass fluxes on the model grid (binary, ca. 12.5MW per year for coarse grid version with 12hourly time step of mass flux fields),

`sub.b` Convection information (detrainment and entrainment rates, vertical diffusion coefficients) on the model grid. (binary, ca. 17.5MW per year for coarse grid version with 12-hourly time step of convection information),

`plandf.d` Fraction of land in each grid cell. This file is not needed by the model code directly, but this information can be useful in the tracer specific source module `trace0`.

`restart.b` Complete model state (control variables, tracer mass and accumulator arrays). Binary. Needed only for a model restart (selected with `istart`=10).

`saveold.b` Complete model state (control variables, tracer mass and accumulator arrays). Binary. Needed if tracer masses are to be initialized from the results of a previous model run (selected with `istart`=3).

4.3 MODEL OUTPUT

4.3.1 Diagnostic output

The tracer model writes several datasets with diagnostic information:

`Logfile` A summary of the values of control variables and of tasks performed during the model run is written on standard output.

`tables1.d` Tables with conservation statistics of the tracer(s) and zonally averaged tracer mixing ratios.

`tables2.d` Tables with conservation statistics of the air mass.

`check.d` Mixing ratio time series at a set of check locations.

`debug.d` Detailed protocol of model run.

Logfile

The logfile of the job contains a detailed description of the actions performed during the model run (e.g. values of control parameters used in the run, start and stop of the model etc.). An annotated example is given below.

```
------------------------------------------------------------------------
Warning: no access to tty; thus no job control in this shell...
TM-Run:  001d    Tue Sep  7 09:04:51 DST 1993
tracer.f:
trace0.f:
advecg.f:
subscalg.f:
controlg.f:
diagnosg.f:
obtaining meteo fields --- Tue Sep  7 09:06:29 DST 1993
for year 87 and grid cg from unitree directory /pf/m/m211037/mdata
/wrk/nqs.+++++4KE9
local: wind.b remote: wind87bn.b
file retrieved
local: sub.b remote: sub87bn.b
file retrieved
local: plandf.d remote: plandf.d
file retrieved
/wrk/nqs.+++++4KE9
total 56836
-rw-------   1 m211037  b34002     13997 Sep  7 09:15 plandf.d
-rw-------   1 m211037  b34002   136548960 Sep  7 09:15 sub.b
-rw-------   1 m211037  b34002   96103880 Sep  7 09:09 wind.b
1
     Global Atmospheric Tracer Model TM2, Version 8.4
 Max-Planck-Institut fuer Meteorologie, D-2000 Hamburg 13

 TM2 run 001d  A Demo Run
      A simple example                                                  8.4
 ------------------------------------------------------------------------
 reading mass flux fields from file :
/wrk/nqs.+++++4KE9/wind.b
 reading subscal info fields from file :
/wrk/nqs.+++++4KE9/sub.b
 reading landfraction info field from file :
/wrk/nqs.+++++4KE9/plandf.d
  &inputz
  icalendo=3
  istart=1
  idatei=1987,1,1,0,0,0
  idatee=1998,1,1,0,0,0
  noindc=2
  indc(1,1)=1,indc(2,1)=1,indc(3,1)=1
  indc(1,2)=5,indc(2,2)=12,indc(3,2)=4
  ncheck=86400
  fscale(1)=2.4167e6
  ndiagp1=-2
  ndiagp2=-2
  nwrite=31536000
  ndyn=14400
  nconv=14400
  nchem=14400
  nsrce=14400
  czeta=1.
  czetak=1.
```

```
 &end
 1987-jan- 1   0: 0: 0    read mass fluxes
 control parameters:
 &INPUTZ  ISTART = 1,  NDYN = 14400,  NCONV = 14400,  NDIAG = 43200,
 NCHEM = 14400,  NSRCE = 14400,  NREAD = 43200,
  NWRITE = 31536000,  NINST = 0,  NCHECK = 86400,  NDIFF = 0,  ICALENDO = 3,
  IYEAR0 = 1980,  IDATEI = 1987,  2*1,  3*0,
  IDATEE = 1991,  2*1,  3*0,  IDATET = 1980,  2*1,  3*0,  NDIAGP1 = -2,
  NDIAGP2 = -2,  CZETA = 1.0,  CZETAK = 1.0,
  DSCALE = 250000.0,  FSCALE = 2416700.0,  NOINDC = 2,
  INDC = 3*1,  5,  12,  4,  24*0,  CDEBUG = F,  &END
 1987-jan- 1   0: 0: 0    model start up complete
 1987-jan- 1   0: 0: 0    model status saved on unit   1
 1987-jan- 1   0: 0: 0    read mass fluxes
 1987-jan- 1   0: 0: 0    read convinfo
 max courant: 0.5248455836223,  0.4550566720212,  0.8995209133343
 approx. cpu(s)/yr: 167.5620958065
 max courant: 0.6187645286795,  0.4247402728316,  1.035181207091
 approx. cpu(s)/yr: 167.9622027344
 max courant: 0.6106787129583,  0.41581962929,  1.127631425898
 approx. cpu(s)/yr: 167.3584974194
 max courant: 0.5728627992465,  0.429060083934,  0.9273945660685
 approx. cpu(s)/yr: 167.2246242917
 max courant: 0.5976953099661,  0.4201883239467,  0.8308831712863
 approx. cpu(s)/yr: 167.4968517843
 max courant: 0.5588564526637,  0.4194243602377,  0.6904747023827
 approx. cpu(s)/yr: 167.7801220521
 max courant: 0.8021882632903,  0.4743169779536,  1.273082802353
 approx. cpu(s)/yr: 167.6579929839
 max courant: 0.5831089628295,  0.5502029877368,  0.5842571780405
 approx. cpu(s)/yr: 167.3897845968
 max courant: 0.6488052923099,  0.4215924112803,  0.8071766089785
 approx. cpu(s)/yr: 166.6620569688
 max courant: 0.6652703837046,  0.4516391476343,  0.7843785212473
 approx. cpu(s)/yr: 166.6888170161
 max courant: 0.5793283814902,  0.3999424746342,  0.6492396380216
 approx. cpu(s)/yr: 167.6824689896
 max courant: 0.5157829247041,  0.449052394582,  0.8168995507033
 approx. cpu(s)/yr: 167.0823566633
 1988-jan- 1   0: 0: 0    model status saved on unit   1
 1988-jan- 1   0: 0: 0    model status saved on unit   3


 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 program has terminated normally.
 1988-jan- 1   0: 0: 0    final time
 no of time steps:     2190 cpu (s):          670.10 cpu/step:       0.07649497


 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------

 STOP  (called by EXITUS )
 CP: 670.128s,  Wallclock: 2084.808s,  10.7% of 3-CPU Machine
 HWM mem: 587468, HWM stack: 43670, Stack overflows: 0
End of Job --- Tue Sep  7 09:51:07 DST 1993
---------------------------------------------------------------------
```

Conservation statistics of the tracer(s)

The file `tables1.d` contains tables of various conservation statistics for each tracer. For each latitude interval of the model grid, for each hemisphere and for the entire globe the instantaneous tracer masses and the changes due to the individual transport and source/sink processes are listed. Furthermore, for each tracer *i* a simple latitude-height table is written of the zonally averaged tracer mixing ratio defined as “kg tracer/kg air” multiplied by the factor `fscale(i)`. The control variable `ndiagp1` selects the output and averaging time interval for the compilation of these tables.

Conservation statistics of the air masses

The conservation statistics of the air mass is written into a set of tables similar to those for the individual tracers (file `tables2.d`). The output interval of these tables is also controlled by the value of `ndiagp1`. These tables are determined directly from the input air mass flux fields. They may be inspected for checking the conservation of air mass of the air mass flux fields.

Mixing ratio time series at check locations

Up to 10 check locations can be defined using control array `indc(3,10)` which is included in common block `m1`. Thereby the triplet (`indc(1,k), indc(2,k), indc(3,k)`) denotes the (*i*,*j*,*l*) indices of check location *k*, (where *k*=1,`noindc`). Depending on the value of the control variable `ncheck`, time series of the mixing ratios of all tracers are written onto file `check.d`. The mixing ratios are defined as “kg tracer/kg air” multiplied by the factor `fscale(i)` for tracer number *i*.

Debugging output

A detailed protocol of all tasks performed by the model is written on file `debug.d` if the control switch `debug` is set to `True`. This option is inteded primarily for the debugging of the transport model code itself and is of little use during normal operation. The amount of information written is very large. Therefore debugging should be used (if at all) only for short test runs of a few days or less.

4.3.2 Binary output datasets

Two types of binary output datasets are generated: restart files and files containing full mixing ratio fields.

Restart files (`save1.b, save2.b, save.b`)

The model writes restart files in binary form alternatingly on files `save1.b` and `save2.b` at fixed time intervals specified by the control variable `nwrite`. Furthermore, at the end of a run a final restart file `save.b` is written. The restart files contain all the needed information for a restart, i.e. the time information, the control variables contained in the main common block `m1`, the header information, the tracer mass fields and its slopes, and the accumulator arrays contained in common block `m4`.

Restart files may be used for two purposes:

1. A model run may be continued by setting `istart=10`. In this case the model restarts from the restart-file `restart.b`. All control parameter values are read from the restart file and used in the continuation run unless overridden by explicit values specified on the job script in namelist `inputz`.

2. The tracer mass distribution as recorded in a particular restart file can also be used as initial distribution of the tracer in a different model run by setting `istart=3`. During the initialization the model reads in this case from the restart file `saveold.b` only the tracer masses but ignores all the other information stored in the restart file.

Mixing ratio files (`mix.b, mmix.b`)

If requested, the model writes the simulated instantaneous tracer mixing ratio fields (controlled by variable `ninst`) in binary form on the file `mix.b`. The output interval (in seconds) is given by the value of `ninst`. The variable `itaut`, which can be set by specifying the control variable `idatet`, defines the time after which instantaneous output is written on `mix.b`. This allows for economically writing instantaneous output only after some model spin-up period.

The model computes time averaged tracer mixing ratio fields by sampling the tracer masses every `ndiag` seconds. These averaged fields are output in binary form to the file `mmix.b`. The time averaging interval can be selected by the control variable `ndiagp2`.

Tracer mixing ratios as output on `mix.b` and `mmix.b` are defined as “kg tracer/kg air” multiplied by the scaling factor `fscale(i)` for tracer number *i*.

The binary output mixing ratio fields can be read from a postprocessing program in the following way:

```
       parameter (im=36,jm=24,lm=9,ntrace=1)

      real x(im,jm,lm,ntrace)
      integer itau,idate(6)
      ...
      do 1000 k=1,1000
        read(10,end=1999) itau,idate,x
1000  continue
1999  continue
```

`itau` and `idate` contain the time/date stamp of the following mixing ratio field in the standard calendar convention (see section 2.2). Note that in the case of time averaged mixing ratio fields the time/date stamp recorded with each field is the time/date when the field is written, i.e. the first time instant after the averaging time interval. For example in the case of monthly averaged fields, the recorded time/date stamp reflects the beginning of the month immediately after the averaging period.

REFERENCES

[Heimann and Keeling, 1989] Heimann M. and Keeling C.D., 1989. A three dimensional model of atmospheric CO_2 transport based on observed winds: 2. Model description and simulated tracer experiments. In: Peterson DH (ed) *Aspects of Climate Variability in the Pacific and the Western Americas*. American Geophysical Union, Washington DC, 277–303.

[Louis 1979] Louis, J. F., 1979. A parametric model of vertical eddy fluxes in the atmosphere. *Boundary Layer Meteorology, 17*, 187-202.

[Prather et al., 1987] Prather M., McElroy, M., Wofsy, S., Russell, G., Rind, D., 1987. Chemistry of the global troposphere: Fluorocarbons as tracers of air motion. *J. Geophys. Res., 92*, 6579–6613.

[Russell and Lerner, 1981] Russell, G. and Lerner, J., 1981. A new finite-differencing scheme for the tracer transport equation. *J. Appl. Meteo., 20*, 1483–1498.

[Tiedke 1989] Tiedke, M., 1989. A comprehensive mass flux scheme for cumulus parametrization in large-scale models. *Mon. Wea. Rev., 117*, 1779–1800.

APPENDIX A LIST OF SYMBOLS

Table A.1 List of symbols

Symbol		Defining Equation	Units
ρ	Density of air	(2.1.1.1)	kg m^{-3}
Q	General tracer volume source	(2.1.1.1)	kg tracer m^{-3} s^{-1}
χ	Tracer mixing ratio	(2.1.1.1)	kg tracer $(kg\ air)^{-1}$
$\vec{u}$	Three-dimensional windvector	(2.1.1.1)	m s^{-1}
$\vec{u}_h$	Horziontal windvector	(2.1.1.3)	m s^{-1}
w	Vertical velocity	(2.1.1.3)	m s^{-1}
σ	Vertical coordinate	(2.1.3.1)	
n	Tracer mass (in gridbox)	(2.1.4.1)	kg tracer
m	Air mass in gridbox	(2.1.4.1)	kg air
n_x, n_y, n_z	Tracer mass slopes	(2.1.4.2)	kg tracer
F_{vsub}	Subgridscale vertical tracer flux	(2.1.5.2)	kg tracer m^{-2} s^{-1}
C	Convection matrix	(2.1.5.9)	
M_v	Updraft air mass flux	(2.1.5.13)	kg air m^{-2} s^{-1}
M_d	Downdraft air mass flux	(2.1.5.13)	kg air m^{-2} s^{-1}
F_u	Tracer flux in updraft	(2.1.5.13)	kg tracer m^{-2} s^{-1}
F_d	Tracer flux in downdraft	(2.1.5.13)	kg tracer m^{-2} s^{-1}
F_s	Tracer flux induced by subsidence	(2.1.5.13)	kg tracer m^{-2} s^{-1}
D_u	Detrainment rate of air out of updraft	(2.1.5.14)	kg air m^{-3} s^{-1}
E_u	Entrainment rate of air into updraft	(2.1.5.14)	kg air m^{-3} s^{-1}
E_d	Entrainment rate of air into downdraft	(2.1.5.16)	kg air m^{-3} s^{-1}
D_d	Detrainment rate of air out of downdraft	(2.1.5.16)	kg air m^{-3} s^{-1}

Table A.1 List of symbols

Symbol		Defining Equation	Units
$E_{u,\,l}$	Air mass entrainment into updraft at height level l	(2.1.5.25)	kg air s^{-1}
$D_{u,\,l}$	Air mass detrainment out of updraft at height level l	(2.1.5.25)	kg air s^{-1}
F_{vdiff}	Tracer flux induced by vertical diffusion	(2.1.5.35)	kg tracer m^{-2} s^{-1}
K_v	Vertical diffusion coefficient	(2.1.5.35)	m^{-2} s^{-1}
i	Zonal index	(3.1.1)	
j	Meridional index	(3.1.2)	
l	Vertical index	Table 1	
λ	Tracer specific growth or decay rate	(4.1.1.2)	s^{-1}

APPENDIX B METEOROLOGICAL DATA PREPROCESSING

B.1 AIR MASS FLUXES

TM2 requires as input fields the horizontal air mass fluxes crossing the gridboundaries during each meteorological time step. These fluxes are calculated from fields of geopotential heights, horizontal wind velocity and surface pressure. The data processing is performed in three steps, described in the following subsections.

B.1.1 Interpolation/integration of the horizontal mass fluxes over the sides of the gridboxes

For each instant of time of the original meteorological analyses the horizontal mass fluxes are obtained by vertical-meridional, and vertical-zonal integration of $\rho\vec{u}_h$ using trapezoidal integration formulae.

B.1.2 Adjustment of proper time staggering

TM2 assumes that the mass flux fields are valid during an entire meteorological time step, while the surface pressure, which determines the atmospheric mass distribution, is defined at the beginning and at the end of a meteorological time step. In practice, the meteorological analyses contain all fields defined for the same instant of time. The required temporal staggering of the air mass fluxes is obtained by time averaging the mass fluxes computed from the analyses at the beginning and at the end of the meteorological timestep.

B.1.3 Adjustment for conservation of air mass

The resulting horizontal mass flux fields after the dataprocessing in general do not conserve mass exactly. Therefore an adjustment procedure is applied as described in Heimann and Keeling [1989].

The conservation of mass requires that the vertically integrated air mass convergence equals the surface pressure (p_s) tendency:

$$\frac{\partial p_s}{\partial t} = -g\int_{z_{surf}}^{z_{top}} \nabla \cdot \rho\vec{u}_h dz \qquad \text{(B.1.3.1)}$$

Denoting by $\vec{F}$ the vertically integrated air mass flux and by M the vertically integrated air mass, equation (B.1.3.1) is reexpressed as

$$\frac{dM}{dt} = -\nabla \cdot \vec{F} \qquad \text{(B.1.3.2)}$$

We set

$$\vec{F} = \vec{F}_{obs} + \vec{F}_{corr} \tag{B.1.3.3}$$

where $\vec{F}_{obs}$ are the air mass fluxes as determined from the analyses and $\vec{F}_{corr}$ represents a correction air mass flux to be determined in order to fulfill equation (B.1.3.2).

Expressing $\vec{F}_{corr}$ as the gradient of a scalar potential Θ:

$$\vec{F}_{corr} = \nabla\Theta \tag{B.1.3.4}$$

we obtain the Poisson equation

$$\nabla^2\Theta = -\nabla \cdot \vec{F}_{obs} - \frac{dM}{dt} \tag{B.1.3.5}$$

which has to be solved on a horizontal grid spanning the entire globe. The boundary conditions are that the meridional component of the gradient of Θ vanish at the poles and that Θ be periodic in the zonal direction.

The discrete version of equation (B.1.3.5) is solved efficiently using two-dimensional Fourier transforms. This yields the vertically averaged correction air mass flux which subsequently is distributed uniformly in the vertical dimension.

B.1.4 air mass flux processing programs

The three preprocessing steps described above are performed using the following programs:

`mflux` Interpolation/integration of air mass fluxes.

`cflux` Time staggering.

`pascha` Adjustment for conservation of mass.

Note that, for historical reasons, within these programs the direction of the vertical coordinate grid is reversed compared to TM2, i.e. the vertical levels start at the top of the atmosphere (l=1) and end at the surface ($l=l_{surf}$)!

B.2 SUBGRID SCALE TRANSPORT INFORMATION

TM2 requires as input global fields the vertical diffusion coefficient and the parameters for the cumulus cloud transport (entrainment and detrainment rates into updraft and downdrafts). These are calculated in the preprocessing stage from meteorological analyses of geopotential, surface pressure, horizontal wind, temperature and relative humidity. In addition, surface fluxes of latent heat are needed.

All subgrid scale transport parameters are calculated in every grid column of the original grid of the meteorological analyses. Subsequently the parameters are averaged onto the coarser grid of TM2.

B.2.5 Vertical diffusion coefficients

The stability dependent vertical diffusion coefficients are calculated using the formulae from Louis [1979] as implemented in the ECMWF operational model [ECMWF manual]. The diffusion coefficient is given by

$$K_v = l_h^2 \left|\frac{\partial u}{\partial z}\right| f_h(Ri) \tag{B.2.5.6}$$

where *Ri* is the Richardson number. It is calculated at the layer boundary *k*+1/2 from the analysis variables on the adjacent level centers

$$Ri_{k+1/2} = g(z_k - z_{k+1})\frac{c_{pd}(T_k - T_{k+1}) + g(z_k - z_{k+1})}{|\vec{u}|^2_{k+1/2} c_{pd} T_{k+1/2}} \tag{B.2.5.7}$$

l_h is a mixing length which is calculated as a function of height *z* above ground from

$$l_h = \frac{kz}{1 + \frac{kz}{\lambda_h}} \tag{B.2.5.8}$$

The stability function $f_h(Ri)$ is computed according to the sign of *Ri*. In the stable case (*Ri* > 0):

$$f_h(Ri) = \frac{1}{1 + 3bRi\sqrt{1 + dRi}} \tag{B.2.5.9}$$

In the unstable case (*Ri* <= 0):

$$f_h(Ri) = 1 - \frac{3bRi}{1 + G(Ri)} \tag{B.2.5.10}$$

and the function *G*(*Ri*) is given by

$$G(Ri) = 3bCl_h^2\left[\frac{-Ri}{z}\left(\frac{1}{\Delta z}\left[\left(1 + \frac{\Delta z}{z}\right)^{\frac{1}{3}} - 1\right]\right)^3\right]^{\frac{1}{2}} \tag{B.2.5.11}$$

Δz is the distance between the adjacent model layer centers.

Numerical values for the constants appearing in equations (B.2.5.9)-(B.2.5.11) are given in Table B.1

Table B.1 Parameter values of vertical diffusion formulae

Parameter	Value
b	5.
C	5.

Table B.1 Parameter values of vertical diffusion formulae

Parameter	Value
d	5.
k	0.4
c_{pd}	1005.46 J kg^{-1} K^{-1}
g	9.90665 ms^{-2}
λ_h	438.18 m

B.2.6 Cloud transport parameters

The cloud transport parameter (entrainment and detrainment rates into updraft and downdrafts) are calculated in a simplified form according to the scheme described by Tiedke [1989]. The major simplification is that in the present version (TM2) the downdraft is neglected and set to zero.

The scheme calculates the entrainment and detrainment rates in the following sequence of steps:

1. Detect the cloud base height. This is determined by lifting surface air adiabatically until saturation occurs.
2. Determine the upward water vapour flux at the base of the cloud by adding the large scale horizontal water vapour convergence below the cloud to the evaporation from the surface.
3. Detect presence of cloud. If the calculated upward water vapour flux at the base of the cloud is zero or negative then no cloud is present and the detrainment and entrainment rates are set to zero.
4. Determine if “shallow convection” (large scale horizontal water vapour convergence is negative - in this case convection is driven primarily by the surface evaporation flux only) or “penetrative convection” (otherwise).
5. The air mass flux at the base of the cloud is calculated as the ratio between the water vapour flux at the base of the cloud and the saturation specific humidity at the temperature of the base of the cloud.
6. Set detrainment rates below the cloud base to zero. Entrainment rates below the cloud are set proportional to the below cloud layer thicknesses, such that the sum of the below cloud entrainment rates equal the mass flux at the base of the cloud.
7. Compute cloud parcel ascent within the cloud. Initialize cloud temperature, moisture and liquid water content with values at the base of the cloud. During the ascent in each cloud layer the following sequence is computed:
   (a) Calculate air entrainment and detrainment on layer (set proportional to updraft mass flux with constants according to Tiedke [1989]).
   (b) Adjust cloud parcel temperature, moisture and liquid water content by condensation to force

water vapour at saturation point.

(c) Calculate precipitation out of cloud parcel and adjust liquid water content.

(d) Check for buoyancy of cloud parcel with respect to environmental air.

8. If the cloud parcel is no longer buoyant detrain the remaining cloud air mass flux above the cloud: in the case of penetrative convection 100% are detrained into the first layer above the cloud, and in the case of shallow convection 70% are detrained into first and 30% into the second layer above the cloud.

B.2.7 Programs

The parameters of the subgrid scale transport are calculated in the following programs and subroutines:

`suball`	Main program.
`louis`	Calculates vertical diffusion coefficients
`cloud`	Calculates cloud entrainment and detrainment rates.

Note that, for historical reasons, within these programs the direction of the vertical coordinate grid is reversed compared to TM2, i.e. the vertical levels start at the top of the atmosphere (l=1) and end at the surface ($l=l_{surf}$)!

APPENDIX C COMPUTER IMPLEMENTATION

C.1 MACHINE DEPENDENT ISSUES

TM2 is programmed entirely in the FORTRAN77 programming language without any machine specific language extensions. The subroutines listed in Table C.1 are installation dependent and are not part of the standard model source code distribution.

Table C.1 Machine specific subroutines

	Subroutine	Purpose
TM2:	second	returns machine system time in seconds
Preprocessing:	`C06FJF`	NAG FFT subroutine

The code has been tested on machines with 64bit and 32bit words. It assumes that integer as well as real variables occupy one machine word. This is of important for the main common blocks which are equivalenced to dummy arrays in several subroutines for ease of writing and reading to and from disk.

The simulation time expressed in seconds is kept in integer variables. On a 32bit machine this imposes a maximum simulation time of approximately 68 years until overflow occurs. There are no checks in the code to prevent this overflow.

C.2 CPU REQUIREMENTTYPICAL CPU REQUIREMENTS ONYPICAL CPU REQUIREMENTS ONTS

Typical CPU requirements for the simulation of one inert tracer for a whole year are given in Table C.2. The code vectorizes moderately well. On a single Cray C90 processor the the code with the 5x4x9 resolution achieves an average performance of 220 MFLOP, the standard 10x8x9 version 170 MFLOP.

Table C.2 Typical CPU requirements on different machines (on one processor)

Machine	Model Version	Resolution	Timestep (s)	CPU (s)[a]
Cray-YMP	8.4	10x8x9	14400	170
Cray-YMP	8.4	5x4x9	3600	1800
Cray-YMP	8.4	10x8x19	3600	850
Cray-C90	8.5	10x8x9	14400	45
Cray-C90	8.5	5x4x9	3600	500

a. CPU seconds required for the simulation of one inert tracer for one year.

C.3 LOCAL INSTALLATION AT THE DKRZ IN HAMBURG

At the DKRZ the model source code is located on the department server `regen` in the directory

/pf/m/m211037/m/tm8l5/source

This directory contains also `constants` files configured for one or two tracers and for the “coarse grid” (cg) or “fine grid” (fg) version.

Example jobs are located in the directory

/pf/m/m211037/m/tm8l5/demo

A script `getmeteo` which loads the air mass flux fields from the file archive is found in the directory

/pf/m/m211037/m/tm8l5/scripts

Currently (April 1994) air mass flux and subgridscale parametrizations are available for the years and resolutions as listed in Table C.3.

Table C.3 Available preprocessed ECMWF meteorology to run TM2

Year	Resolution[a]
1982-1985	cg[b]
1986	cg, fg
1987	cg
1990	cge

a. cg = 10x8x9,fg = 5x4x9, cge = 10x8x19
b. uses precomputed monthly averaged convection matrices

APPENDIX D MODEL USERS

Table D.1 Major model users (incomplete)

Institution	Contact (email)
Centre des Faibles Radioactivitées, Gif-sur-Yvette, France	P. Monfray (monfray@asterix.saclay.cea.fr) M. Ramonet (ramonet@asterix.saclay.cea.fr) Y. Balkansky (balkany@asterix.saclay.cea.fr)
Max-Planck-Institut für Luftchemie, Mainz, Germany	R. Hein (hein@mpch-mainz.mpg.d400.de)
Max-Planck-Institut für Meteorologie, Hamburg, Germany	M. Heimann (heimann@dkrz.d400.de) S. Rehfeld (rehfeld@dkrz.d400.de)
Scripps Institution of Oceanography, La Jolla, CA, USA	S. Piper (piper@cdrgsun.ucsd.edu)
KNMI, De Bilt, The Netherlands	H. Kelder P. v. Velthoven (vanvelth@knmi.nl)

An electronic mailinglist is maintained at the DKRZ in Hamburg in order to provide communication between the TM2 model users. The address of the mailing list is `tmlist@dkrz.d400.de`.