MAX-PLANCK-GESELLSCHAFT

The Global Atmospheric Tracer Model TM3

Model Description and User's Manual
Release 3.8a

Martin Heimann and Stefan Körner

Max-Planck-Institut für Biogeochemie, Jena

Technical Reports – Max-Planck-Institut für Biogeochemie 5, 2003

Editor of this report:

Stefan Körner
Max-Planck-Institut für Biogeochemie
Postfach 10 01 64
07701 Jena
Germany
Tel.: +49 (0)3641 576 352
E-mail: Stefan.Koerner@bgc-jena.mpg.de
http://www.bgc-jena.mpg.de

Following authors from the Max Planck Institute for Biochemistry contributed to this manual release of June 11, 2003:

Martin Heimann (original program author)
Stefan Körner (System Description, User's Manual, transport diagnostics)
Ina Tegen (many hints and suggestions, proofreading)
Martin Werner (many hints and suggestions, proofreading)

Title picture: William Turner, "Rain, Steam and Speed" (1844)

Contents

Introduction

The global atmospheric tracer model TM3 is a three-dimensional transport model that solves the continuity equation for an arbitrary number of atmospheric tracers. It uses meteorological output fields from an atmospheric general circulation model or from a weather forecast model.

The tracer advection is calculated using the "slopes scheme" of Russell and Lerner [Russel and Lerner, 1981]. Vertical transport due to convective clouds is computed using a cloud mass flux scheme by Tiedke [Tiedke, 1989]. Turbulent vertical transport is calculated by stability dependent vertical diffusion according to Louis [Louis, 1979].

The model is coded in Fortran 95 and available as source code. For a quick start see section 3.1 on page 20.

Chapter 1

Model Description

1.1 Meteorological Input Data

TM3 solves the continuity equation based on given time-dependent two- or three-dimensional meteorological fields of the surface pressure, wind velocity, air temperature, specific humidity and geopotential. Additionally, evaporation fluxes are needed in order to calculate the sub-gridscale transport by cumulus clouds (see section 1.2.3 and appendix A.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 timestep of 12 hours or shorter.

1.2 Model Physics

The model equations of TM3 are formulated in Eulerian coordinates. For their derivation in this section a Cartesian coordinate system $\{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$ is used for simplicity. The actual model coordinate system is displayed in section 2.2.

1.2.1 Continuity Equation

The atmospheric tracer model TM3 solves the continuity equation for a tracer in flux form

$$\frac{\partial}{\partial t}\rho\chi + \nabla \cdot \rho\vec{u}\chi = Q \tag{1.1}$$

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 grid elements and model timestep by overbars and deviations from these averages with primes, we obtain the equation for the averaged quantities:

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} + \nabla \cdot \bar{\rho}\bar{\vec{u}}\bar{\chi} + \nabla \cdot \bar{\rho}\overline{\vec{u}'\chi'} = \bar{Q} \tag{1.2}$$

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

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} + \nabla_h \cdot \bar{\rho}\overline{\vec{u}_h}\bar{\chi} + \frac{\partial}{\partial z}\bar{\rho}\bar{w}\bar{\chi} + \nabla_h \cdot \bar{\rho}\overline{\vec{u}'_h\chi'} + \frac{\partial}{\partial z}\bar{\rho}\overline{w'\chi'} = \bar{Q} \tag{1.3}$$

where w is the vertical wind velocity and the subscript h refers to horizontal vectors.

The second and third term on the left hand side of equation (1.3) correspond to transport scales that are resolved on the model grid and are termed "advection" (see section 1.2.2). 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 wind and tracer concentration from their grid averages (see section 1.2.4 on page 14). The last term on the left hand side of equation (1.3) represents sub-gridscale 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 1.2.3 on page 9).

The model numerically solves the continuity equation (1.3) in flux form and splits the transport operator in time. Formally, we may view the tracer masses in all the gridboxes as the components of a large vector, $\mathbf{n}(t)$. The discrete version of equation (1.3) is then written as

$$\mathbf{n}(t+\Delta t) = \mathbf{L}_{vconvec}\mathbf{L}_{hdiff}\mathbf{L}_{advec}\left(\mathbf{n}(t) + \mathbf{Q}(t)\Delta t\right) \tag{1.4}$$

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

1.2.2 Advection

Advection, i.e. tracer transport by the three-dimensional airmass fluxes resolved on the model grid, is calculated using the "slopes scheme" developed by Russell and Lerner [Russel and Lerner, 1981]. In this scheme each tracer is represented within the Eulerian gridboxes by the tracer mixing ratio, χ, and and the components of the spatial gradient of the tracer mixing ratio within the gridbox, $(\frac{\partial\chi}{\partial x}, \frac{\partial\chi}{\partial y}, \frac{\partial\chi}{\partial z})$.

For program efficiency the primary variables in the model are not the mixing ratios but the tracer masses, $\mathbf{n}$, and the "slopes" of the tracer mass $(\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z)$ in the gridboxes. Assuming no variations of the air density within a particular gridbox the tracer mass is given by

$$n = \chi m = \chi\rho\Delta x\Delta y\Delta z, \tag{1.5}$$

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

$$n_x = m\frac{\partial\chi}{\partial x}\frac{\Delta x}{2} \tag{1.6}$$

and analogously in the y- and z-direction.

[1]The size of the airmass fluxes determines the maximum permissible value of Δt: The total amount of air moved out of any gridbox during Δt must not exceed the airmass present in the gridbox at the beginning of the timestep (Courant-Friedrichs-Levi criterion).

With the additional information on 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 inacceptable. In regions with steep concentration gradients the calculated tracer mass slopes may become so large that negative tracer masses emerge. If this is deemed unacceptable, this behaviour may be inhibited by limiting the absolute size of the tracer mass slopes (logical input variable limits). This, however, effectively implies a larger numerical diffusion of the advection scheme, but in contrast to some other schemes the transport remains linear.

The slopes scheme requires the airmasses in each gridbox as input data together with the airmass fluxes, i.e. the amount of air crossing the borders of the gridboxes during each time step. For a meteorological dataset (meteorological analyses or climate model output), the fields of airmasses and airmass fluxes are computed on a TM3 model grid during a preprocessing stage (see page 44), stored on disk and then read during tracer model runs.

The three-dimensional advection process is 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 the one-dimensional advection steps of length Δt are further subdivided and performed sequentially in a so-called spatial leap-frog pattern:

1. zonal advection lasting $\frac{1}{4}\Delta t$,
2. meridional advection lasting $\frac{1}{2}\Delta t$,
3. zonal advection lasting $\frac{1}{4}\Delta t$,
4. vertical advection lasting Δt,
5. zonal advection lasting $\frac{1}{4}\Delta t$,
6. meridional advection lasting $\frac{1}{2}\Delta t$,
7. zonal advection lasting $\frac{1}{4}\Delta t$

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. The grid-dependent widths of these bands around south and north poles are hardwired as lat248 in a Fortran source file. It has great influence on the achievable length of the integration timesteps.

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 [Russel and Lerner, 1981]):

1. Calculate the new airmass in the gridbox based on the old airmass and the airmass 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.

1.2.3 Vertical Convection

Sub-gridscale 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 A.2).

Tracer transport by sub-gridscale cumulus clouds is calculated using the massflux scheme of Tiedke [Tiedke, 1989]. Formally this scheme defines a statistical stationary cloud containing an updraft and a downdraft in each vertical grid column. 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 mass is entrained into the up- and downdraft, mixed with the air inside the up- and downdraft and detrained into the surrounding air. Furthermore, the net vertical airmass flux of the up- and downdrafts induces a sub-gridscale subsidence flux of surrounding air outside the cloud. Figure 1.1 shows a schematic sketch of this sub-gridscale transport parametrization.

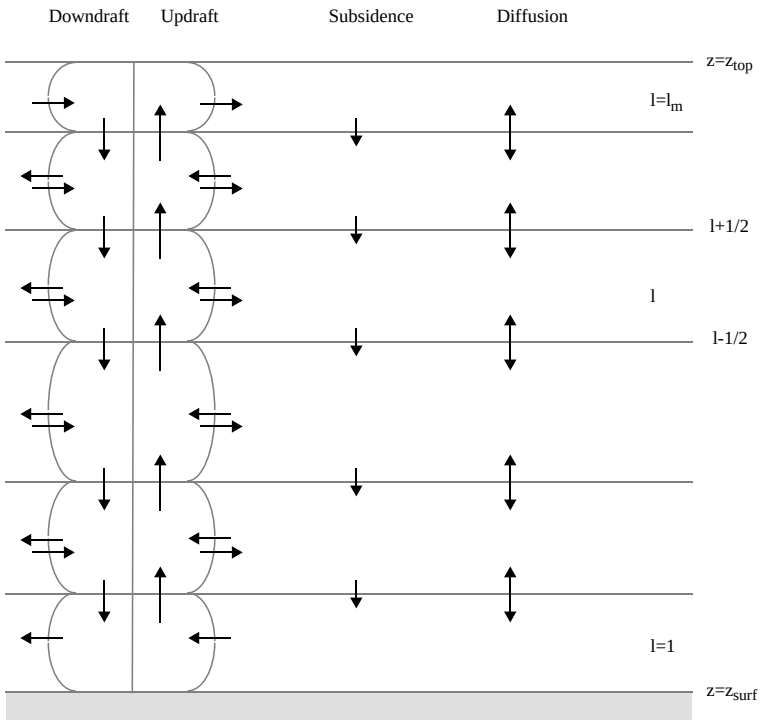


Figure 1.1: Vertical subgridscale processes("vertical convection"), computed in TM3

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 A.2). In a tracer model simulation those coefficients and parameters are 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 the cloud model, with the appropriate en- and detrainment fluxes of the other layers set to zero.

Discretization

Since the tracer transport is linear, the effect of the sub-gridscale transport processes on the tracer masses on the different model levels during one base time step Δt can be represented simply by the multiplication of the vector of tracer masses in a vertical grid column by a "convection matrix" $\mathbf{C}$. The element $[\mathbf{C}]_{lk}$ of this matrix represents the fraction of the tracer mass of the gridbox at level k, which 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 subgrid-scale transport processes. They are recomputed each time the diffusion coefficients and cumulus cloud transport parameters (en- and detrainment rates 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:

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} = -\frac{\partial}{\partial z}\bar{\rho}\overline{\chi' w'} \tag{1.7}$$

during the timestep Δt. The right hand side of equation (1.7) is represented by the divergence of the sub-gridscale tracer fluxes F_{vsub} (defined positive in the upward direction):

$$\frac{\partial}{\partial t}\bar{\rho}\bar{\chi} = -\frac{\partial}{\partial z}F_{vsub} = -\frac{\partial}{\partial z}\left(F_u + F_d + F_s + F_{vdiff}\right) \tag{1.8}$$

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 1.8 over a gridbox at height level l gives

$$\frac{d}{dt}n_l = \left(F_{vsub,l-\frac{1}{2}} - F_{vsub,l+\frac{1}{2}}\right) \tag{1.9}$$

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

$$F_{vsub,l+\frac{1}{2}} = \sum_{k=1}^{l_m} f_{l+\frac{1}{2},k}\, n_k. \tag{1.10}$$

The coefficient $f_{l+\frac{1}{2},k}$ thus represents the fraction of the tracer mass of layer k, which crosses the layer boundary $l+\frac{1}{2}$ by means of sub-gridscale vertical transport processes

per unit time. It is further split into components representing the tracer mass fractions crossing the layer boundary due to each individual sub-gridscale vertical transport process:

$$f_{l+\frac{1}{2},k} = f^u_{l+\frac{1}{2},k} + f^d_{l+\frac{1}{2},k} + f^s_{l+\frac{1}{2},k} + f^{vdiff}_{l+\frac{1}{2},k}. \tag{1.11}$$

Explicit forms of $f^x_{l+\frac{1}{2},k}$ for each of the different vertical sub-gridscale processes are derived in the following subsections.

Equation (1.9) thus can be written for the entire column in matrix form[2]

$$\frac{d}{dt}\mathbf{n} = \mathbf{M} \cdot \mathbf{n}, \tag{1.12}$$

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

$$[\mathbf{M}]_{l,k} = f_{l-\frac{1}{2},k} - f_{l+\frac{1}{2},k}. \tag{1.13}$$

For reasons of stability equation (1.12) is integrated in implicit form:

$$\mathbf{n}(t+\Delta t) = \mathbf{n}(t) + \Delta t \mathbf{M} \cdot \mathbf{n}(t+\Delta t). \tag{1.14}$$

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

$$\mathbf{n}(t+\Delta t) = (\mathbf{I} - \Delta t \mathbf{M})^{-1} \cdot \mathbf{n}(t) = \mathbf{C} \cdot \mathbf{n}(t), \tag{1.15}$$

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

Sub-gridscale transport also affects the slopes of the tracer mass within the gridboxes. The horizontal tracer mass slopes $\mathbf{n}_x$ and $\mathbf{n}_y$ are transported like the tracer mass, i.e. they are updated by

$$\mathbf{n_x}(t+\Delta t) = \mathbf{C} \cdot \mathbf{n_x}(t) \tag{1.16}$$

and

$$\mathbf{n_y}(t+\Delta t) = \mathbf{C} \cdot \mathbf{n_y}(t). \tag{1.17}$$

The vertical slopes are treated differently. It is assumed that the sub-gridscale processes tend to reduce the tracer mass slopes in the vertical direction. In the present version 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

$$n_{z,l}(t+\Delta t) = [\mathbf{C}]_{ll}\ n_{z,l}(t). \tag{1.18}$$

Transport by cumulus clouds

Sub-gridscale transport induced by cumulus clouds is represented by the three components

$$F_u + F_d + F_s = M_u \chi_u + M_d \chi_d - (M_u + M_d)\bar{\chi} \tag{1.19}$$

corresponding to the tracer flux in the updraft (subscript u), in the downdraft (subscript d) and induced by subsidence (subscript s). M_u denotes the updraft airmass flux, M_d the downdraft and $-(M_u + M_d)$ the mass flux by subsidence, all expressed in kg m^{-2} s^{-1}.

[2] In this section the symbol $\mathbf{n}$ refers to the vector of tracer masses of a particular grid column only.

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 A.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:

$$\frac{\partial}{\partial z}M_u = E_u - D_u \tag{1.20}$$

$$\frac{\partial}{\partial z}F_u = E_u\bar{\chi} - D_u\chi_u \tag{1.21}$$

and for the downdraft:

$$\frac{\partial}{\partial z}M_d = E_d - D_d \tag{1.22}$$

$$\frac{\partial}{\partial z}F_d = E_d\bar{\chi} - D_d\chi_d \tag{1.23}$$

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

$$F_u = M_u\chi_u \tag{1.24}$$

$$F_d = M_d\chi_d \tag{1.25}$$

The appropriate boundary conditions are

$$\begin{array}{lll} M_u = F_u = 0 & \text{at } z = z_{surf} \\ M_d = F_d = 0 & \text{at } z = z_{top} \end{array} \tag{1.26}$$

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

$$\int_{z_{surf}}^{z_{top}}(E_u - D_u)dz = 0 \tag{1.27}$$

$$\int_{z_{surf}}^{z_{top}}(E_d - D_d)dz = 0 \tag{1.28}$$

Using 1.24 and 1.25, the equations for the updraft and downdraft tracer fluxes (1.21) and (1.23) can be expressed as

$$\frac{\partial}{\partial z}F_u = E_u\bar{\chi} - D_u\frac{F_u}{M_u} \tag{1.29}$$

$$\frac{\partial}{\partial z}F_d = E_d\bar{\chi} - D_d\frac{F_d}{M_d} \tag{1.30}$$

Updraft

The discretized form of the equation for the updraft airmass flux, (1.20), is obtained by integrating over a gridbox at height level l

$$M_{u,l+\frac{1}{2}} - M_{u,l-\frac{1}{2}} = E_{u,l} - D_{u,l} \tag{1.31}$$

where $E_{u,l}$ and $D_{u,l}$ denote the airmass entrainment and detrainment, respectively (in units of kg s^{-1}), and $M_{u,l+\frac{1}{2}}$ denotes the updraft airmass flux at the model layer boundary $z_{l+\frac{1}{2}}$ (in units of kg $^{-1}$). Similarly the discrete version of the equation for the updraft tracer mass flux, (1.29), becomes

$$F_{u,l+\frac{1}{2}} - F_{u,l-\frac{1}{2}} = E_{u,l}\bar{\chi}_l - D_{u,l}\frac{F_{u,l+\frac{1}{2}}}{M_{u,l+\frac{1}{2}}} \tag{1.32}$$

where we have used an implicit formulation on the right hand side, for stability reasons. Solving for $F_{u,l+\frac{1}{2}}$ and using (1.31) we obtain

$$F_{u,l+\frac{1}{2}} = (F_{u,l-\frac{1}{2}} + E_{u,l}\frac{n_l}{m_l})(1 - \frac{D_{u,l}}{M_{u,l-\frac{1}{2}} + E_{u,l}}) \tag{1.33}$$

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

Equations (1.31) and (1.33) represent recursive relations that may be solved by starting from the surface using the boundary conditions (1.26) and working upwards. It is easily seen that, using this procedure, $F_{u,l+\frac{1}{2}}$ at any layer boundary is a linear function of the tracer masses $n_l, l = 1, \ldots, l_m$ of all the gridboxes in the grid column under consideration, in the form of equation (1.10).

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

$$\begin{aligned} f^u_{\frac{1}{2},k} &= 0 \quad \forall\ k = 1, \ldots, l_m \\ f^u_{l+\frac{1}{2},k} &= (f^u_{l-\frac{1}{2},k} + \delta_{k,l}\frac{E_{u,l}}{m_l})(1 - \frac{D_{u,l}}{M_{u,l-\frac{1}{2}} + E_{u,l}}) \end{aligned} \tag{1.34}$$

where

$$\delta_{k,l} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{otherwise} \end{cases} \tag{1.35}$$

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

$$\begin{aligned} f^d_{l_m+\frac{1}{2},k} &= 0 \quad \forall\ \ k = 1, \ldots, l_m \\ f^d_{l-\frac{1}{2},k} &= (f^d_{l+\frac{1}{2},k} - \delta_{k,l}\frac{E_{d,l}}{m_l})(1 + \frac{D_{d,l}}{M_{d,l+\frac{1}{2}} - E_{d,l}}) \end{aligned} \tag{1.36}$$

Subsidence

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

$$F_s = -(M_u + M_d)\bar{\chi} \tag{1.37}$$

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

$$F_{s,l+\frac{1}{2}} = -(M_{u,l+\frac{1}{2}} + M_{d,l+\frac{1}{2}})\frac{n_{l+1}}{m_{l+1}} \tag{1.38}$$

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

$$f^s_{l+\frac{1}{2},k} = -\frac{M_{u,l+\frac{1}{2}} + M_{d,l+\frac{1}{2}}}{m_{l+1}}\delta_{l+1,k} \tag{1.39}$$

Vertical diffusion

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

$$F_{vdiff} = -A\rho K\frac{\partial\bar{\chi}}{\partial z} \tag{1.40}$$

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

$$F_{vdiff,l+\frac{1}{2}} = -A\frac{\rho_{l+\frac{1}{2}}}{z_{l+1} - z_l}K_{l+\frac{1}{2}}(\frac{n_{l+1}}{m_{l+1}} - \frac{n_l}{m_l}) \tag{1.41}$$

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

$$f^{vdiff}_{l+\frac{1}{2},k} = -A\frac{\rho_{l+\frac{1}{2}}}{z_{l+1} - z_l}K_{l+\frac{1}{2}}(\frac{\delta_{l+1,k}}{m_{l+1}} - \frac{\delta_{l,k}}{m_l}) \tag{1.42}$$

Global tuning of convection parametrization

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

1.2.4 Horizontal Diffusion

Simulation experiments with earlier versions of the transport model (TM1, see Heimann and Keeling [Heimann and Keeling, 1989]) indicated that the interhemispheric transport on annual average was too weak, depending on the meteorological fields used to drive the model, compared to that inferred from other tracer studies (notably ^{85}Kr and ^{11}F). 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. [Prather et al., 1987]. The strength of this additional term was controlled by a global parameter (a length scale) 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 TM3 the code for the horizontal diffusion has been removed.

Chapter 2

System Description

2.1 Overview

The TM3 model system consists of two components: The preprocessing programs MFLUX, SUBSCALE and PASCHA, which generate the airmass fluxes and sub-gridscale transport information on the tracer model grid from meteorological fields, and TM3, the actual tracer model code itself. In practice the steps MFLUX, SUBSCALE and PASCHA are performed only once for a particular meteorological dataset and transport model geometry configuration. Figure 2.1 shows the TM3 information flow.

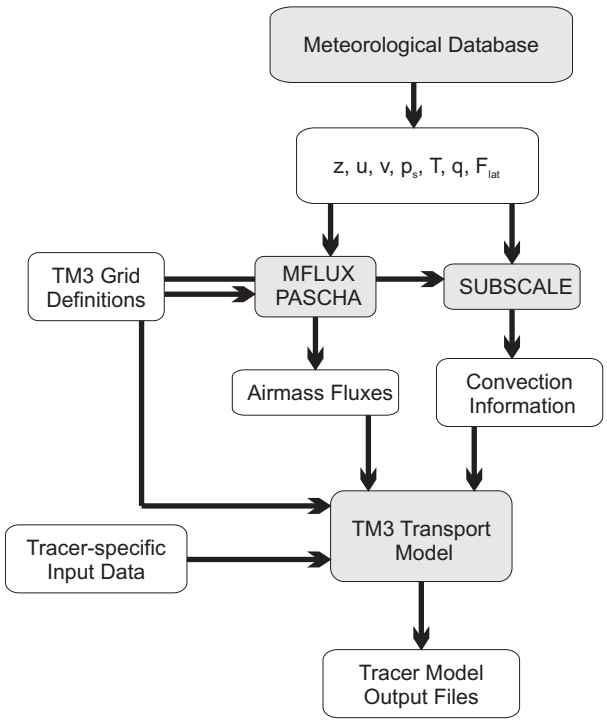


Figure 2.1: TM3 model information flow

The preprocessing modules are described in appendix A.1 and A.2. The following descrip-

tion focuses on the code of the TM3 tracer transport model itself.

2.2 Coordinate Systems and Time Steps

TM3 uses an equidistant latitude-longitude grid, with i_m boxes in the zonal, j_m boxes in the meridional and l_m layers in the vertical dimension. Currently there exist five grid versions:

notation	identifier	i_m	j_m	l_m
coarse	cg	36	24	9
fine	fg	72	48	19
very fine	vfg	192	96	28/31
extra fine	xfg	320	161	31

In the zonal direction the longitudes ϕ_i of the grid box centers are located at

$$\phi_i = -180^\circ + (i-1)\Delta x \qquad i = 1, .., i_m \tag{2.1}$$

with a grid spacing of $\Delta x = 360^\circ / i_m$.

In the meridional direction the j_m grid boxes have an extent of $\Delta y = 180^\circ/(j_m - 1)$. They are spaced such that the latitudes Θ_j of the grid box centers are located at

$$\Theta_j = -90^\circ + (j-1)\Delta y \qquad j = 1, .., j_m \tag{2.2}$$

The first and last (indices $j = 1$ and $j = j_m$) grid box are centered on the poles itself and have a meridional extent of $\Delta y/2$. These polar boxes are not divided in the zonal direction and are referenced with the zonal index $i = 1$. For indices $j = 1$ and $j = j_m$ the elements with the zonal indices $i > 1$ are undefined.

The horizontal TM3 grid layouts are shown in appendix B on page 51.

In the vertical dimension TM3 uses a hybrid coordinate defined by two real coefficients a and b per level l and the pressure equation

$$p_l = a_l + b_l \cdot p_s \ , \quad l = 0 \ldots l_m,$$

where p_s is the surface pressure. Depending on the vertical coordinate system of the underlying meteorological forcing fields, these coefficients can be chosen so that the vertical coordinate is purely σ-like ($\sigma = p/p_{surf}$) near the ground and purely pressure-like at the top of the atmosphere. This is the case for ERA/ECMWF meteorological fields. NCEP/NCAR fields have pure σ-coordinates, which are represented in TM3's hybrid coordinate system as special case with $a_l = 0$.

The vertical box boundaries in hybrid coordinates and in meters above ground are tabulated in appendix C on page 61.

Transport is calculated using a base time step Δt as defined in section 1.2.1. The availability of the meteorological data defines a second time step, Δt_{met}, which must be a multiple of the base time step Δt.

2.3 Program Modules and Source Code Files

A standard TM3 code archive comes with the following directories and contents:

doc	contains this manual, a change log and a commented template of the control parameter input file tm3.in,
examples	contains simple example user files,
src	contains the TM3 Fortran sources
shared	contains source files shared with the preprocessing programs

The root directory contains a Makefile, which the user can copy to his work directory and adjust to his needs.

Generally, the files were given the name of the contained Fortran module, if possible. These modules were named, in turn, according to their function. The individual files and their purposes are tabulated in the following:

File	Purpose	optional
advection.f90	advection module	
aux_flux.f90	handling of surface source fluxes	yes
aux_grid.f90	auxiliary routines for grid calculations	yes
convection.f90	convection module	
diagnostics.f90	diagnostic routines (air and tracers)	
griddefs_<metID>_<gridID>.f90	defines constants related to grid and meteorology	
mass_conservation.f90	mass conservation check	
time_and_date.f90	date/time/calendar calculations	
tm3global.f90	main global TM3 constants and variables	
tm3io_flux.f90	I/O of surface flux files (unformatted)	yes
tm3io_flux_netCDF.f90	I/O of surface flux files (netCDF)	yes
tm3io_mix_netCDF.f90	I/O of instantaneous tracer mixing ratios (netCDF)	yes
tm3io_mix_unformatted.f90	output of instantaneous tracer mixing ratios (unformatted)	
tm3io_mixstn_ascii.f90	output of mixing ratios at stations	
tm3io_mixstn_unformatted.f90	output of mixing ratios at stations	yes
tm3io_prepro.f90	input of preprocessed driver fields	
tm3io_schedule_old.f90	output of mean mixing ratios at station locations	yes
tm3io_tmass_unformatted.f90	output of instantaneous tracer masses	
tm3main.f90	main loop and misc.	
tm3netCDF.f90	TM3 netCDF interface	yes

2.4 Global Variables and Constants

The principal physical variables are defined in one single Fortran module TM3global, contained in source file tm3global.f90. All physical subroutines operate on these variables, as well as the user-specified code for sources, sinks and/or chemistry. For details see the listing in appendix D on page 65, or the documented module src/tm3global.f90 itself.

The runtime parameters of TM3 are combined in a Fortran namelist inputz, which is also declared in module TM3global. At program start, TM3 tries to read a text file tm3.in from the current directory, where these parameters have to be set according to the Fortran namelist input rules. Defaults apply for variables not specified there. A documented template can be found under doc/tm3.in, which is listed in appendix E on page 71. More details are explained in section 3.4.1 on page 29.

2.5 The Model Main Loop

The file tm3main.f90 contains, among others, the main program and the subroutine mainloop with the principal model loop. The main loop is entered after initialization actions, and after it is exited, shut-down operations are executed. The sequence of performed tasks is listed in Table 2.1.

Each task is controlled by variables contained in the Fortran namelist inputz. These control parameters specify e.g. the time steps for the particular task. Since the model simulation time is divided in increments of the base advection time step ndyn, the values of each of these time steps must be set to a multiple of ndyn. This is checked during program startup.

2.6 Time and Calendar Calculations

The basic time unit of the model is the second. The basic time step is defined by variable ndyn which describes the length of an advection timestep. The counter itau contains the current simulation time in seconds since 0:00:00 UT, January 1, iyear0. Functions are provided to convert simulation time instants expressed in seconds to an easily readable 6-element integer vector (/year, month, day, hour, minute, second/), or to a character string.

Four different calendar options are available, which can be selected by the control parameter 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 default is 2. The choice of the calendar kind depends on the meteorological data to be used: (1) is appropriate with climate model output which uses 360 days per year, (2) for simulations using multiyear meteorological analyses, (3) and (4) for simulations over several years but by cycling repeatedly through the meteorology of a particular year.

Subroutine	**Timing by Variable**	**Task**
start	-	TM3 initialization
trace0	-	user-specific initializations
trace1	-	user-specific initializations (if istart = 10+i)
begin of main loop		
read_prepro	nread	reads meteorological fields from storage medium
init_advection	nread	initializes advection module
init_convection	nread	initializes convection module
interp_prepro	ndyn	interpolates some of the input fields in time
chem1	nchem	updates tracer fields χ by chemistry process 1
chem2	nchem	updates tracer fields χ by chemistry process 2
source1	nsrce	updates tracer fields χ by source process 1
source2	nsrce	updates tracer fields χ by source process 2
advection_step	ndyn	calculates horizontal advection
hdiff	ndiff	calculates horizontal diffusion
convec_step	nconv	calculates vertical mixing by convection and vertical diffusion
acc_diag	ndiag	accumulates diagnostics
inctime	ndyn	increments model time by one basic time step ndyn and updates calendar
write_cnsrv	ndiagp1	writes out mass conservation check
write_diag	ndiagp2	writes out mean field diagnostics
writc	ncheck	writes out tracer mixing ratio $\bar{\chi}$ at checkpoint locations
write_mix	ninst	writes out instantaneous tracer mixing ratio field $\bar{\chi}$
write_mixstn	ninststn	writes out instantaneous tracer mixing ratio $\bar{\chi}$ at station locations
write_tmass	ninsttmass	writes out instantaneous tracer mass field $\bar{\chi} \cdot m$
end of main loop		
tracee	-	user-specific shut-down
finish	-	TM3 shut-down

Table 2.1: Calling Sequence in TM3

Chapter 3

User's Manual

The TM3 program code is written with the endeavour to follow the Fortran 95 standard[1] as much as possible. The only known portability issues concern the "make" process using GNU make and the use of the "/" or "\" character as filename separators. The program has been successfully built and run "as is" on numerous UNIX-, MS-Windows- and Apple Macintosh OS-X platforms.

Since release 3.6a, the TM3 program features symmetric multiprocessing, which is especially useful for higher resolutions. This option is selected by a special make target (type make without any target for a complete list) and OpenMP environment variables. When many processors are used, the performance might depend strongly on whether the user's routines are parallelized efficiently or not. Multiprocessing with TM3 is explained in more detail in sec. 3.6.2 on p. 38.

If preprocessed files are transferred from one platform to another, the user has to take care of matching integer and real data representations (width, IEEE or not, big or little endian), as well as the sequential record structure, which is not part of the Fortran 95 standard.

All setup and run actions can take place in an arbitrary working directory. The link to the TM3 code is made by specifying its path in the Makefile.

3.1 Building and Running TM3 - Overview

For the following, a working Fortran 95 compiler and the GNU make utility are prerequisite.

3.1.1 Building a TM3 Executable

The user needs three files in a working directory: Two user-supplied Fortran files and a Makefile.

Step 1: Setting up the user Fortran source files

For building an executable, TM3 needs two user-specified Fortran source files, user-constants.f90 and userroutines.f90. In the former, the user has to define only the precision of certain TM3 variables and the number of tracers ntrace. In the latter

[1] Language constructs defined in the Fortran 95 Technical Reports (TR) are not used.

reside the user codes for sources/sinks/chemistry, together with user variables. Examples are given in the examples directory. The preparation of these files is described below (sec. 3.2 and 3.3).

Step 2: Setting up the Makefile
A Makefile template resides in the root directory of the TM3 tar archive. It must be copied to the working directory. Adjust the commented *Configuration Section* inside the Makefile copy according to your needs.

Step 3: Running the GNU make command make or gmake
After adjusting the Makefile, type "`make` <target>" (or "`gmake` <target>") on the command line, where <target> is the kind of executable you want to build. A list of possible targets (optimized/debugging, single processor/multiprocessor etc.) is displayed when typing "`make`" (or "`gmake`") without target. Normally, the make process produces an executable named `tm3`.

3.1.2 Running TM3

Standard prerequisites for running TM3 are the executable, the meteorology driver files and a runtime parameter file tm3.in, specifying the runtime parameters.

A template for the runtime parameter file tm3.in can be found in the doc directory. Not all parameters need to be specified there, one can delete lines or leave them commented out, if the defaults can be used. For more details see sec. 3.4.1.

Make sure that all files and directories specified really exist. Then start the executable tm3.

3.2 Module userconstants

The small module userconstants in file userconstants.f90 defines the numerical precisions of the standard TM3 I/O files (pages 30 and 33), and TM3's internal precision. By default, everything is set to standard REAL. Furthermore, the number of tracer species is selected here (parameter ntrace).
The user can find examples in the examples directory.

Example

```
!******************************************************************************
!  mandatory user-defined TM3 constants
!******************************************************************************

MODULE userconstants

!------------------------------------------------------------------------------
!
!  Description
!  -----------
!
!  Only the TM3 parameters below need to be specified.
!  User-variables and parameters are better defined in the file
!  'userroutines.f90'.
!
!------------------------------------------------------------------------------
!  History
!  -------
!
!  06.05.2002  Initial free format version. Stefan Koerner
!
!  $Id: userconstants.f90,v 1.3 2002/08/19 17:21:21 tpobw Exp $
!------------------------------------------------------------------------------

   IMPLICIT NONE


   !--- Fortran kinds of real TM3 variables

   INTEGER, PARAMETER :: k4 = KIND(1e0), &         ! default REAL
                         k8 = KIND(1d0), &         ! double precision
                         ki = k4,        &         ! prepro input fields
                         kf = k4,        &         ! flux input fields
                         kc = k4,        &         ! calculation
                         ko = k4                   ! mix output fields


   !--- number of tracers

   INTEGER, PARAMETER :: ntrace=1


END MODULE userconstants
```

3.3 Module **userroutines**

In the module userroutines (file userroutines.f90) the user can place his source-, sink- and chemistry-process calculations, as well as user-defined variables and parameters. This module can be set up almost entirely according to the user's needs; the TM3 core modules need access only to ten `PUBLIC` subroutines, to be declared without dummy arguments:

trace0	is called during program startup by subroutine start to initialize the module, e.g. opening tracer input files, initializing tracer fields,
trace1	is called from subroutine start after trace0, if istart = 10+i,
source1	is called every nsrce seconds for calculation of source1,
source2	is called every nsrce seconds for calculation of source2,
chem1	is called every nchem seconds for calculation of chemistry1,
chem2	is called every nchem seconds for calculation of chemistry2,
everystep	is called every step (ndyn) for auxiliary operations (e.g. reading data),
tracee	is called at program end by subroutine finish for a user shut-down,
save_userstatus	in case of a job continuation: called after main loop completion instead of tracee to save the user module status,
restore_userstatus	in case of a job continuation: called from subroutine start instead of trace0 (and possibly trace1) to restore the status of the user module.

The user can write arbitrary additional subroutines or data into the module (remember to `SAVE` all variables that should retain their values across invocations). The diagnostic routines keep track separately the tracer conservation statistics for each of the four possible source and chemistry processes, but in principle the calculations assigned to source1/2 and chem1/2 can be chosen arbitrarily.

The important physical fields like tracer mass, slopes, air mass etc. are all declared and documented in the module TM3global, residing in file src/tm3global.f90 and listed in appendix D on page 65. They can get into the current programming scope by a `USE` statement. The tracer masses and slopes must be modified within userroutines according to the processes under consideration.

The modification of the tracer slopes by emissions requires information about the emission distribution within a grid box. With a given emission field inside a box, built up by tracer emission during a time interval Δt_s, it is consistent with the advection scheme to perform a least-square of a 3d linear function to the sum of old and emitted tracer distribution. The modification of the slopes is then determined by their definition in eq. 1.6.

Some standard emissions are treated in the following.

3.3.1 Example 1: Volume Source

Let a source strength for grid element (i, j, l) be denoted as Q_{ijl} and given in kg/s. The tracer mass fields in kg before and after the emission are denoted by n^k_{ijl} and n^{k+1}_{ijl}. Then the tracer mass field is updated by

$$n^{k+1}_{ijl} = n^k_{ijl} + Q_{ijl}\Delta t_s,$$

where Δt_s is the length of the emission time step.

This update corresponds to the following line in subroutine source1:

```
rm(i,j,l,1) = rm(i,j,l,1) + q(i,j,l)*nsrce
```

In TM3, nsrce is the identifier for the source1 and source2 timestep and a runtime parameter.

3.3.2 Example 2: Sink Proportional to Tracer Concentration

In the case of a decay process with mass loss proportional to the mass itself, $n(t + \Delta t) = n(t)\exp(-\lambda\,\Delta t)$ in continuous time, like e.g. radioactove decay, the update equation for sufficiently small timesteps Δt_s reads

$$n^{k+1}_{ijl} = n^k_{ijl}\,(1 - \lambda\Delta t_s).$$

The slopes nx, ny and nz, defined by eq. 1.6, have to be multiplied by the same factor:

$$\begin{aligned} nx^{k+1}_{ijl} &= nx^k_{ijl}\,(1 - \lambda\Delta t_s) \\ ny^{k+1}_{ijl} &= ny^k_{ijl}\,(1 - \lambda\Delta t_s) \\ nz^{k+1}_{ijl} &= nz^k_{ijl}\,(1 - \lambda\Delta t_s). \end{aligned}$$

In case of a large decay rate, e.g. if $\lambda\Delta t_s > \frac{1}{2}$, an implicit discretization might be required in order to prevent instabilities or/and negative concentrations. For the tracer mass the implicit update equation reads

$$n^{k+1}_{ijl} = n^k_{ijl}\,/(1 + \lambda\Delta t_s),$$

with according equations for the slopes.

The explicit update equations correspond to the following lines in subroutine chem1:

```
 rm(i,j,l,1) =  rm(i,j,l,1) * (1.-lambda*nchem)
rxm(i,j,l,1) = rxm(i,j,l,1) * (1.-lambda*nchem)
rym(i,j,l,1) = rym(i,j,l,1) * (1.-lambda*nchem)
rzm(i,j,l,1) = rzm(i,j,l,1) * (1.-lambda*nchem)
```

In TM3, nchem is the identifier for the chem1 and chem2 timestep and a runtime parameter.

3.3.3 Example 3: Surface Source

Let the source strength for the surface grid element (i, j) be denoted as Q_{ij} and given in kg/s. The tracer mass fields in kg before and after the emission are again denoted by n^k_{ijl} and n^{k+1}_{ijl}. Then the tracer mass field is updated by

$$n^{k+1}_{ij1} = n^k_{ij1} + Q_{ij}\Delta t_s,$$

where Δt_s is the length of the emission time step. Only the surface grid elements are recomputed.

If one can assume a homogeneous emission inside the surface elements, the slopes are not changed by this process. But if one approximates the vertical emission density to be delta-distributed on the surface, $\rho(z) = \delta(z)Q\Delta t_s$, with $\int_0^\infty \delta(z)dz = 1$, the new slopes are computed by a least-square fit to the sum of old tracer distribution and the emitted profile. This scheme is consistent with the scheme used in the advection algorithm by Russel and Lerner [Russel and Lerner, 1981]. As result the vertical slope nz is updated by

$$nz^{k+1}_{ij1} = nz^k_{ij1} - 3Q_{ij}\Delta t_s.$$

Depending on the total tracer mass in the box n_{ij1}, this update might generate negative mixing ratios (the mixing ratio varies in the vertical between $(n_{ij1} - |nz_{ij1}|)/m_{ij1}$ and $(n_{ij1} + |nz_{ij1}|)/m_{ij1}$, if zero nx and ny slopes are assumed; m_{ij1} is the air mass). There are two solutions: Either one uses the update formula above and corrects for negative mixing ratios, e.g. by setting the runtime parameter limits to `.TRUE.`, or one uses an update that is the best achievable fit to the delta emission under the side condition not to generate new negative mixing ratios. The latter yields

$$nz^{k+1}_{ij1} = nz^k_{ij1} - Q_{ij}\Delta t_s.$$

A Commented Code for Example 3

```
!******************************************************************************
!  user-supplied subroutines and user-variables
!******************************************************************************

MODULE userroutines

!------------------------------------------------------------------------------
!
!  Purpose
!  -------
!
!  This module reads at model startup a temporally constant source field <flux>
!  from file 'foss2_fg.d', which is supposed to contain a surface emission flux
!  field in kg/year. In subroutine source1, this source emits the tracer into
!  the lowest model layer.
!
!
!  Description
!  -----------
!
!  The following module contains all user-supplied subroutines associated
```

```
!  with sources and sinks (due to chemical reactions) and possibly
!  with the initial tracer fields.
!
!  trace0               called from subroutine 'start' to initialize the user
!                       module (e.g. opening tracer input files, possibly
!                       initializing tracer fields)
!  trace1               called from subroutine 'start' after 'trace0', if
!                       <istart> = 10+i
!  source1              called every <nsrce> seconds for calculation of source1
!  source2              called every <nsrce> seconds for calculation of source2
!  chem1                called every <nchem> seconds for calculation of
!                       chemistry1
!  chem2                called every <nchem> seconds for calculation of
!                       chemistry2
!  everystep            called in every step for auxiliary operations (e.g.
!                       reading of user data)
!  tracee               called after completion of the main loop for possible
!                       final operations
!  save_userstatus      in case of a job continuation: called after main loop
!                       completion instead of 'tracee' to save the user module
!                       status
!  restore_userstatus   in case of a job continuation: called from subroutine
!                       'start' instead of 'trace0' (and possibly 'trace1') to
!                       restore the status of the user module
!
!-------------------------------------------------------------------------------
!  $Id: userroutines.f90,v 1.6 2003/06/11 11:24:15 tpobw Exp $
!-------------------------------------------------------------------------------
   USE userconstants, ONLY: kc, &               ! computational precision
                            ntrace              ! tracer number
   USE TM3global, ONLY: im, jm, lm,        &    ! grid dimensions
                        rm, rxm, rym, rzm, &    ! tracer field and slopes
                        nsrce,             &    ! time interval for source calc.
                        path_in_s               ! path to input source data
   IMPLICIT NONE

   PRIVATE


   !--- subroutines called by the TM3 core

   PUBLIC :: trace0, trace1,    &
             source1, source2,  &
             chem1, chem2,      &
             everystep, tracee, &
             save_userstatus,   &
             restore_userstatus


   !--- private user variables

   REAL(kc), SAVE :: flux(im,jm)               ! see also the module 'aux_flux'
                                               !    for useful flux routines


   !--- define user parameters to be read together with the runtime
   !--- parameter list (tm3.in)

   !NAMELIST /userparams/ ...


CONTAINS


   !-------------------------------------------------------------------------
   !  initialize source module
```

```
!
!  Example: read in stationary source strength field <flux>
!           units: kg/year
!-------------------------------------------------------------------------
SUBROUTINE trace0

   USE TM3io_prepro, ONLY: get_free_lun      ! utility subroutine
   IMPLICIT NONE

   INTEGER :: my_unit

!-------------------------------------------------------------------------
   my_unit = get_free_lun()                  ! get a free unit number
   OPEN (my_unit,FILE=TRIM(path_in_s)//'foss2_fg.d', STATUS='OLD')
   READ (my_unit,*) flux
   CLOSE (my_unit)

   flux = flux/(365.*24.*3600.)              ! transform to kg/s

END SUBROUTINE trace0



!-------------------------------------------------------------------------
!  this subroutine is called at program startup after 'trace0', if
!  <istart> = 10+i
!-------------------------------------------------------------------------
SUBROUTINE trace1
   IMPLICIT NONE
END SUBROUTINE trace1



!-------------------------------------------------------------------------
!  this subroutine changes the tracer mass and its slopes
!  source process 1
!
!  Example: add tracer to first layer
!           source given in kg/gridbox/year
!-------------------------------------------------------------------------
SUBROUTINE source1

   IMPLICIT NONE

   INTEGER  :: i, j, n
   REAL(kc) :: x

!-------------------------------------------------------------------------
   DO n=1,ntrace
      DO j=1,jm
         DO i=1,im
            x = flux(i,j)*nsrce               ! nsrce is the emission timestep
            rm (i,j,1,n) = rm (i,j,1,n) + x
            rzm(i,j,1,n) = rzm(i,j,1,n) - x   ! surface emission
         ENDDO
      ENDDO
   ENDDO

END SUBROUTINE source1



!-------------------------------------------------------------------------
!  this subroutine changes the tracer mass and its slopes
!  source process 2
!-------------------------------------------------------------------------
```

```
SUBROUTINE source2
   IMPLICIT NONE
END SUBROUTINE source2


!-----------------------------------------------------------------------
!  this subroutine changes the tracer mass and its slopes by chemistry
!  process 1
!-----------------------------------------------------------------------
SUBROUTINE chem1
   IMPLICIT NONE
END SUBROUTINE chem1


!-----------------------------------------------------------------------
!  this subroutine changes the tracer mass and its slopes by chemistry
!  process 2
!-----------------------------------------------------------------------
SUBROUTINE chem2
   IMPLICIT NONE
END SUBROUTINE chem2


!-----------------------------------------------------------------------
!  procedure called after every time step
!-----------------------------------------------------------------------
SUBROUTINE everystep
   IMPLICIT NONE
END SUBROUTINE everystep


!-----------------------------------------------------------------------
!  procedure called after completion of main loop
!-----------------------------------------------------------------------
SUBROUTINE tracee
   IMPLICIT NONE
END SUBROUTINE tracee


!-----------------------------------------------------------------------
!  in case of a job continuation: called after main loop completion
!  instead of 'tracee' to save the user module status
!-----------------------------------------------------------------------
SUBROUTINE save_userstatus
   IMPLICIT NONE
END SUBROUTINE save_userstatus


!-----------------------------------------------------------------------
!  in case of a job continuation: called from subroutine 'start'
!  instead of 'trace0' (and possibly 'trace1') to restore the status of
!  the user module
!-----------------------------------------------------------------------
SUBROUTINE restore_userstatus
   IMPLICIT NONE

   !  The only user variable to be restored if a continuation run starts
   !  is flux. It's done by reading the flux file again.

   CALL trace0
```

```
    END SUBROUTINE restore_userstatus

END MODULE userroutines
```

Other examples of userroutines can be found in the examples directory.

3.4 Program Inputs

3.4.1 Runtime Parameter File tm3.in

In the formatted sequential (ASCII) file tm3.in the user has to specify TM3's runtime parameters. tm3.in is being read from the working directory during program start. It must contain specifications for items of TM3's namelist inputz, which compiles all TM3 runtime parameters. Defaults apply for variables not quoted there, hence only those items need to be specified for which the defaults can not be used. The runtime parameters of namelist inputz are defined in the module for global variables TM3global.

The user can append his own namelist with runtime parameters for his user-defined routines to tm3.in.

A documented example with the complete list of runtime parameters, their meaning, and default values can be found in the doc directory. It is listed in appendix E on page 71. The following paragraph describes one of the most important runtime parameters.

The Start Mode

The integer runtime parameter istart defines the startup behaviour of TM3. It can be set to the following values:

1 Cold start
Initial tracer fields are set to 0. They can afterwards be partially or totally overwritten in subroutine trace0. No other status fields are read.

2 Cold start with initial tracer fields
Initial tracer fields are read from file
<path_out>//<jobid>//'modelstatus.b'. No other status fields are read.

3 Job chain mode
The complete model status, including tracer fields, but excluding the runtime parameters of namelist inputz, is read at program start from file
<path_out>//<jobid>//'modelstatus.b'. Useful for automatic job chains in time limited queues (see also sec. 3.6.3).

10+i As above, with i = 1,2 or 3. In addition the user-routine trace1 will be called after trace0.

3.4.2 Meteorological Input Files and their Presumed Organization

TM3 is an off-line model that reads the necessary meteorological fields for tracer transport from a storage medium. Normally those fields are produced by a preprocessing step, described in appendixes A.1 and A.2.

For these input files as well as for the output files, TM3 applies a naming convention, with ".b" as filename suffix for unformatted files and ".d" for formatted (ASCII) files.

The unformatted ".b"-files read by TM3 have to be of Fortran type "sequential unformatted". Each record is of "TM3 type", i.e. it is written by statements like

```
INTEGER  :: itau, idate(6)
REAL (ki):: array(dim1,dim2[,dim3])
  ...
WRITE (lun) itau, idate, array
```

itau and idate represent two reference times for the 2- or 3d array, itau in seconds since midnight January 1, <iyear0> of the preprocessing run, and idate as 6-dimensional integer vector (/year, month, day, hour, minute, second/) (see section 2.6). Positioning the read pointer inside TM3 to the requested reference time is done using idate only.

Assuming 4 bytes per `REAL` variable, TM3 reads in particular

Field	**Default filenames for grid res. fg**	**Dim**	**Size/yr in MB for 6-hourly meteorology**	**Unit**
entrainment and detrainment rates for the convection module	eu_fg.b, ed_fg.b, du_fg.b, dd_fg.b	3	4×366	kg/m^{2}s
vertical diffusion coefficient	k_fg.b	3	385	m^2/s
geopotential height	z.fg	3	385	m
time-staggered surface pressure and mass flux rates in x- and y-direction (cyclic meteorology)	stag_ps_fg.b, stag_pu_fg.b, stag_pv_fg.b	2, 3, 3	19+2×366	Pa, kg/s
time-staggered surface pressure and mass flux rates in x- and y-direction (continuous meteorology)	stagc_ps_fg.b, stagc_pu_fg.b, stagc_pv_fg.b	2, 3, 3	19+2×366	do.
massflux information for adaptive integration stepwidth mode (optional)	maxndyn_fg.d, maxndync_fg.d	1	0.06	s
temperature, spec. humidity (both optional)	t_fg.b, q_fg.b	3, 3	2×366	K, kg/kg

The files are organized per year for cg and fg, and per month for higher resolutions, to avoid file sizes beyond 2 GB. They must reside in a single directory with the reference year YYYY as directory name for grid resolutions cg and fg, and with reference year and month YYYYMM as name for higher grid resolutions.

For a "cyclic mode", using the meteorology of a particular year for simulations over multiple years, airmass flux fields are read in that differer from those for the continuous mode by the record of the last reference time of the year. Their file names are stag_*.b for cyclic meteorology instead of stagc_*.b for the continuous.

Example

```
$ pwd
/Net/Groups/AG_Heimann/tm3/prepro/fg/NCEP
$ ls
1955 1959 1963 1967 1971 1975 1979 1983 1987 1991 1995 1999
1956 1960 1964 1968 1972 1976 1980 1984 1988 1992 1996 2000
1957 1961 1965 1969 1973 1977 1981 1985 1989 1993 1997 2001
1958 1962 1966 1970 1974 1978 1982 1986 1990 1994 1998 2002
$ ls 2001
@job0.log     eu_fg.b        q_fg.b         stagc_pu_fg.b
@job1_fg.log  k_fg.b         stag_ps_fg.b   stagc_pv_fg.b
dd_fg.b       maxndyn_fg.d   stag_pu_fg.b   t_fg.b
du_fg.b       maxndync_fg.d  stag_pv_fg.b   z_fg.b
ed_fg.b       p_fg.b         stagc_ps_fg.b
```

The path of these annually or monthly input directories must be specified in the runtime parameter file tm3.in, namelist variable path_in_p, without trailing slash. The path parts for resolution and meteorology are appended by the program. Hence, for the preceding example the path specification in tm3.in reads

```
path_in_p = '/Net/Groups/AG_Heimann/tm3/prepro'
```

3.4.3 Other Standard Input Files

plandf_<gridID>.d Fraction of land in each grid cell (optional)
This file is not needed by the model code itself, but this information might be useful in the tracer specific subroutines.
Control parameters in namelist inputz: read_plandf, name_plandf
Default: read_plandf=.FALSE. (do not read <name_plandf>)

modelstatus.b Complete model status or initial tracer fields only
The complete form is always written at a normal program end. It is needed for a model cold start with initial tracer fields (selected with istart=2), or a continuation run (selected with istart=3).
For a user-defined startup tracer distribution (istart=2), the user needs an initial file modelstatus.b in the directory <path_out> with the following shape (TM3 record structure):

```
INTEGER  ::  itaur, idater(6)
REAL(kc) ::  rm(im,jm,lm,ntrace), rxm(im,jm,lm,ntrace), &
             rym(im,jm,lm,ntrace), rzm(im,jm,lm,ntrace)
WRITE (lun) itaur, idater, rm, rxm, rym, rzm
```

The date idater is checked at program startup and a warning is issued, if it does not match the program start date idatei of namelist inputz. itaur is not checked.
For an automatic job chain (istart=3) see sec. 3.6.3 or the explanations in doc/tm3.in.
Control parameters in namelist inputz: istart, nsave, jobid, path_out
Default: Do not read modelstatus.b at startup (istart=1, cold start)

modelstatus.d Restart information for job chains (optional)
This formatted file contains the next model start date idatei of the form

```
INTEGER :: idatei(6)
```

as well as user service data. See sec. 3.6.3 on page 40 or the file doc/tm3.in for more

information.
Control parameters in namelist inputz: istart, nsave, jobid, path_out.

stationlist.d Station coordinates list (optional)
For output of tracer mixing ratio timeseries at station locations. It must be a text file with one line per station of the form

name lat lon height

Here, name is a character string with up to 12 characters, lat and lon give the horizontal coordinates in degrees north and east, and height is counted in meters above sea level. Alternatively, height can be set to a negative integer; then its absolute value is being taken as the model layer index l (useful to force the station to be in the lowest layer by setting height to -1).
Lines with the token # or ! as first non-blank character are interpreted as comments.
Control parameters in namelist inputz: name_stationlist

3.5 Program Outputs

The standard output files are written to an output directory specified by the runtime parameter path_out in namelist inputz. All file names quoted here are relative to that path.

For most output files there exists a corresponding runtime parameter in inputz, where a different name can be specified. This section refers to the default names. They get the value of the runtime parameter string jobid as prefix, with an underscore appended. The default is jobid = '' (nothing will be prefixed).

The naming conventions and record structure of the formatted and unformatted files, described in sec. 3.4.2, also apply for the output files.

TM3 has a convention for those runtime parameters in inputz that control output time intervals. If such a parameter is set to an integer greater than 0, this means "interval in seconds since 0:00:00 UT, January 1, iyear0". 0 means "no output", -1 "daily at 0:00", -2 "monthly at 1st 0:00", and -3 "yearly, January 1 0:00".

3.5.1 Output to the Standard Output Device

During runtime, TM3 prints a description of the performed actions to standard output.

Example

```
Reading model control parameters from "tm3.in".. ok

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

   Global Atmospheric Tracer Model TM3, version 3.8
   Max Planck Institute for Biogeochemistry, Jena
   Fine grid (fg, 560 km), NCEP meteorology

   Starting job: ""



------------------------------------------------------------------------
1998-Jan-01  00:00:00    opening output files:
   "output/mix.b"
1998-Jan-01  00:00:00    opening met input files
1998-Jan-01  00:00:00    opening mass conservation output files:
   "output/consrv_tracer.d"
   "output/consrv_air.d"
1998-Jan-01  00:00:00    opening diagnostic output files:
   "output/zonal_avrg.d"
   "output/mmix.b"
1998-Jan-01  00:00:00    *** starting main model loop
1998-Jan-02  00:00:00    maxc:  0.461 0.242 0.515
1998-Jan-03  00:00:00    maxc:  0.396 0.216 0.483
1998-Jan-04  00:00:00    maxc:  0.350 0.207 0.481
1998-Jan-05  00:00:00    maxc:  0.425 0.209 0.424
1998-Jan-06  00:00:00    maxc:  0.471 0.242 0.582
1998-Jan-07  00:00:00    maxc:  0.592 0.471 1.233,  ncflv: 0 0 12
1998-Jan-08  00:00:00    maxc:  0.608 0.268 0.500
1998-Jan-08  00:00:00    *** main model loop finished
1998-Jan-08  00:00:00    status arrays saved to "output/modelstatus.b"



------------------------------------------------------------------------
```

```
------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------


Job "" completed normally.

End condition:                  finish date has been reached
Number of timesteps:            252
Cumul. number of CFL violations: 12
CPU time:                       0 h 1 min 4 s
CPU per step:                   .25 s
CPU per tracer and year:        0 h 55 min


------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
```

In this example the runtime parameter ndiagp1 was set to -1, hence the daily maximum Courant numbers[2] maxc for the three space dimensions are printed. In case CFL criterion violations happened during the last ndiagp1 time interval, then counters for Courant numbers > 1 are appended for the three space dimensions (ncflv). After finishing the main loop the CPU statistics is printed, which helps in scheduling larger jobs.

3.5.2 Tracer-Related Main Output

Mixing Ratios $\bar{\chi}$

By default, the model writes the simulated instantaneous tracer mixing ratio fields in the file mix.b. The output interval in seconds is given by the runtime parameter ninst of namelist inputz. The runtime parameter idatet defines the date after which instantaneous output shall begin; this allows for economically writing instantaneous output after some model spin-up period.

Tracer mixing ratios written to mix.b have the unit "kg tracer/kg air", and are multiplied by the scaling factor fscale(i) (runtime parameter) for tracer i. If unformatted output is selected in the Makefile, the record structure is "TM3 type" (sec. 3.4.2), i.e. it can be read as:

```
REAL    :: mix(im,jm,lm,ntrace) ! spatial grid dim. and tracer number
INTEGER :: tau, date(6)
...
OPEN (UNIT=10, FILE=..., FORM='UNFORMATTED', STATUS='OLD')
READ (10) tau, date, mix
```

If netCDF output is selected in the Makefile instead, instantaneous mixing ratio fields are written to a netCDF file mix.nc. This file contains

- longitudes, latitudes, level numbers, tau (TM3 reference time),
- the vertical hybrid coordinate coefficients a and b,
- the land fraction array,

[2] See footnote on p. 7.

- the grid cell area array,
- the instantaneous mixing ratios,
- the instantaneous surface source fluxes, if any,
- the instantaneous air mass, and
- the instantaneous surface pressure.

For a detailed description see the comment header of src/tm3io_mix_netCDF.f90.

Station Output

If TM3 finds a station list at program startup (see sec. 3.4.3), it writes time series of tracer mixing ratios at all locations specified in the station list. Values are linearly interpolated in 3 dimensions between the 8 surrounding grid boxes next to the station location. The units are the same as for the 3d field output, described in the two preceding sections.

In the default case (unformatted sequential output to file mixstn.b), a matrix with the values for all tracers for all stations is written for each reference time. It can be read as:

```
REAL    :: mix(stations_n,ntrace) ! number of stations, tracer number
INTEGER :: itau, idate(6)
...
OPEN (UNIT=10, FILE=..., FORM='UNFORMATTED', STATUS='OLD')
READ (10) itau, idate, mix
```

If the Makefile is set up to produce formatted (ASCII) output instead, instantaneous mixing ratios are written to several files mix_*stationname*.d, one file per station. Each reference time corresponds to one line, with itau at column 1, the reference date idate at columns 2–7 and the mixing ratios of all tracers starting from column 8.

Diagnostic mixing ratios for some gridpoints with known indices (i,j,l) can be written directly in file check.d (see sec. 3.5.3).

Tracer Mass Fields

The model can also write the simulated instantaneous tracer mass fields in the file tmass.b (default name). The output interval in seconds is given by the runtime parameter ninsttmass in namelist inputz. The runtime parameter idatet defines the date after which instantaneous output begins. As for the mixing ratios, this allows for economically writing instantaneous output after some model spin-up period.

Tracer mixing mass written to tmass.b has the unit "kg tracer". The record structure is "TM3 type" as for the mixing ratios, and can be read as:

```
REAL    :: tmass(im,jm,lm,ntrace) ! spatial grid dim. and tracer number
INTEGER :: itau, idate(6)
...
OPEN (UNIT=10, FILE=..., FORM='UNFORMATTED', STATUS='OLD')
READ (10) itau, idate, tmass
```

3.5.3 Diagnostic Output

The tracer model writes several ASCII and binary files with diagnostic information:

mmix.b Time-averaged tracer mixing ratio 3d fields (switched on by default),

zonal_avrg.d Tables with zonally and temporally averaged tracer mixing ratios (switched on by default),

consrv_tracer.d Tables with conservation statistics of the tracers, and zonally averaged tracer mixing ratios (switched on by default),

consrv_air.d Tables with conservation statistics of the air mass (switched on by default),

check.d Mixing ratio timeseries at a set of check locations (switched off by default), and

debug.d Detailed protocol of the model run (switched off by default).

Time-Averaged Mixing Ratios

The model computes time averaged tracer mixing ratio fields by sampling the tracer masses every ndiag seconds (runtime parameter), starting at idatem (runtime parameter). These averaged fields are written in a file mmix.b by default. The time averaging interval can be selected by the runtime parameter ndiagp2.

As with the instantaneous mixing ratios, mmix.b contains tracer fields with unit "kg tracer/kg air", multiplied by the scaling factor fscale(i) (runtime parameter) for tracer i.

This file has the same structure as the instantaneous mixing ratio file - see above. Note that in this case 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. E.g. in the case of monthly averaged fields, the recorded time/date stamp shows the beginning of the month immediately *after* the averaging period.

Zonally and Temporally Averaged Tracer Mixing Ratios

The zonally averaged tracer mixing ratios are written in a set of ASCII tables with default filename zonal_avrg.d. The output interval is controlled by the value of runtime parameter ndiagp2.

Tracer Conservation Statistics

The file consrv_tracer.d contains ASCII tables of various conservation statistics for each tracer. The instantaneous tracer masses and their changes due to the individual transport and source/sink processes are listed here, for each latitude interval of the model grid, for each hemisphere and for the entire globe. Furthermore, for each tracer a simple latitude-height table of the zonally averaged tracer mixing ratio is written, with units "kg tracer/kg air" and multiplied by the factor fscale(i) (runtime parameter) for tracer i. The runtime parameter ndiagp1 selects the output and averaging time intervals for these tables.

Air Mass Conservation Statistics

The conservation statistics of the air mass is written in a set of ASCII tables similar to those for the individual tracers. The default filename is consrv_air.d. The output interval is also controlled by the value of runtime parameter ndiagp1. These tables are determined directly from the input airmass flux fields. They can be inspected to check the conservation of air mass.

Mixing Ratio Timeseries at Check Locations

Up to 10 check locations can be defined with the runtime parameter array indc(3,10), which is included in namelist inputz. The triplet indc(:,k) denotes the (i, j, l) grid indices of check location k, where $k = 1$,noindc (runtime parameter). Depending on the value of the runtime parameter ncheck, time series of the mixing ratios of all tracers are written in file check.d. The mixing ratios have the unit "kg tracer/kg air", multiplied by the factor fscale(i) (runtime parameter) for tracer i.

Debugging Output

If the runtime parameter debug is set to `.TRUE.`, a detailed protocol of all tasks performed by the model is written in file debug.d. This option is intended primarily for the debugging of the transport model code itself and of little use during normal operation. The amount of information written in debug.d is very large. Therefore this switch should be used only for short test runs of length less than a few days.

3.5.4 Other Output

tm3.in_complete At startup the complete list of *all* runtime parameters is written in a file tm3.in_complete, including those not specified in tm3.in. If such a file already exists it will be overwritten.

ps.b An instantaneous surface pressure field for possible vertical interpolation of the mixing ratio output to pressure or sigma levels in a postprocessing stage. This output is triggered at the same time instants as the output of the instantaneous mixing ratio fields, and consistent with the instantaneous mass and mixing ratio fields.
The output is controlled by the runtime parameter cwrtps and switched off by default.

modelstatus.b When finishing a run normally, TM3 writes its complete status, excluding the runtime parameters, in a file modelstatus.b. The tracer fields and slopes after the last integration time step are also contained therein. If a file of that name exists, it will be overwritten. In conjunction with the job chain mode (istart=3, described in sec. 3.6.3), this file can be used as input for continuation runs, or, in its abridged form, for "cold starts with initial tracer distributions (istart=2)". The file structure is described in sec. 3.4.3.

modelstatus.d If the user has selected the job chain mode with istart=3, TM3 writes an output file modelstatus.d, after the runtime parameter time nsave has passed by without reaching the model end date idatee. Its existence indicates the necessity

of a continuation run. After reaching the end date idatee this file is deleted. See sec. 3.4.3 and sec. 3.6.3 for more details.

3.6 Programming Hints

3.6.1 Useful Auxiliary Subroutines

TM3 comes with several subroutines helping the user in fulfilling his tasks:

Module	File	Description
aux_flux	aux_flux.f90	subroutines for handling surface source fluxes
aux_grid	aux_grid.f90	auxiliary routines for grid calculations
tm3io_schedule	tm3io_schedule.f90	subroutines for calculation/output of mean mixing ratios measured at station locations according to a given time schedule
time_and_date	time_and_date.f90	routines for date/time/calendar calculations

In addition, the function get_free_lun() in module tm3io_prepro provides a free logical I/O unit number at run time.

For details see the description in the source files.

3.6.2 Multiprocessing with TM3

Since release 3.6R2, TM3 features symmetric multiprocessing with OpenMP 1.0 (see www.openmp.org), especially useful for higher resolutions.

Remark 1 *With TM3 release 3.8, the computational work is divided among the tracers. Hence the speed gain is largest if either the number of tracers n is a multiple of the number of processors p, or the fraction n/p is large compared to 1.*

Remark 2 *If running TM3 in parallel, don't change parallelized TM3 code parts unless having understood how the related OpenMP constructs work. The same applies for all data involved.*

Setting up a Multiprocessing Executable

The OpenMP programming interface includes source code constructs in the shape of special Fortran comments, library calls and environment variables. Commonly, OpenMP parallelized source code is written in a manner that retains the capability of compiling and running the code on a single CPU, without any additional precautions.

Setting up a multiprocessing executable is platform dependent. Usually it is done by quoting special compiler options, or by calling a special compiler. For the platforms supported by the Makefile, this is ready-for-use. Type "`make`" without targets to see what is implemented.

Running a parallel TM3

Generally, at run time an OpenMP-instructed executable accesses environment variables that influence the behaviour of its parallel regions. The user needs to set only one,

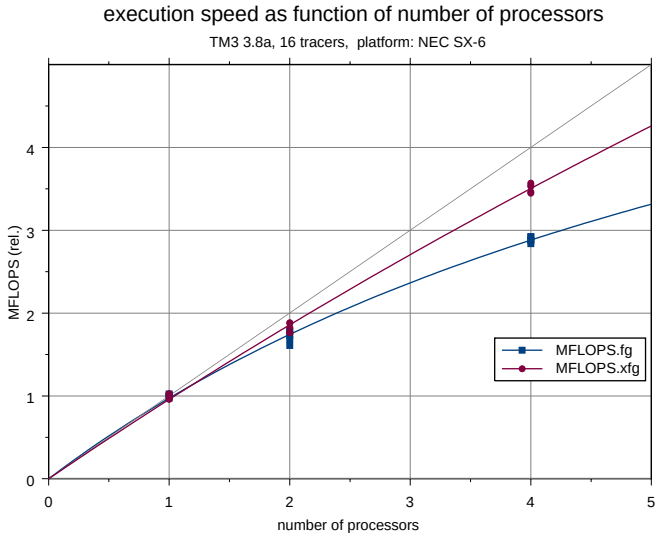
OMP_NUM_THREADS, which specifies the number of threads (CPUs) to be used. Under a UNIX shell, the command `export OMP_NUM_THREADS=4`, e.g., instructs the TM3 executable to use 4 processors. The default is platform dependent.
At start-up with OMP_NUM_THREADS=4, TM3 gives a note about multiprocessing alike:

```
-------------------------------------------------------

   Global Atmospheric Tracer Model TM3, release 3.8
   Max Planck Institute for Biogeochemistry, Jena
   Grid version: fine grid, NCEP meteorology
   Multiprocessing with 4 threads
   ...

-------------------------------------------------------
```

TM3 presumes to have the specified number of CPUs entirely for itself. If this is not the case, speed degradations are likely (the loop schedules are set to "static"). The achievable speedup depends on platform, compiler, TM3 setup parameters and the user code. Test results with empty user-routines are shown in the figure below.



The Influence of Serial Program Parts

Running TM3 on more than one CPU has effect only on those program parts that have been parallelized. With more processors, the parts that are executed serially become a performance bottleneck.

Example In serial mode, a program executes over 20 seconds in serial program regions and 80 seconds in program regions that can also be run in parallel. I.e., 20 % of the program's execution time is not parallelized. When run on 4 processors the total execution time is 40 s, assuming that the parallel parts are accelerated by a factor of 4. But 50 %

of the execution time is now spent in the small part that is not parallelized. 3 CPUs are idle at half of the total execution time.
To achieve a reasonable speedup even with few processors, it was necessary to parallelize large parts of TM3. If one has serially programmed, runtime expensive user routines, then the total speedup will be seriously restricted. The only way to achieve higher model speedup is parallelizing these parts as well.

3.6.3 Job Chains

Job chains, i.e. a (repeated) cycle of computation and finishing with status saving, are explicitly supported by TM3, making the work with time-limited job queueing systems easier. The controlling runtime parameters are istart and nsave. istart needs to be set to 3 for job chains.

In TM3, the existence of a file <path_out>//<jobid>//modelstatus.d is used as indicator if this is a continuation run within a job chain or the start of it. The user can employ this as end condition in a self-calling job script.

IF this file does NOT EXIST, then the start date is that of tm3.in. Initial tracer fields are read from the file <path_out>//<jobid>/modelstatus.b, if existent, or otherwise set to 0.

IF the file EXISTS, then the namelist inputz is first read from file tm3.in. Afterwards the start date idatei is overwritten by the date stored in <path_out>//<jobid>//modelstatus.d, saved by a previous model run at time intervals of <nsave> seconds, counted from 01-Jan-<iyear0> 00:00:00.

Initial tracer fields are read from file <path_out>//<jobid>//modelstatus.b, if existent. That file will always be written after a model run, independently of what istart is set to.

The continuation parameter file <path_out>//<jobid>//modelstatus.d contains the start date in the form `YYYY MM DD hh mm ss` as first line. The last line contains all prepro directory identifiers needed by the next continuation cycle, e.g. `1999 2000 2001` or `199911 199912 200001`. This is useful for getting prepro files out of an archive by shell script commands, like shown in the jobscript example of sec. 3.6.4. After reaching the end date idatee, TM3 deletes this file.

Furthermore, two TM3global variables support the initialization of user-defined subroutines with istart=3:

jobcont is set to `.TRUE.` at program startup if a job continuation is present, i.e., if modelstatus.d exists, and to `.FALSE.` otherwise,

newsrun is set to `.TRUE.` at program startup and to `.FALSE.` after the first timestep.

For saving and restoring user data for the next job cycle, the subroutines save_userstatus and restore_userstatus in module userroutines can be utilized. They are called instead of trace0, trace1 and tracee in case TM3 detects a job continuation. The user can do the saving in save_userstatus by contructs like

```
SUBROUTINE save_userstatus
   IMPLICIT NONE
   INTEGER :: lun

   !--- job chain: save user fields
```

```
   lun = get_free_lun()                                  ! get a free unit number
   OPEN (UNIT=lun, FILE=TRIM(path_out)//'userfields.b', FORM='UNFORMATTED', &
         STATUS='REPLACE')
   WRITE (lun) <the user's data>
   CLOSE (lun)

END SUBROUTINE save_userstatusEND IF
```

and read the fields accordingly in subroutine restore_userstatus.

3.6.4 Jobscript Example

The following example shows work-alikes of the TM3 features

- parallel processing with 4 processors,
- automatic job chaining with istart=3,
- using the service information of modelstatus.d to copy only the required meteo files out of an archive.

Platform-dependent are the Korn shell syntax and the usage of a NQS job queueing environment.

```
#@$-s /bin/ksh
#@$-c 4                     # number of cpus, default is 1
#@$-lM 4GB                  # memory limit per request
#@$-lT 3:45:00              # time limit per request
#@$-r mylabel               # job label
#@$-eo                      # send stderr to stdout
#

cd "$QSUB_WORKDIR"


#--- the system provides a very fast file system,
#--- where later the prepro files are copied to

ln -s $TMPDIR input


#--- set OpenMP environment

export OMP_NUM_THREADS=4


#--- get the proper prepro files from an archive

if [ ! -f output/modelstatus.d ]; then
   # initial run
   dirlist="199901 199902 199903"
else
   # continuation run: get the necessary directory names
   # from the last line of the service file 'modelstatus.d'
   dirlist=‘tail -1 output/modelstatus.d‘
fi

for dir in $dirlist; do
   for ppfile in eu ed du dd stag_ps ... ; do
      [mf]cp my_storagedir/${dir}/${ppfile}_xfg.b input
   done
done
```

```
...


#--- Set up the TM3 runtime parameter list.
#--- For parameters not specified the defaults are ok.
#--- Fortran namelist I/O syntax is required here.

cat > tm3.in << EOD
&inputz
!
! 1st Most frequently user-changed parameters
! -----------------------------------------
!
   path_in_p = 'input/'            ! path to input prepro (met.) data
   path_in_s = 'input/'            ! path to input source data
   path_out  = 'output/'           ! path to output

   idatei  = 1999 1 1 0 0 0        ! for start of model run
   idatee  = 2000 12 31 18 0 0     ! for end of model run

   ...
!
! 2nd Other Adjustable Parameters
! -------------------------------
!
   istart = 3                      ! do job chaining
   nsave  = 7257600                ! 3x28 days, in seconds

   ...
/
EOD


#--- Start TM3.
#--- Output is already redirected by NQS.

./tm3


#--- some cleanup

rm -r input
rm tm3.in


#--- If the job hasn't come to an end, send this
#--- script again to the queueing system.
#--- Leave an opportunity to kill this job gracefully,
#--- in case something goes wrong.

if [ -f output/modelstatus.d ]; then
   cat < /dev/null > kill_me_now
   sleep 20
   rm kill_me_now
   qsub <the_name_of_this_script>
else
   (send a mail, e.g.)
fi
```

3.6.5 Migration Guidelines for Prior TM3 Releases

TM3 3.7 and older

1. Since release 3.8, the TM3 source code is written in Fortran 95 free format. The Makefile accesses the files named userconstants.f90 and userroutines.f90, but for fixed format code this could be changed in the Makefile to the suffix ".f".
   Programming in free format is a lot more convenient, hence the recommendation is to convert fixed format user code to free format. There exist many converters around. In case of doubt the user can contact the TM3 support (see appendix G).

2. Compared to earlier releases, some files and modules are new or have been renamed. E.g., the Fortran module containing the most important physical fields like tracer masses etc. is now tm3global, rather than variables or tm3variables. Source code files have the name of the module contained within, wherever possible.

3. The module encapsulation has been advanced. Previously global program entities may now require an `USE` statement for the module defining that entity.

4. A log of changes can be found in doc/CHANGES.

TM3 3.8

To get the sources compiled, three empty subroutines trace1, save_userstatus and restore_userstatus need to be added to the module userroutines. Furthermore, the format of some outputs was slightly changed - see the file doc/CHANGES.

3.6.6 CPU expense

The CPU times of TM3 runs depend largely on platform, setup parameters (e.g. number of tracers, grid resolution, number of used processors, the user-routines, compiler options) and runtime parameters (e.g. time steps, whether limits is set or not, usage of variable timesteps). TM3 version 3.8a is optimized to run efficiently on the vector computer NEC SX-6 at the *Deutsches Klimarechenzentrum Hamburg*, and effectivity tests yield the result that an effective application on superscalar architectures requires code adjustments, in particular in the advection module.

The following table shows CPU times T for one year integration period with a conservative setup of TM3 release 3.8a for one tracer, ERA-15 meteorology and empty user-routines, using 1 CPU on the NEC SX-6 at the *Deutsches Klimarechenzentrum Hamburg* and on a standard PC:

platform \ grid res.	fg	vfg	xfg
NEC SX-6	9 min	2 h 50 min	12 h
PC AMD Athlon 1.3 GHz	1h 30 min	68 h	.

Appendix A

Meteorological Data Preprocessing

A.1 Airmass Fluxes

TM3 requires as input fields the horizontal airmass fluxes crossing the gridbox boundaries during each meteorological time step. These fluxes are calculated from fields of geopotential heights, horizontal wind velocities and surface pressure. The data processing is performed in three steps described in the following subsections.

A.1.1 Interpolation/Integration of the Horizontal Mass Fluxes over the Grid Box Sides

For each instant of time of the original meteorological analyses the horizontal massfluxes are obtained by vertical-meridional, respectively vertical-zonal integration of $\rho\vec{u}_h$ using trapezoidal integration formulae (program MFLUX).

A.1.2 Ensuring Mass Conservation

After interpolating the horizontal massflux fields and surface pressure the resulting fields are not exactly mass conserving any more. To restore this fundamental property a correction procedure is applied as follows.

The conservation of mass requires that the divergence of the airmass flux density, vertically integrated, equals the surface pressure tendency:

$$\frac{\partial p_s}{\partial t} = g \int_{z_{surf}}^{z_{top}} \nabla \cdot \rho\vec{u}_h dz \tag{A.1}$$

This is derived from combining the continuity equation with the hydrostatic approximation. Denoting by $\vec{F}$ the vertically integrated airmass flux density, equation (A.1) is reexpressed as

$$\frac{1}{g}\frac{\partial p_s}{\partial t} = \nabla \cdot \vec{F} \ . \tag{A.2}$$

We split

$$\vec{F} = \vec{F}_{obs} + \vec{F}_{corr} \tag{A.3}$$

where $\vec{F}_{obs}$ are the airmass flux densities as determined from the analyses and $\vec{F}_{corr}$ represents a correction to be determined in order to fulfil equation (A.2).

For convenience, we consider a time interval $t \in [0, \Delta t]$. To compute a correction for $\vec{F}$ only, we have to assume $\partial p_s/\partial t$ as known, in fact we define

$$\left.\frac{\partial p_s}{\partial t}\right|_{t=\Delta t/2} = \frac{p_s(\Delta t) - p_s(0)}{\Delta t}.$$

To be consistent, the airmass flux densities $\vec{F}$ are also considered from now at $t = \Delta t/2$ by $\vec{F}_{obs}(\Delta t/2) = \left(\vec{F}_{obs}(\Delta t) + \vec{F}_{obs}(0)\right)/2$ (time staggering).

Hence the correction flux equation becomes

$$\nabla \cdot \vec{F}_{corr} = \frac{1}{g}\frac{p_s(\Delta t) - p_s(0)}{\Delta t} - \nabla \cdot \vec{F}_{obs} \ .$$

This is only the divergence of $\vec{F}_{corr}$. The rotational part of $\vec{F}_{corr}$ is not defined by this and set to zero in order to obtain minimal corrections. We make a potential ansatz

$$\vec{F}_{corr} = \nabla\Theta \tag{A.4}$$

and obtain a Poisson equation for the 2d field Θ:

$$\Delta\Theta = \frac{1}{g}\frac{p_s(\Delta t) - p_s(0)}{\Delta t} - \nabla \cdot \vec{F}_{obs} \ , \tag{A.5}$$

which has to be solved for 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 (A.5) is solved efficiently using two-dimensional Fourier transforms. This yields the vertically summed correction airmass flux field which is subsequently distributed in vertical direction proportionally to the absolute values of the x- and y-components of $\vec{F}_{obs}$.

A.1.3 Airmass Flux processing Programs

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

`MFLUX`	Interpolation/integration of airmass fluxes
`PASCHA`	Time staggering and adjustment for conservation of mass
`SUBSCALE`	Interpolation of non-flux fields

Note that, for historical reasons, in SUBSCALE the direction of the vertical coordinate grid is reversed compared to TM3, i.e. the vertical levels start at the top of the atmosphere ($l = 1$) and end at the surface ($l = l_{surf}$).

A.2 Sub-Gridscale Transport Information

TM3 requires as input global fields the vertical diffusion coefficient and the parameters for the cumulus cloud transport (ent- and detrainment rates into up- and downdraft). These are calculated in the preprocessing stage from meteorological analyses of geopotential, surface pressure, horizontal wind, temperature and specific humidity. Furthermore surface fluxes of latent heat are needed.

All sub-gridscale 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 TM3.

A.2.1 Vertical Diffusion Coefficients

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

$$K_v = l_h^2 \left| \frac{\partial u}{\partial z} \right| f_h(Ri) \tag{A.6}$$

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

$$Ri_{k+1/2} = g(z_k - z_{k+1}) \frac{c_{pd}(T_k - T_{k+1}) + g(z_k - z_{k+1})}{|\vec{u}|^2_{k+1/2} c_{pd} T_{k+1/2}} \tag{A.7}$$

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

$$l_h = \frac{kz}{1 + \frac{kz}{\lambda_h}} \tag{A.8}$$

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

$$f_h(Ri) = \frac{1}{1 + 3bRi\sqrt{1 + dRi}} \tag{A.9}$$

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

$$f_h(Ri) = 1 - \frac{3bRi}{1 + G(Ri)} \tag{A.10}$$

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

$$G(Ri) = 3bCl_h^2 \sqrt{\frac{-Ri}{z} \left\{ \frac{1}{\Delta z} \left[\left(1 + \frac{\Delta z}{z} \right)^{1/3} - 1 \right] \right\}^3} \tag{A.11}$$

Δz is the distance between the adjacent model layer centers.

Numerical values for the constants appearing in equations (A.8)-(A.11) are given in the following table:

Parameter	Value
b	5.
C	5.
d	5.
k	0.4
c_{pd}	1005.46 J kg^{-1} K^{-1}
g	9.80665 m s^{-2}
λ_h	438.18 m

A.2.2 Cloud Transport Parameter Determination

The cloud transport parameters (ent- and detrainment rates into up- resp. downdraft) are calculated according to the scheme described in [Tiedke, 1989]. This scheme calculates the entrainment and detrainment rates in the sequence of steps described in the following subsections.

The scheme detects and parameterizes three types of convective clouds: deep or penetrative, shallow and mid-level convective clouds. Their presence in a model grid column is diagnosed by different conditions, and each has different cloud parameters (see table A.2).

The physical constants employed are listed in table A.1. All air properties of the environment are defined on the model level boundaries.

Parameter	Symbol	Value	Units
Gas constant of dry air	R_d	287.05	J kg^{-1} K^{-1}1
Gas constant of water vapour	R_v	461.51	J kg^{-1} K^{-1}
	ϵ	R_d/R_v	
	C_p	1005.46	J kg^{-1} K^{-1}
	L_v	2.5008 10^6	J kg^{-1}

Table A.1: Physical constants used for the cloud calculations

	Cloudtype		
	D + ML	shallow	
ϵ_u	10^{-4}	3 10^{-4}	m^{-1}
δ_u	10^{-4}	3 10^{-4}	m^{-1}
ϵ_d	2 10^{-4}		m^{-1}
δ_d	2 10^{-4}		m^{-1}
β	0.0	0.15	
γ	0.3		
α	0.2		
k_p	2 10^{-3}		s^{-1}
z_{min}	1500		m

Table A.2: Values of the parameters in the cloud parametrization, see text. D + ML: "deep and mid level".

Detection of cloud base height

The base of a potential convective cloud is determined by lifting surface air dry adiabatically until saturation occurs. "Surface air" thereby is defined as an air parcel with the physical properties of the surface layer except for a tunable humidity, which can be selected between grid average humidity (parameter $\alpha = 0$) and 100% saturation (parameter $\alpha = 1$). The operational value is $\alpha = 0.2$, which was determined empirically by comparisons with cloud cover and height extent of the ERA-15 ECMWF analyses.

During the ascent, the temperature of the air parcel is modified from one layer boundary to the next:

$$T_l = T_{l+1} - g(z_l - z_{l+1})/C_p \tag{A.12}$$

while the moisture remains unchanged:

$$q_l = q_{l+1} \tag{A.13}$$

As soon as saturation is detected ($q > q_{sat}$) condensation is computed by solving the implicit equation for the temperature T_s of the air parcel at saturation:

$$C_p(T - T_s) = L(q - q_{sat}(T_s, p)) \tag{A.14}$$

where T, q are temperature and moisture of the air parcel prior to condensation. The solution of (A.14) is obtained iteratively by Newton's method.

The buoyancy of the air parcel with respect to the environment is checked in terms of their virtual temperature, defined as:

$$T_v = T(1 + q\frac{1-\epsilon}{\epsilon}) \tag{A.15}$$

If the parcel is buoyant, a cloud is detected, else there is no convective or shallow cloud in that particular grid column.

Determination of the updraft air flux at the cloud base

If a cloud base exists, the vertically integrated horizontal water vapour convergence below the cloud is computed:

$$\text{CONV} = -\int_{p=p_s}^{p=p_{cb}} \nabla \cdot \rho \mathbf{v} q dp \tag{A.16}$$

where p_{cb} denotes the pressure of the cloud base. If this convergence is positive, a penetrative or a midlevel convective cloud exists, else a shallow convection cloud exists.

If a cloud is detected, the water vapour flux at the base of the cloud is determined by adding to the large scale water vapour convergence below the cloud the evaporation from the surface. Since also the downdraft (if existing) delivers moisture to the below cloud air volume, this contribution is included in an iterative way by computing firstly the cloud properties without the contribution from the downdraft. In a second iteration the moisture contribution from the downdraft is included. The airmass flux at the base of the cloud is then determined 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.

Finally, detrainment rates into the updraft below the cloud base are set to zero and 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 massflux at the base of the cloud.

Computation of cloud properties in up- and downdraft

Cloud temperature, moisture and liquid water content are initialized with values at the base of the cloud. Subsequently the air parcel ascent is followed by solving the equations

(8) in [Tiedke, 1989] for the updraft mass flux F_u, dry static energy $s_u = C_p T_u + gz$, moisture q_u and liquid water content l_u:

$$\frac{\partial F_u}{\partial z} = E_u - D_u \tag{A.17}$$

$$\frac{\partial F_u s_u}{\partial z} = E_u \bar{s} - D_u s_u + L\bar{\rho} c_u \tag{A.18}$$

$$\frac{\partial F_u q_u}{\partial z} = E_u \bar{q} - D_u q_u - \bar{\rho} c_u \tag{A.19}$$

$$\frac{\partial F_u l_u}{\partial z} = -D_u l_u + \bar{\rho} c_u - \bar{\rho} G_p \tag{A.20}$$

where c_u is the net condensation produced in the updraft. Overbars indicate environmental values. Inserting equation (A.17) into the equations for temperature, moisture and liquid water content yields:

$$F_u \frac{\partial s_u}{\partial z} = E_u(\bar{s} - s_u) + L\bar{\rho} c_u \tag{A.21}$$

$$F_u \frac{\partial q_u}{\partial z} = E_u(\bar{q} - q_u) - \bar{\rho} c_u \tag{A.22}$$

$$F_u \frac{\partial l_u}{\partial z} = -E_u l_u + \bar{\rho} c_u - \bar{\rho} G_p \tag{A.23}$$

which are solved in discretized form going from one layer boundary to the next above. Similar equations are used for the downdraft computation (equations 17 in [Tiedke, 1989]). In each cloud layer the following sequence is computed:

1. Air ent- and detrainment rates are set proportional to the updraft massflux with constants ϵ_u and δ_u as given in table A.2:

   $$E_u = \epsilon_u F_u \Delta z \tag{A.24}$$

   and

   $$D_u = \delta_u F_u \Delta z \tag{A.25}$$

   where F_u denotes the updraft massflux at the lower boundary of the layer and Δz the layer thickness (in m).

2. Cloud air parcel temperature, moisture and liquid water content are first adiabatically lifted and then adjusted by the entrainment of environmental air. This results in preliminary values:

   $$T_l = T_{l+1} - g\Delta z / C_p + \epsilon_u \Delta z (\bar{T}_l - T_{l+1}) \tag{A.26}$$

   $$q_l = q_{l+1} + \epsilon_u \Delta z (\bar{q}_l - q_{l+1}) \tag{A.27}$$

   $$l_l = l_{l+1} + \epsilon_u \Delta z (-q_{l+1}) \tag{A.28}$$

3. Condensation is computed and the parcel temperature increased to force the cloud water vapour at saturation. This is performed with the same iteration procedure as described above (equation (A.14)). The condensate, $q - q_{sat}(T_s, p)$, is added to the liquid water content of the cloud air parcel.

4. If the cloud layer is higher than z_m above the cloud base, precipitation rate out of the layer is computed from

$$G_p = k_p l F_u \tag{A.29}$$

and the liquid water content adjusted using an implicit formulation in order to prevent negative liquid water values (superscript +, resp. - indicate values after, resp. before adjustment):

$$l^+ = l^- - l^- \frac{\bar{\rho} G_p}{1 + \bar{\rho} G_p} \tag{A.30}$$

5. Check for level of free sinking (LFS) where the cumulus downdraft starts. The LFS is assumed to be the highest model level where a mixuter of equal parts of cloud air and environmental air (at wet bulb temperature) becomes negative byoyant with trespect to environmental air.

6. Compute downdraft if an LFS is detected and and only if there is precipitation (i.e. above z_{min}). This is performed similar to the calculations described above with an initial downdraft mass flux at the LFS of $\gamma F_{u,cb}$. Ent- and detrainment rates are defined by the constants ϵ_d and δ_d. The downdraft air parcel is forced to remain at saturation by evaporation of liquid water. Below the cloud base, the downdraft air mass flux is detrained in all layers proportional to the layer thicknesses.

7. The ascent of the updraft cloud parcel is continued above the LFS if the air parcel is still buoyant.

If the cloud parcel is no longer buoyant then the remaining airmass flux is detrained in the two layers above the cloud with a fraciton of β in the first and $(1 - \beta)$ in the second layer to simulate the effect of cloud overshoots.

A.2.3 Programs

The parameters of the sub-gridscale transport are calculated in the following programs and subroutines:

SUBSCALE	main program
louis	calculates vertical diffusion coefficients
cloud	calculates cloud ent- and detrainment rates

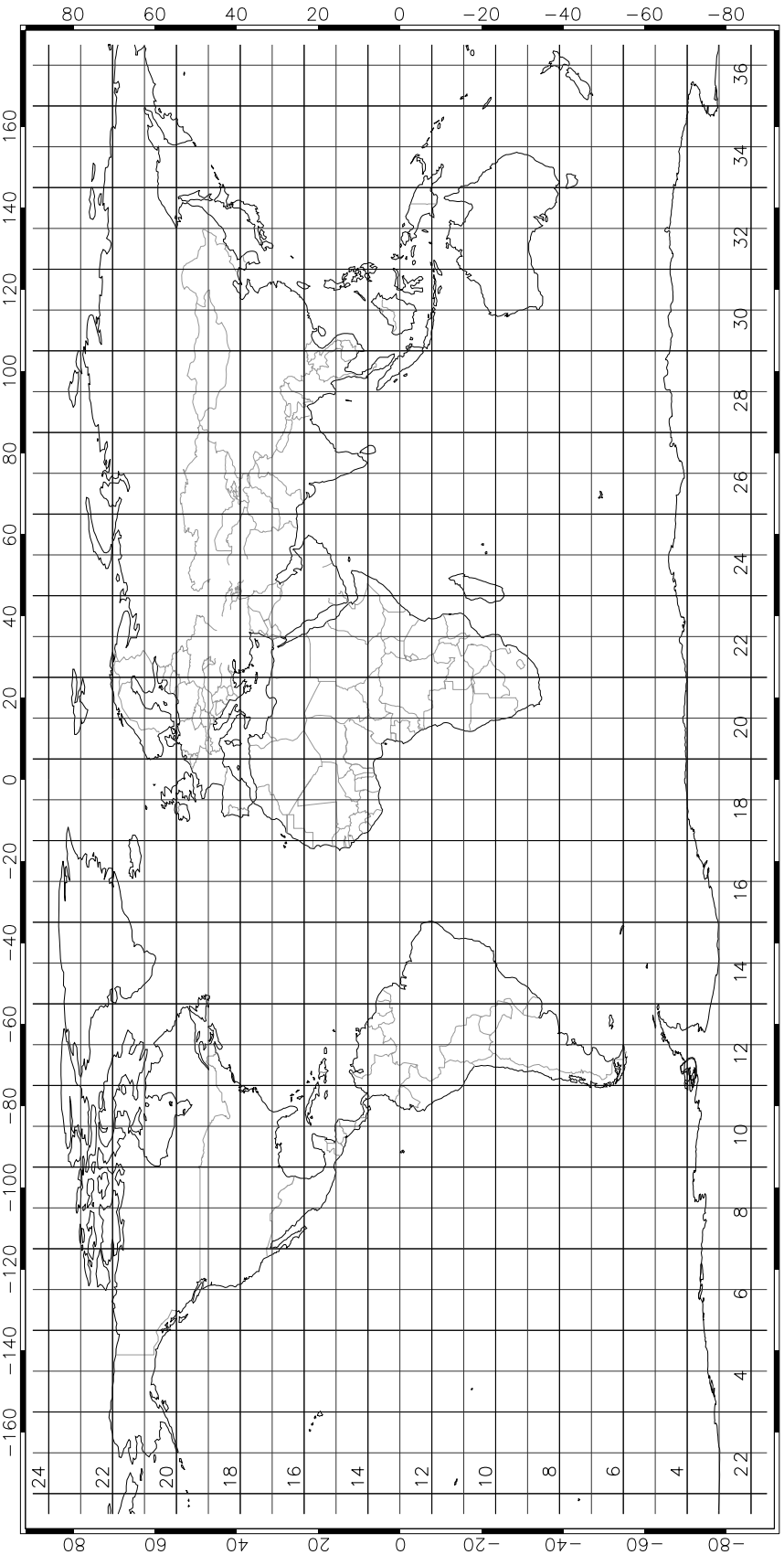
Note that, for historical reasons, within this program the direction of the vertical coordinate grid is reversed compared to TM3, i.e. the vertical levels start at the top of the atmosphere ($l = 1$) and end at the surface ($l = l_{surf}$)!

Appendix B

Horizontal Grid Layouts

Inside the plots, the indices in x- and y-direction are for those grid boxes that have the next left/lowest bold reference line as left/lower boundary. See the next pages.

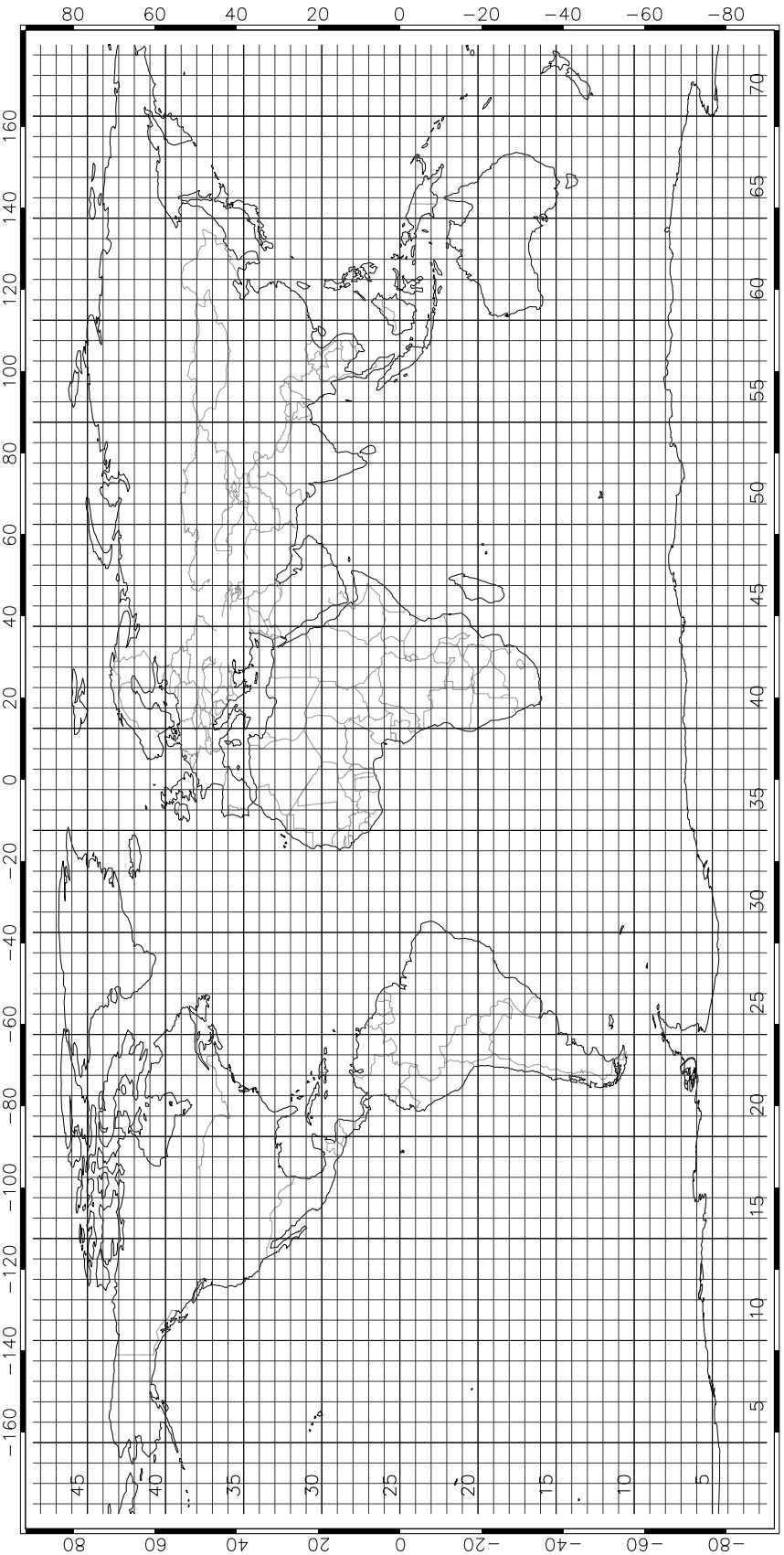
The auxiliary subroutines coords2indices and indices2coords in the Fortran module aux_grid can be used for converting TM3 grid indices to real-world coordinates and vice versa.



Grid layout for resolution cg

i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]
1	175.00	(-90.00)	10	-95.00	-23.48	19	-5.00	46.96	28	85.00
2	-175.00	-86.09	11	-85.00	-15.65	20	5.00	54.78	29	95.00
3	-165.00	-78.26	12	-75.00	-7.83	21	15.00	62.61	30	105.00
4	-155.00	-70.43	13	-65.00	0.00	22	25.00	70.43	31	115.00
5	-145.00	-62.61	14	-55.00	7.83	23	35.00	78.26	32	125.00
6	-135.00	-54.78	15	-45.00	15.65	24	45.00	86.09	33	135.00
7	-125.00	-46.96	16	-35.00	23.48	25	55.00		34	145.00
8	-115.00	-39.13	17	-25.00	31.30	26	65.00		35	155.00
9	-105.00	-31.30	18	-15.00	39.13	27	75.00		36	165.00

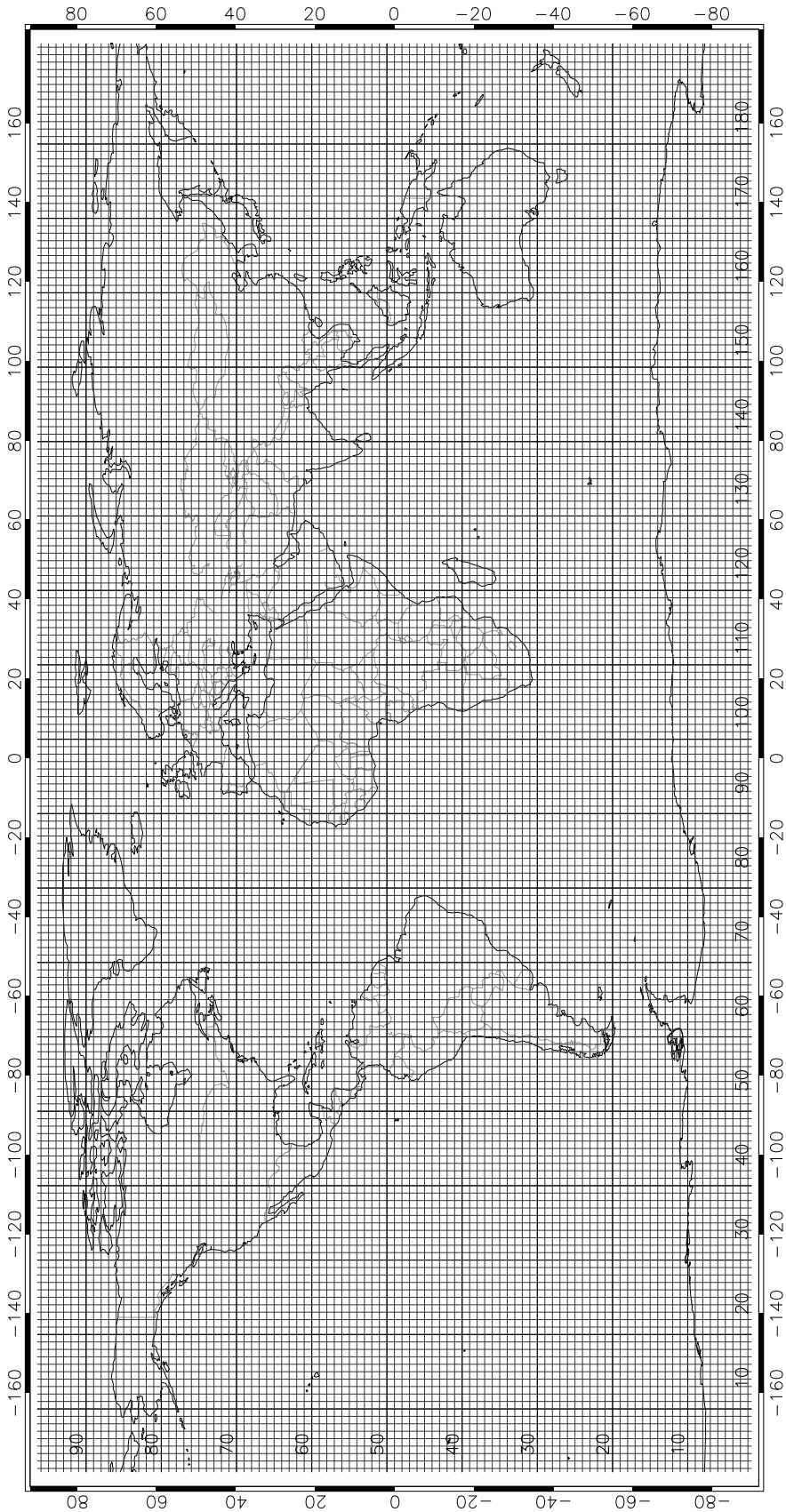
Table B.1: Western/southern grid box boundaries for the TM3 cg grid (36x24L9).



Grid layout for resolution fg

i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]
1	177.50	(-90.00)	19	-92.50	-22.98	37	-2.50	45.96	55	87.50
2	-177.50	-88.09	20	-87.50	-19.15	38	2.50	49.79	56	92.50
3	-172.50	-84.26	21	-82.50	-15.32	39	7.50	53.62	57	97.50
4	-167.50	-80.43	22	-77.50	-11.49	40	12.50	57.45	58	102.50
5	-162.50	-76.60	23	-72.50	-7.66	41	17.50	61.28	59	107.50
6	-157.50	-72.77	24	-67.50	-3.83	42	22.50	65.11	60	112.50
7	-152.50	-68.94	25	-62.50	0.00	43	27.50	68.94	61	117.50
8	-147.50	-65.11	26	-57.50	3.83	44	32.50	72.77	62	122.50
9	-142.50	-61.28	27	-52.50	7.66	45	37.50	76.60	63	127.50
10	-137.50	-57.45	28	-47.50	11.49	46	42.50	80.43	64	132.50
11	-132.50	-53.62	29	-42.50	15.32	47	47.50	84.26	65	137.50
12	-127.50	-49.79	30	-37.50	19.15	48	52.50	88.09	66	142.50
13	-122.50	-45.96	31	-32.50	22.98	49	57.50		67	147.50
14	-117.50	-42.13	32	-27.50	26.81	50	62.50		68	152.50
15	-112.50	-38.30	33	-22.50	30.64	51	67.50		69	157.50
16	-107.50	-34.47	34	-17.50	34.47	52	72.50		70	162.50
17	-102.50	-30.64	35	-12.50	38.30	53	77.50		71	167.50
18	-97.50	-26.81	36	-7.50	42.13	54	82.50		72	172.50

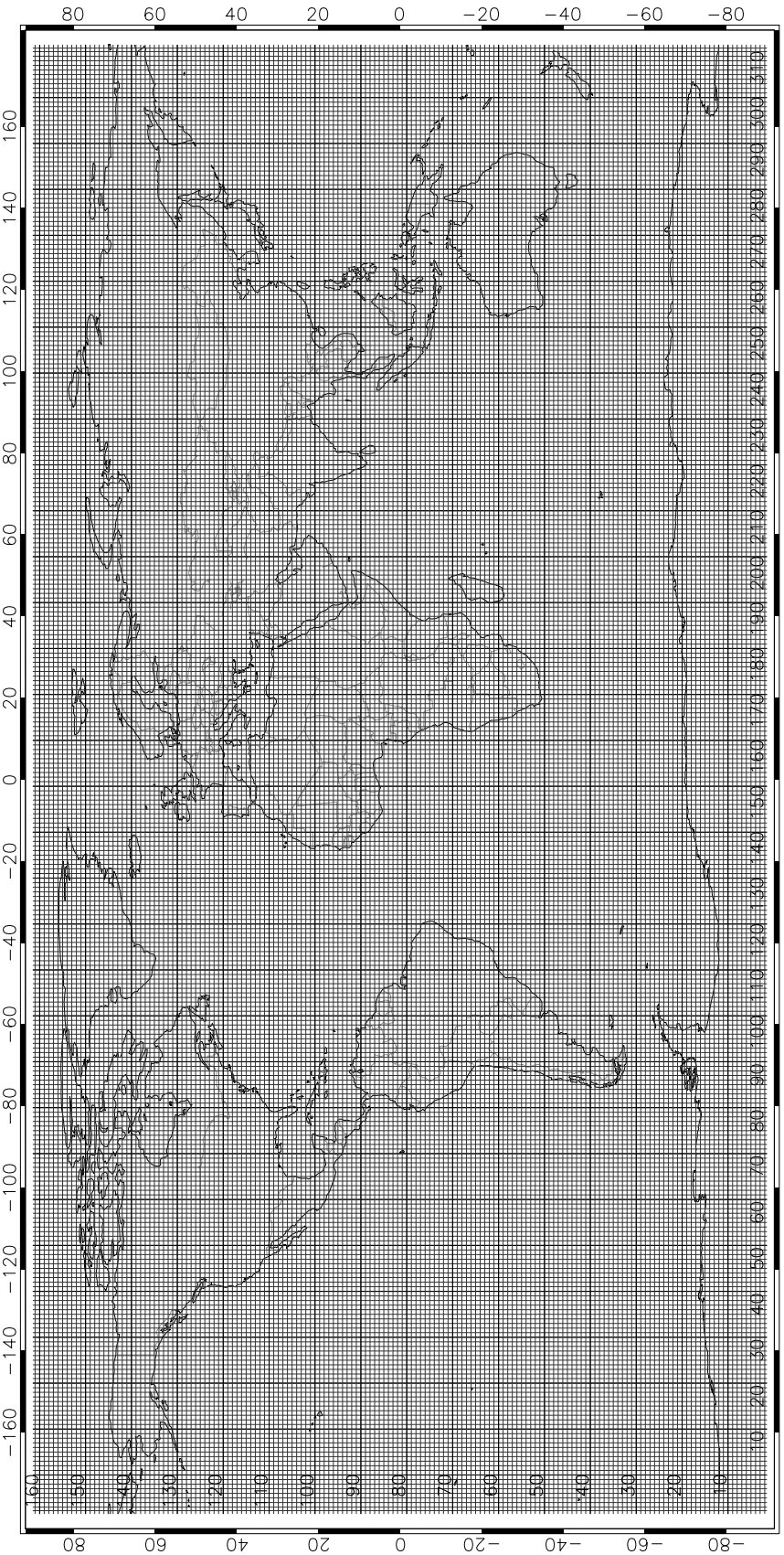
Table B.2: Western/southern grid box boundaries for the TM3 fg grid (72x48L19).



Grid layout for resolution vfg

i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]	i/j	lon [°]
1	179.06	(-90.00)	49	-90.94	0.00	97	-0.94	145	89.06
2	-179.06	-89.05	50	-89.06	1.89	98	0.94	146	90.94
3	-177.19	-87.16	51	-87.19	3.79	99	2.81	147	92.81
4	-175.31	-85.26	52	-85.31	5.68	100	4.69	148	94.69
5	-173.44	-83.37	53	-83.44	7.58	101	6.56	149	96.56
6	-171.56	-81.47	54	-81.56	9.47	102	8.44	150	98.44
7	-169.69	-79.58	55	-79.69	11.37	103	10.31	151	100.31
8	-167.81	-77.68	56	-77.81	13.26	104	12.19	152	102.19
9	-165.94	-75.79	57	-75.94	15.16	105	14.06	153	104.06
10	-164.06	-73.89	58	-74.06	17.05	106	15.94	154	105.94
11	-162.19	-72.00	59	-72.19	18.95	107	17.81	155	107.81
12	-160.31	-70.11	60	-70.31	20.84	108	19.69	156	109.69
13	-158.44	-68.21	61	-68.44	22.74	109	21.56	157	111.56
14	-156.56	-66.32	62	-66.56	24.63	110	23.44	158	113.44
15	-154.69	-64.42	63	-64.69	26.53	111	25.31	159	115.31
16	-152.81	-62.53	64	-62.81	28.42	112	27.19	160	117.19
17	-150.94	-60.63	65	-60.94	30.32	113	29.06	161	119.06
18	-149.06	-58.74	66	-59.06	32.21	114	30.94	162	120.94
19	-147.19	-56.84	67	-57.19	34.11	115	32.81	163	122.81
20	-145.31	-54.95	68	-55.31	36.00	116	34.69	164	124.69
21	-143.44	-53.05	69	-53.44	37.89	117	36.56	165	126.56
22	-141.56	-51.16	70	-51.56	39.79	118	38.44	166	128.44
23	-139.69	-49.26	71	-49.69	41.68	119	40.31	167	130.31
24	-137.81	-47.37	72	-47.81	43.58	120	42.19	168	132.19
25	-135.94	-45.47	73	-45.94	45.47	121	44.06	169	134.06
26	-134.06	-43.58	74	-44.06	47.37	122	45.94	170	135.94
27	-132.19	-41.68	75	-42.19	49.26	123	47.81	171	137.81
28	-130.31	-39.79	76	-40.31	51.16	124	49.69	172	139.69
29	-128.44	-37.89	77	-38.44	53.05	125	51.56	173	141.56
30	-126.56	-36.00	78	-36.56	54.95	126	53.44	174	143.44
31	-124.69	-34.11	79	-34.69	56.84	127	55.31	175	145.31
32	-122.81	-32.21	80	-32.81	58.74	128	57.19	176	147.19
33	-120.94	-30.32	81	-30.94	60.63	129	59.06	177	149.06
34	-119.06	-28.42	82	-29.06	62.53	130	60.94	178	150.94
35	-117.19	-26.53	83	-27.19	64.42	131	62.81	179	152.81
36	-115.31	-24.63	84	-25.31	66.32	132	64.69	180	154.69
37	-113.44	-22.74	85	-23.44	68.21	133	66.56	181	156.56
38	-111.56	-20.84	86	-21.56	70.11	134	68.44	182	158.44
39	-109.69	-18.95	87	-19.69	72.00	135	70.31	183	160.31
40	-107.81	-17.05	88	-17.81	73.89	136	72.19	184	162.19
41	-105.94	-15.16	89	-15.94	75.79	137	74.06	185	164.06
42	-104.06	-13.26	90	-14.06	77.68	138	75.94	186	165.94
43	-102.19	-11.37	91	-12.19	79.58	139	77.81	187	167.81
44	-100.31	-9.47	92	-10.31	81.47	140	79.69	188	169.69
45	-98.44	-7.58	93	-8.44	83.37	141	81.56	189	171.56
46	-96.56	-5.68	94	-6.56	85.26	142	83.44	190	173.44
47	-94.69	-3.79	95	-4.69	87.16	143	85.31	191	175.31
48	-92.81	-1.89	96	-2.81	89.05	144	87.19	192	177.19

Table B.3: Western/southern grid box boundaries for the TM3 vfg grid (192x96L31/L28).



Grid layout for resolution xfg

i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]
1	179.44	(-90.00)	81	-90.56	-0.56	161	-0.56	89.44	241	89.44
2	-179.44	-89.44	82	-89.44	0.56	162	0.56		242	90.56
3	-178.31	-88.31	83	-88.31	1.69	163	1.69		243	91.69
4	-177.19	-87.19	84	-87.19	2.81	164	2.81		244	92.81
5	-176.06	-86.06	85	-86.06	3.94	165	3.94		245	93.94
6	-174.94	-84.94	86	-84.94	5.06	166	5.06		246	95.06
7	-173.81	-83.81	87	-83.81	6.19	167	6.19		247	96.19
8	-172.69	-82.69	88	-82.69	7.31	168	7.31		248	97.31
9	-171.56	-81.56	89	-81.56	8.44	169	8.44		249	98.44
10	-170.44	-80.44	90	-80.44	9.56	170	9.56		250	99.56
11	-169.31	-79.31	91	-79.31	10.69	171	10.69		251	100.69
12	-168.19	-78.19	92	-78.19	11.81	172	11.81		252	101.81
13	-167.06	-77.06	93	-77.06	12.94	173	12.94		253	102.94
14	-165.94	-75.94	94	-75.94	14.06	174	14.06		254	104.06
15	-164.81	-74.81	95	-74.81	15.19	175	15.19		255	105.19
16	-163.69	-73.69	96	-73.69	16.31	176	16.31		256	106.31
17	-162.56	-72.56	97	-72.56	17.44	177	17.44		257	107.44
18	-161.44	-71.44	98	-71.44	18.56	178	18.56		258	108.56
19	-160.31	-70.31	99	-70.31	19.69	179	19.69		259	109.69
20	-159.19	-69.19	100	-69.19	20.81	180	20.81		260	110.81
21	-158.06	-68.06	101	-68.06	21.94	181	21.94		261	111.94
22	-156.94	-66.94	102	-66.94	23.06	182	23.06		262	113.06
23	-155.81	-65.81	103	-65.81	24.19	183	24.19		263	114.19
24	-154.69	-64.69	104	-64.69	25.31	184	25.31		264	115.31
25	-153.56	-63.56	105	-63.56	26.44	185	26.44		265	116.44
26	-152.44	-62.44	106	-62.44	27.56	186	27.56		266	117.56
27	-151.31	-61.31	107	-61.31	28.69	187	28.69		267	118.69
28	-150.19	-60.19	108	-60.19	29.81	188	29.81		268	119.81
29	-149.06	-59.06	109	-59.06	30.94	189	30.94		269	120.94
30	-147.94	-57.94	110	-57.94	32.06	190	32.06		270	122.06
31	-146.81	-56.81	111	-56.81	33.19	191	33.19		271	123.19
32	-145.69	-55.69	112	-55.69	34.31	192	34.31		272	124.31
33	-144.56	-54.56	113	-54.56	35.44	193	35.44		273	125.44
34	-143.44	-53.44	114	-53.44	36.56	194	36.56		274	126.56
35	-142.31	-52.31	115	-52.31	37.69	195	37.69		275	127.69
36	-141.19	-51.19	116	-51.19	38.81	196	38.81		276	128.81
37	-140.06	-50.06	117	-50.06	39.94	197	39.94		277	129.94
38	-138.94	-48.94	118	-48.94	41.06	198	41.06		278	131.06
39	-137.81	-47.81	119	-47.81	42.19	199	42.19		279	132.19
40	-136.69	-46.69	120	-46.69	43.31	200	43.31		280	133.31
41	-135.56	-45.56	121	-45.56	44.44	201	44.44		281	134.44
42	-134.44	-44.44	122	-44.44	45.56	202	45.56		282	135.56
43	-133.31	-43.31	123	-43.31	46.69	203	46.69		283	136.69
44	-132.19	-42.19	124	-42.19	47.81	204	47.81		284	137.81
45	-131.06	-41.06	125	-41.06	48.94	205	48.94		285	138.94
46	-129.94	-39.94	126	-39.94	50.06	206	50.06		286	140.06
47	-128.81	-38.81	127	-38.81	51.19	207	51.19		287	141.19
48	-127.69	-37.69	128	-37.69	52.31	208	52.31		288	142.31
49	-126.56	-36.56	129	-36.56	53.44	209	53.44		289	143.44
50	-125.44	-35.44	130	-35.44	54.56	210	54.56		290	144.56
51	-124.31	-34.31	131	-34.31	55.69	211	55.69		291	145.69
52	-123.19	-33.19	132	-33.19	56.81	212	56.81		292	146.81
53	-122.06	-32.06	133	-32.06	57.94	213	57.94		293	147.94
54	-120.94	-30.94	134	-30.94	59.06	214	59.06		294	149.06
55	-119.81	-29.81	135	-29.81	60.19	215	60.19		295	150.19
56	-118.69	-28.69	136	-28.69	61.31	216	61.31		296	151.31
57	-117.56	-27.56	137	-27.56	62.44	217	62.44		297	152.44
58	-116.44	-26.44	138	-26.44	63.56	218	63.56		298	153.56
59	-115.31	-25.31	139	-25.31	64.69	219	64.69		299	154.69
60	-114.19	-24.19	140	-24.19	65.81	220	65.81		300	155.81

i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]	lat [°]	i/j	lon [°]
61	-113.06	-23.06	141	-23.06	66.94	221	66.94		301	156.94
62	-111.94	-21.94	142	-21.94	68.06	222	68.06		302	158.06
63	-110.81	-20.81	143	-20.81	69.19	223	69.19		303	159.19
64	-109.69	-19.69	144	-19.69	70.31	224	70.31		304	160.31
65	-108.56	-18.56	145	-18.56	71.44	225	71.44		305	161.44
66	-107.44	-17.44	146	-17.44	72.56	226	72.56		306	162.56
67	-106.31	-16.31	147	-16.31	73.69	227	73.69		307	163.69
68	-105.19	-15.19	148	-15.19	74.81	228	74.81		308	164.81
69	-104.06	-14.06	149	-14.06	75.94	229	75.94		309	165.94
70	-102.94	-12.94	150	-12.94	77.06	230	77.06		310	167.06
71	-101.81	-11.81	151	-11.81	78.19	231	78.19		311	168.19
72	-100.69	-10.69	152	-10.69	79.31	232	79.31		312	169.31
73	-99.56	-9.56	153	-9.56	80.44	233	80.44		313	170.44
74	-98.44	-8.44	154	-8.44	81.56	234	81.56		314	171.56
75	-97.31	-7.31	155	-7.31	82.69	235	82.69		315	172.69
76	-96.19	-6.19	156	-6.19	83.81	236	83.81		316	173.81
77	-95.06	-5.06	157	-5.06	84.94	237	84.94		317	174.94
78	-93.94	-3.94	158	-3.94	86.06	238	86.06		318	176.06
79	-92.81	-2.81	159	-2.81	87.19	239	87.19		319	177.19
80	-91.69	-1.69	160	-1.69	88.31	240	88.31		320	178.31

Table B.4: Western/southern grid box boundaries for the TM3 xfg grid (320x161L31).

Appendix C

Vertical Coordinate Tables

The following tables contain for each TM3 grid resolution the yearly and globally averaged pressures and heights for the TM3 model layer midpoints and layer boundaries. l denotes the TM3 layer number (— for a boundary), η the vertical hybrid coordinate $\eta = a/p_0 + b$ (reference pressure $p_0 = 985$ hPa), and p and h the respective time- and space-averaged pressures and heights.

cg grid, met. fields: NCEP/NCAR reanalysis 1998

l	η	p/hPa	h/m
—	0.000	0	81057.5
9	0.030	30	49686.0
—	0.061	60	18388.9
8	0.103	102	15914.1
—	0.146	143	13485.9
7	0.200	197	12044.5
—	0.255	251	10651.1
6	0.325	320	9357.8
—	0.396	390	8107.8
5	0.478	471	6483.0
—	0.560	552	4890.2
4	0.650	641	3726.8
—	0.740	729	2582.3
3	0.806	794	1866.2
—	0.871	858	1159.9
2	0.908	895	766.8
—	0.945	931	378.0
1	0.972	958	188.4
—	1.000	985	0.0

fg grid, met. fields: NCEP/NCAR reanalysis 1998

l	η	p/hPa	h/m
—	0.000	0	81058
19	0.010	10	53062
—	0.020	20	25094
18	0.030	30	22698
—	0.041	40	20317
17	0.051	50	19345
—	0.061	60	18389
16	0.081	80	17507
—	0.102	100	16641
15	0.112	111	15822
—	0.123	121	15020
14	0.146	144	14244
—	0.170	167	13486
13	0.197	194	12751
—	0.225	221	12032
12	0.256	252	11333
—	0.287	283	10651
11	0.342	336	9989
—	0.396	390	9343
10	0.436	429	8137
—	0.475	468	6944
9	0.518	510	5912
—	0.560	552	4890
8	0.605	596	4064
—	0.649	640	3246
7	0.695	684	2630
—	0.740	729	2019
6	0.785	773	1588
—	0.829	817	1160
5	0.870	857	872
—	0.910	897	585
4	0.928	914	481
—	0.945	931	378
3	0.959	945	295
—	0.973	958	213
2	0.983	968	147
—	0.992	977	82
1	0.996	981	41
—	1.000	985	0

vfg NCEP grid, met. fields: NCEP/NCAR reanalysis 1998

l	η	p/hPa	h/m
—	0.000	0	81058
28	0.003	3	57239
—	0.007	6	33421
27	0.010	10	30913
—	0.014	14	28404
26	0.018	18	26749
—	0.023	23	25094
25	0.029	28	23799
—	0.035	34	22504
24	0.042	41	21411
—	0.049	48	20317
23	0.058	57	19353
—	0.067	66	18389
22	0.078	77	17515
—	0.089	88	16641
21	0.103	101	15830
—	0.117	115	15020
20	0.133	131	14253
—	0.149	147	13486
19	0.168	166	12759
—	0.188	185	12032
18	0.210	207	11342
—	0.233	229	10651
17	0.259	255	9997
—	0.284	280	9343
16	0.313	308	8725
—	0.341	336	8108
15	0.372	367	7526
—	0.403	397	6944
14	0.436	429	6406
—	0.469	462	5869
13	0.502	494	5380
—	0.535	527	4890
12	0.568	560	4452
—	0.601	592	4014
11	0.633	624	3629
—	0.665	655	3246
10	0.694	684	2914
—	0.724	713	2582
9	0.751	740	2301
—	0.778	766	2019
8	0.801	789	1784
—	0.825	813	1549
7	0.846	833	1354
—	0.866	853	1160
6	0.884	871	1001
—	0.901	888	842
5	0.916	902	714
—	0.931	917	585
4	0.943	928	481
—	0.955	940	378
3	0.964	950	295
—	0.974	960	213
2	0.982	967	147
—	0.990	975	82
1	0.995	980	41
—	1.000	985	0

vfg and xfg ERA grid, met. fields: ERA-15 reanalysis 1993

l	η	p/hPa	h/m
—	0.000	0	75760.8
31	0.010	10	50717.5
—	0.020	20	25674.2
30	0.030	30	23481.8
—	0.041	40	21289.4
29	0.051	50	20024.3
—	0.061	60	18759.3
28	0.071	70	17876.5
—	0.081	80	16993.7
27	0.091	90	16309.1
—	0.102	100	15624.6
26	0.112	111	15045.7
—	0.123	121	14466.8
25	0.134	132	13948.0
—	0.146	143	13429.2
24	0.158	155	12949.6
—	0.170	167	12469.9
23	0.183	180	12019.4
—	0.196	193	11568.9
22	0.210	207	11142.7
—	0.225	221	10716.5
21	0.240	236	10311.5
—	0.255	251	9906.5
20	0.271	267	9520.1
—	0.287	283	9133.6
19	0.305	300	8763.2
—	0.322	317	8392.8
18	0.340	335	8036.9
—	0.358	352	7681.0
17	0.377	371	7338.7
—	0.396	390	6996.3
16	0.415	409	6667.0
—	0.435	428	6337.6
15	0.455	448	6020.7
—	0.475	468	5703.9
14	0.496	489	5399.0
—	0.517	510	5094.2
13	0.539	531	4801.1
—	0.560	552	4508.0
12	0.583	574	4226.4
—	0.605	595	3944.9
11	0.627	618	3674.9
—	0.649	640	3405.0
10	0.672	662	3146.9
—	0.695	684	2888.9
9	0.717	707	2643.6
—	0.740	729	2398.3
8	0.763	751	2167.1
—	0.785	773	1935.9
7	0.807	795	1720.7
—	0.829	817	1505.4
6	0.850	838	1308.4
—	0.871	858	1111.4
5	0.891	878	935.7
—	0.910	897	760.1
4	0.928	914	610.1
—	0.945	931	460.2
3	0.959	945	341.5
—	0.973	958	222.8
2	0.983	968	142.8
—	0.992	977	62.8
1	0.996	981	31.4
—	1.000	985	0.0

Appendix D

Global Program Variables

```
!*****************************************************************************
!  Main global TM3 constants and variables.
!
!  $Id: tm3global.f90,v 1.15 2003/06/11 11:25:44 tpobw Exp $
!*****************************************************************************

MODULE TM3global

   USE userconstants, ONLY: kc, ki, kf, ko, &
                            ntrace
   USE griddefs, ONLY: im, jm, lm, TM_res_ID, met_ID, ndyn_default
   IMPLICIT NONE

   SAVE


!=============================================================================
!  runtime parameters, collected in namelist 'inputz'
!
!  Convention for output time intervals:
!     >0      means "interval in seconds",
!      0             no output,
!     -1             "daily at 0:00",
!     -2             "monthly at 1st 0:00",
!     -3             "yearly, January 1 0:00".
!
!=============================================================================

!
!  1st Most frequently user-changed parameters
!  -------------------------------------------
!
   CHARACTER(32)   :: jobid  = ''                          ! label: job ID (prefixing output file names)
   CHARACTER(100)  :: xlabel = ''                          ! label of model run
   CHARACTER(100)  :: path_in_p = 'prepro'                 ! path to input prepro (met.) data
   CHARACTER(100)  :: path_in_s = 'input'                  ! path to input source data
   CHARACTER(100)  :: path_out  = 'output'                 ! path to output
   REAL(kc)        :: fscale(ntrace) = 1.0_kc              ! scale factor for output of mixing ratios:
                                                           !    converts AmountTracer/kgAir into
                                                           !    application units

   !--- date (yr,month,day,hour,min,sec)
   INTEGER         :: idatei(6) = (/1988, 1, 1, 0, 0, 0/)  ! for start of model run
   INTEGER         :: idatee(6) = (/1988, 1, 3, 0, 0, 0/)  ! for end of model run
   INTEGER         :: idatet(6) = (/0, 1, 1, 0, 0, 0/)     ! after which instantaneous
                                                           !    output to <name_mix> is written;
                                                           !    set year to 0 to write from
                                                           !    the beginning of run

   !--- computational time intervals in seconds (0=switched off)
   INTEGER         :: ndyn = ndyn_default                  ! length of full advection step
                                                           !    (basic time step of integration)
   INTEGER         :: nconv = ndyn_default                 ! interval for convection calculation
   INTEGER         :: nsrce = ndyn_default                 ! interval for source calculations (SUBR
                                                           !    'source1','source2')
   INTEGER         :: nchem = ndyn_default                 ! interval for chemistry calculations (SUBR
                                                           !    'chem1', 'chem2')

   !--- output time intervals in seconds (convention applies)
   INTEGER         :: ninst = 21600                        ! interval for output of instantaneous tracer
                                                           !    mixing ratio on file <name_mix>
   INTEGER         :: ninststn = 21600                     ! interval for output of instantaneous
                                                           !    tracer mixing ratio at station locations
```

```
                                                    !    into file <name_mixstn>
   INTEGER         :: ninsttmass = 0                ! interval for output of instantaneous tracer
                                                    !    mass on file <name_tmass>

!
!  2nd Other Adjustable Parameters
!  -------------------------------
!
   !--- mass flux input file names (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/)
   CHARACTER(20)    :: name_pu = 'stag_pu_'//TM_res_ID//'.b' ! for repeated met. year (if iyearm/=0)
   CHARACTER(20)    :: name_pv = 'stag_pv_'//TM_res_ID//'.b' !            "
   CHARACTER(20)    :: name_ps = 'stag_ps_'//TM_res_ID//'.b' !            "
   CHARACTER(20)    :: name_maxndyn = 'maxndyn_'//TM_res_ID//'.d' ! max. timesteps 'ndyn'
   CHARACTER(20)    :: name_pu_c = 'stagc_pu_'//TM_res_ID//'.b' ! for consecut. met. years (if iyearm==0)
   CHARACTER(20)    :: name_pv_c = 'stagc_pv_'//TM_res_ID//'.b' !             "
   CHARACTER(20)    :: name_ps_c = 'stagc_ps_'//TM_res_ID//'.b' !             "
   CHARACTER(20)    :: name_maxndync_c = 'maxndync_'//TM_res_ID//'.d' ! max. timesteps 'ndyn'

   !--- convection info input file names (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/)
   CHARACTER(20)    :: name_eu     = 'eu_'//TM_res_ID//'.b'
   CHARACTER(20)    :: name_du     = 'du_'//TM_res_ID//'.b'
   CHARACTER(20)    :: name_ed     = 'ed_'//TM_res_ID//'.b'
   CHARACTER(20)    :: name_dd     = 'dd_'//TM_res_ID//'.b'
   CHARACTER(20)    :: name_vdiff  = 'k_'//TM_res_ID//'.b'
   CHARACTER(20)    :: name_height = 'z_'//TM_res_ID//'.b'

   !--- additional (optional) files (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/ for t and q)
   CHARACTER(20)    :: name_t = 't_'//TM_res_ID//'.b'         ! temperature
   CHARACTER(20)    :: name_q = 'q_'//TM_res_ID//'.b'         ! humidity
   LOGICAL          :: creadt = .FALSE.                       ! read temperature field?
   LOGICAL          :: creadq = .FALSE.                       ! read humidity field?
   CHARACTER(100)   :: name_plandf = 'plandf_'//TM_res_ID//'.b' ! landfraction (with complete path)
   LOGICAL          :: read_plandf = .FALSE.                  ! read in landfraction file?
   CHARACTER(100)   :: name_stationlist = 'input/stationlist.d' ! table of station coordinates

   !--- output file names (relative to <path_out>/)
   CHARACTER(20)    :: name_mix     = 'mix.b'                 ! instantaneous mixing ratios
   CHARACTER(20)    :: name_mixstn  = 'mixstn.b'              ! instantaneous mixing ratios at stations
   CHARACTER(20)    :: name_tmass   = 'tmass.b'               ! instantaneous tracer mass field
   CHARACTER(20)    :: name_psout   = 'ps.b'                  ! instantaneous surface pressure field
   LOGICAL          :: cwrtps       = .FALSE.                 ! write instantaneous surface pressure field?

   !--- start/restart option
   INTEGER          :: istart = 1                             ! 1 Cold start
                                                              !      Initial tracer fields are set to 0.
                                                              !      They can afterwards be partially or
                                                              !      totally overwritten in subroutine
                                                              !      'trace0'.
                                                              !      No other status fields are read.
                                                              ! 2 Cold start with initial tracer fields
                                                              !      Initial tracer fields are read from
                                                              !      file
                                                              !      <path_out>//<jobid>//'modelstatus.b'.
                                                              ! 3 Job chain mode
                                                              !      By TM3, the existence of a file
                                                              !      <path_out>//<jobid>//'modelstatus.d'
                                                              !      is used as indicator if this is a
                                                              !      continuation run within a job chain or
                                                              !      the start of it.
                                                              !      The user can employ it as end
                                                              !      condition in a self-calling job
                                                              !      script.
                                                              !         IF this file does NOT EXIST, then
                                                              !      the start date is that of 'tm3.in'.
                                                              !      Initial tracer fields are read from
                                                              !      the file
                                                              !      <path_out>//<jobid>//'modelstatus.b',
                                                              !      if existent, or otherwise set to 0.
                                                              !         IF the file EXISTS, then the
                                                              !      namelist 'inputz' is first read from
                                                              !      file 'tm3.in'. Afterwards the start
                                                              !      date <idatei> is overwritten by the
                                                              !      date in
                                                              !      <path_out>//<jobid>//'modelstatus.d',
                                                              !      saved by a previous model run after
                                                              !      time interval <nsave>.
                                                              !         Initial tracer fields are read from
                                                              !      file
                                                              !      <path_out>//<jobid>//'modelstatus.b',
                                                              !      if existent.
                                                              !      That file will always be written after
                                                              !      a model run, independent of what
                                                              !      istart is set to.
                                                              !         The continuation parameter file
                                                              !      <path_out>//<jobid>//'modelstatus.d'
                                                              !      contains the start date in the form
                                                              !      YYYY MM DD hh mm ss as first line.
                                                              !      The last line contains all prepro
                                                              !      directory identifiers needed by the
                                                              !      next continuation cycle, e.g.
```

```
                                                !      '1999 2000 2001' or
                                                !      '199911 199912 200001' (without
                                                !      quotes).
                                                !      This is useful for getting prepro
                                                !      files out of an archive by shell
                                                !      script commands (see the manual for an
                                                !      example).
                                                ! 10+i As above, with i = 1,2 or 3. In
                                                !      addition the user routine 'trace1'
                                                !      will be called after 'trace0'.
                                                !
   INTEGER         :: nsave = 0                 ! Only with istart=3: Time interval in
                                                !    seconds since <idatei> for stopping TM3
                                                !    and writing the restart
                                                !    files
                                                !    <path_out>//<jobid>//'modelstatus.d' (if
                                                !    the end date 'idatee' has not been
                                                !    reached) and
                                                !    <path_out>//<jobid>//'modelstatus.b'.
                                                !    If the end date <idatee> has been
                                                !    reached,
                                                !    the program finishes having
                                                !    <path_out>//<jobid>//'modelstatus.d'
                                                !    deleted; thus that file can taken as
                                                !    indicator for the necessity of a new job
                                                !    job cycle in a recursive job script.

   !--- calendar
   INTEGER         :: iyear0 = 0                ! base year for calendar
                                                !    calculations (time is internally
                                                !    represented as seconds since
                                                !    01-Jan-<iyear0> 00:00:00).
                                                !    NOTE: because of overflow problems
                                                !    on a 32 bit machine, only years
                                                !    in the range <iyear0> ... <iyear0>+65
                                                !    can be represented.
                                                !    iyear0=0:  use year of begin of
                                                !               run (=idatei(1)) as base year
   INTEGER         :: iyearm = 0                ! year whose meteorology shall be used
                                                !    iyearm=0:  use true year
                                                !    else:      repeatedly use <iyearm>
   INTEGER         :: icalendo = 2              ! calendar type
                                                !    1   permanent 360 day year
                                                !    2   real calendar
                                                !    3   permanent 365 day year
                                                !    4   permanent 366 day year

   !--- diagnostics
   CHARACTER(20)   :: name_mmix    = 'mmix.b'   ! mean mixing ratios
   CHARACTER(20)   :: name_tables1 = 'consrv_tracer.d' ! diagnostic table 1: tracer conservation
   CHARACTER(20)   :: name_tables2 = 'consrv_air.d' ! diagnostic table 2: air conservation
   CHARACTER(20)   :: name_tables3 = 'zonal_avrg.d' ! diagnostic table 3: mean mixing ratios
   CHARACTER(20)   :: name_debug   = 'debug.d'  ! debugging info
   LOGICAL         :: cdebug = .FALSE.          ! write debug info (to <name_debug>)?
   INTEGER         :: idatem(6) = (/0, 1, 1, 0, 0, 0/) ! time for starting the computation of time
                                                !    averaged fields (files <name_mmix.b> and
                                                !    <name_tables3>); set year to 0 to start
                                                !    from the beginning of run
   INTEGER         :: ndiag   = 21600           ! sampling interval for mean quantities
   INTEGER         :: ndiagp1 =    -1           ! interval for output of conservation
                                                !    statistics on files <name_tables1> and
                                                !    <name_tables2>, and of the Courant
                                                !    statistics to <kmain>
   INTEGER         :: ndiagp2 =    -1           ! interval for output of time averaged
                                                !    fields on files <name_mmix.b>,
                                                !    <name_tables3>

   !--- checkpoints
   CHARACTER(20)   :: name_check = 'check.d'    ! output filename of mixing ratios at
                                                !     checkpoints (relative to <path_out>)
   INTEGER         :: ncheck = 0                ! interval for output of tracer mix ratios
                                                !    at checkpoints
   INTEGER         :: noindc = 0                ! actual number of checkpoints (max. 10)
   INTEGER         :: indc(3,10) = 0            ! indices of checkpoints (i,j,l)

   !--- computational parameters
   LOGICAL         :: limits = .FALSE.          ! prevent negative tracer masses?
   REAL(kc)        :: couranttarget = -1.0_kc   ! If set to something between 0 and 1,
                                                !   adaptive integration stepwidth is
                                                !   switched on, which =tries= not to
                                                !   exceed <couranttarget> (preliminary
                                                !   implementation).
   REAL(kc)        :: czeta = 1.0_kc            ! scaling factor for convection
   REAL(kc)        :: czetak = 1.0_kc           ! scaling factor for vertical diffusion
   INTEGER         :: ndiff = 0                 ! interval for horizontal diffusion
                                                !    (currently not implemented)

!---
   NAMELIST /inputz/                                &
      !--- 1st part
```

```
      jobid, xlabel,                                                &
      path_in_p, path_in_s, path_out, fscale,                       &
      idatei, idatee, idatet,                                       &
      ndyn, nconv, nsrce, nchem,                                    &
      ninst, ninststn, ninsttmass,                                  &
      !--- 2nd part
      name_pu,   name_pv,   name_ps,   name_maxndyn,                &
      name_pu_c, name_pv_c, name_ps_c, name_maxndyn_c,              &
      name_eu, name_du, name_ed, name_dd, name_vdiff, name_height,  &
      name_t, name_q, creadt, creadq,                               &
      name_plandf, read_plandf, name_stationlist,                   &
      name_mix, name_mixstn, name_tmass, name_psout, cwrtps,        &
      istart, nsave,                                                &
      iyear0, iyearm, icalendo,                                     &
      name_mmix, name_tables1, name_tables2, name_tables3, name_debug, &
      cdebug, idatem, ndiag, ndiagp1, ndiagp2,                      &
      name_check, ncheck, noindc, indc,                             &
      limits, couranttarget, czeta, czetak, ndiff
!--- total: 66 variables

!-------------------------------------------------------------------------------
!  end of 'inputz' runtime parameters
!-------------------------------------------------------------------------------


!-------------------------------------------------------------------------------
!  geometrical constants
!-------------------------------------------------------------------------------

   REAL(kc), PARAMETER ::  pi    = 3.14159265358979323846264338327950288419716939937510_kc, &
                           twopi = 2.0_kc*pi,                    &
                           dlon  = twopi/im,                     &
                           dlat  = .5_kc*twopi/(jm-1)



!-------------------------------------------------------------------------------
!  I/O related constants
!-------------------------------------------------------------------------------

!--- I/O units for standard TM3 routines

   INTEGER, PARAMETER :: kmain   = 6         ! main control output
   INTEGER, PARAMETER :: kdebug  = 10        ! debug output

                                             ! windmassflux fields
   INTEGER, PARAMETER :: kwind1   = 11, &    !   fmu
                         kwind2   = 12, &    !   fmv
                         kwind3   = 13, &    !   ps
                         kmaxndyn = 14       !   max. timesteps 'ndyn'

                                             ! convection info
   INTEGER, PARAMETER :: kconv1  = 15, &     !   entrainment updraft
                         kconv2  = 16, &     !               downdraft
                         kconv3  = 17, &     !   detrainment updraft
                         kconv4  = 18, &     !               downddraft
                         kconv5  = 19, &     !   vertical diffusion coefficient
                         kconv6  = 20        !   height of level boundaries

                                             ! additional (optional) field
   INTEGER, PARAMETER :: kaddt   = 21        !   temperature
   INTEGER, PARAMETER :: kaddq   = 22        !   humidity

                                             ! output files
   INTEGER, PARAMETER :: kcheck  = 31        ! mix-ratios at check locations
   INTEGER, PARAMETER :: kdiag1  = 32        ! diagnostic tables 1 (tracer info)
   INTEGER, PARAMETER :: kdiag2  = 33        ! diagnostic tables 2 (air mass)
   INTEGER, PARAMETER :: kdiag3  = 34        ! diagnostic tables 3 (mean field)
   INTEGER, PARAMETER :: kmix    = 35        ! instantaneous mixing ratio fields
   INTEGER, PARAMETER :: kmeanx  = 36        ! averaged mixing ratio fields
   INTEGER, PARAMETER :: kmixstn = 37        ! mixing ratios at stations
   INTEGER, PARAMETER :: ktmass  = 38        ! instantaneous tracer mass fields
   INTEGER, PARAMETER :: kps     = 39        ! instantaneous surface pressure field


!--- I/O units for additional packages and/or user-supplied routines

   INTEGER, PARAMETER :: kflux   = 40        ! kflux+1 ... kflux+ntrace
                                             ! reserved for flux I/O

   INTEGER, PARAMETER :: kuser   = kflux+1+ntrace
                                             ! standard TM3 does not use
                                             ! unit numbers 'kuser' or larger
                                             ! NOTE: following non-standard
                                             ! packages do use units
                                             ! above 'kuser':
                                             !    tm3io_mixstn_ascii.f

   CHARACTER(*), PARAMETER :: pathsep = '/'  ! file path separator character
```

```
!-------------------------------------------------------------------------
!  internal control variables
!  (initialized during SUBROUTINE start)
!-------------------------------------------------------------------------

!---                                     times (in seconds since 1st-jan-year0, 00:00:00)
   INTEGER  :: itau                        !   current model time
   INTEGER  :: idate(6)                    !   date corresponding to itau
   INTEGER  :: itaui                       !   time corresponding to idatei
   INTEGER  :: itaue                       !   time corresponding to idatee
   INTEGER  :: itaut                       !   time corresponding to idatet
   INTEGER  :: itaum                       !   time corresponding to idatem
   LOGICAL  :: newyr                       !   .true.  if at beginning of a new year and at TM3 start
   LOGICAL  :: newmonth                    !   .true.  if at beginning of a new month and at TM3 start
   LOGICAL  :: newday                      !   .true.  if at beginning of a new day and at TM3 start
                                           !     (i.e. at 00Z)
   LOGICAL  :: newsrun                     !   .true.  if at beginning of a new run
   INTEGER  :: nstep                       !   advection step counter
   LOGICAL  :: adapt_ndyn                  !   flag whether adaptive integration stepwidth is
                                           !      in use (set by range testing of <courant_limit>)
   INTEGER  :: maxndyn                     !   ndyn value for which the max. Courant number
                                           !      is 1 (for adaptive integration stepwidth)
   REAL(kc) :: cpu0                        !   process time at beginning of run (in seconds)
   INTEGER  :: nistep                      !   number of integration steps per meteo. step
   INTEGER  :: istep                       !   current integration step within meteo. step
   LOGICAL  :: jobcont                     !   .true. if a job continuation is present
                                           !     (istart=3 and saved parameter file
                                           !     'modelstatus.d' exists)


!-------------------------------------------------------------------------
!  main fields
!-------------------------------------------------------------------------

!---                                     geometry fields
   REAL(kc) :: areag                       ! surface of the globe
   REAL(kc) :: dsig(lm)                    ! thickness of model layers in sigma units
                                           !   (positive numbers!)
   REAL(kc) :: dxyp(jm)                    ! areas of grid cells (in m**2)
   REAL(kc) :: sin_lon(im)                 ! sin(longitude)
   REAL(kc) :: cos_lon(im)                 ! cos(longitude)
   REAL(kc) :: sin_lat(jm)                 ! sin(latitude)
   REAL(kc) :: cos_lat(jm)                 ! cos(latitude)
   REAL(kc) :: lambda_lb(im)               ! longitude of western grid cell boundaries in radian
   REAL(kc) :: mu_lb(jm)                   ! sine of southern grid cell boundaries
   REAL(kc) :: pland(im,jm)                ! land fraction in each grid cell

!---                                     prepro input fields
   REAL(ki) :: pu(im,jm,lm)                ! W-E mass flux
   REAL(ki) :: pv(im,jm,lm)                ! N-S mass flux
   REAL(ki) :: ps(im,jm)                   ! surface pressure
   REAL(ki) :: eu0(im,jm,lm), &            ! entrainment updraft    [previous]
               eu1(im,jm,lm)               !                        [next]
   REAL(ki) :: du0(im,jm,lm), &            ! detrainment updraft    [previous]
               du1(im,jm,lm)               !                        [next]
   REAL(ki) :: ed0(im,jm,lm), &            ! entrainment downdraft  [previous]
               ed1(im,jm,lm)               !                        [next]
   REAL(ki) :: dd0(im,jm,lm), &            ! detrainment downdraft  [previous]
               dd1(im,jm,lm)               !                        [next]
   REAL(ki) :: ak0(im,jm,0:lm), &          ! diffusion coefficient  [previous]
               ak1(im,jm,0:lm)             !                        [next]
   REAL(ki) :: gph(im,jm,0:lm)             ! geopotential height    [current]
   REAL(ki) :: gph0(im,jm,0:lm)            !                        [previous]
   REAL(ki) :: gph1(im,jm,0:lm)            !                        [next]
   REAL(ki) :: t(im,jm,lm)                 ! temperature (optional) [current]
   REAL(ki) :: t0(im,jm,lm)                !                        [previous]
   REAL(ki) :: t1(im,jm,lm)                !                        [next]
   REAL(ki) :: q(im,jm,lm)                 ! humidity (optional)    [current]
   REAL(ki) :: q0(im,jm,lm)                !                        [previous]
   REAL(ki) :: q1(im,jm,lm)                !                        [next]

!---                                     air mass field (kg)
   REAL(kc) :: m0(im,jm,lm)                ! previous
   REAL(kc) :: m(im,jm,lm)                 ! current
   REAL(kc) :: m1(im,jm,lm)                ! next

!---                                     pressure values (Pascal) at hybrid level boundaries
   REAL(kc) :: phlb(im,jm,lm+1), &         ! current
               phlb1(im,jm,lm+1)           ! next timestep


!---                                     main model fields
   REAL(kc) ::  rm(im,jm,lm,ntrace)        ! tracer mass (kg)
   REAL(kc) :: rxm(im,jm,lm,ntrace)        ! x-slope of tracer:
   REAL(kc) :: rym(im,jm,lm,ntrace)        !    m*(dchi/dx)*delta_x/2 (kg)
   REAL(kc) :: rzm(im,jm,lm,ntrace)        ! y-slope of tracer:
                                           !    m*(dchi/dy)*delta_y/2 (kg)
                                           ! z-slope of tracer:
                                           !    m*(dchi/dz)*delta_z/2 (kg)
```

```
!---                                    mixing ratios

   REAL(ko) :: mix(im,jm,lm,ntrace)          ! instantaneous mixing ratios
                                             !   updated by call to 'get_mix'
                                             !     mix = fscale * rm/m
                                             !   despite the special grid structure
                                             !     within the fields m and rm,
                                             !     all entries of the array mix
                                             !     are well-defined

!---                                    temporal interpolation
   REAL(kc) :: w0, w1                        ! current weights


!-------------------------------------------------------------------------------
!  station data
!-------------------------------------------------------------------------------

!---                                    as read from station list
   CHARACTER(12), ALLOCATABLE :: stations_name(:)           ! name
   REAL(kc), ALLOCATABLE      :: stations_lat(:)            ! latitude (degrees N)
   REAL(kc), ALLOCATABLE      :: stations_lon(:)            ! longitude (degrees E)
   REAL(kc), ALLOCATABLE      :: stations_h(:)              ! height (masl)

!---                                    calculated in 'init_stations'
   INTEGER                    :: stations_n                 ! # of stations
   INTEGER, ALLOCATABLE       :: stations_inw(:)            ! coordinates of 4
   INTEGER, ALLOCATABLE       :: stations_ine(:)            !   surrounding grid cells
   INTEGER, ALLOCATABLE       :: stations_isw(:)
   INTEGER, ALLOCATABLE       :: stations_ise(:)
   INTEGER, ALLOCATABLE       :: stations_jnw(:)
   INTEGER, ALLOCATABLE       :: stations_jne(:)
   INTEGER, ALLOCATABLE       :: stations_jsw(:)
   INTEGER, ALLOCATABLE       :: stations_jse(:)
   REAL(kc), ALLOCATABLE      :: stations_di(:)             ! fractional distance from
   REAL(kc), ALLOCATABLE      :: stations_dj(:)             !   SW grid cell

!---                                    mixing ratios
   REAL(ko), ALLOCATABLE      :: mixstn(:,:)


END MODULE TM3global
```

Appendix E

Runtime Parameters

The following commented runtime parameter template can be found under doc/tm3.in.

```
!=========================================================================
!
!  TM3 runtime parameter template
!
!  Uncomment and edit the parameters which need adjustment.
!  Every parameter is listed with the initialization expression it gets in the
!  Fortran source code. If uncommented, this has to be changed to NAMELIST I/O
!  syntax where necessary.
!
!  Convention for output time intervals:
!     >0      means "interval in seconds",
!      0              no output,
!     -1              "daily at 0:00",
!     -2              "monthly at 1st 0:00",
!     -3              "yearly, January 1 0:00".
!
!=========================================================================

&inputz
!
!  1st Most frequently user-changed parameters
!  -------------------------------------------
!
!   jobid  = ''                                ! label: job ID (prefixing output file names)
!   xlabel = ''                                ! label of model run
!   path_in_p = 'prepro'                       ! path to input prepro (met.) data
!   path_in_s = 'input'                        ! path to input source data
!   path_out  = 'output'                       ! path to output
!   fscale = 1.0_kc                            ! scale factor for output of mixing ratios:
!                                              !    converts AmountTracer/kgAir into
!                                              !    application units

!                                              !--- date (yr,month,day,hour,min,sec)
!   idatei = 1988, 1, 1, 0, 0, 0               ! for start of model run
!   idatee = 1988, 1, 3, 0, 0, 0               ! for end of model run
!   idatet = 0, 1, 1, 0, 0, 0                  ! after which instantaneous
!                                              !    output to <name_mix> is written;
!                                              !    set year to 0 to write from
!                                              !    the beginning of run

!                                              !--- computational time intervals in seconds (0=switched off)
!   ndyn = ndyn_default                        ! length of full advection step
!                                              !    (basic time step of integration)
!   nconv = ndyn_default                       ! interval for convection calculation
!   nsrce = ndyn_default                       ! interval for source calculations (SUBR
!                                              !    'source1','source2')
!   nchem = ndyn_default                       ! interval for chemistry calculations (SUBR
!                                              !    'chem1', 'chem2')

!                                              !--- output time intervals in seconds (convention applies)
!   ninst = 21600                              ! interval for output of instantaneous tracer
!                                              !    mixing ratio on file <name_mix>
!   ninststn = 21600                           ! interval for output of instantaneous
!                                              !    tracer mixing ratio at station locations
!                                              !    into file <name_mixstn>
!   ninsttmass = 0                             ! interval for output of instantaneous tracer
!                                              !    mass on file <name_tmass>

!
!  2nd Other Adjustable Parameters
!  -------------------------------
```

```
!
!   !--- mass flux input file names (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/)
!   name_pu = 'stag_pu_'//TM_res_ID//'.b'        ! for repeated met. year (if iyearm/=0)
!   name_pv = 'stag_pv_'//TM_res_ID//'.b'        !            "
!   name_ps = 'stag_ps_'//TM_res_ID//'.b'        !            "
!   name_maxndyn = 'maxndyn_'//TM_res_ID//'.d'   ! max. timesteps 'ndyn'
!   name_pu_c = 'stagc_pu_'//TM_res_ID//'.b'     ! for consecut. met. years (if iyearm==0)
!   name_pv_c = 'stagc_pv_'//TM_res_ID//'.b'     !            "
!   name_ps_c = 'stagc_ps_'//TM_res_ID//'.b'     !            "
!   name_maxndyn_c = 'maxndync_'//TM_res_ID//'.d'! max. timesteps 'ndyn'

!   !--- convection info input file names (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/)
!   name_eu     = 'eu_'//TM_res_ID//'.b'
!   name_du     = 'du_'//TM_res_ID//'.b'
!   name_ed     = 'ed_'//TM_res_ID//'.b'
!   name_dd     = 'dd_'//TM_res_ID//'.b'
!   name_vdiff  = 'k_'//TM_res_ID//'.b'
!   name_height = 'z_'//TM_res_ID//'.b'

!   !--- additional (optional) files (relative to <path_in_p>/<TM_res_ID>/<met_ID>/<year>/ for t and q)
!   name_t = 't_'//TM_res_ID//'.b'               ! temperature
!   name_q = 'q_'//TM_res_ID//'.b'               ! humidity
!   creadt = .FALSE.                             ! read temperature field?
!   creadq = .FALSE.                             ! read humidity field?
!   name_plandf = 'plandf_'//TM_res_ID//'.b'     ! landfraction (with complete path)
!   read_plandf = .FALSE.                        ! read in landfraction file?
!   name_stationlist = 'input/stationlist.d'     ! table of station coordinates

!                                                !--- output file names (relative to <path_out>/)
!   name_mix     = 'mix.b'                       ! instantaneous mixing ratios
!   name_mixstn  = 'mixstn.b'                    ! instantaneous mixing ratios at stations
!   name_tmass   = 'tmass.b'                     ! instantaneous tracer mass field
!   name_psout   = 'ps.b'                        ! instantaneous surface pressure field
!   cwrtps       = .FALSE.                       ! write instantaneous surface pressure field?

!                                                !--- start/restart option
!   istart = 1                                   ! 1 Cold start
!                                                !      Initial tracer fields are set to 0.
!                                                !      They can afterwards be partially or
!                                                !      totally overwritten in subroutine
!                                                !      'trace0'.
!                                                !      No other status fields are read.
!                                                ! 2 Cold start with initial tracer fields
!                                                !      Initial tracer fields are read from
!                                                !      file
!                                                !      <path_out>//<jobid>//'modelstatus.b'.
!                                                ! 3 Job chain mode
!                                                !      By TM3, the existence of a file
!                                                !      <path_out>//<jobid>//'modelstatus.d'
!                                                !      is used as indicator if this is a
!                                                !      continuation run within a job chain or
!                                                !      the start of it.
!                                                !      The user can employ it as end
!                                                !      condition in a self-calling job
!                                                !      script.
!                                                !         IF this file does NOT EXIST, then
!                                                !      the start date is that of 'tm3.in'.
!                                                !      Initial tracer fields are read from
!                                                !      the file
!                                                !      <path_out>//<jobid>//'modelstatus.b',
!                                                !      if existent, or otherwise set to 0.
!                                                !         IF the file EXISTS, then the
!                                                !      namelist 'inputz' is first read from
!                                                !      file 'tm3.in'. Afterwards the start
!                                                !      date <idatei> is overwritten by the
!                                                !      date in
!                                                !      <path_out>//<jobid>//'modelstatus.d',
!                                                !      saved by a previous model run after
!                                                !      time interval <nsave>.
!                                                !         Initial tracer fields are read from
!                                                !      file
!                                                !      <path_out>//<jobid>//'modelstatus.b',
!                                                !      if existent.
!                                                !      That file will always be written after
!                                                !      a model run, independent of what
!                                                !      istart is set to.
!                                                !         The continuation parameter file
!                                                !      <path_out>//<jobid>//'modelstatus.d'
!                                                !      contains the start date in the form
!                                                !      YYYY MM DD hh mm ss as first line.
!                                                !      The last line contains all prepro
!                                                !      directory identifiers needed by the
!                                                !      next continuation cycle, e.g.
!                                                !      '1999 2000 2001' or
!                                                !      '199911 199912 200001' (without
!                                                !      quotes).
!                                                !      This is useful for getting prepro
!                                                !      files out of an archive by shell
!                                                !      script commands (see the manual for an
!                                                !      example).
```

```
!                                       ! 10+i As above, with i = 1,2 or 3. In
!                                       !      addition the user routine 'trace1'
!                                       !      will be called after 'trace0'.
!                                       !
!   nsave = 0                           ! Only with istart=3: Time interval in
!                                       !    seconds since <idatei> for stopping TM3
!                                       !    and writing the restart
!                                       !    files
!                                       !    <path_out>//<jobid>//'modelstatus.d' (if
!                                       !    the end date 'idatee' has not been
!                                       !    reached) and
!                                       !    <path_out>//<jobid>//'modelstatus.b'.
!                                       !    If the end date <idatee> has been
!                                       !    reached,
!                                       !    the program finishes having
!                                       !    <path_out>//<jobid>//'modelstatus.d'
!                                       !    deleted; thus that file can taken as
!                                       !    indicator for the necessity of a new job
!                                       !    job cycle in a recursive job script.

!                                       !--- calendar
!   iyear0 = 0                          ! base year for calendar
!                                       !    calculations (time is internally
!                                       !    represented as seconds since
!                                       !    01-Jan-<iyear0> 00:00:00).
!                                       !    NOTE: because of overflow problems
!                                       !    on a 32 bit machine, only years
!                                       !    in the range <iyear0> ... <iyear0>+65
!                                       !    can be represented.
!                                       !    iyear0=0:  use year of begin of
!                                       !               run (=idatei(1)) as base year
!   iyearm = 0                          ! year whose meteorology shall be used
!                                       !    iyearm=0:  use true year
!                                       !    else:      repeatedly use <iyearm>
!   icalendo = 2                        ! calendar type
!                                       !    1   permanent 360 day year
!                                       !    2   real calendar
!                                       !    3   permanent 365 day year
!                                       !    4   permanent 366 day year

!                                       !--- diagnostics
!   name_mmix    = 'mmix.b'             ! mean mixing ratios
!   name_tables1 = 'consrv_tracer.d'    ! diagnostic table 1: tracer conservation
!   name_tables2 = 'consrv_air.d'       ! diagnostic table 2: air conservation
!   name_tables3 = 'zonal_avrg.d'       ! diagnostic table 3: mean mixing ratios
!   name_debug   = 'debug.d'            ! debugging info
!   cdebug = .FALSE.                    ! write debug info (to <name_debug>)?
!   idatem = 0, 1, 1, 0, 0, 0           ! for starting the computation of time
!                                       !    averaged fields (files <name_mmix.b> and
!                                       !    <name_tables3>); set year to 0 to start
!                                       !    from the beginning of run
!   ndiag   = 21600                     ! sampling interval for mean quantities
!   ndiagp1 =    -1                     ! interval for output of conservation
!                                       !    statistics on files <name_tables1> and
!                                       !    <name_tables2>, and of the Courant
!                                       !    statistics to <kmain>
!   ndiagp2 =    -1                     ! interval for output of time averaged
!                                       !    fields on files <name_mmix.b>,
!                                       !    <name_tables3>

!                                       !--- checkpoints
!   name_check = 'check.d'              ! output filename of mixing ratios at
!                                       !     checkpoints (relative to <path_out>/)
!   ncheck = 0                          ! interval for output of tracer mix ratios
!                                       !    at checkpoints
!   noindc = 0                          ! actual number of checkpoints (max. 10)
!   indc = 0                            ! indices of checkpoints (i,j,l)

!                                       !--- computational parameters
!   limits = .FALSE.                    ! prevent negative tracer masses?
!   couranttarget = -1.0_kc             ! If set to something between 0 and 1,
!                                       !   adaptive integration stepwidth is
!                                       !   switched on, which =tries= not to
!                                       !   exceed <couranttarget> (preliminary
!                                       !   implementation).
!   czeta = 1.0_kc                      ! scaling factor for convection
!   czetak = 1.0_kc                     ! scaling factor for vertical diffusion
!   ndiff = 0                           ! interval for horizontal diffusion
!                                       !    (currently not implemented)
/
```

Appendix F

Diagnostic Results

F.1 The Effects of Varying Grid Resolutions

F.1.1 Surface Point Emissions

The setup of this synthetic simulation comprised a stationary tracer emission of 1 kg/s into a single surface pixel over a period of a month, starting at 1st January or July 1993. The emission sites were selected according to different mean atmospheric environments: Paris, Manaus (Brazil), Bombay, Los Angeles and Zotino (Siberia). TM3 was run in version 3.8a and fed with ERA-15 reanalysis data of all available resolutions, i.e. cg, fg, vfg and xfg.

Surface Layer Maps of Monthly Means

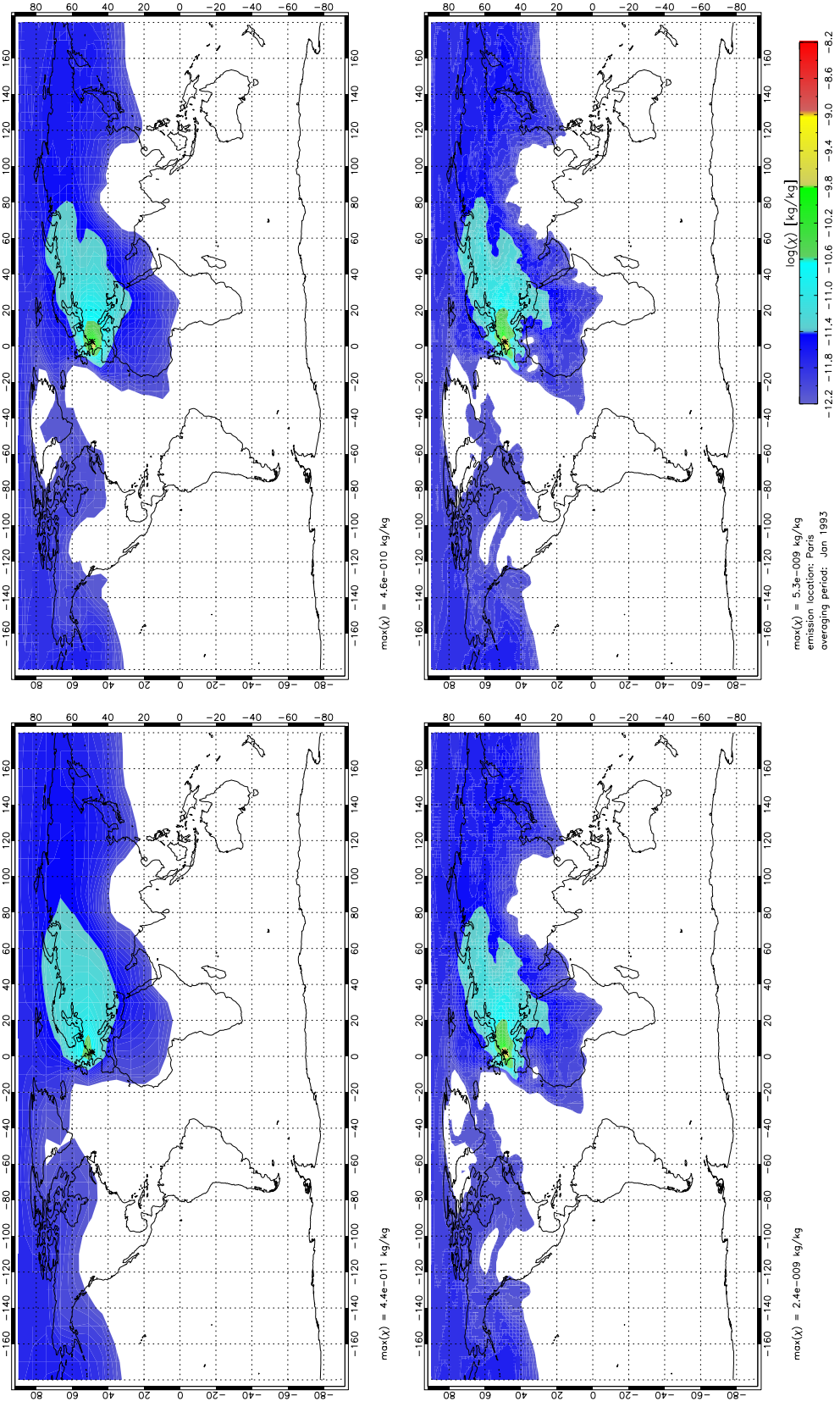
The following pages 76–95 show surface-layer maps of monthly mixing ratio means $\bar{\chi}$ on the left doublepage, respectively, and the relative deviations compared to xfg results $(\bar{\chi} - \bar{\chi}_{\mathrm{xfg}})/\bar{\chi}_{\mathrm{xfg}}$ on the corresponding right doublepage. The resolutions are cg (top left), fg (top right) and vfg (bottom left) versus xfg (bottom right), the finest available. In the deviation plots the monthly mean for xfg itself is displayed, and grid cells with a mixing ratio in the corresponding xfg cell of less than 10^{-4} times the global xfg maximum were masked out. Furthermore, relative deviations greater than 2.33 were reset to this value, or with other words, the last class in the deviation color bar is open to high values.

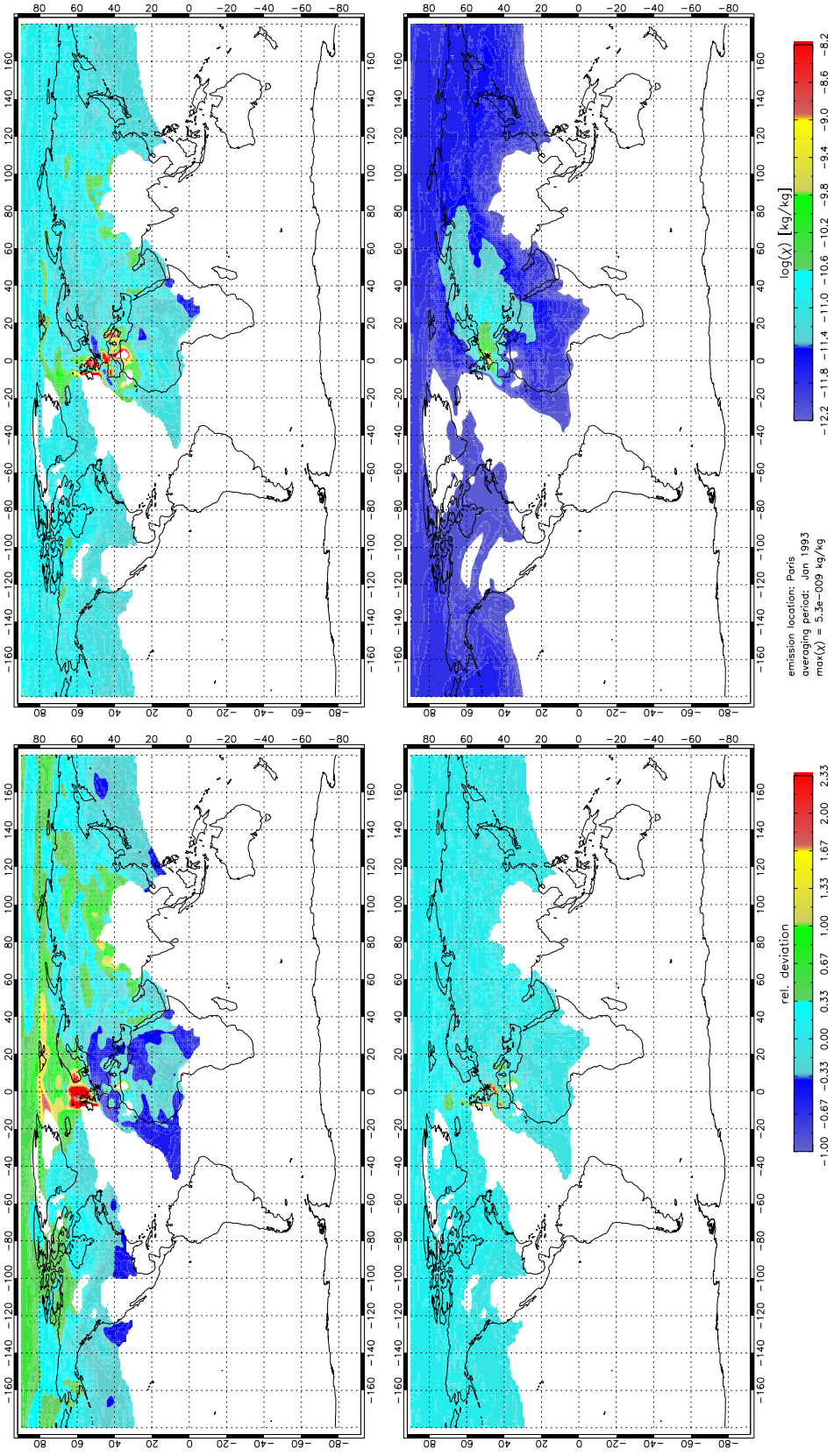
Surface-Layer Maps of Instantaneous Mixing Ratios

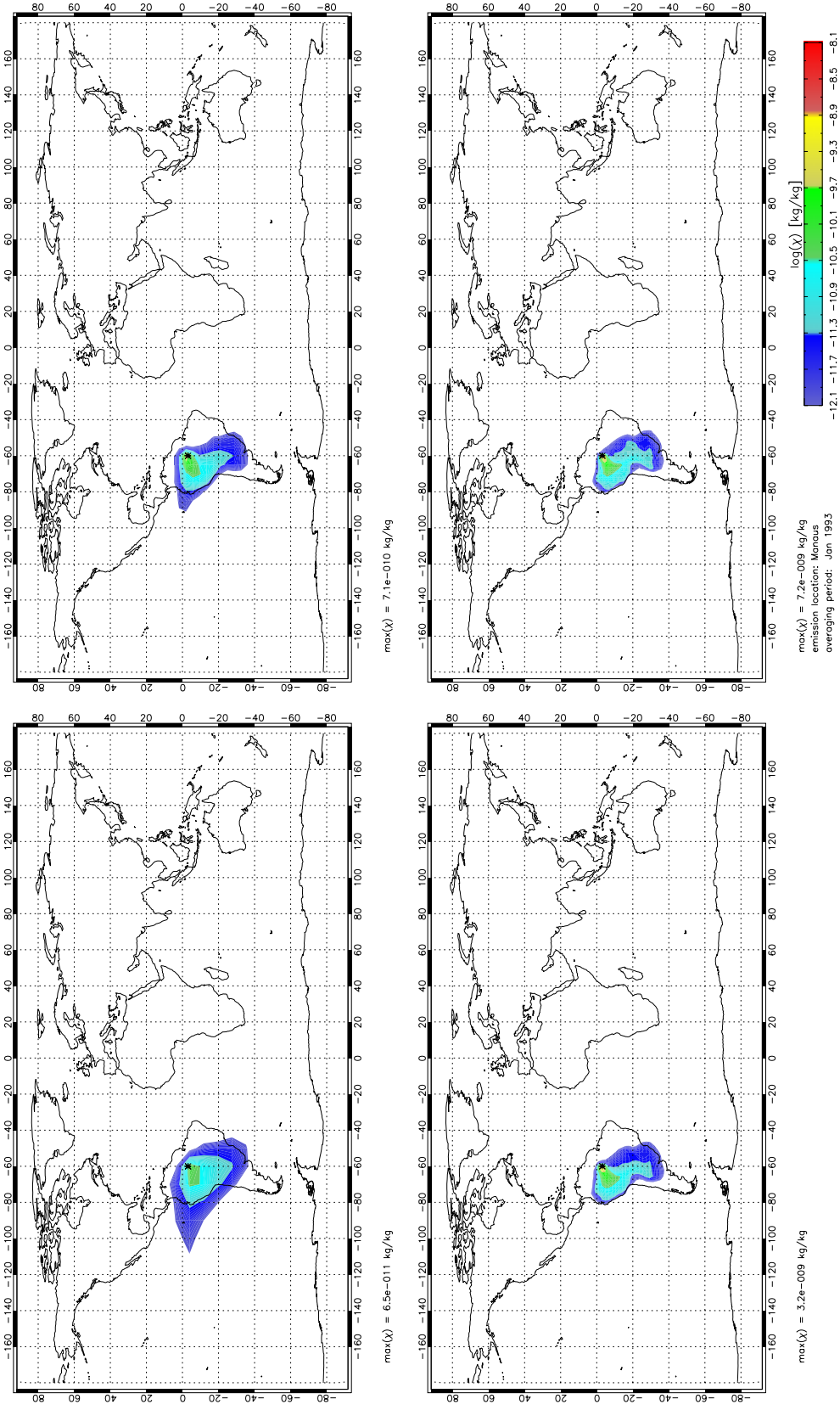
The relative deviations of *instantaneous* mixing ratios after the simulated month were in the same order of magnitude (not shown). Map panels of instantaneous surface-layer mixing ratios for the different resolutions are shown on pages 96–105.

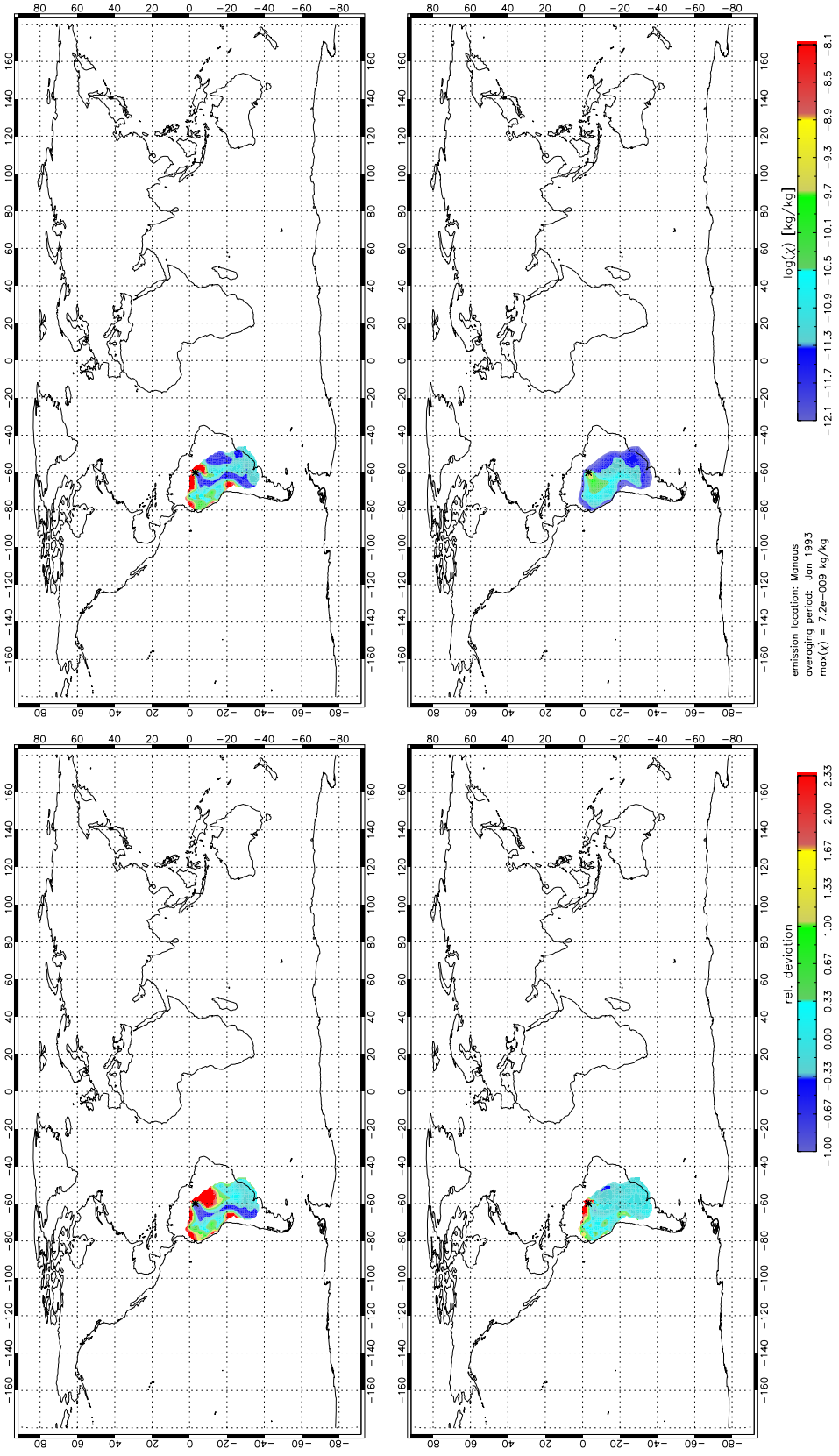
Meridional Vertical Cut in Pressure Coordinates

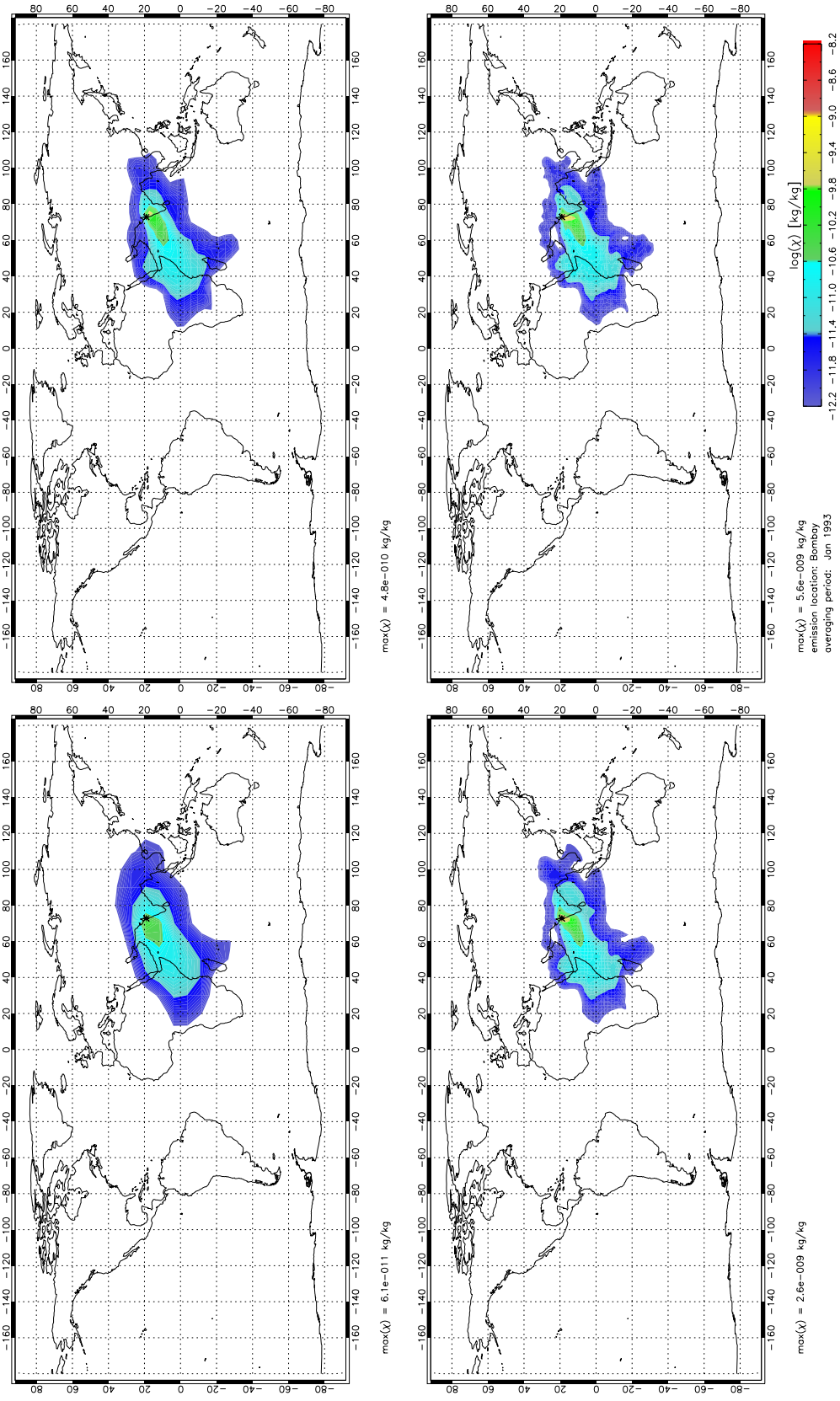
For the emission location Manaus, vertical meridional cuts of instantaneous tracer mixing ratios are shown on pages 106–107.

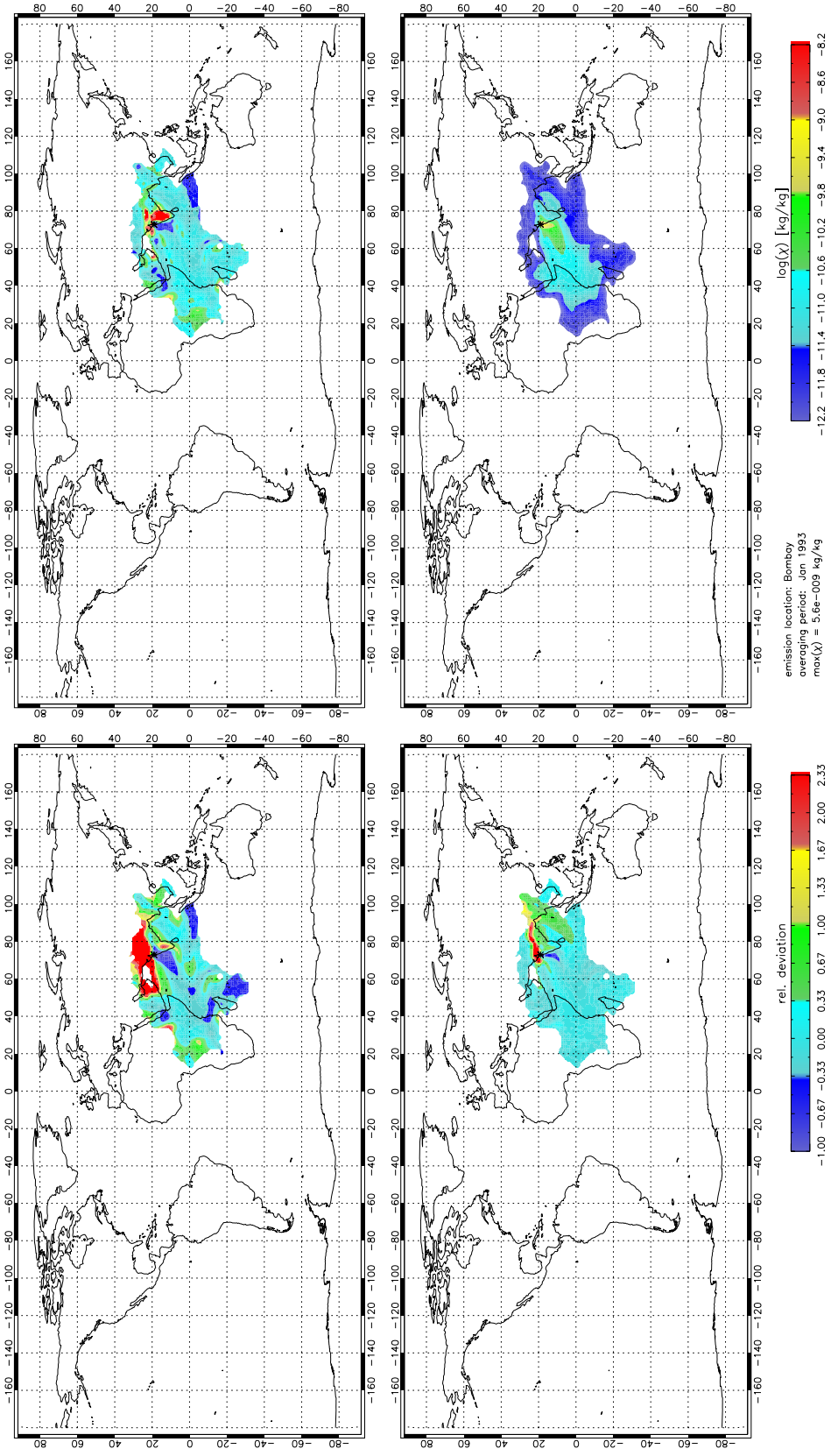

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
max(χ) = 5.3e−009 kg/kg
emission location: Paris
averaging period: Jan 1993
max(χ) = 4.6e−010 kg/kg
max(χ) = 4.4e−011 kg/kg
max(χ) = 2.4e−009 kg/kg

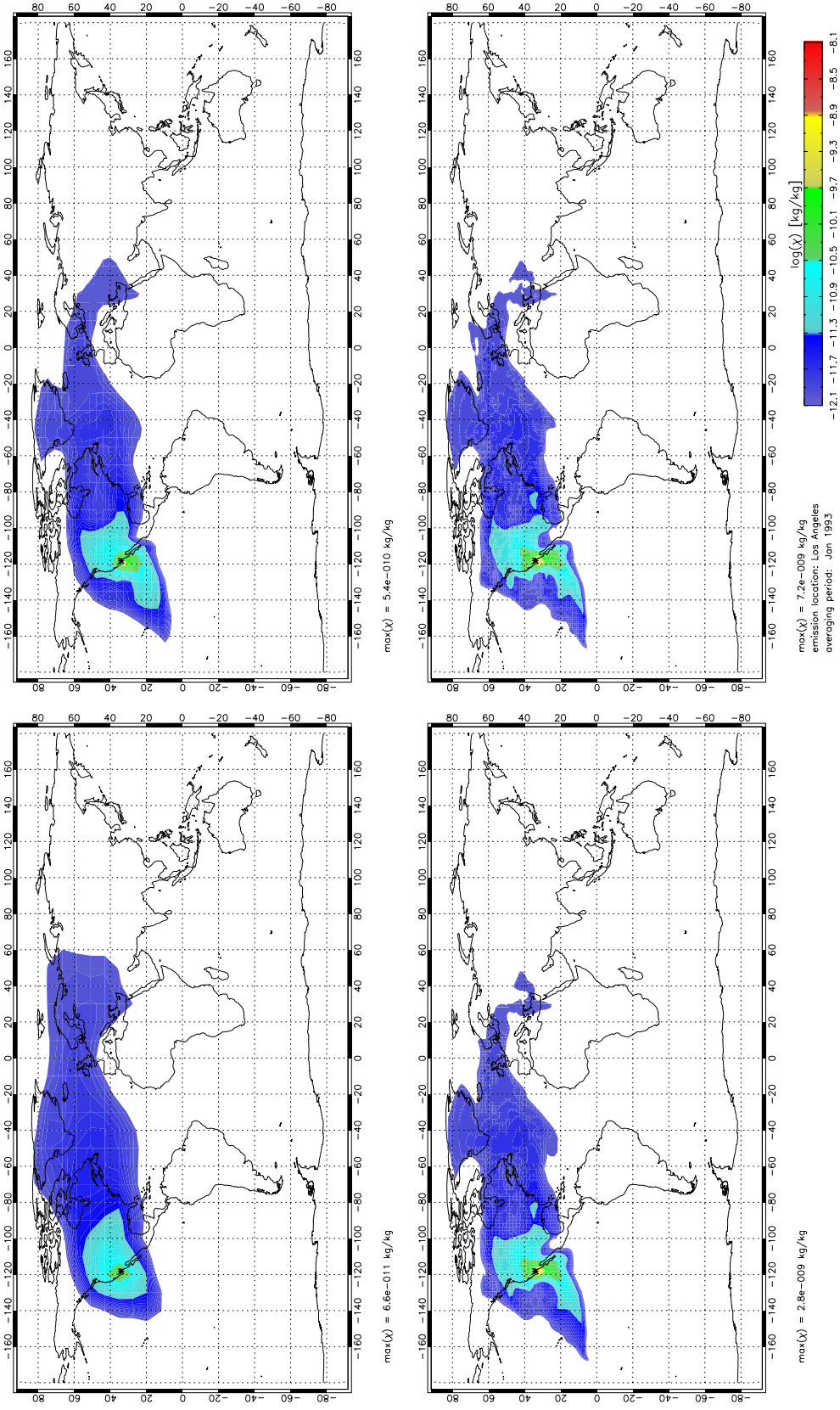

log⟨χ⟩ [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
emission location: Paris
averaging period: Jan 1993
max⟨χ⟩ = 5.3e−009 kg/kg
rel. deviation
2.33
2.00
1.67
1.33
1.00
0.67
0.33
0.00
−0.33
−0.67
−1.00

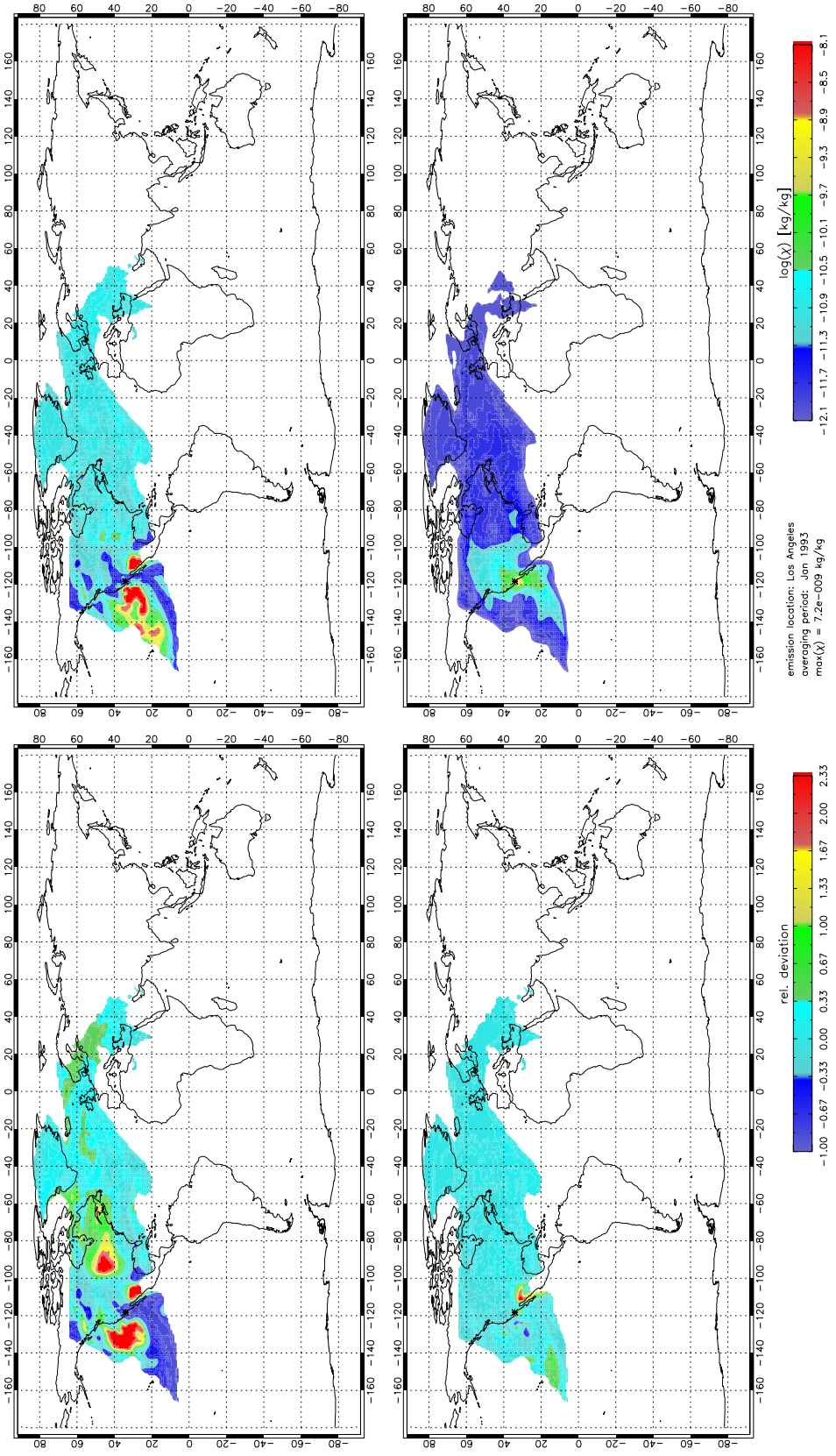

log(χ) [kg/kg]
−8.1 −8.5 −8.9 −9.3 −9.7 −10.1 −10.5 −10.9 −11.3 −11.7 −12.1
max(χ) = 7.2e−009 kg/kg
emission location: Manaus
averaging period: Jan 1993
max(χ) = 7.1e−010 kg/kg
max(χ) = 6.5e−011 kg/kg
max(χ) = 3.2e−009 kg/kg

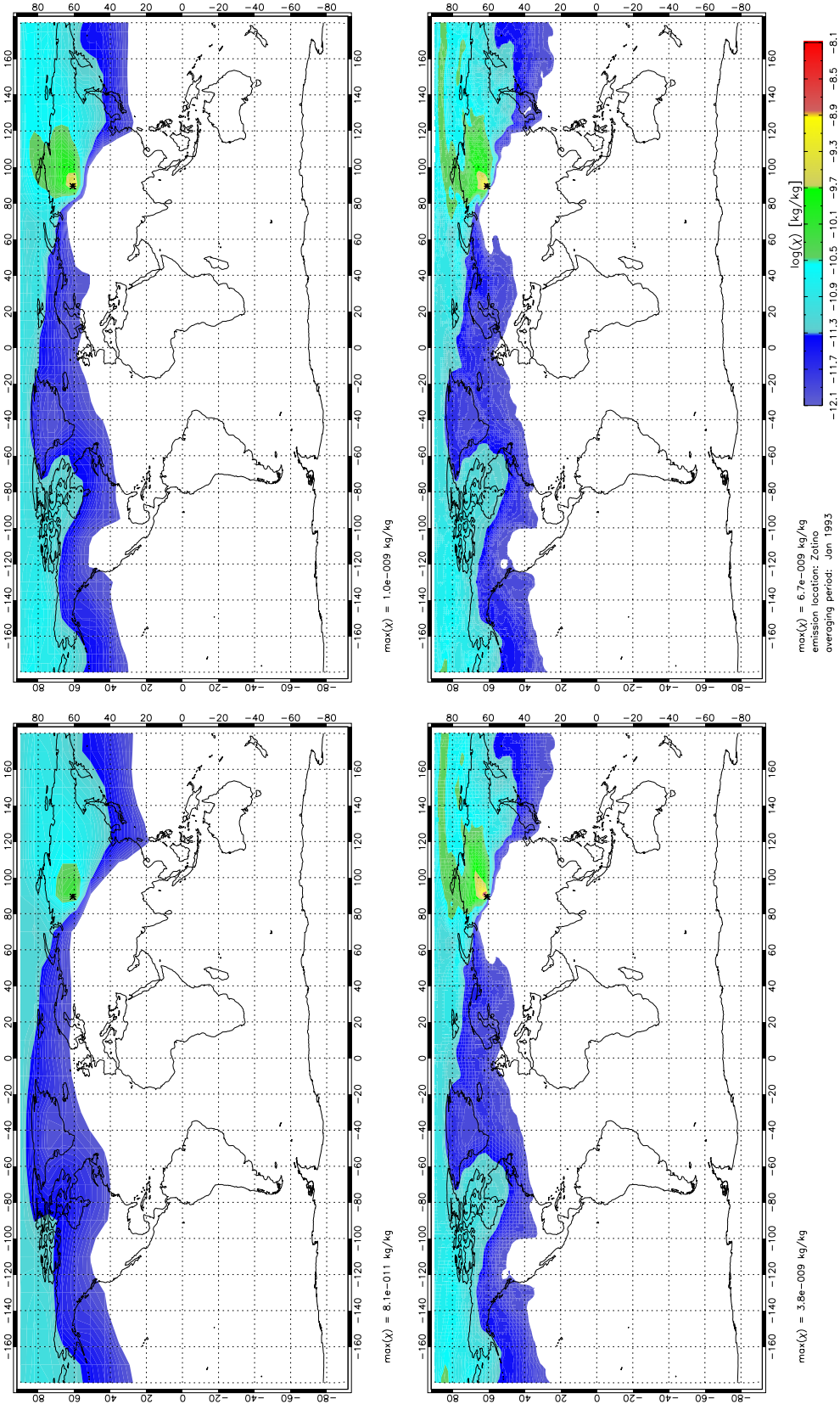

log(χ) [kg/kg]
−12.1 −11.7 −11.3 −10.9 −10.5 −10.1 −9.7 −9.3 −8.9 −8.5 −8.1
emission location: Manaus
averaging period: Jan 1993
max(χ) = 7.2e−009 kg/kg
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33

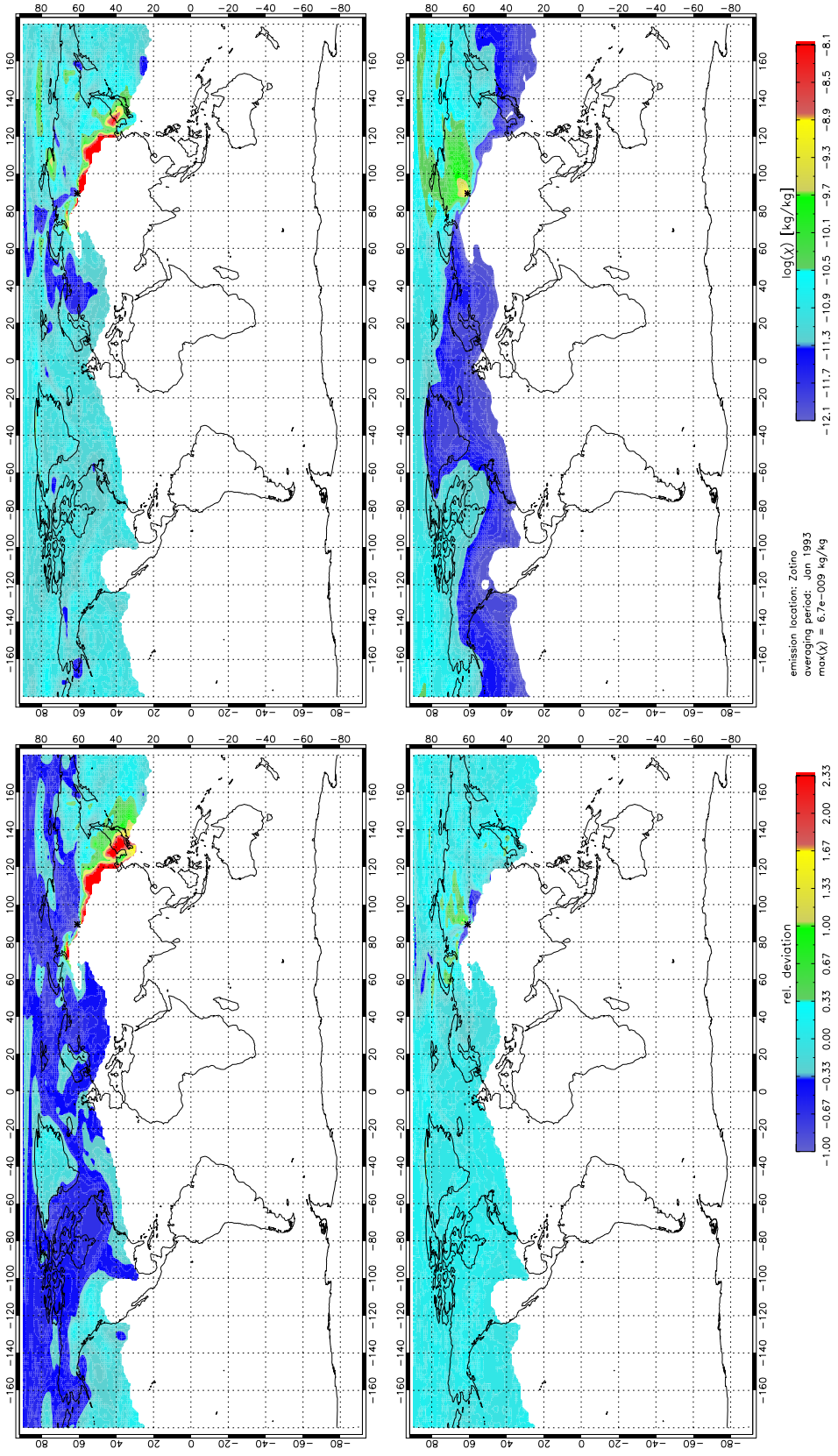

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
max(χ) = 5.6e−009 kg/kg
emission location: Bombay
averaging period: Jan 1993
max(χ) = 4.8e−010 kg/kg
max(χ) = 6.1e−011 kg/kg
max(χ) = 2.6e−009 kg/kg

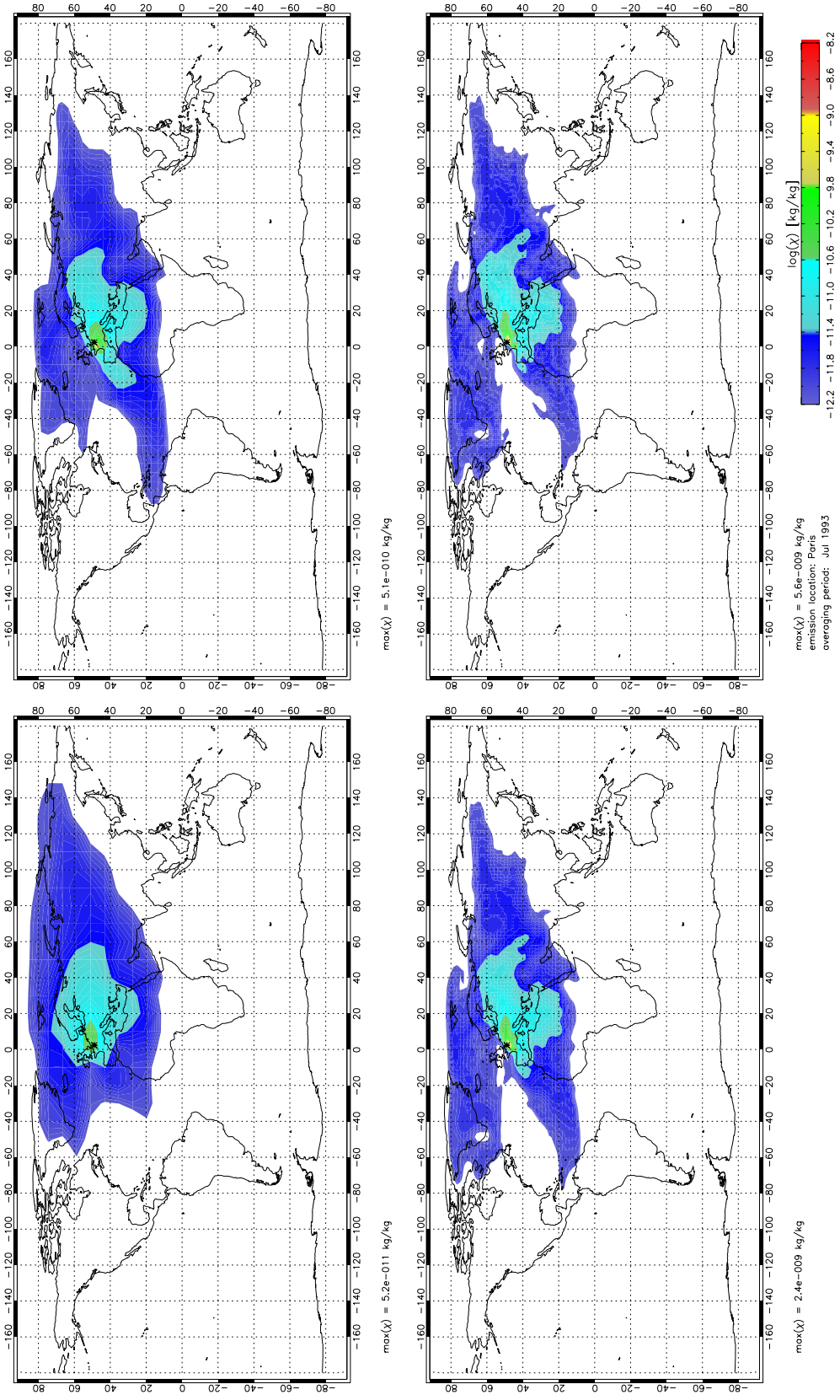

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
emission location: Bombay
averaging period: Jan 1993
max(χ) = 5.6e−009 kg/kg
rel. deviation
2.33
2.00
1.67
1.33
1.00
0.67
0.33
0.00
−0.33
−0.67
−1.00

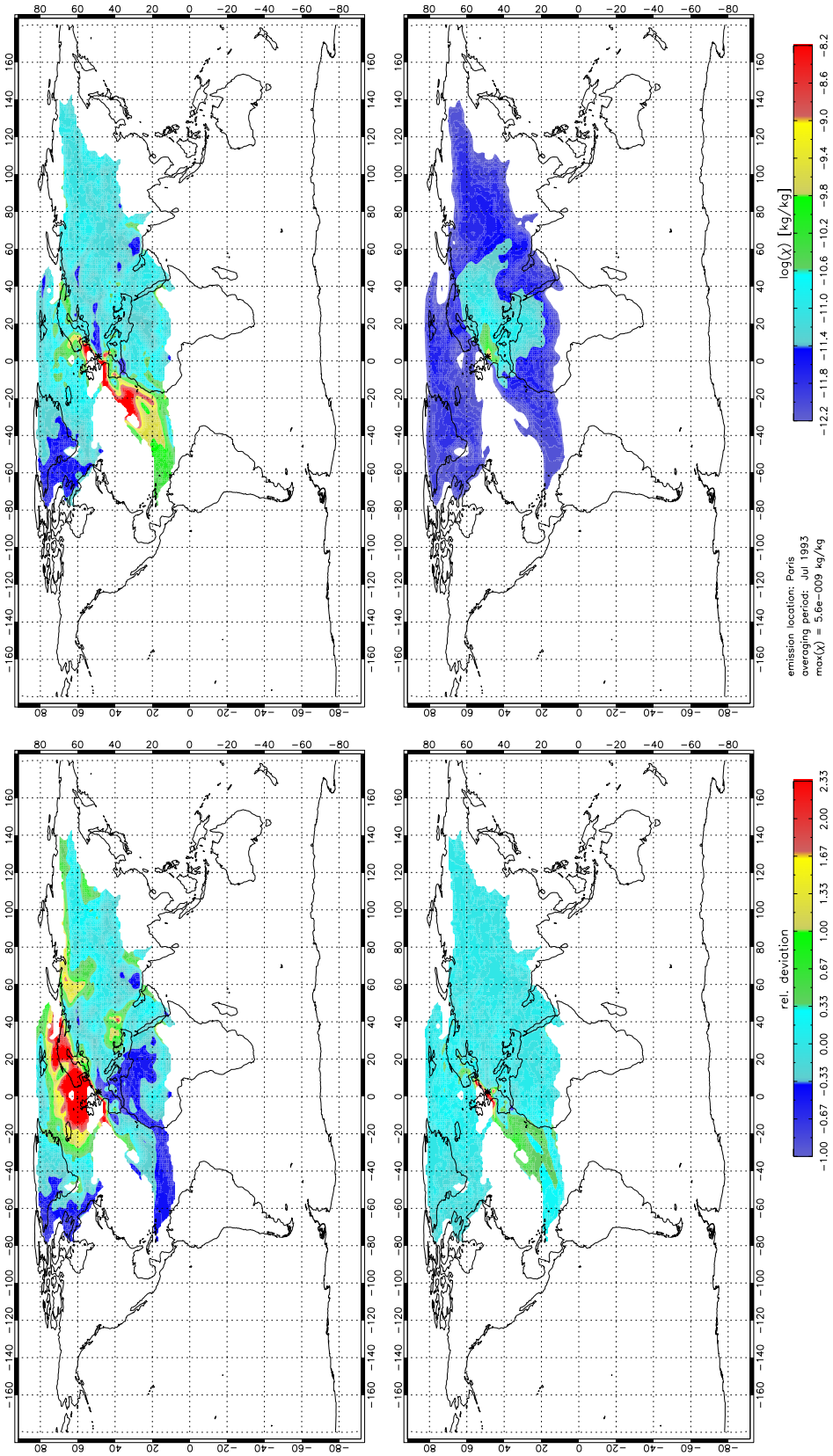

log(χ) [kg/kg]
−8.1
−8.5
−8.9
−9.3
−9.7
−10.1
−10.5
−10.9
−11.3
−11.7
−12.1
max(χ) = 7.2e−009 kg/kg
emission location: Los Angeles
averaging period: Jan 1993
max(χ) = 5.4e−010 kg/kg
max(χ) = 6.6e−011 kg/kg
max(χ) = 2.8e−009 kg/kg

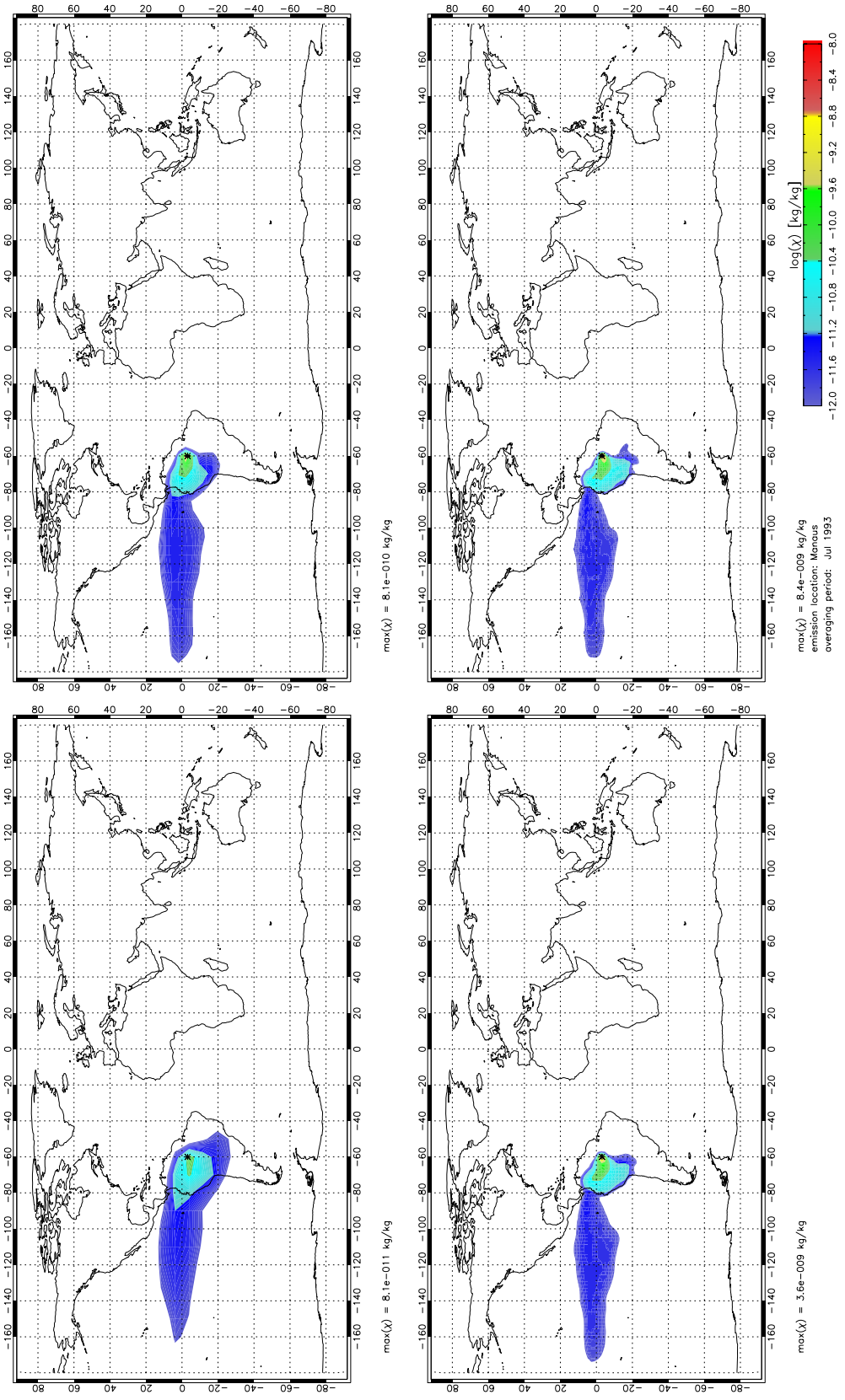

log(χ) [kg/kg]
−12.1 −11.7 −11.3 −10.9 −10.5 −10.1 −9.7 −9.3 −8.9 −8.5 −8.1
emission location: Los Angeles
averaging period: Jan 1993
max(χ) = 7.2e−009 kg/kg
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33

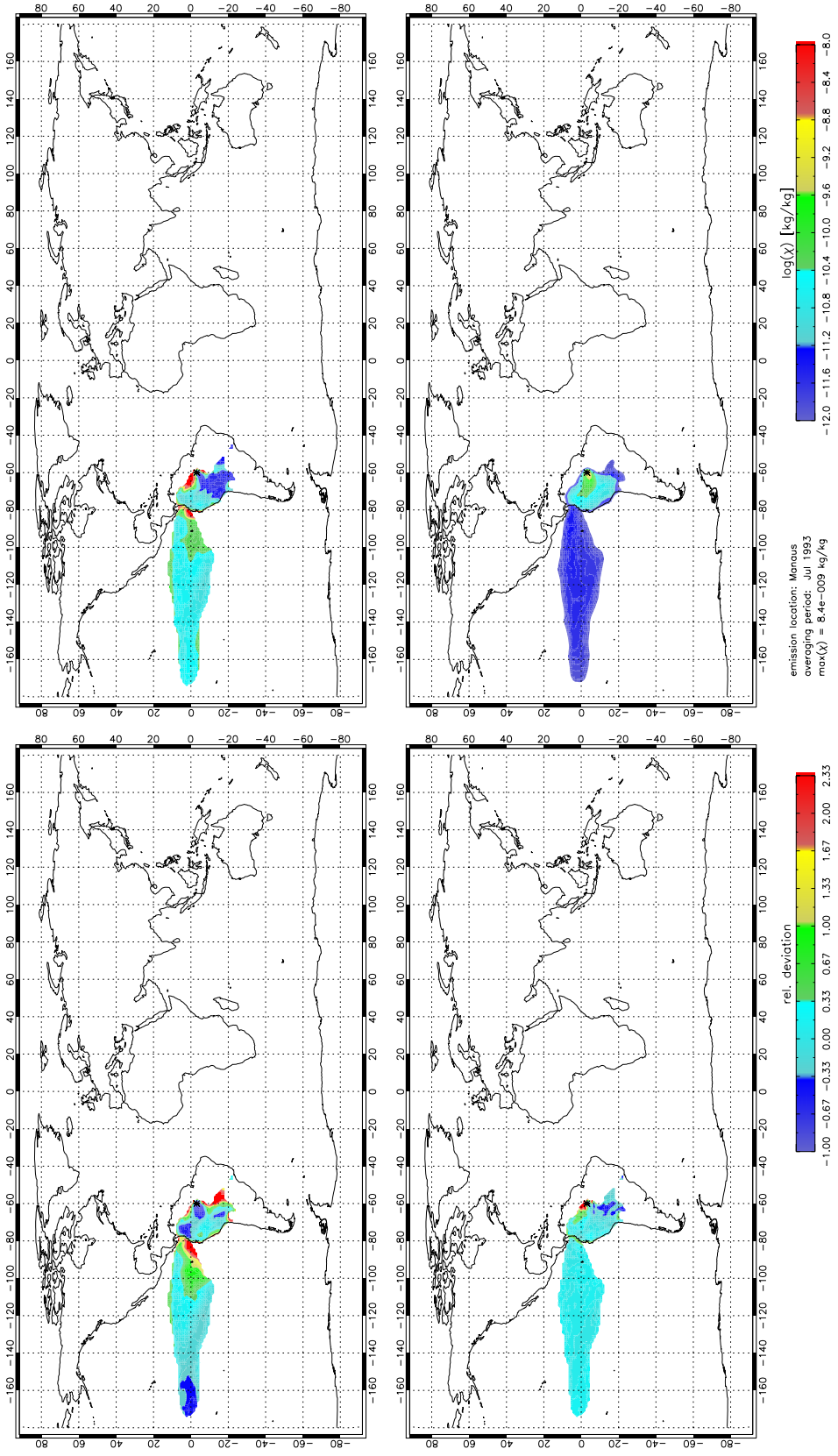

log(χ) [kg/kg]
−8.1 −8.5 −8.9 −9.3 −9.7 −10.1 −10.5 −10.9 −11.3 −11.7 −12.1
max(χ) = 6.7e−009 kg/kg
emission location: Zotino
averaging period: Jan 1993
max(χ) = 1.0e−009 kg/kg
max(χ) = 8.1e−011 kg/kg
max(χ) = 3.8e−009 kg/kg

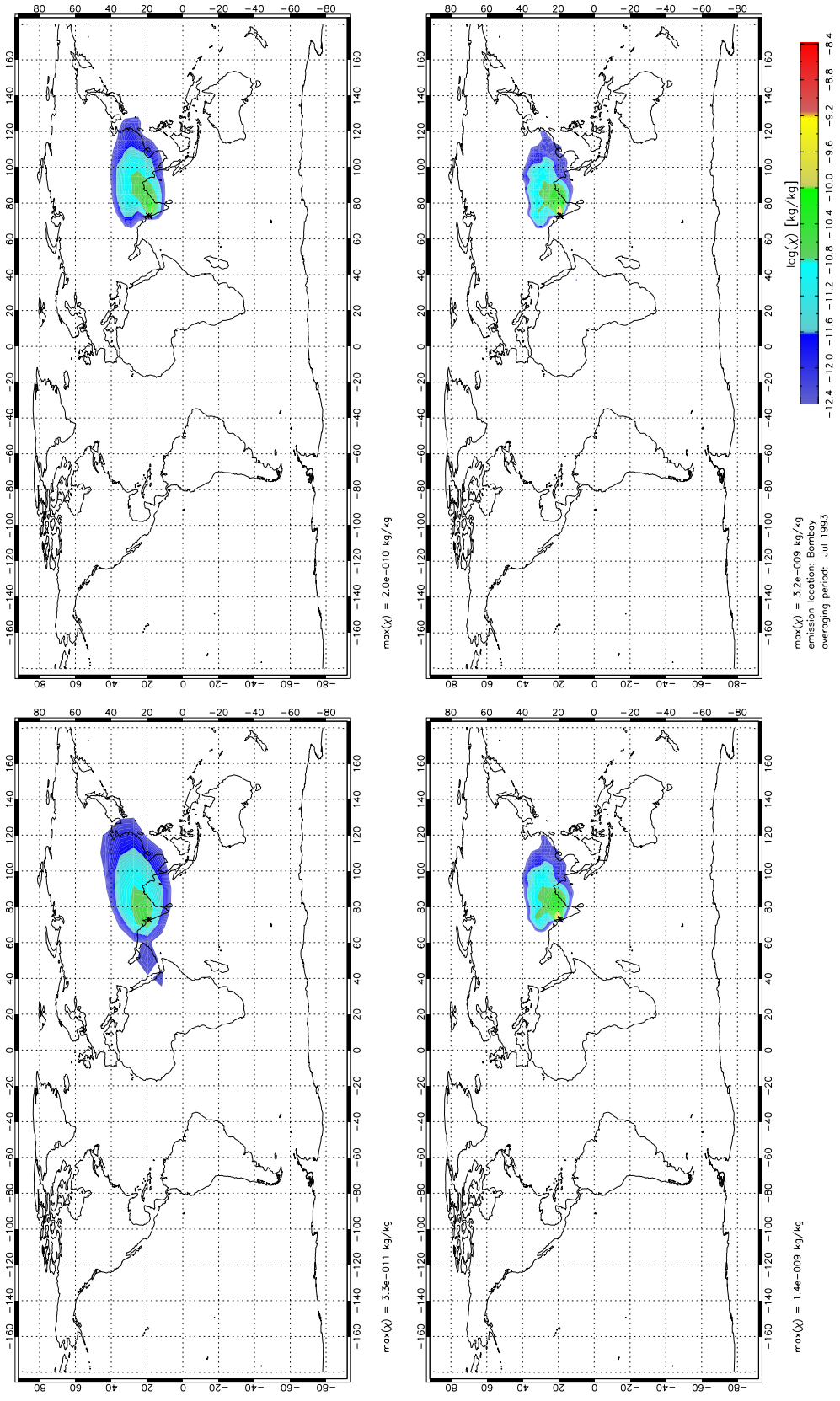

log(χ) [kg/kg]
−12.1 −11.7 −11.3 −10.9 −10.5 −10.1 −9.7 −9.3 −8.9 −8.5 −8.1
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33
emission location: Zotino
averaging period: Jan 1993
max(χ) = 6.7e−009 kg/kg

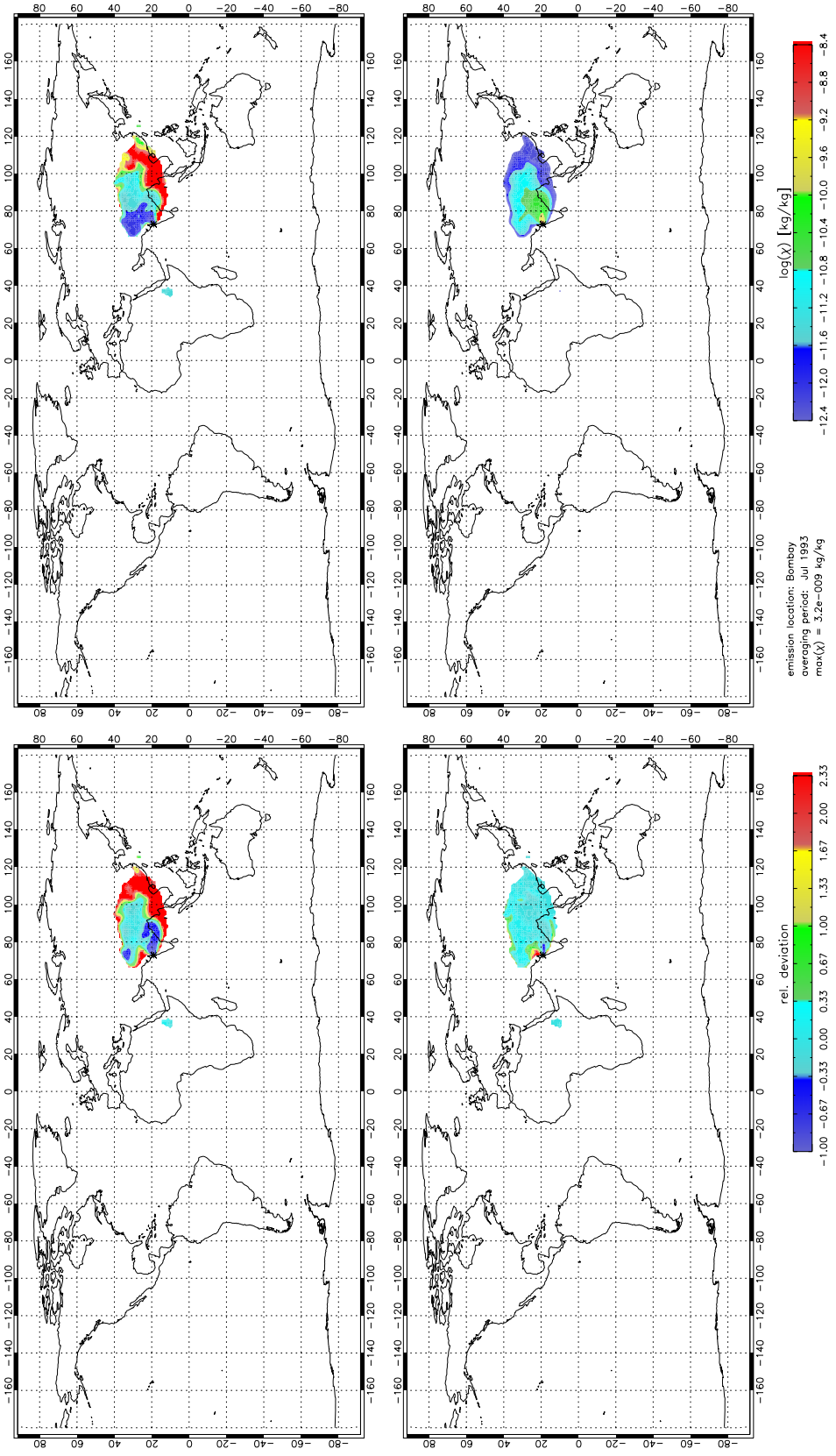

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
max(χ) = 5.6e−009 kg/kg
emission location: Paris
averaging period: Jul 1993
max(χ) = 5.1e−010 kg/kg
max(χ) = 5.2e−011 kg/kg
max(χ) = 2.4e−009 kg/kg

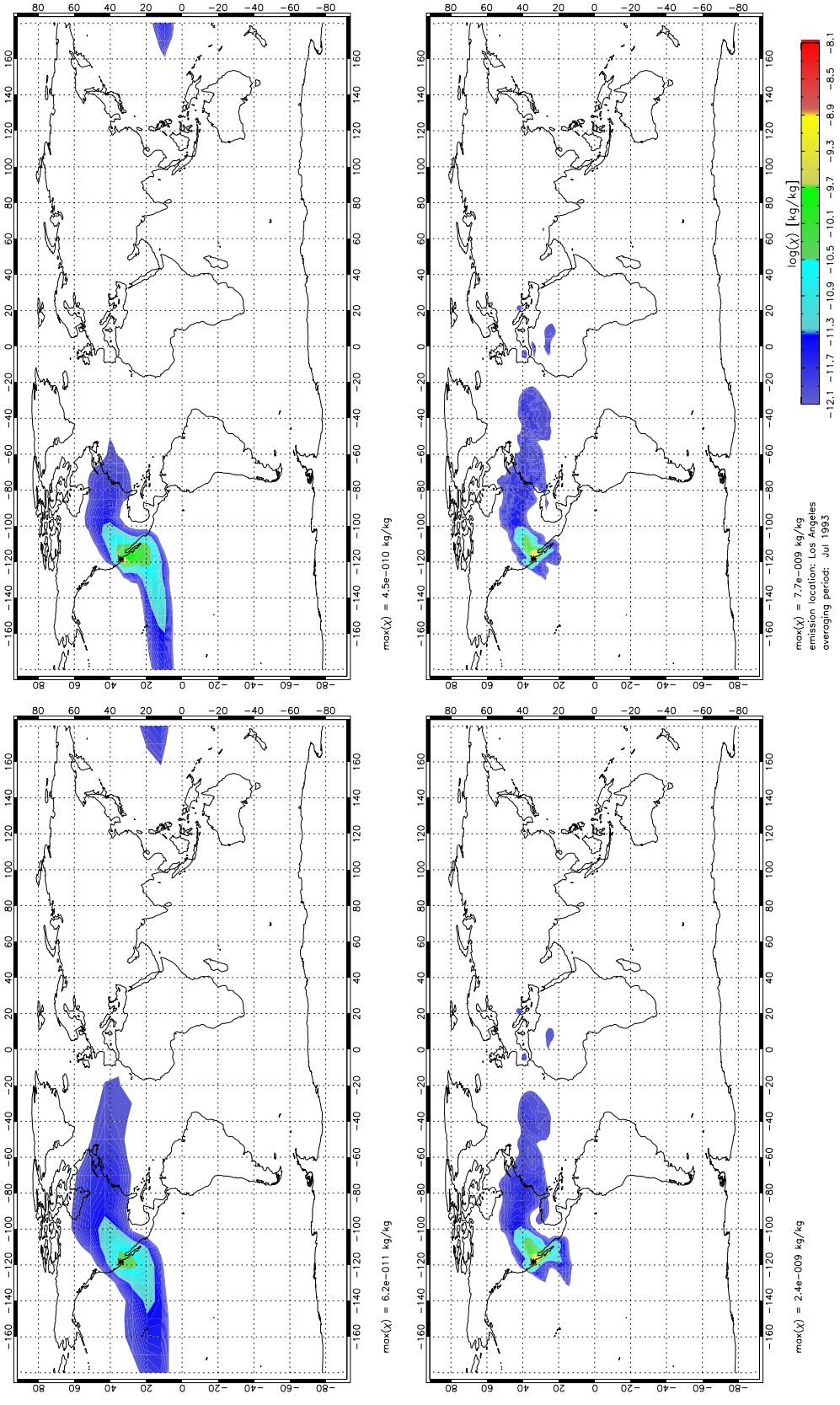

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
emission location: Paris
averaging period: Jul 1993
max(χ) = 5.6e−009 kg/kg
rel. deviation
2.33
2.00
1.67
1.33
1.00
0.67
0.33
0.00
−0.33
−0.67
−1.00

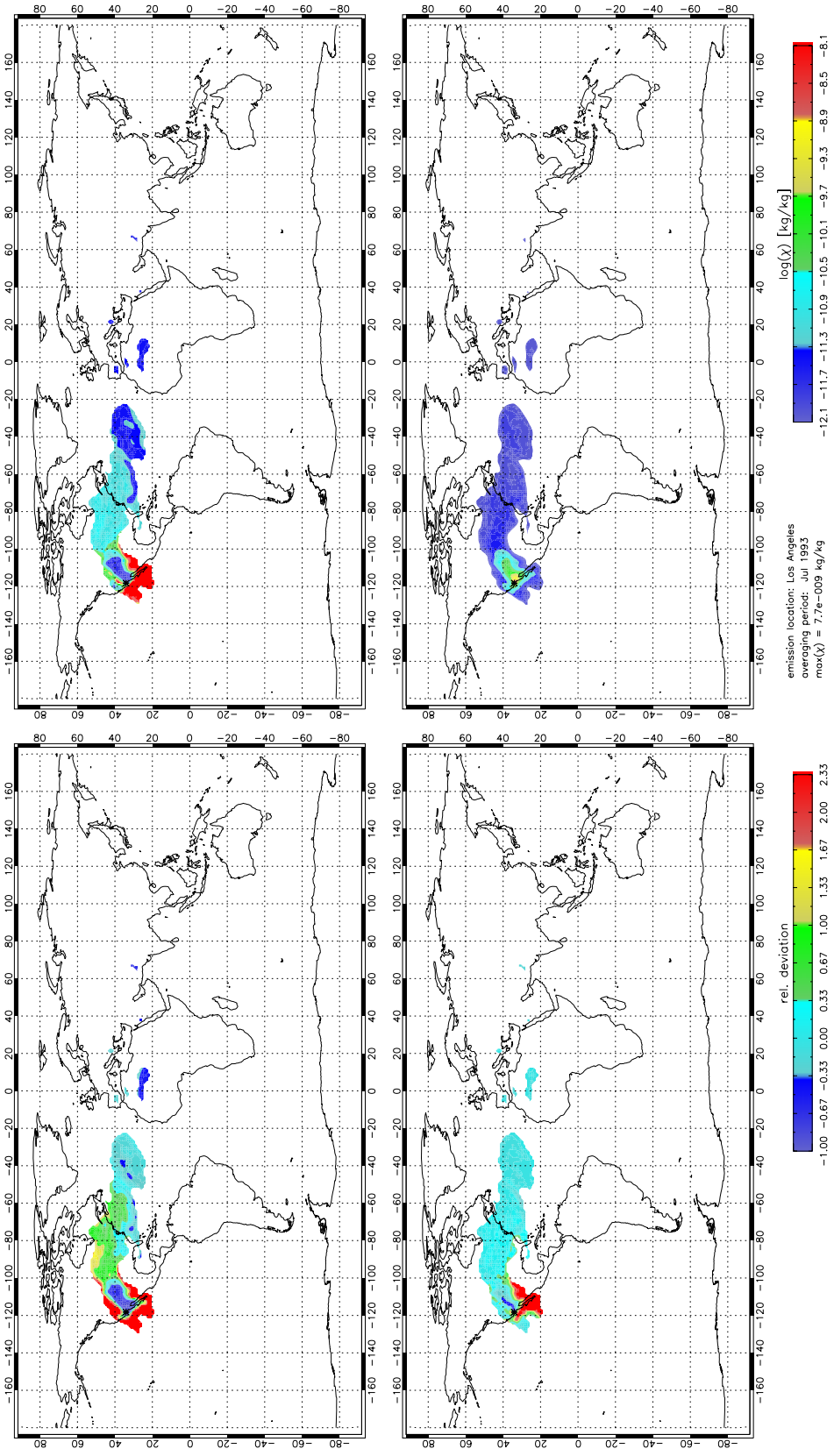

log(χ) [kg/kg]
−8.0
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
max(χ) = 8.4e−009 kg/kg
emission location: Manaus
averaging period: Jul 1993
max(χ) = 8.1e−010 kg/kg
max(χ) = 8.1e−011 kg/kg
max(χ) = 3.6e−009 kg/kg

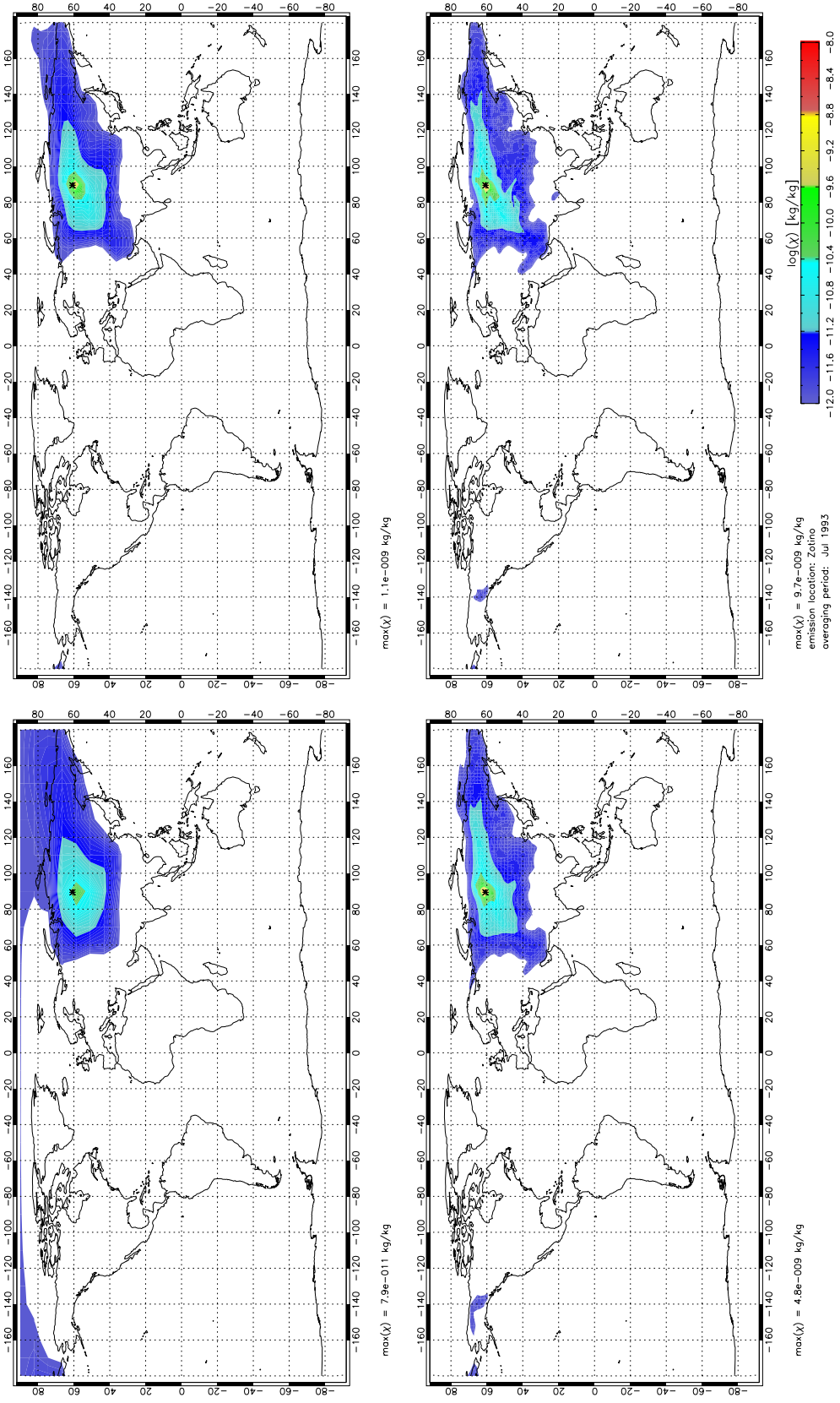

log(χ) [kg/kg]
−12.0 −11.6 −11.2 −10.8 −10.4 −10.0 −9.6 −9.2 −8.8 −8.4 −8.0
emission location: Manaus
averaging period: Jul 1993
max(χ) = 8.4e−009 kg/kg
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33

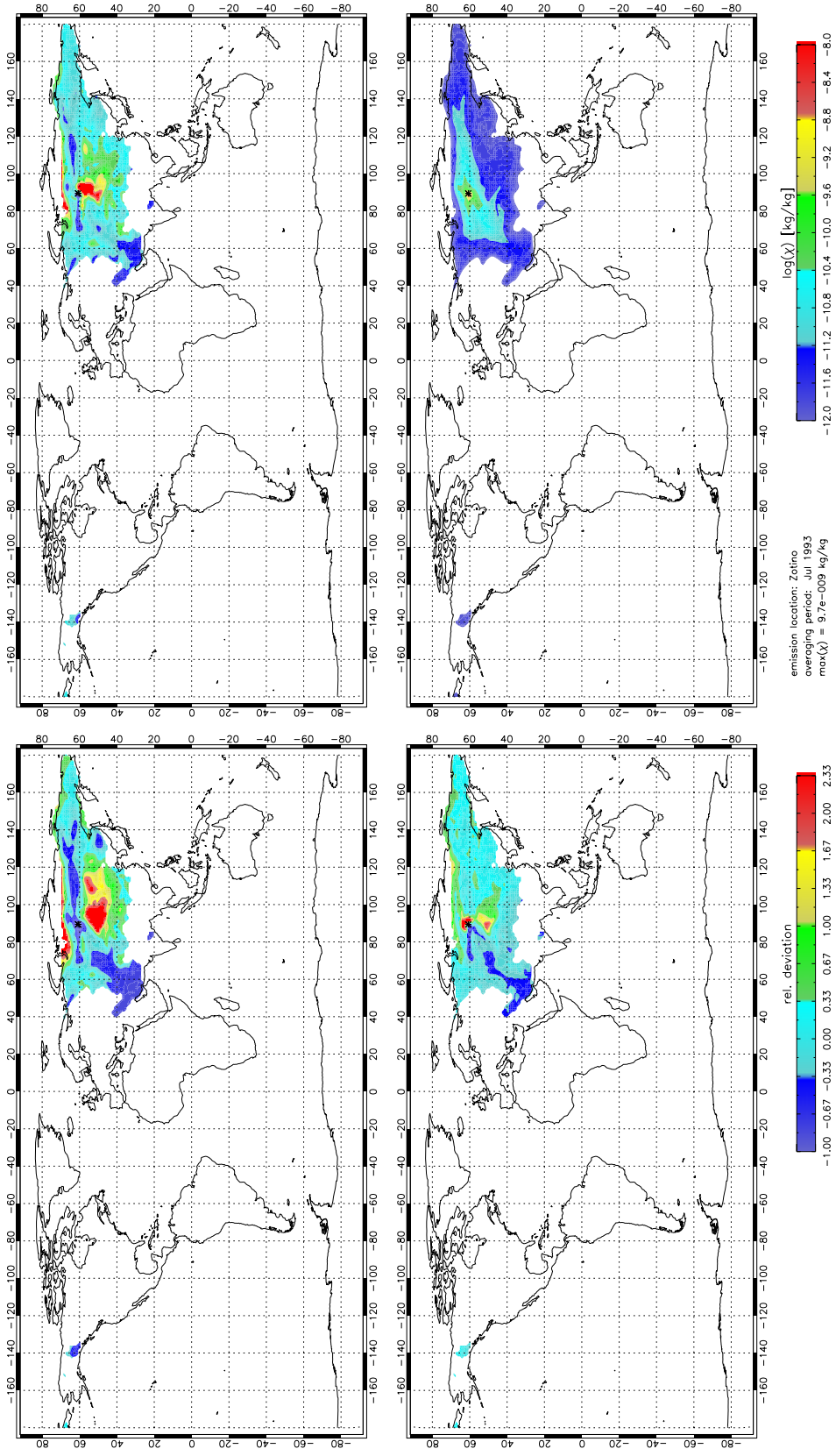

log(χ) [kg/kg]
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
−12.4
max(χ) = 2.0e−010 kg/kg
max(χ) = 3.2e−009 kg/kg
emission location: Bombay
averaging period: Jul 1993
max(χ) = 3.3e−011 kg/kg
max(χ) = 1.4e−009 kg/kg

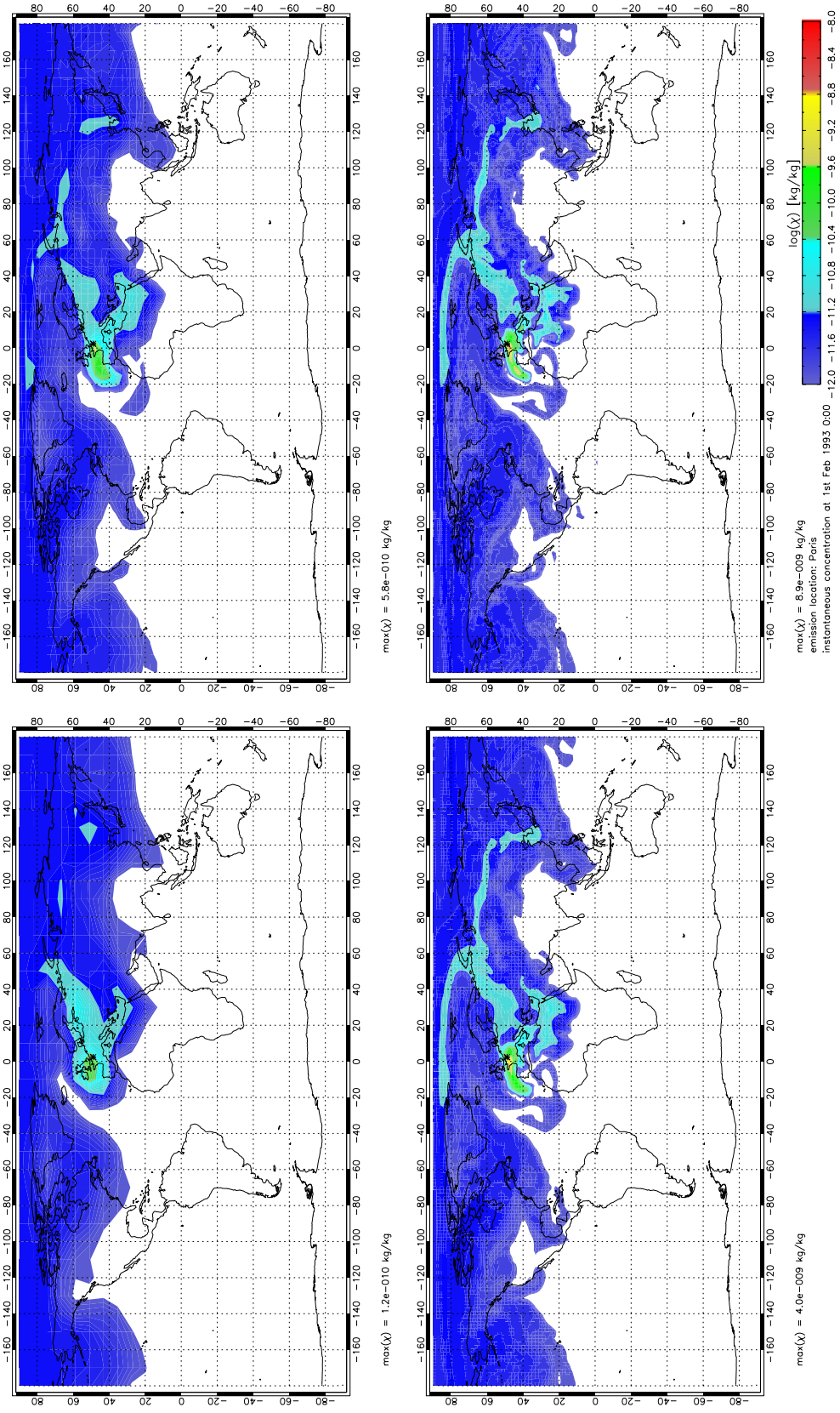

log(χ) [kg/kg]
−12.4 −12.0 −11.6 −11.2 −10.8 −10.4 −10.0 −9.6 −9.2 −8.8 −8.4
emission location: Bombay
averaging period: Jul 1993
max(χ) = 3.2e−009 kg/kg
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33

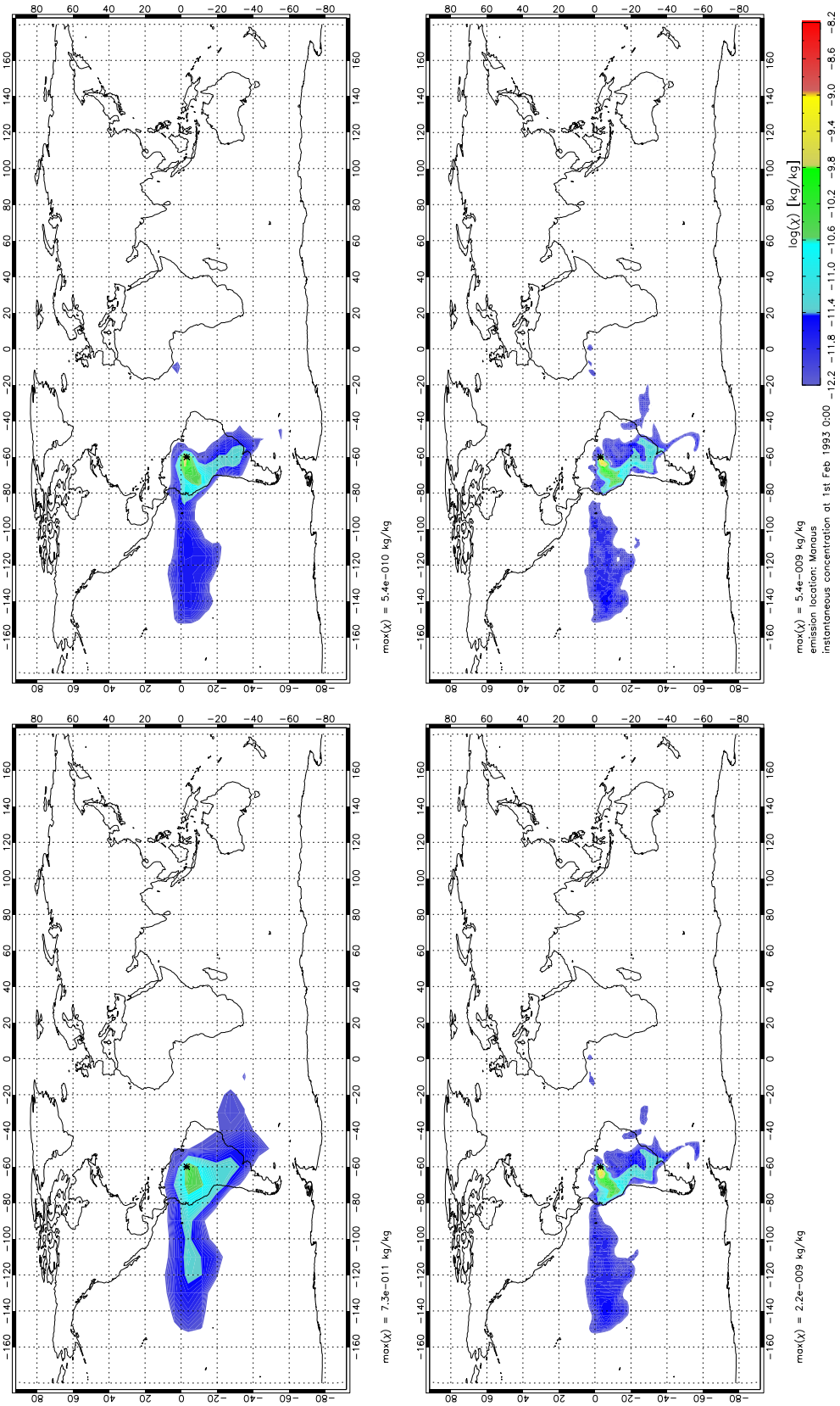

log(χ) [kg/kg]
−12.1 −11.7 −11.3 −10.9 −10.5 −10.1 −9.7 −9.3 −8.9 −8.5 −8.1
max(χ) = 7.7e−009 kg/kg
emission location: Los Angeles
averaging period: Jul 1993
max(χ) = 4.5e−010 kg/kg
max(χ) = 6.2e−011 kg/kg
max(χ) = 2.4e−009 kg/kg

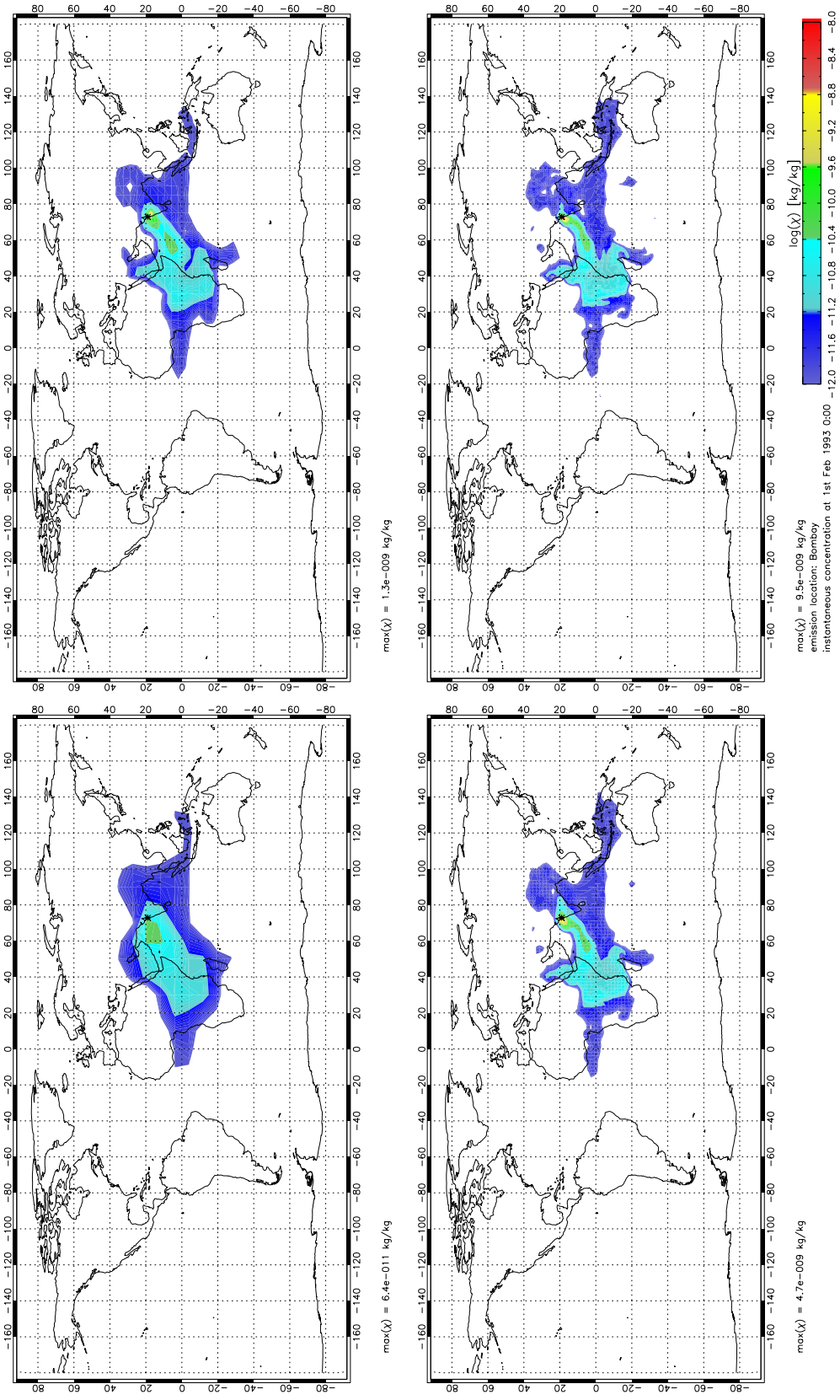

log(χ) [kg/kg]
−12.1 −11.7 −11.3 −10.9 −10.5 −10.1 −9.7 −9.3 −8.9 −8.5 −8.1
emission location: Los Angeles
averaging period: Jul 1993
max(χ) = 7.7e−009 kg/kg
rel. deviation
−1.00 −0.67 −0.33 0.00 0.33 0.67 1.00 1.33 1.67 2.00 2.33

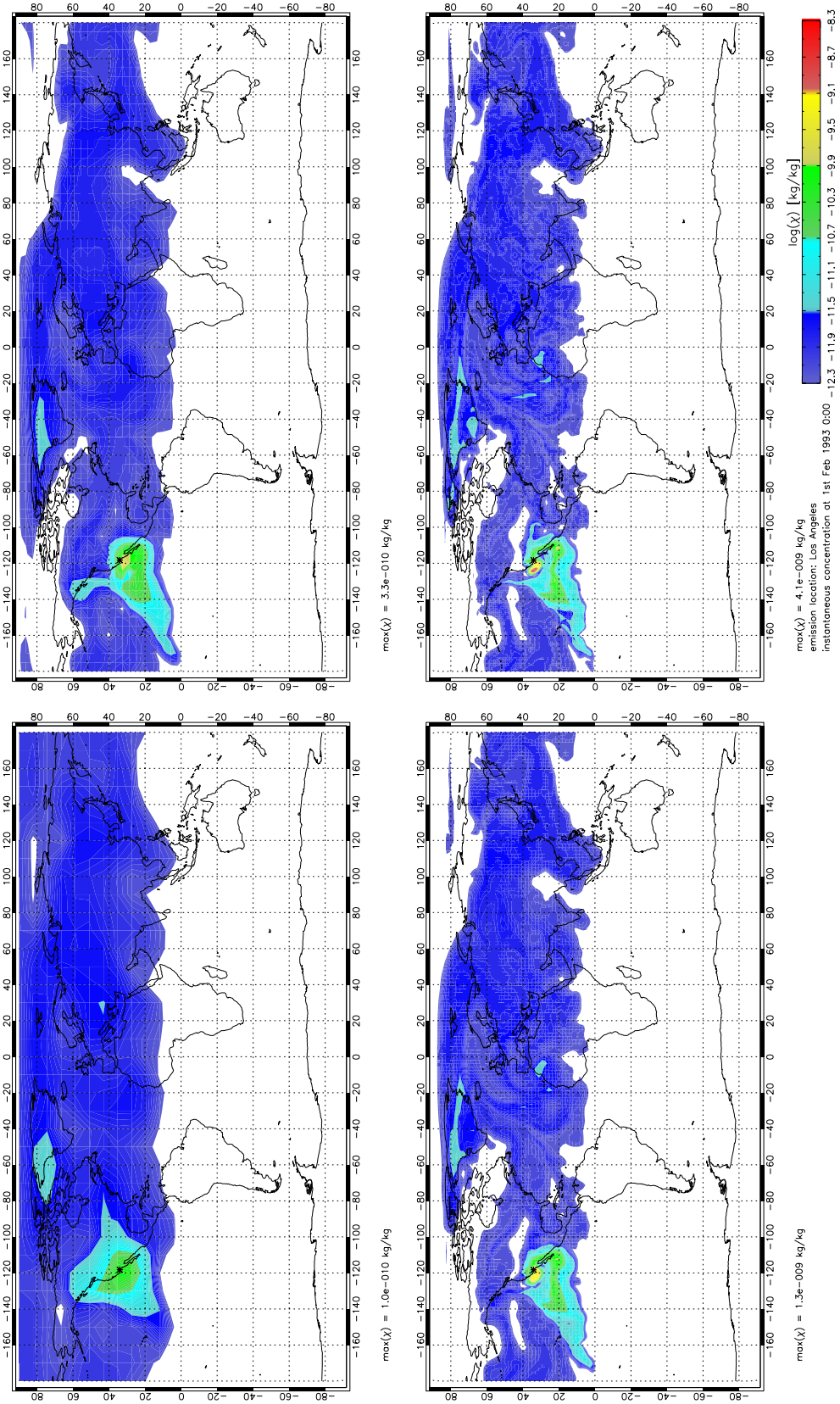

log(χ) [kg/kg]
−12.0 −11.6 −11.2 −10.8 −10.4 −10.0 −9.6 −9.2 −8.8 −8.4 −8.0
max(χ) = 9.7e−009 kg/kg
emission location: Zotino
averaging period: Jul 1993
max(χ) = 1.1e−009 kg/kg
max(χ) = 7.9e−011 kg/kg
max(χ) = 4.8e−009 kg/kg

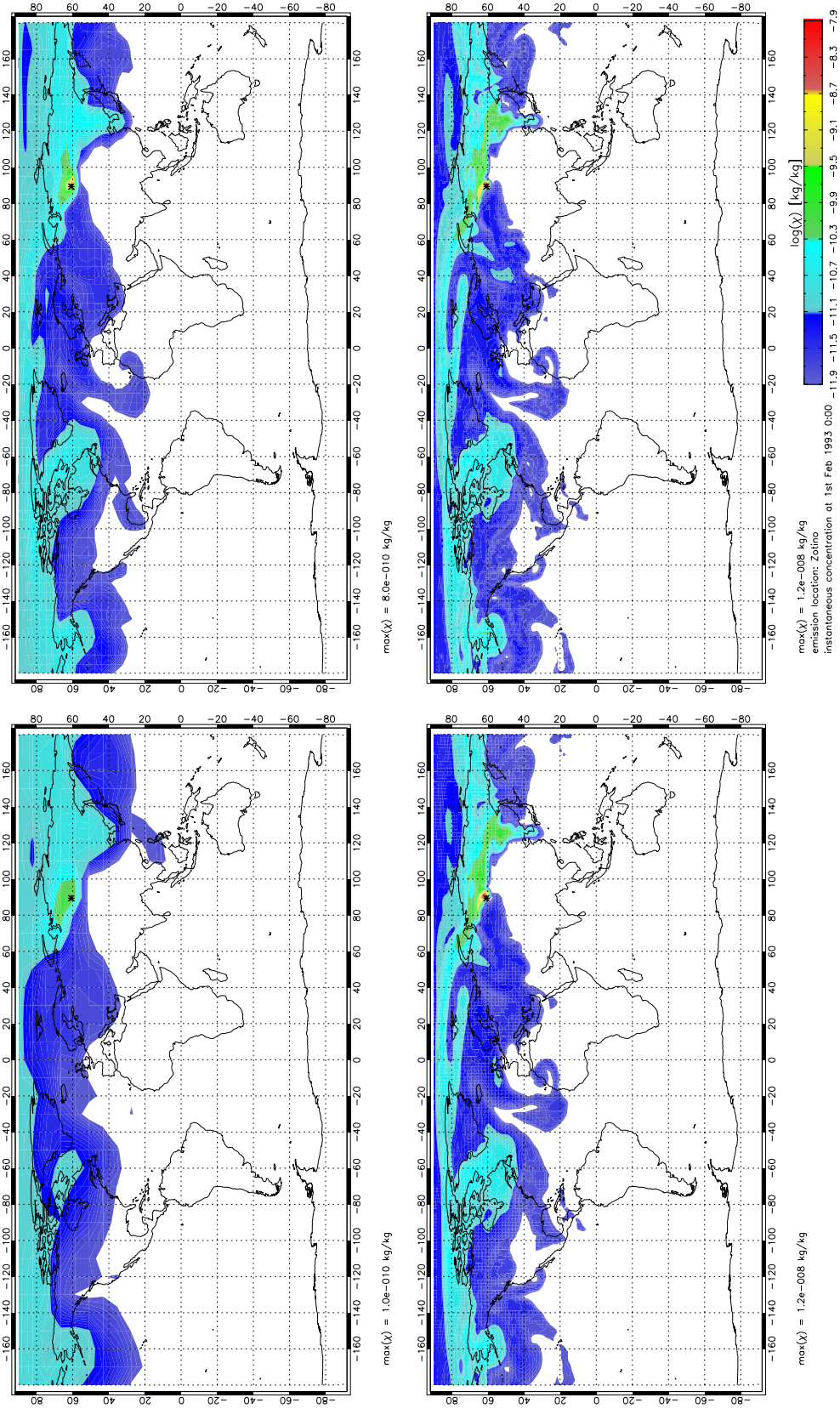

log(χ) [kg/kg]
−8.0
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
emission location: Zotino
averaging period: Jul 1993
max(χ) = 9.7e−009 kg/kg
rel. deviation
2.33
2.00
1.67
1.33
1.00
0.67
0.33
0.00
−0.33
−0.67
−1.00

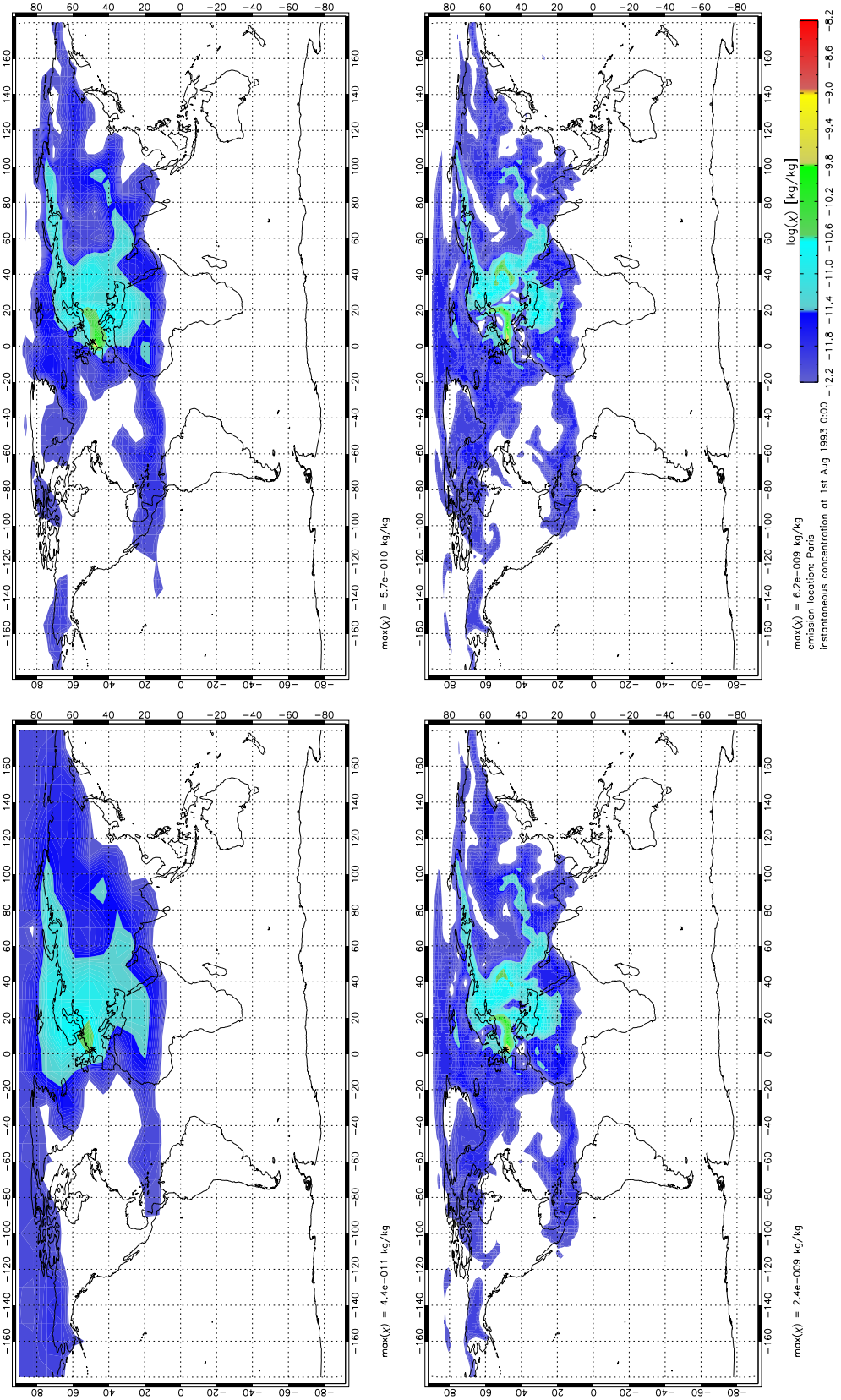

log(χ) [kg/kg]
−8.0
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
max(χ) = 8.9e−009 kg/kg
emission location: Paris
instantaneous concentration at 1st Feb 1993 0:00
max(χ) = 5.8e−010 kg/kg
max(χ) = 1.2e−010 kg/kg
max(χ) = 4.0e−009 kg/kg

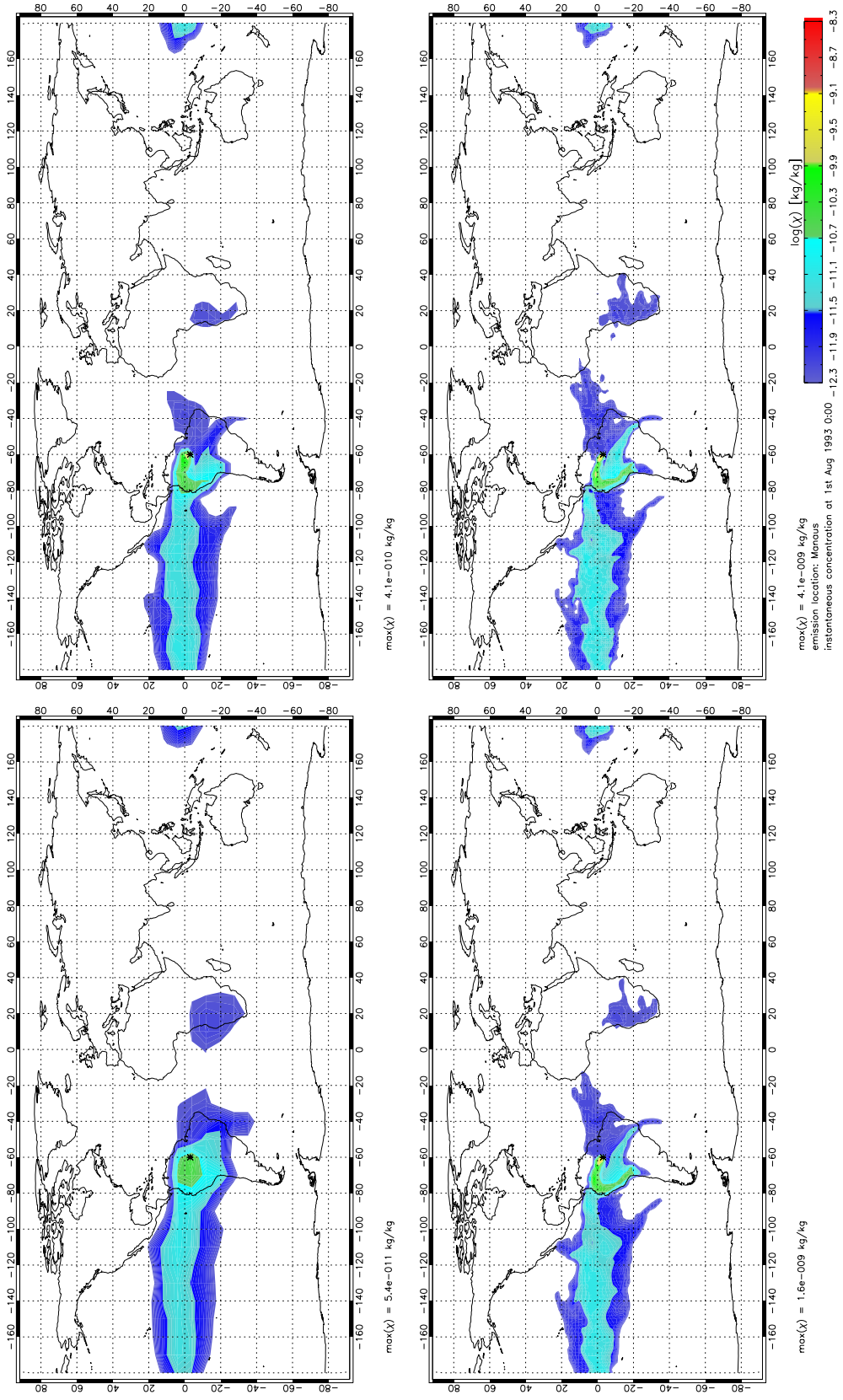

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
max(χ) = 5.4e−009 kg/kg
emission location: Manaus
instantaneous concentration at 1st Feb 1993 0:00
max(χ) = 5.4e−010 kg/kg
max(χ) = 7.3e−011 kg/kg
max(χ) = 2.2e−009 kg/kg

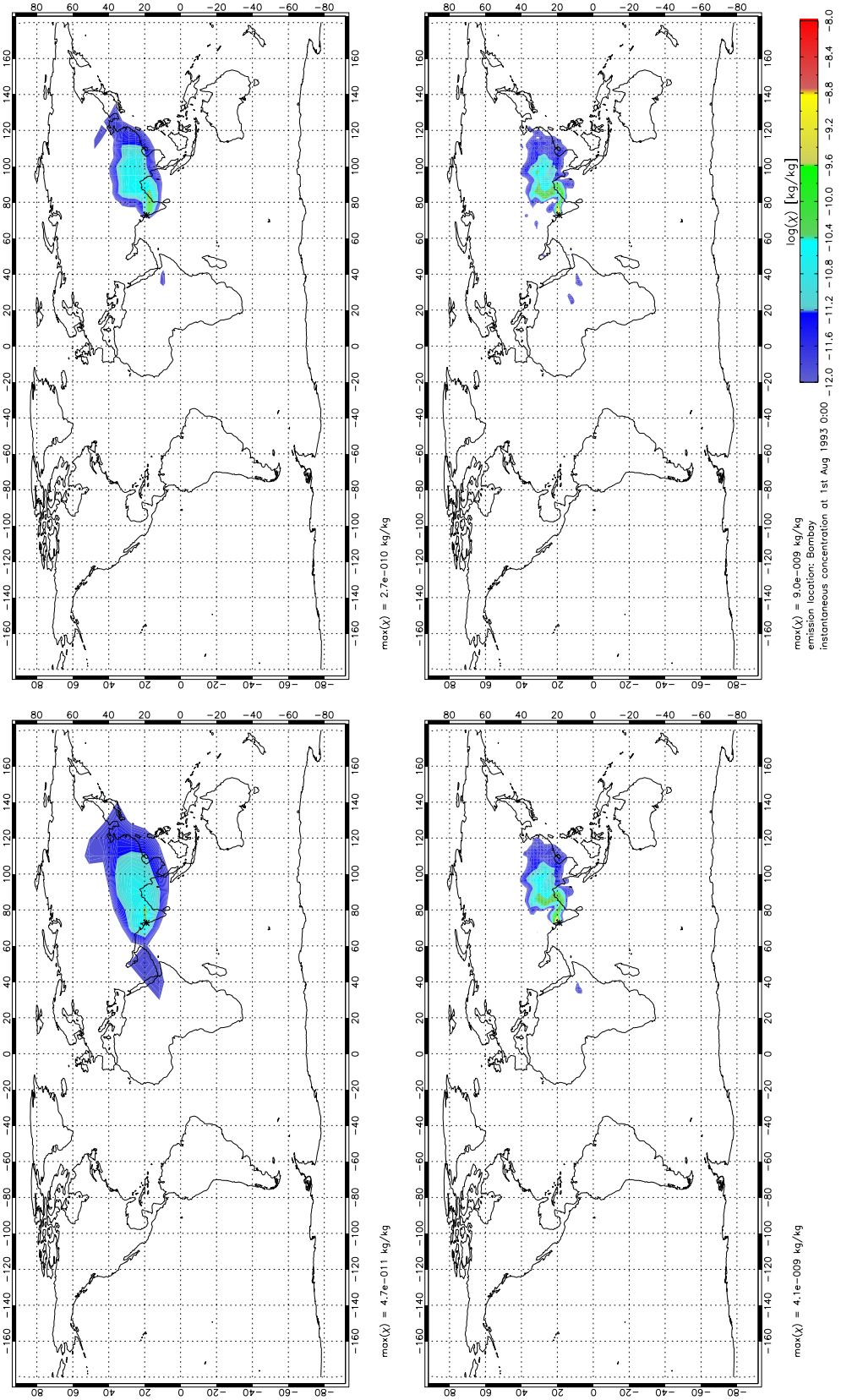

log(χ) [kg/kg]
−8.0
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
max(χ) = 9.5e−009 kg/kg
emission location: Bombay
instantaneous concentration at 1st Feb 1993 0:00
max(χ) = 1.3e−009 kg/kg
max(χ) = 6.4e−011 kg/kg
max(χ) = 4.7e−009 kg/kg

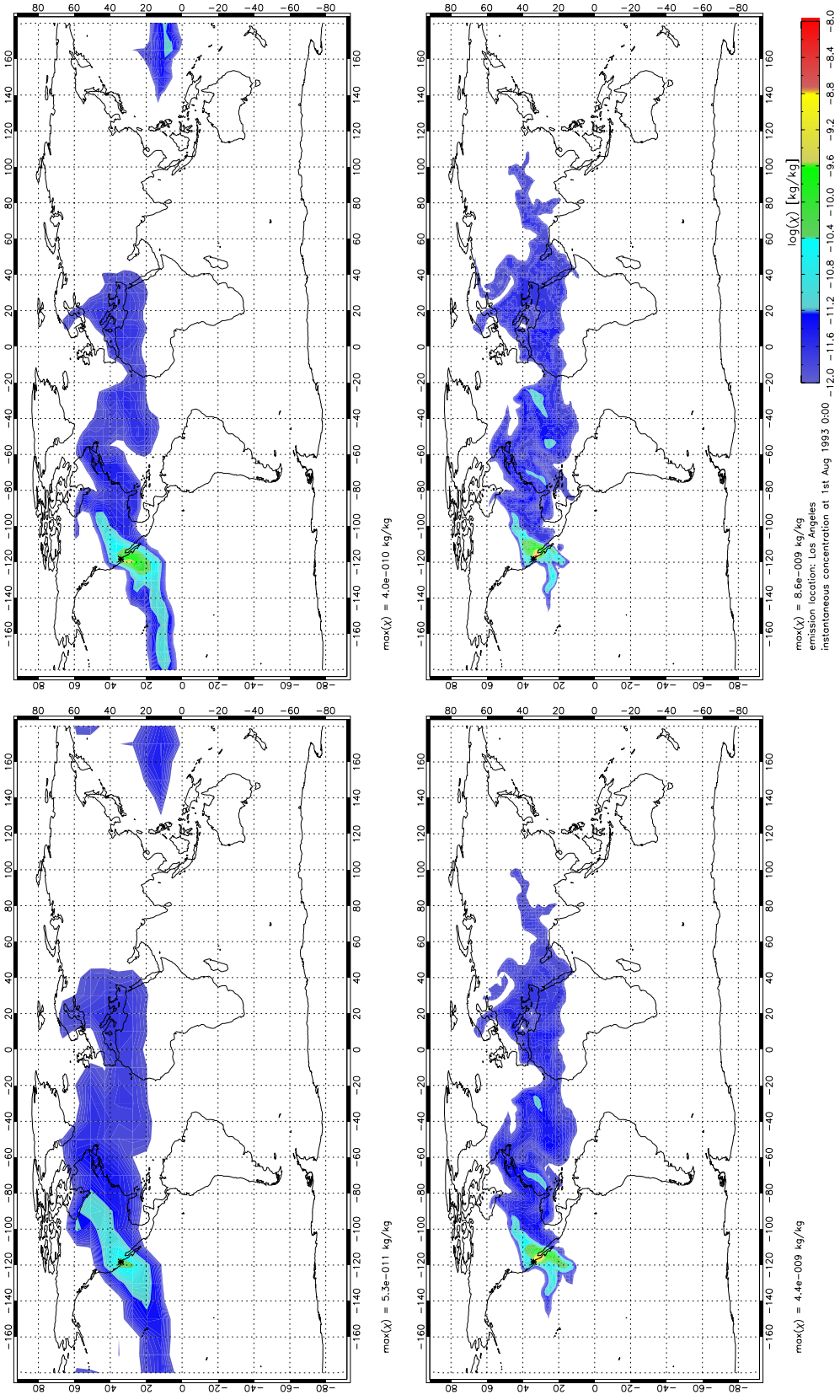

log(χ) [kg/kg]
−8.3
−8.7
−9.1
−9.5
−9.9
−10.3
−10.7
−11.1
−11.5
−11.9
−12.3
max(χ) = 4.1e−009 kg/kg
emission location: Los Angeles
instantaneous concentration at 1st Feb 1993 0:00
max(χ) = 3.3e−010 kg/kg
max(χ) = 1.0e−010 kg/kg
max(χ) = 1.3e−009 kg/kg

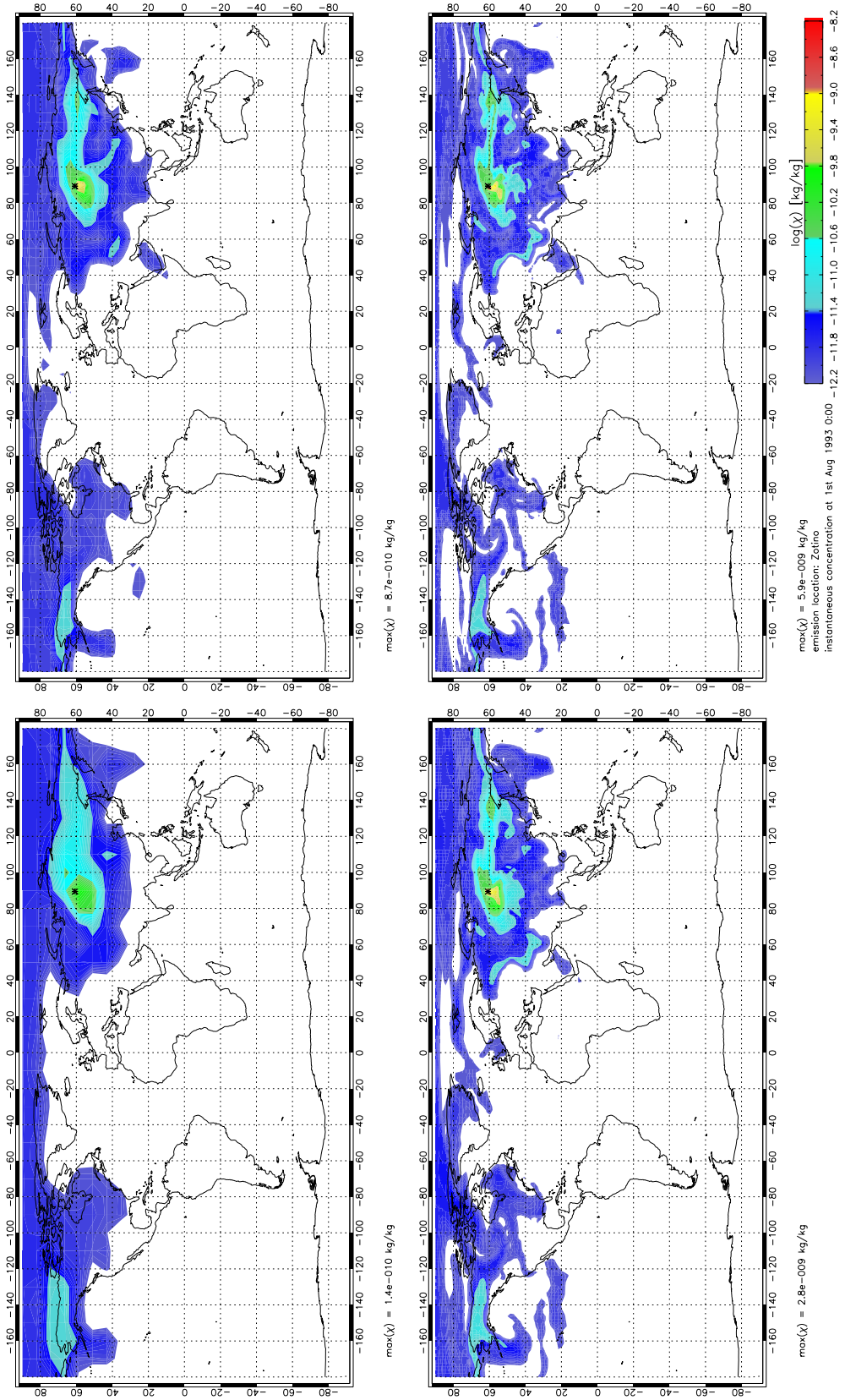

log(χ) [kg/kg]
−7.9
−8.3
−8.7
−9.1
−9.5
−9.9
−10.3
−10.7
−11.1
−11.5
−11.9
max(χ) = 1.2e−008 kg/kg
emission location: Zotino
instantaneous concentration at 1st Feb 1993 0:00
max(χ) = 8.0e−010 kg/kg
max(χ) = 1.2e−008 kg/kg
max(χ) = 1.0e−010 kg/kg

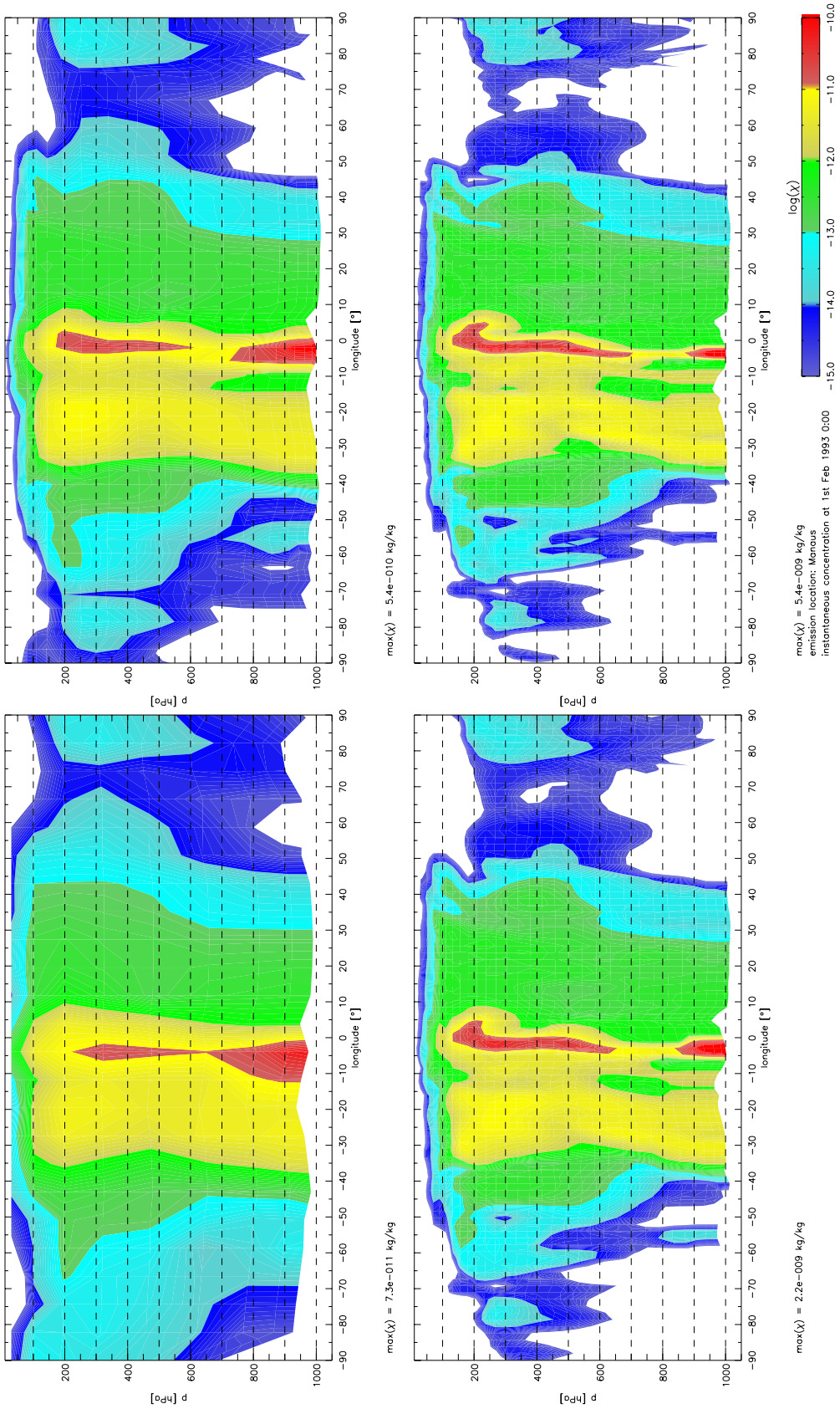

log(χ) [kg/kg]
−8.2
−8.6
−9.0
−9.4
−9.8
−10.2
−10.6
−11.0
−11.4
−11.8
−12.2
max(χ) = 6.2e−009 kg/kg
emission location: Paris
instantaneous concentration at 1st Aug 1993 0:00
max(χ) = 5.7e−010 kg/kg
max(χ) = 4.4e−011 kg/kg
max(χ) = 2.4e−009 kg/kg

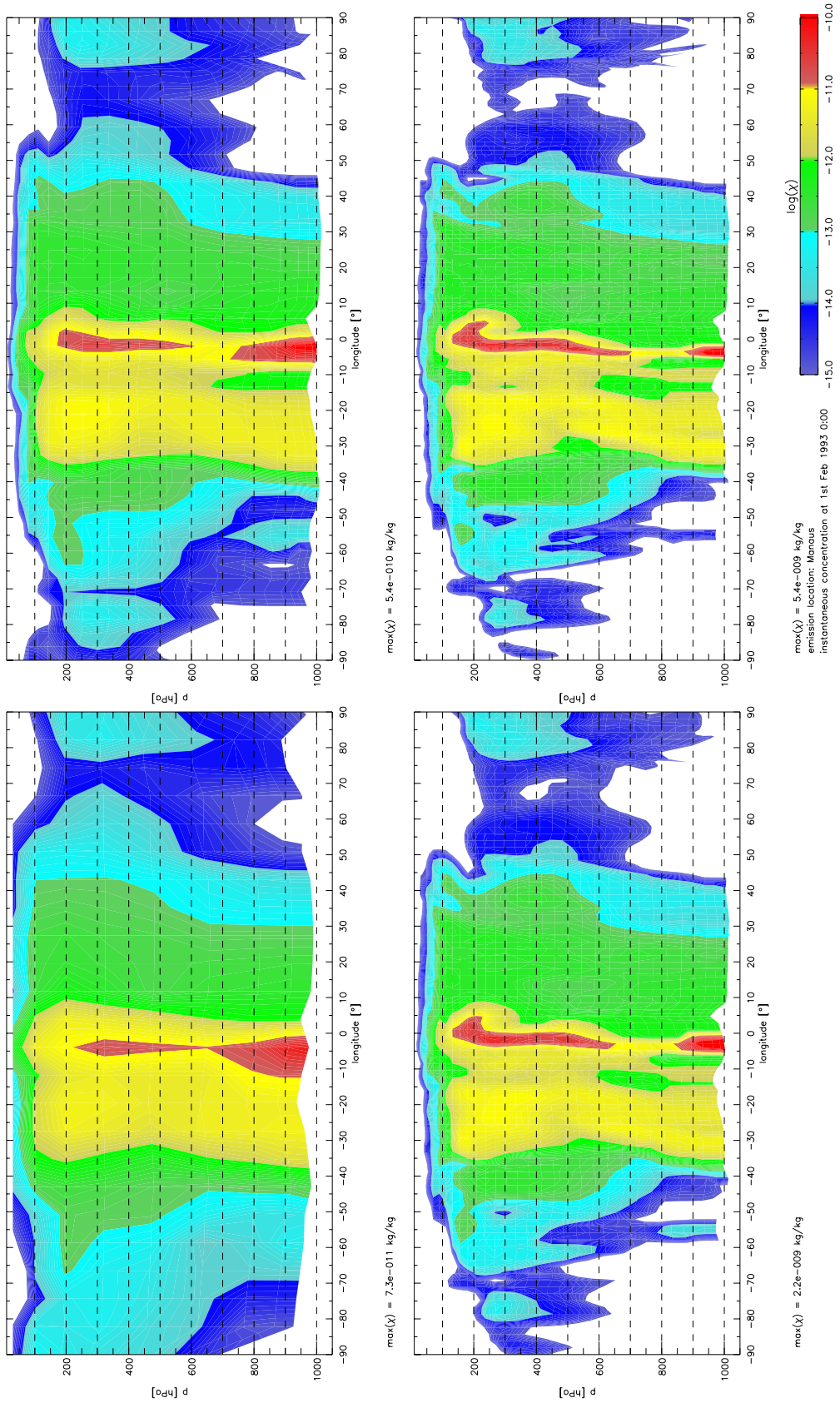

log(χ) [kg/kg]
−8.3
−8.7
−9.1
−9.5
−9.9
−10.3
−10.7
−11.1
−11.5
−11.9
−12.3
max(χ) = 4.1e−009 kg/kg
emission location: Manaus
instantaneous concentration at 1st Aug 1993 0:00
max(χ) = 4.1e−010 kg/kg
max(χ) = 5.4e−011 kg/kg
max(χ) = 1.6e−009 kg/kg


log(χ) [kg/kg]
max(χ) = 2.7e−010 kg/kg
max(χ) = 9.0e−009 kg/kg
emission location: Bombay
instantaneous concentration at 1st Aug 1993 0:00
max(χ) = 4.7e−011 kg/kg
max(χ) = 4.1e−009 kg/kg


log(χ) [kg/kg]
−8.0
−8.4
−8.8
−9.2
−9.6
−10.0
−10.4
−10.8
−11.2
−11.6
−12.0
max(χ) = 8.6e−009 kg/kg
emission location: Los Angeles
instantaneous concentration at 1st Aug 1993 0:00
max(χ) = 4.0e−010 kg/kg
max(χ) = 5.3e−011 kg/kg
max(χ) = 4.4e−009 kg/kg


log(χ) [kg/kg]
-8.2
-8.6
-9.0
-9.4
-9.8
-10.2
-10.6
-11.0
-11.4
-11.8
-12.2
max(χ) = 5.9e-009 kg/kg
emission location: Zolino
instantaneous concentration at 1st Aug 1993 0:00
max(χ) = 8.7e-010 kg/kg
max(χ) = 1.4e-010 kg/kg
max(χ) = 2.8e-009 kg/kg


p [hPa]
longitude [°]
max(χ) = 5.4e−010 kg/kg
max(χ) = 7.3e−011 kg/kg
max(χ) = 2.2e−009 kg/kg
max(χ) = 5.4e−009 kg/kg
emission location: Manaus
instantaneous concentration at 1st Feb 1993 0:00
log(χ)


max(χ) = 5.4e−010 kg/kg
max(χ) = 5.4e−009 kg/kg
emission location: Manaus
instantaneous concentration at 1st Feb 1993 0:00
log(χ)
−10.0
−11.0
−12.0
−13.0
−14.0
−15.0
max(χ) = 7.3e−011 kg/kg
max(χ) = 2.2e−009 kg/kg
p [hPa]
longitude [°]

F.1.2 Radon

This simulation setup comprised a stationary emission of ^{222}Rn with 1 atom/cm^2 over land for the simulation intervals July and December 1993. One month spin-up preceded, respectively. The ice-covered areas Antarctica, Greenland, Iceland, Baffin Island, Svalbard and Novaja Semlia were assumed to have zero emission. TM3 was run in version 3.8 and fed with the ERA-15 reanalysis data of all available resolutions, i.e. cg, fg, vfg and xfg.

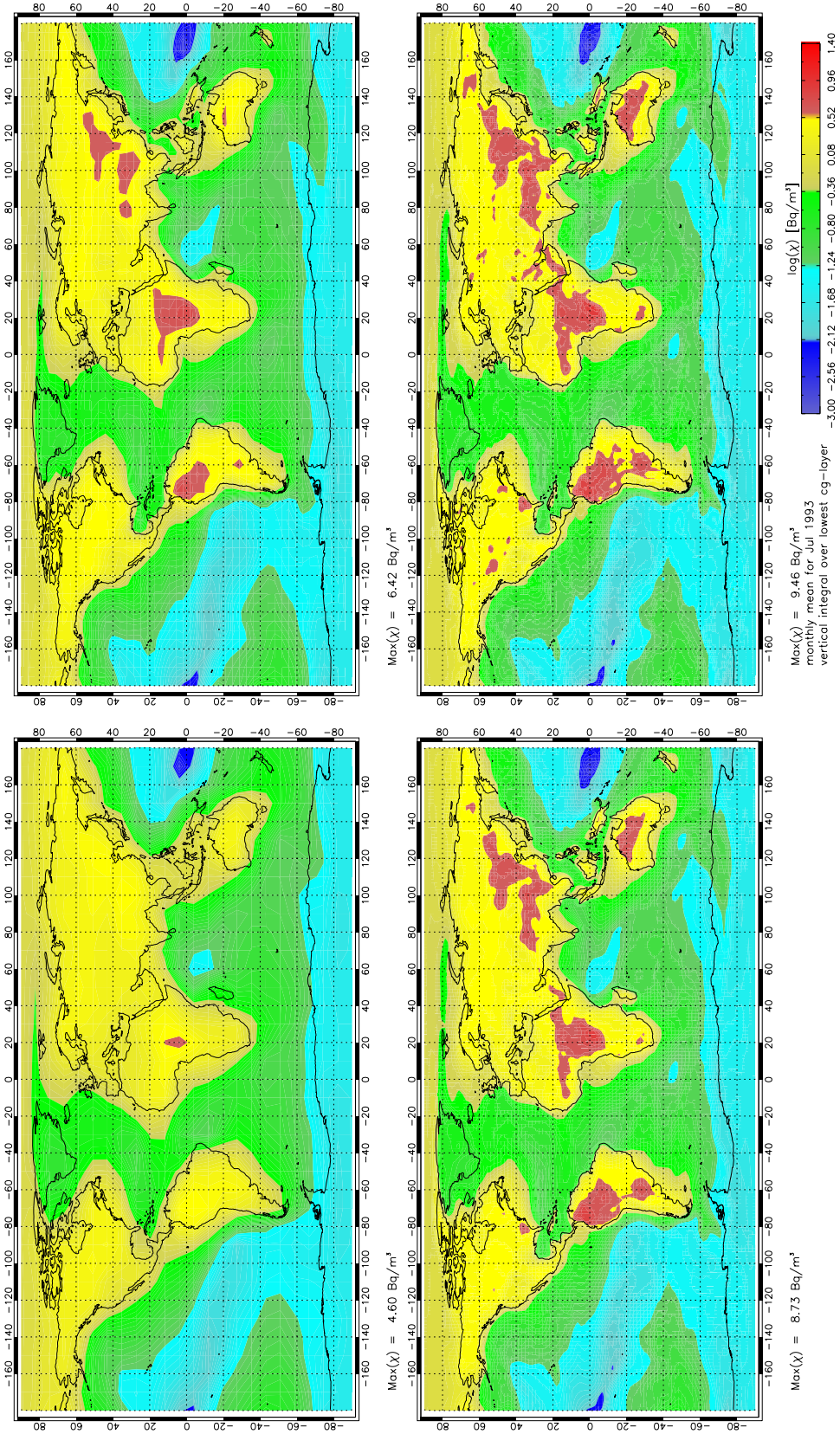
Following results are shown on the subsequent pages:

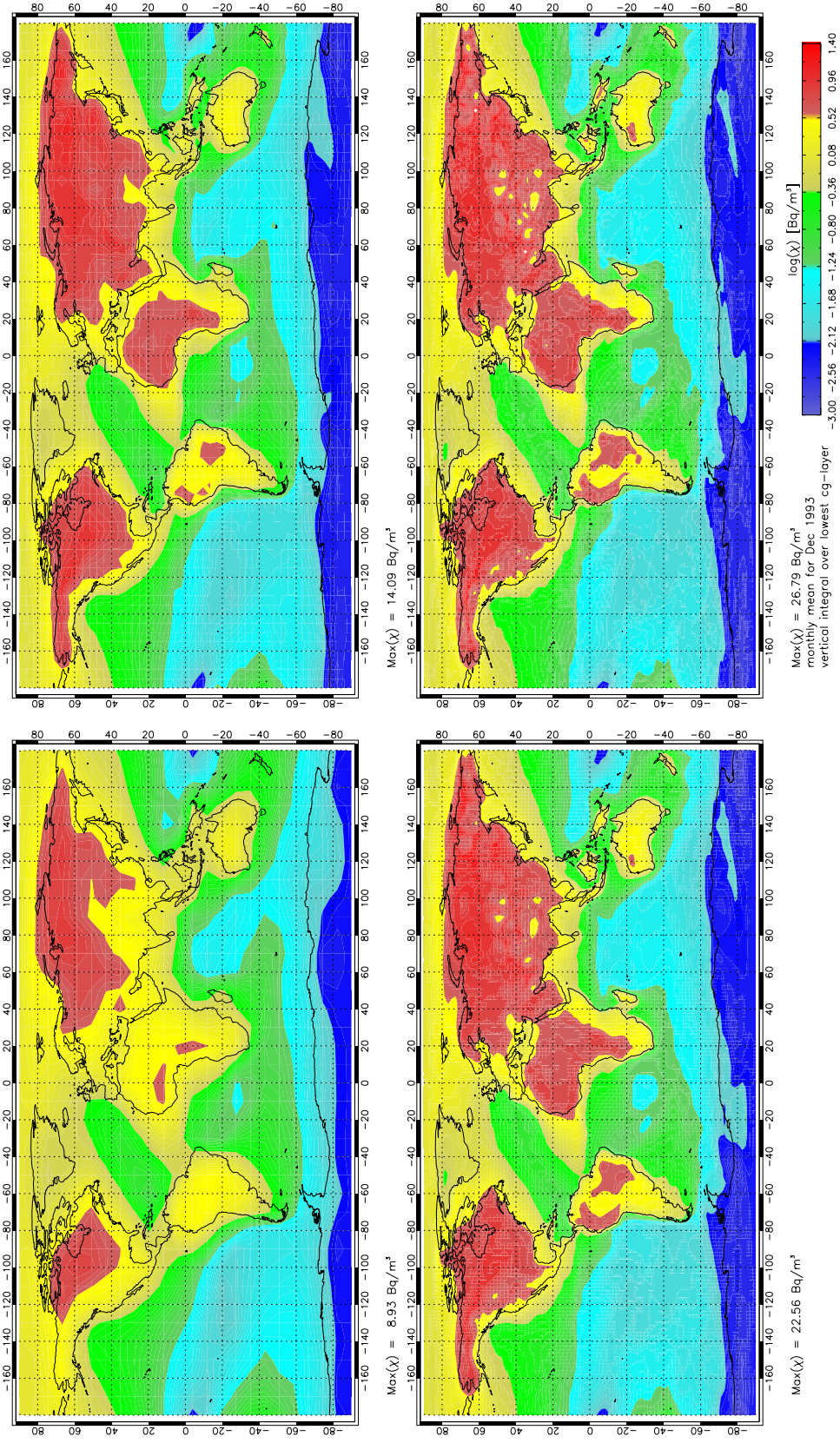
- surface maps of the monthly mean activity, vertically integrated over the thickness of the lowest cg layer for better comparison (pp 110–111),
- zonal averages of the monthly mean activities (pp 112–113),
- vertical cuts at the latitudes 50°N and 0° (pp 114–117), and
- monthly mean budgets of ^{222}Rn for eight different atmospheric pools: Eurasia, North America, North Atlantic and North Pacific, with each area subdivided in the vertical into boundary layer and free troposphere (page 118).

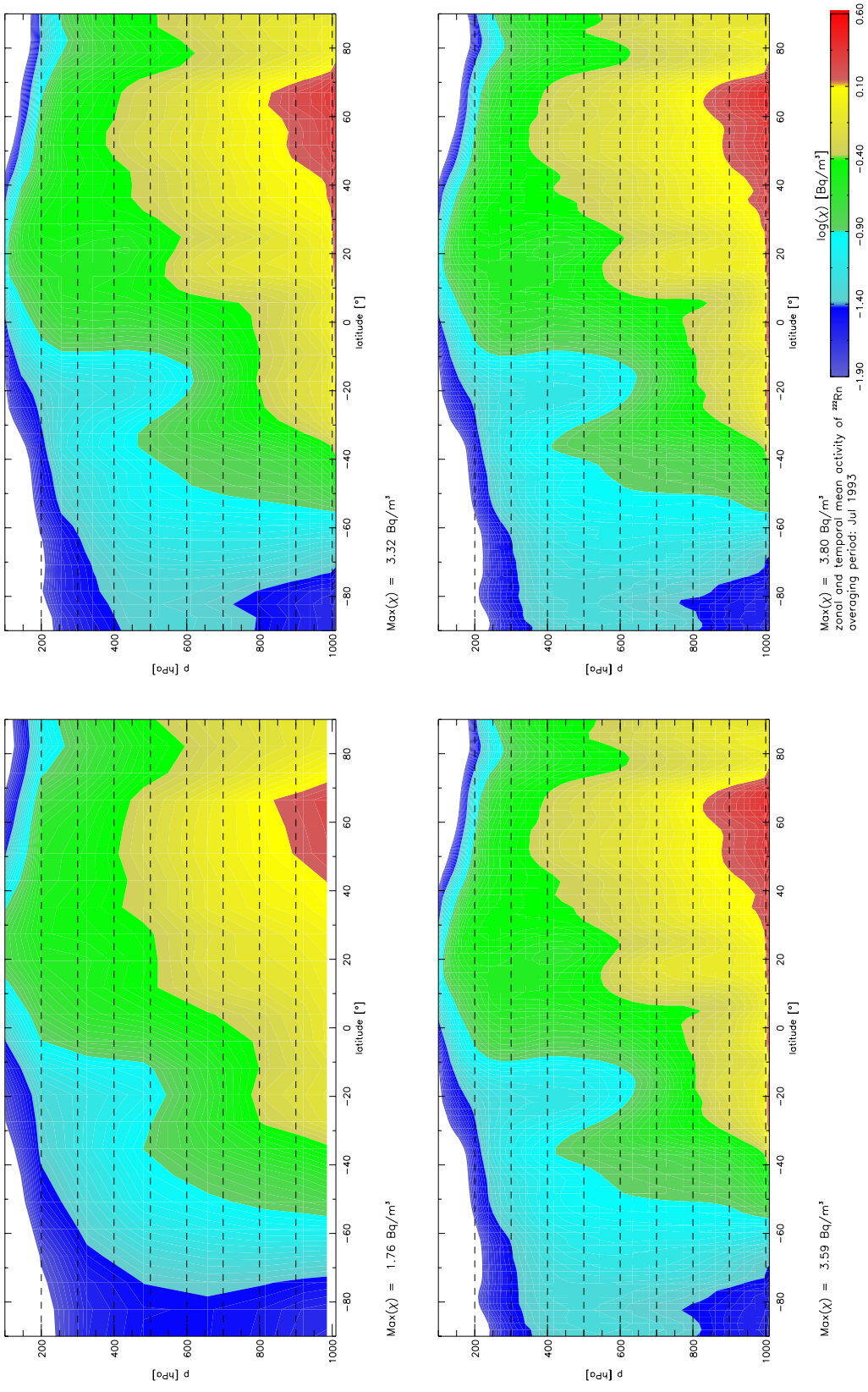
Within the panels the different resolution maps are placed as follows:

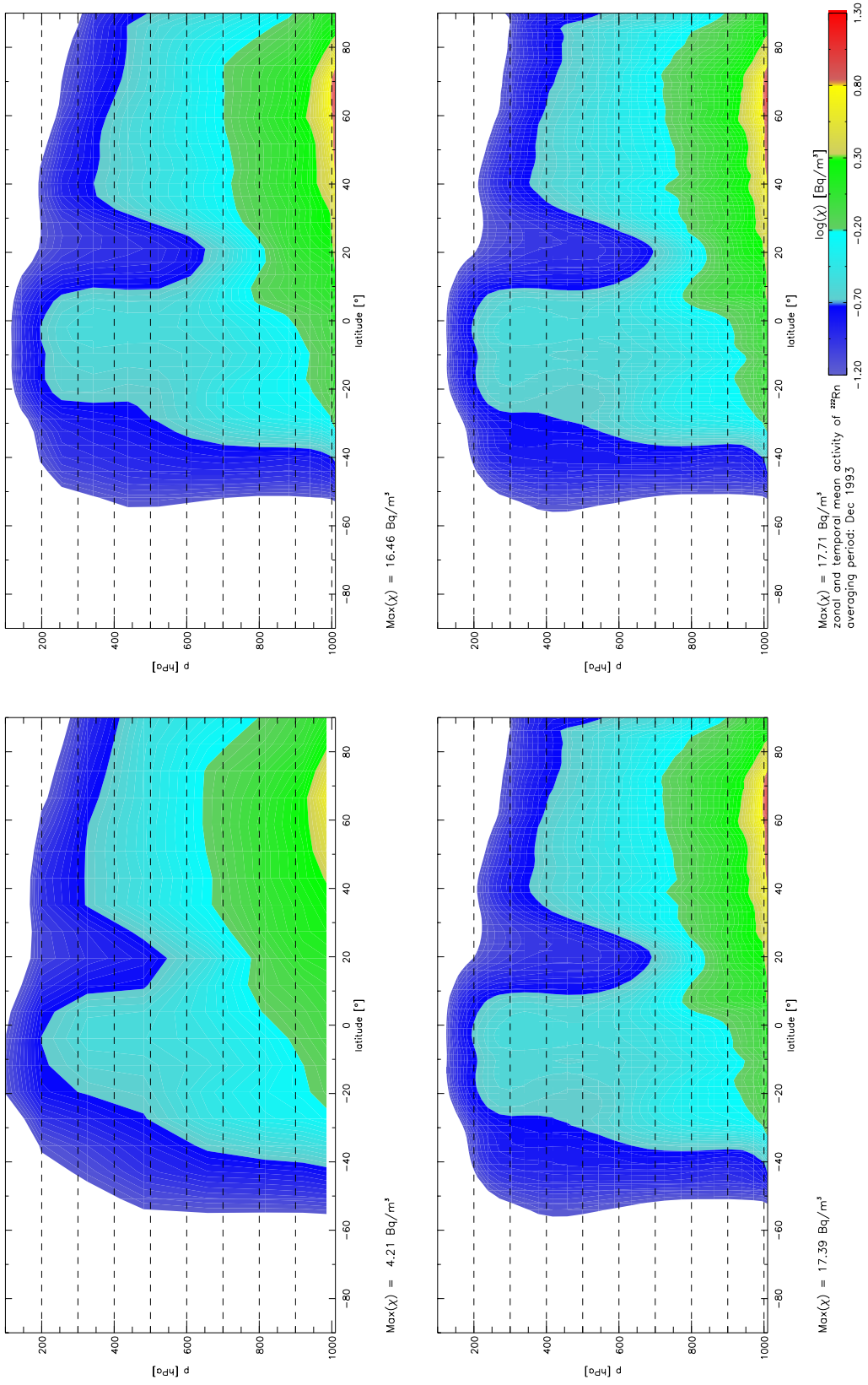
cg	fg
vfg	xfg

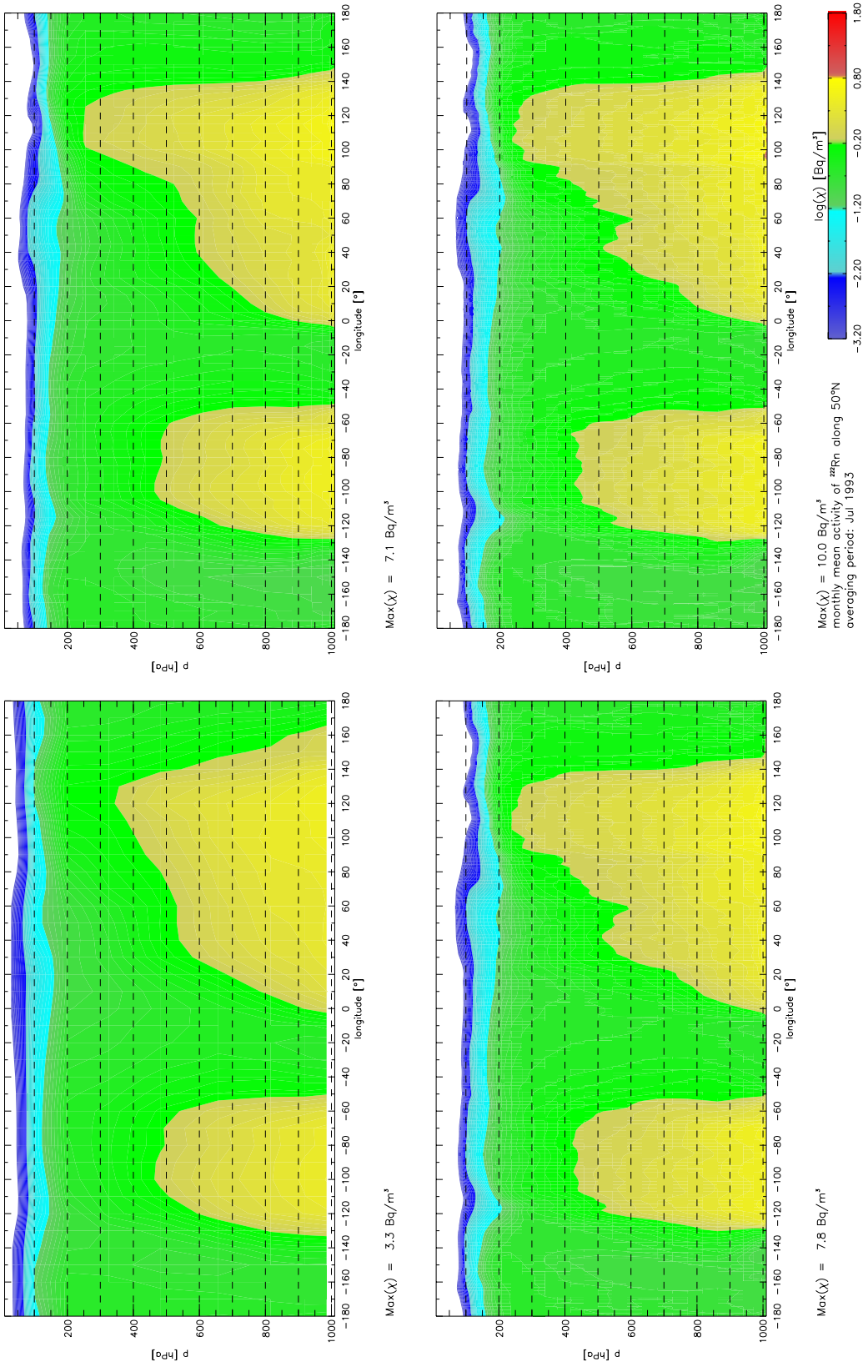
.

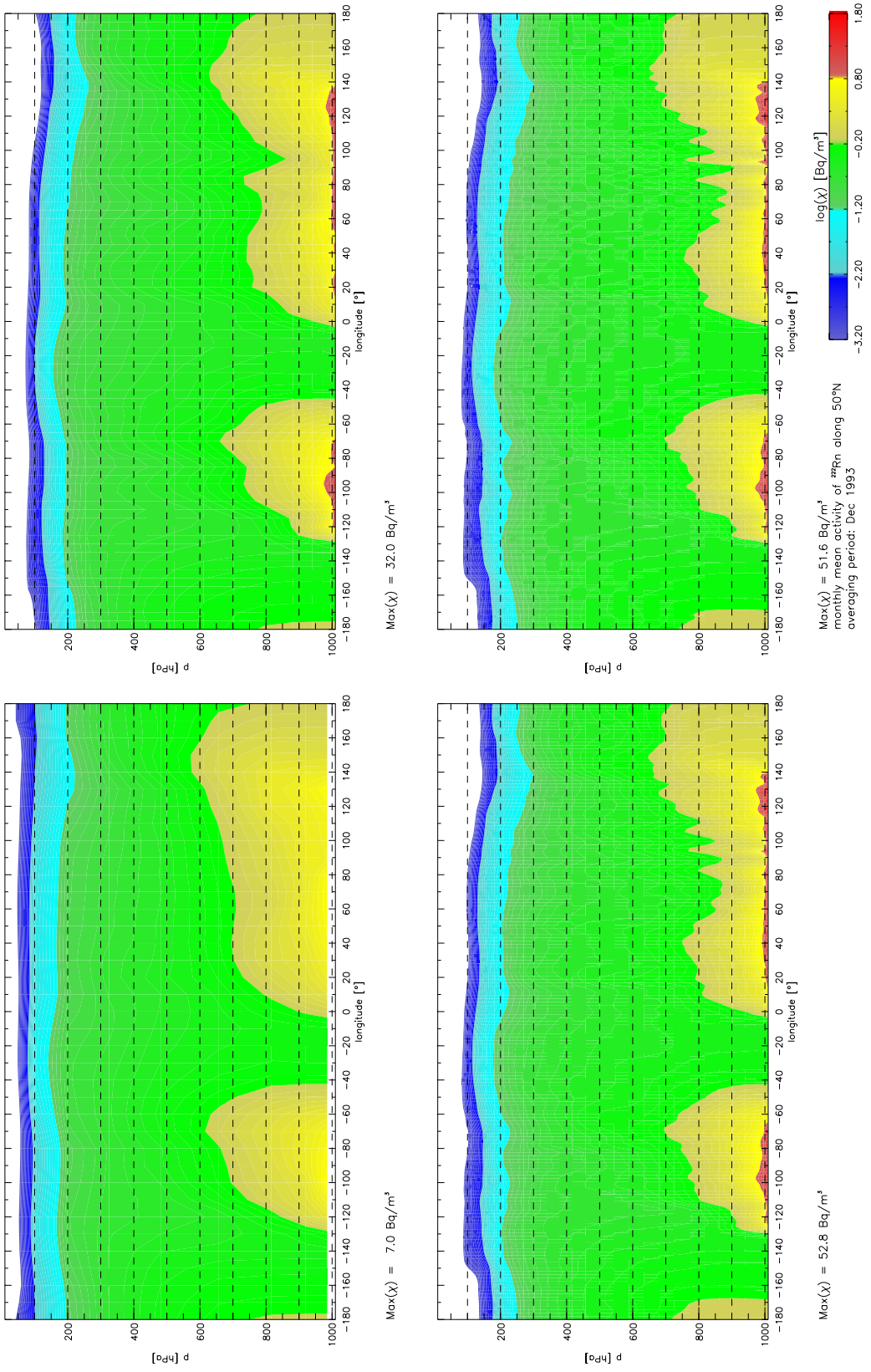

Max(χ) = 6.42 Bq/m³
Max(χ) = 9.46 Bq/m³
monthly mean for Jul 1993
vertical integral over lowest cg-layer
log(χ) [Bq/m³]
-3.00 -2.56 -2.12 -1.68 -1.24 -0.80 -0.36 0.08 0.52 0.96 1.40
Max(χ) = 4.60 Bq/m³
Max(χ) = 8.73 Bq/m³

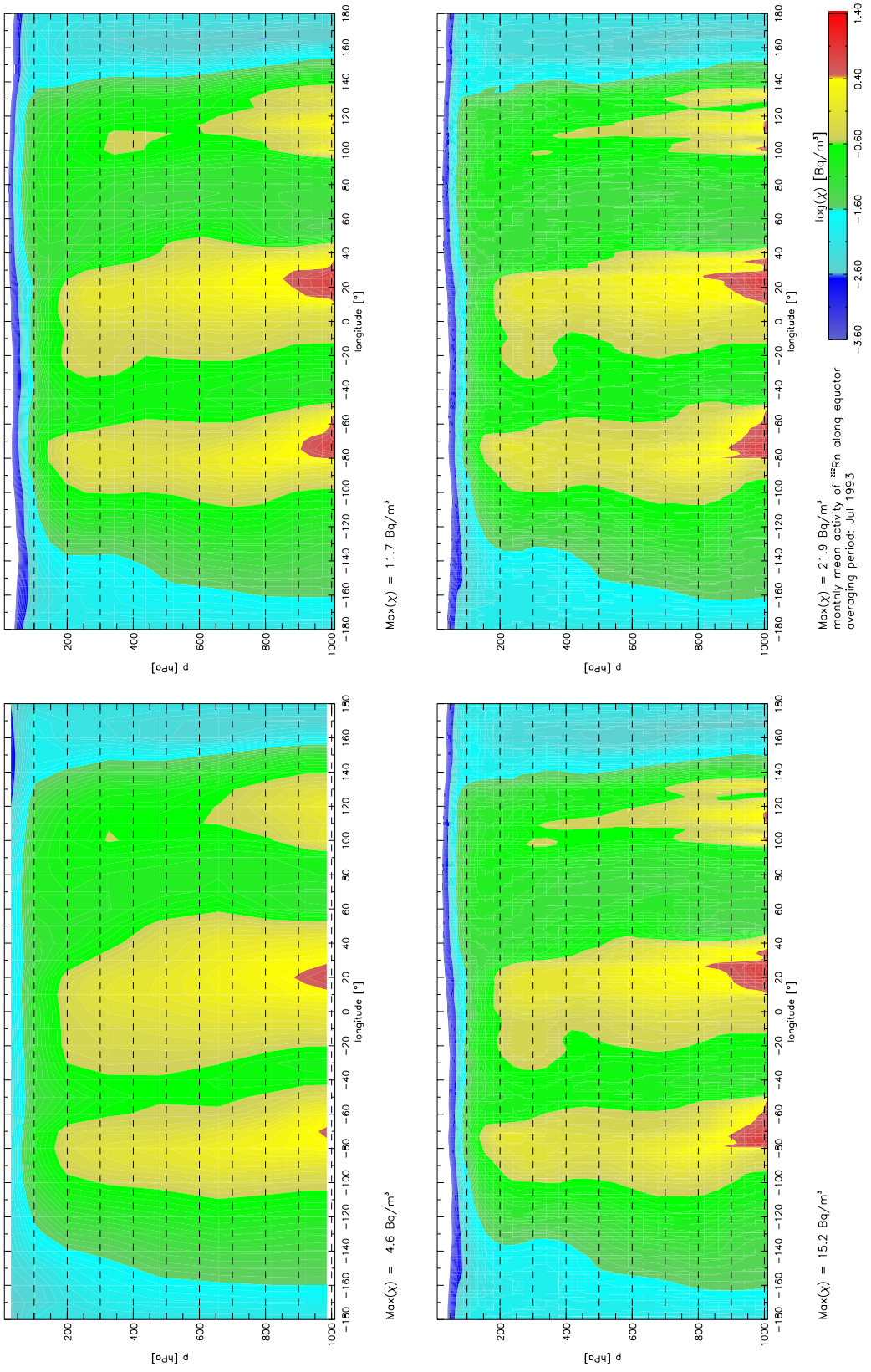

Max(χ) = 14.09 Bq/m³
Max(χ) = 26.79 Bq/m³
monthly mean for Dec 1993
vertical integral over lowest cg-layer
log(χ) [Bq/m³]
-3.00 -2.56 -2.12 -1.68 -1.24 -0.80 -0.36 0.08 0.52 0.96 1.40
Max(χ) = 8.93 Bq/m³
Max(χ) = 22.56 Bq/m³

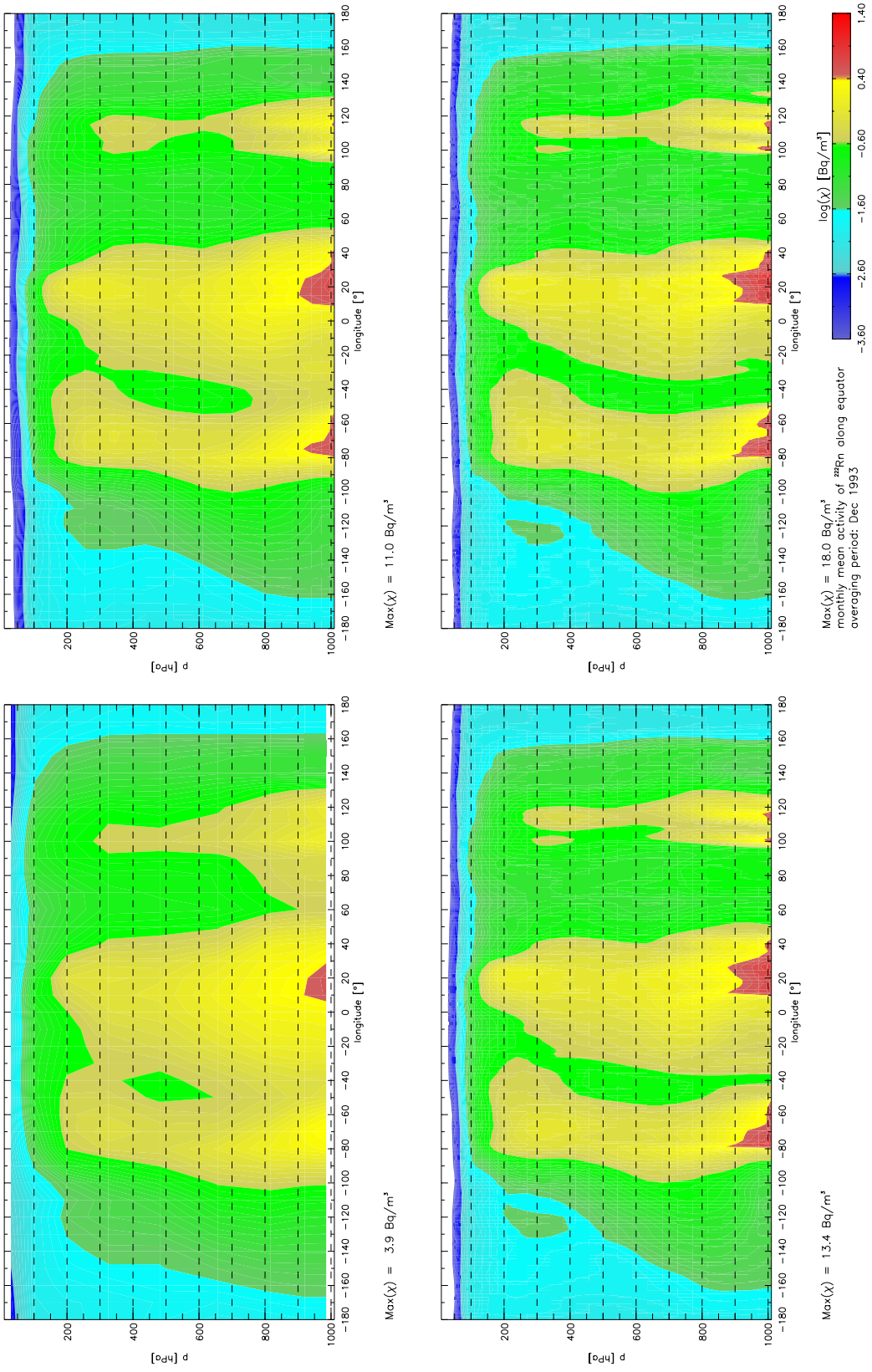

p [hPa]
latitude [°]
Max(χ) = 3.32 Bq/m³
Max(χ) = 3.80 Bq/m³
zonal and temporal mean activity of ²²²Rn
averaging period: Jul 1993
log(χ) [Bq/m³]
0.60
0.10
−0.40
−0.90
−1.40
−1.90
Max(χ) = 1.76 Bq/m³
Max(χ) = 3.59 Bq/m³


p [hPa]
latitude [°]
Max(χ) = 16.46 Bq/m³
Max(χ) = 17.71 Bq/m³
zonal and temporal mean activity of ²²²Rn
averaging period: Dec 1993
log(χ) [Bq/m³]
Max(χ) = 4.21 Bq/m³
Max(χ) = 17.39 Bq/m³


p [hPa]
longitude [°]
Max(χ) = 7.1 Bq/m³
Max(χ) = 10.0 Bq/m³
monthly mean activity of ²²²Rn along 50°N
averaging period: Jul 1993
log(χ) [Bq/m³]
Max(χ) = 3.3 Bq/m³
Max(χ) = 7.8 Bq/m³


Max(χ) = 32.0 Bq/m³
Max(χ) = 51.6 Bq/m³
monthly mean activity of ²²²Rn along 50°N
averaging period: Dec 1993
log(χ) [Bq/m³]
−3.20
−2.20
−1.20
−0.20
0.80
1.80
Max(χ) = 7.0 Bq/m³
Max(χ) = 52.8 Bq/m³
p [hPa]
longitude [°]


p [hPa]
longitude [°]
Max(χ) = 11.7 Bq/m³
Max(χ) = 21.9 Bq/m³
monthly mean activity of ²²²Rn along equator
averaging period: Jul 1993
log(χ) [Bq/m³]
Max(χ) = 4.6 Bq/m³
Max(χ) = 15.2 Bq/m³


p [hPa]
longitude [°]
Max(χ) = 11.0 Bq/m³
Max(χ) = 18.0 Bq/m³
monthly mean activity of ²²²Rn along equator
averaging period: Dec 1993
log(χ) [Bq/m³]
Max(χ) = 3.9 Bq/m³
Max(χ) = 13.4 Bq/m³

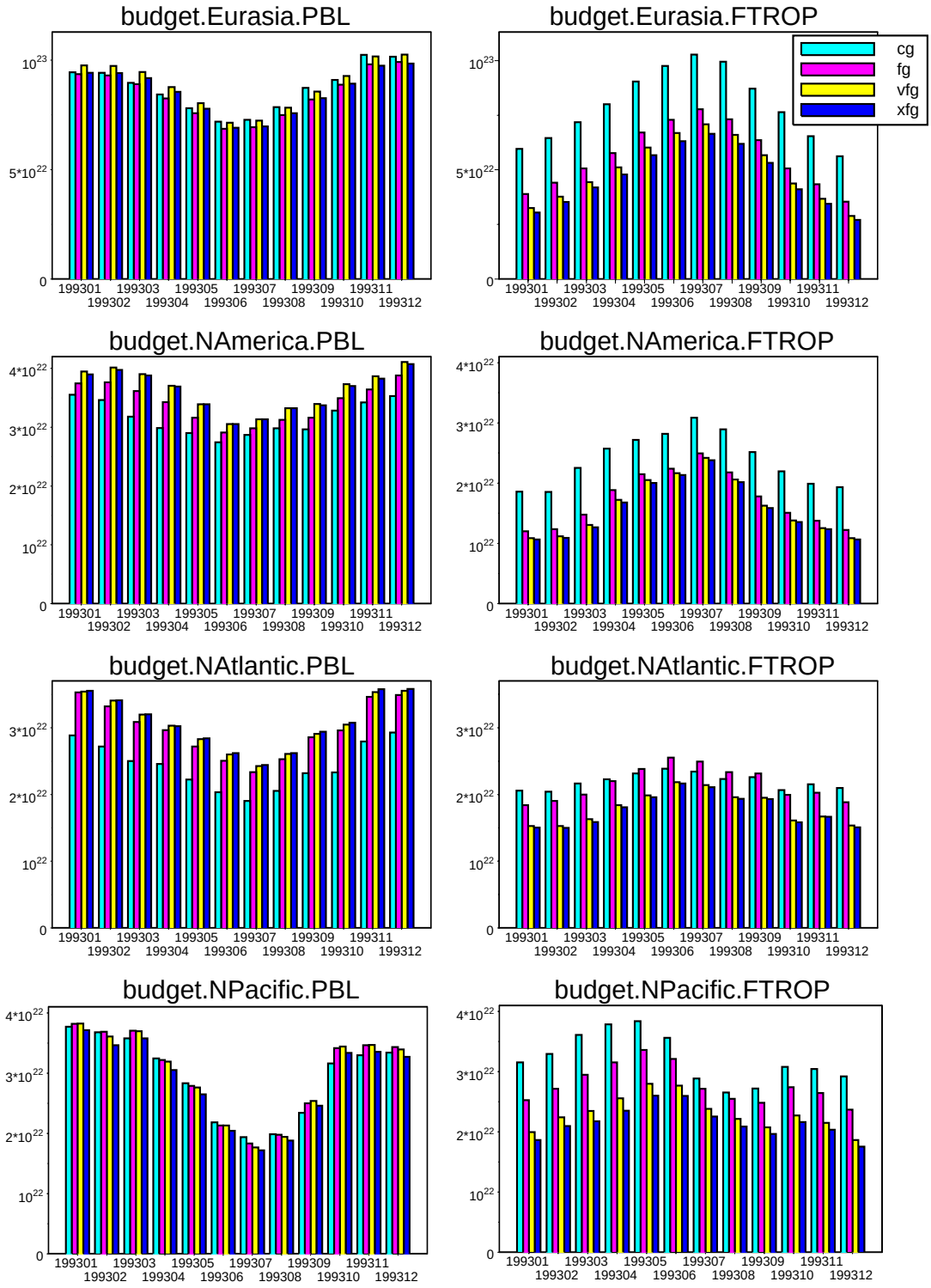


Figure F.1: Monthly mean budgets of ^{222}Rn in kg for eight different pools: Eurasia, North America, North Atlantic and North Pacific, with each area subdivided in the vertical into boundary layer and free troposphere. The boundary layer has been approximated by the collectivity of model layers with a monthly mean pressure greater than 700 hPa.

F.1.3 Sulfur Hexafluoride SF_6

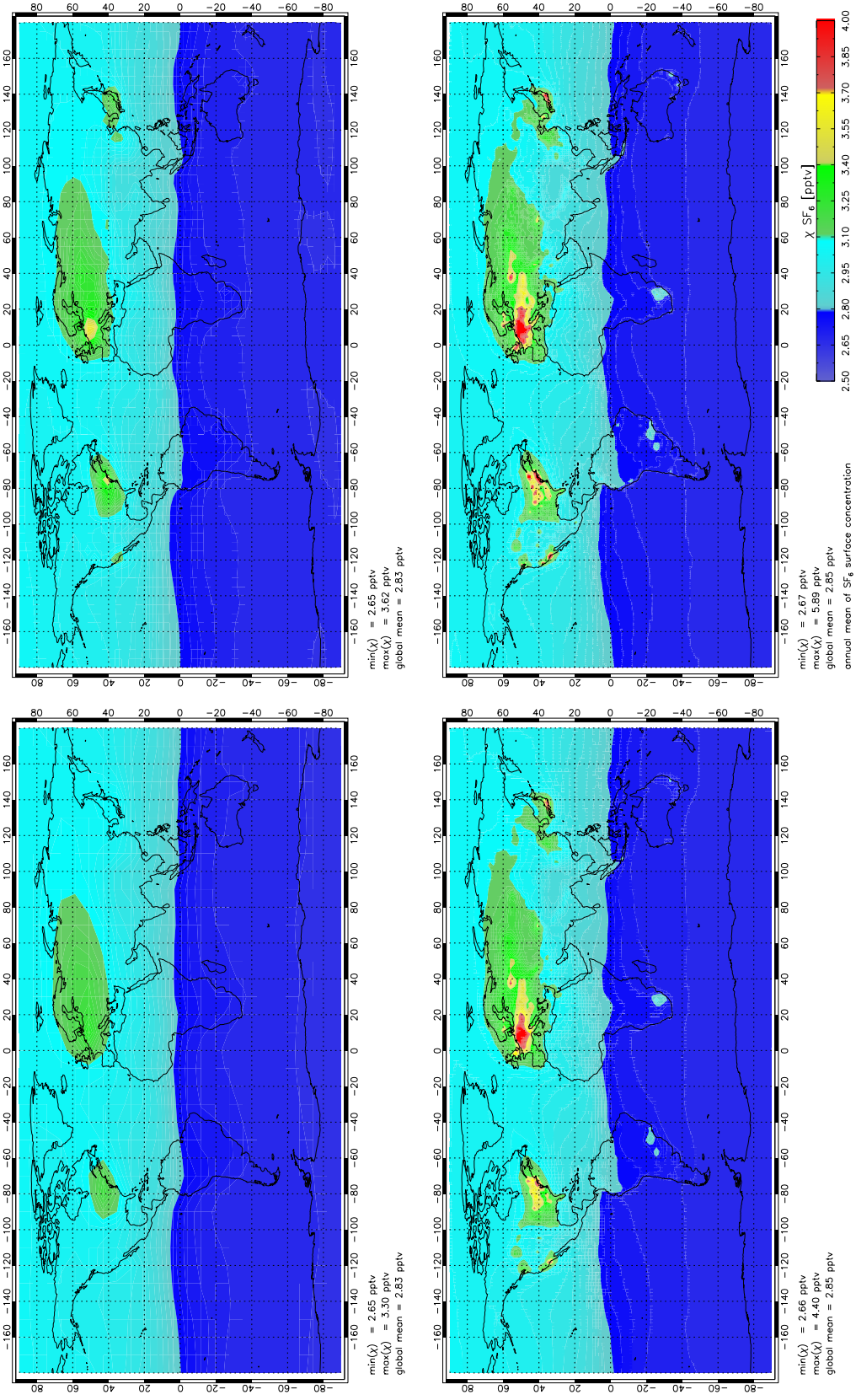
The setup for this simulation followed the TransCom 2 protocol [Denning et al., 1999], with the exception of using corrected SF_6 emissions and initial mixing ratios from start instead of applying post-adjustments to the simulation results. The year under consideration was 1993 with four years spin-up, starting from a uniform SF_6 mixing ratio of 1.93 pptv. TM3 was run in version 3.8 and fed with ERA-15 reanalysis data of all available resolutions, i.e. cg, fg, vfg and xfg.

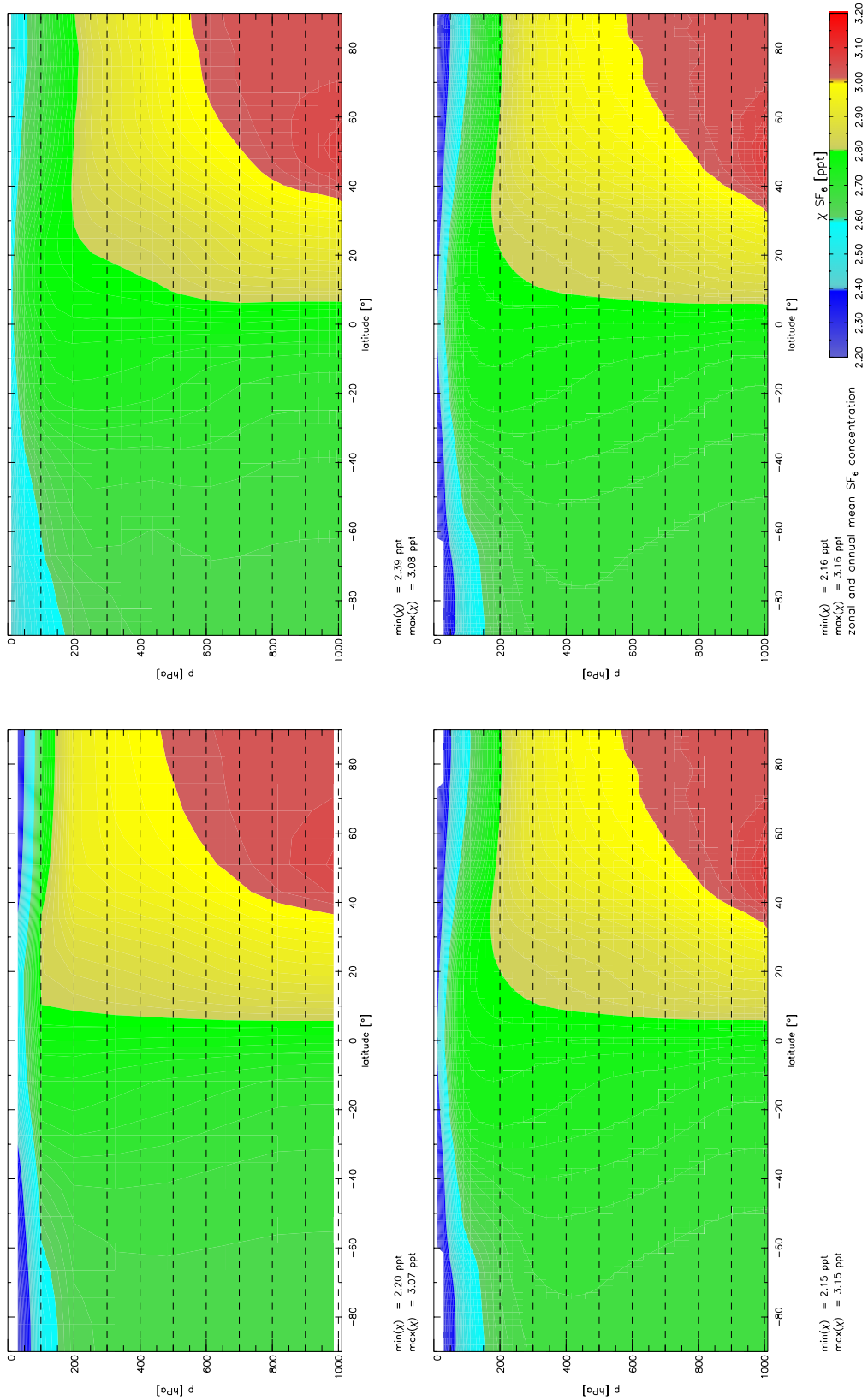
Surface Maps and Zonal Averages

The following two pages show maps of SF_6 annual mean surface mixing ratios and zonal averages of the vertical SF_6 distribution in pressure coordinates, for the four experimental resolutions, arranged in the picture panels according to resolutions like

cg	fg
vfg	xfg

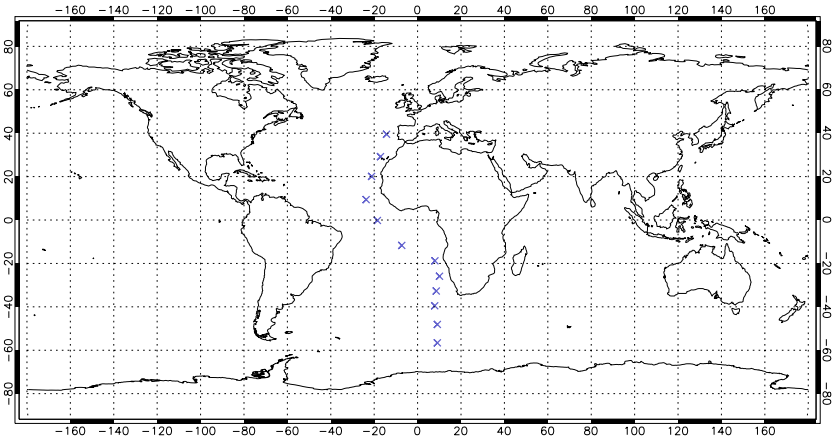
.


χ SF6 [pptv]
2.50 2.65 2.80 2.95 3.10 3.25 3.40 3.55 3.70 3.85 4.00
min(χ) = 2.65 pptv
max(χ) = 3.62 pptv
global mean = 2.83 pptv
min(χ) = 2.67 pptv
max(χ) = 5.89 pptv
global mean = 2.85 pptv
annual mean of SF6 surface concentration
min(χ) = 2.65 pptv
max(χ) = 3.30 pptv
global mean = 2.83 pptv
min(χ) = 2.66 pptv
max(χ) = 4.40 pptv
global mean = 2.85 pptv


p [hPa]
latitude [°]
min(χ) = 2.39 ppt
max(χ) = 3.08 ppt
min(χ) = 2.16 ppt
max(χ) = 3.16 ppt
zonal and annual mean SF6 concentration
χ SF6 [ppt]
min(χ) = 2.20 ppt
max(χ) = 3.07 ppt
min(χ) = 2.15 ppt
max(χ) = 3.15 ppt

Atlantic Transect

High-volume surface air spot samples were collected during an atlantic cruise of the research ship *Polarstern* in November 1993 from 40°N to 56°S [Maiss et al., 1996]. Their locations are shown in the following picture:



These measurements are compared in figure F.2 with November mean values, computed by TM3 at the same coordinates. They show remarkably little resolution dependency. Offsets of $\sim$ -1.1 ppt, estimated by weighted least-square fits to the observations, have been subtracted from the model values. The vertical bars show the standard deviations of the respective TM3 November time series.

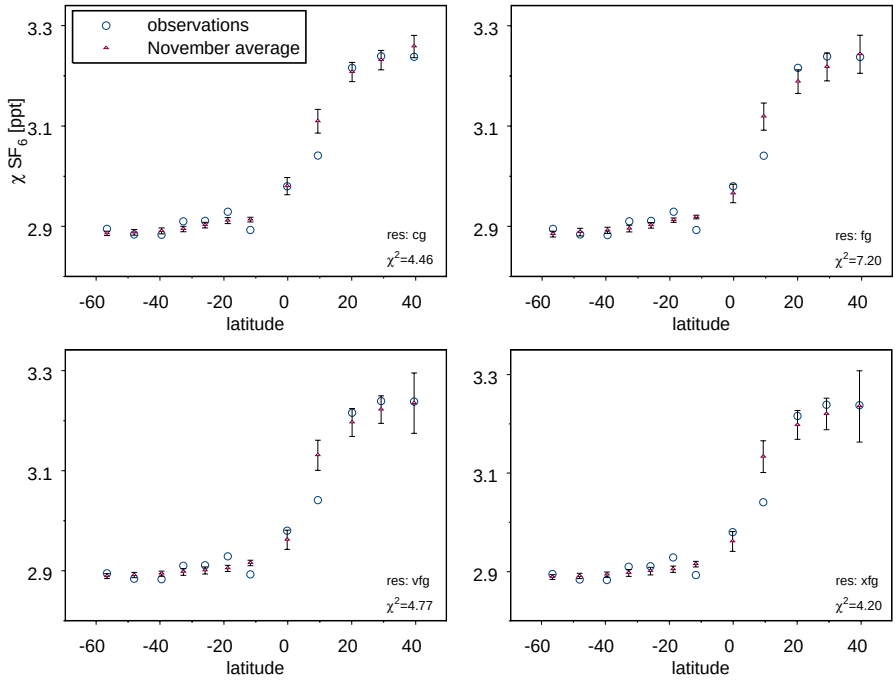
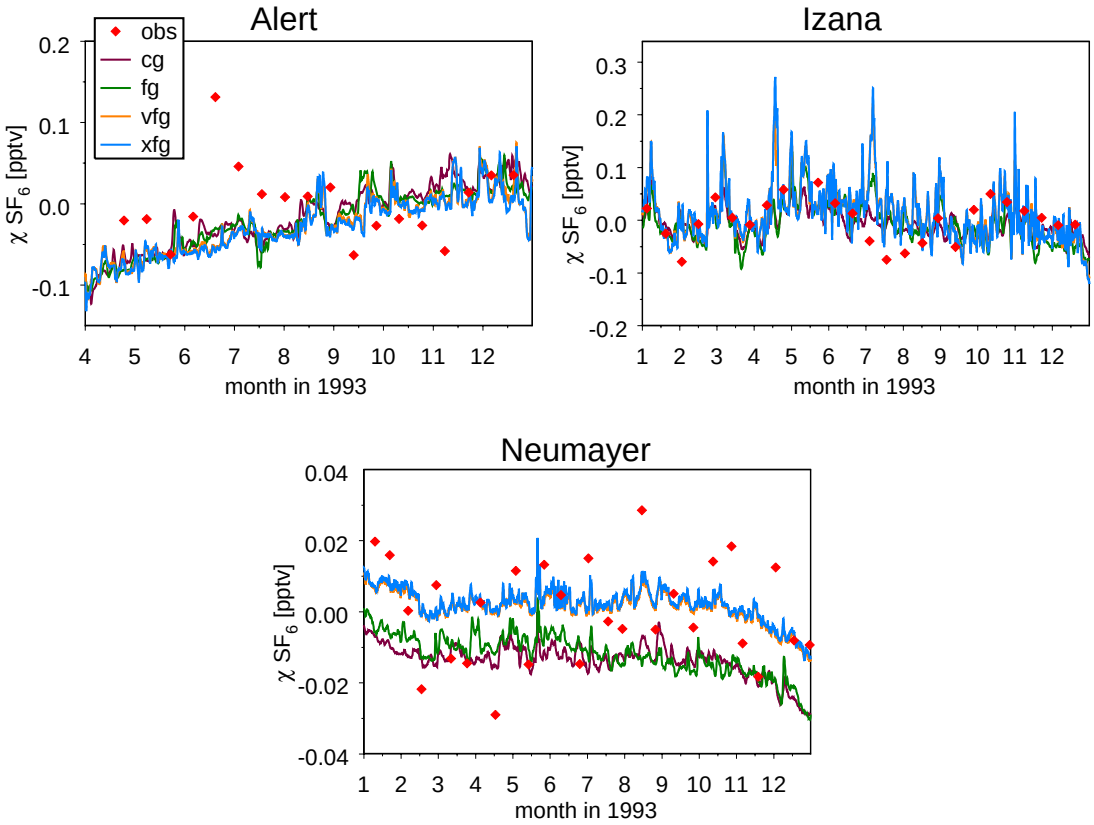


Figure F.2: Comparison of *Polarstern* transect measurements with TM3 November means

Time-Series at Background Stations

In the following panel, SF_6 mixing ratio time-series for three sampling sites are depicted and compared with modelling results, for Alert (82.45°N 62.51°W), Izaña (28.3°N 16.48°W) and Neumayer (70.66°S 8.25°W) (from [Maiss et al., 1996]). Offsets of ~ 0.1 pptv were added to the modelling results, and a quadratic trend has been subtracted from modelled and observed mixing ratios.



Furthermore, the annual mean mixing ratio difference between the northern (arctic) background station Alert and the southern (antarctic) background station Neumayer are given below:

	obs	cg	fg	vfg	xfg
$\overline{\Delta\chi}$ [pptv]	0.396	0.381	0.377	0.362	0.359

Interhemispheric Transport

Interhemispheric Exchange Times τ_{ex} This quantity serves as measure for the resistance of the tropical circulation system (including the intertropical convergence zone ITCZ) to interhemispheric air mass exchange. It is derived from a linear ansatz between the north-south tracer mass flux and the hemispheric tracer mass difference:

$$\bar{F}_{NS} = \frac{1}{\tau_{ex}} \left(\bar{M}_N - \bar{M}_S\right).$$

Here, $\bar{F}_{NS}$ denotes the temporally averaged mass flux from the northern to the southern hemisphere and $\bar{M}_{N/S}$ the temporally averaged hemispheric tracer masses.

For the present computational experiment, the averaging period stretches across the year 1993. A "two boxes" approximation was used:

$$\tau_{ex} = 2\frac{\bar{M}_N - \bar{M}_S}{\bar{E}_N - \bar{E}_S - \overline{\left(\frac{d(M_N(t)-M_S(t))}{dt}\right)}},$$

where $\bar{E}_N$ and $\bar{E}_S$ denote the emission rate in the northern resp. southern hemisphere. The anually averaged time derivative in the denominator was approximated by linear regression of the time series $M_{N/S}(t)$, with typical regression coefficients $r^2 \simeq 0.7$. The results show a resolution dependency:

resolution	cg	fg	vfg	xfg
τ_{ex} [y]	1.18	0.951	0.878	0.865

Pole-to-Pole Surface Mixing Ratio Difference This quantity was computed as annual average for 1993:

resolution	cg	fg	vfg	xfg
$\bar{\chi}_{NP} - \bar{\chi}_{SP}$ [pptv]	0.3897	0.3865	0.3692	0.3668

F.2 Regional Transport: The ATMES-II Modelling Exercise

Tracer release experiments represent an outstanding opportunity for the verification of atmospheric transport models because of two reasons. Firstly, by controlled release of an easily detectable agent the main disadvantage of other verification methods, the lack of exact knowledge about flux strength and location of the tracer source, is removed. Secondly, by organizing a dense sampling network prior to the release, the real atmospheric tracer distribution at the surface or in 3d may be rather well known..

ETEX, the *E*uropean *T*racer *EX*periment, included two releases of cyclic perfluorocarbons at Monterfil in the North-West of France (48°03'30"N, 2°00'30"W), with westerly flows over western and central Europe during the experiment. The first release started at 16:00 UTC on October 23, 1994, and lasted 11 hours and 50 minutes. 340 kg of PMCH (perfluoromethylcyclohexane) were released. 168 ground-level sampling stations in western and eastern Europe established the sampling network with an average spacing between two sampling stations of about 80 km (figure F.3). The sampling timestep was 3 hours on average. Airborne campaigns with three aircrafts complemented the ground network.

About two years after the ETEX releases, the ATMES-II (Atmospheric Transport Model Evaluation Study II) modelling exercise was launched. Participants were required to calculate the concentration fields of the first ETEX tracer experiment using ECMWF analyzed meteorological data as input of their own atmospheric transport model. The requirement of using ECMWF data should point out the differences of the participating models in respect of transport calculation and usage of the meteorological data. But this requirement was not compulsory.

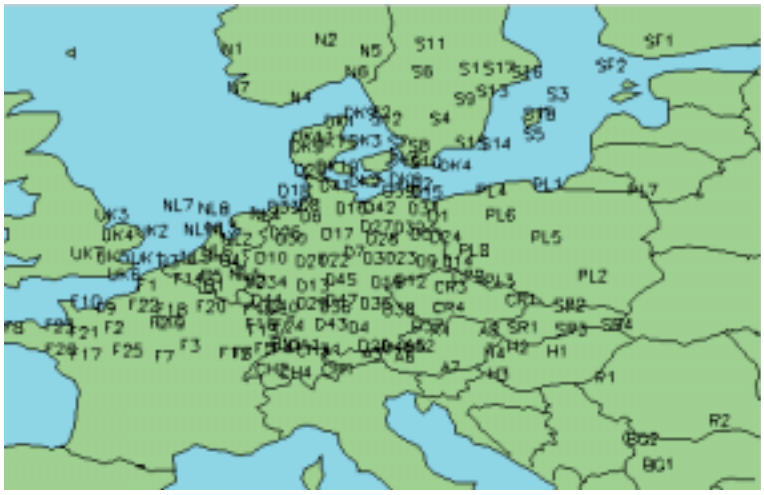


Figure F.3: The ETEX sampling network with the respective station-IDs (figure taken from [ETEX homepage]).

The TM3 simulation setup for the ATMES-II modelling exercise comprised a model run for the concerning time interval with the highest available input data resolution for that time, the vfg resolution of the NCEP/NCAR reanalysis data (1.875°x1.875°, 28 layers) with TM3 3.8a. A three-hourly concentration output at all 168 ground-level station locations, synchronous to the reference times of the observations, constituted the data base for the analyses in the subsequent paragraphs. The observational dataset (version etex1_v1.1.960505) was obtained from [ETEX homepage].

Analysis of the Model Results

The following definitions of statistical indicators follow those of the ATMES-II report [Mosca et al., 1998], in order to be comparable with other modelling results. The analysis is subdivided into global analysis, where time- and spacial information is disregarded, spacial analysis, where the time instants are fixed, and temporal analysis, where the locations are fixed.

As one can observe in figure F.4 on p. 128, the point tracer source creates a plume with sharp gradients on small scales. This situation makes high demands on the model's advection package. Furthermore, to resolve spatial structures comparable in scale with the station mesh-width of about 80 km, a model resolution of at least 40 km would be required[1]. With the available horizontal resolution of about 135 km x 210 km one cannot expect too much.

Definitions

In the following list of computed indicators, χ denotes the tracer concentration, P_i the prediction (model value) at time/location i, and M_i the measurement at time/location i.

[1]With a standard finite differences transport scheme. The situation with TM3 is slightly better, because it uses additional sub-scale information of the tracer distribution within the grid cells.

B Bias, the average difference between paired predictions and measurements, $B = \frac{1}{N}\sum_i (P_i - M_i)$, is an indicator for over- or underestimation.

F_{ex} Factor of exceedance. Let $N_{P>M}$ be the number of (P_i, M_i)-pairs where the predictions exceed the measurements. The factor of exceedance is the fraction of predictions that are too large, normalized to give zero in case of an equal number of over- and underpredictions: $F_{ex} = \frac{N_{P>M}}{N} - \frac{1}{2}$. It can take values between $-\frac{1}{2}$ (everything is underpredicted) and $+\frac{1}{2}$ (everything is overpredicted).

FA_2 The FA_k coefficient is the fraction of predictions within a factor k from the measurements, $FA_k = prob\,(P \leq k \cdot M \wedge P \geq M/k)$, for non-zero measurement values. FA_k can take values between 0 and 1.

$NMSE$ The normalized mean square error $NMSE$ is an index for the magnitude of the deviations between predictions and measurements: $NMSE = \frac{1}{NPM}\sum_i (P_i - M_i)^2$.

$\bar{P}$, $\bar{M}$ Mean

ϱ Pearson's linear correlation coefficient, $-1 \leq \varrho \leq 1$.

t_a Arrival time of the tracer cloud at a fixed location. Earliest time instant with $\chi(t) \geq \chi_{max}/e$.

t_d Duration time of the tracer cloud at a fixed location. Time interval with $\chi(t) \geq \chi_{max}/e$.

Global Analysis

FA_2	F_{ex}	$\bar{P}$ [ng/m^3]	$\bar{M}$ [ng/m^3]	ϱ	B [ng/m^3]	$NMSE$
0.315	0.35	0.15	0.095	0.43	0.057	16.8

Numerical diffusion, probably induced by the comparatively coarse model resolution, leads to overpredictions at many stations. The factor of exceedance F_{ex} indicates 85 % overpredictions. A positive bias B of 0.057 ng/m^3 indicates this trend also. That the bias B does not equal the difference of the means originates from missing values in the observational dataset. Only 31.5 % of the predictions are within a factor of 2 around the measurements. The correlation coefficient of $\varrho^2 = 0.18$ is small compared to other ATMES-II models.

Spatial Analysis

This part of the analysis is restricted to a figure of concentration contours at different times after the release start (fig. F.4, p. 128). Probably numerical diffusion, induced by the coarse model resolution, is the reason for comparatively small maximum tracer concentrations on the modelling side, as well as for oversized horizontal dimensions of the tracer cloud. The splitting into a north-western and south-eastern part, as visible in the observations from 36 hours after release start, could not be reproduced by the model at all.

Temporal Analysis

The temporal analysis was carried out for 11 stations, forming approximately two arcs perpendicular to the main cloud trajectory, the first arc at a distance of approximately 600 km, the second at a distance of about 1200-1400 km from the source (figure F.5, p. 129).

Station-ID	$\Delta\chi_{\max}$ [ng/m^3]	Δt_a [h]	Δt_d [h]	ϱ	B [ng/m^3]	*NMSE*
NL05	-1.35	-18	15	-0.05	0.02	6.8
B05	0.56	-15	15	0.29	0.24	16
NL01	-0.43	-12	9	0.31	0.14	5.8
D44	0.66	0	0	0.98	0.16	2.7
DK05	-1.70	-6	12	0.77	-0.19	12
DK02	-0.27	-6	9	0.78	0.02	1.2
D42	-1.01	-6	3	0.72	-0.11	2
D05	-0.18	0	3	0.94	0.02	0.27
PL03	-0.32	-3	3	0.84	-0.03	1.2
CR03	-0.05	0	3	0.92	0.03	0.64
H02	-0.36	0	12	0.76	-0.03	3.8

Underpredictions of the peak height at all stations but two characterize the scene here (negative $\Delta\chi_{\max}$). The tracer front arrives at the first arc far too early, what might be attributed again to numerical diffusion, as well as the high positive bias values especially for the stations of the 1st arc. The duration of the front is too long, partially much too long.

F.2.1 Conclusions

Underpredictions of the peak concentrations but mainly overpredictions in the "global" average, tracer fronts that arrive too early and overlong front durations indicate that the used model resolution is too coarse for this kind of experiment, leading to numercial diffusion as prominent error source. One main feature of the large-scale transport, the splitting into two sub-clouds over the Netherlands, is not reproduced at all.

This experiment will be repeated when more highly-resolved preprocessing data from the ECMWF ERA-40 reanalysis are available.

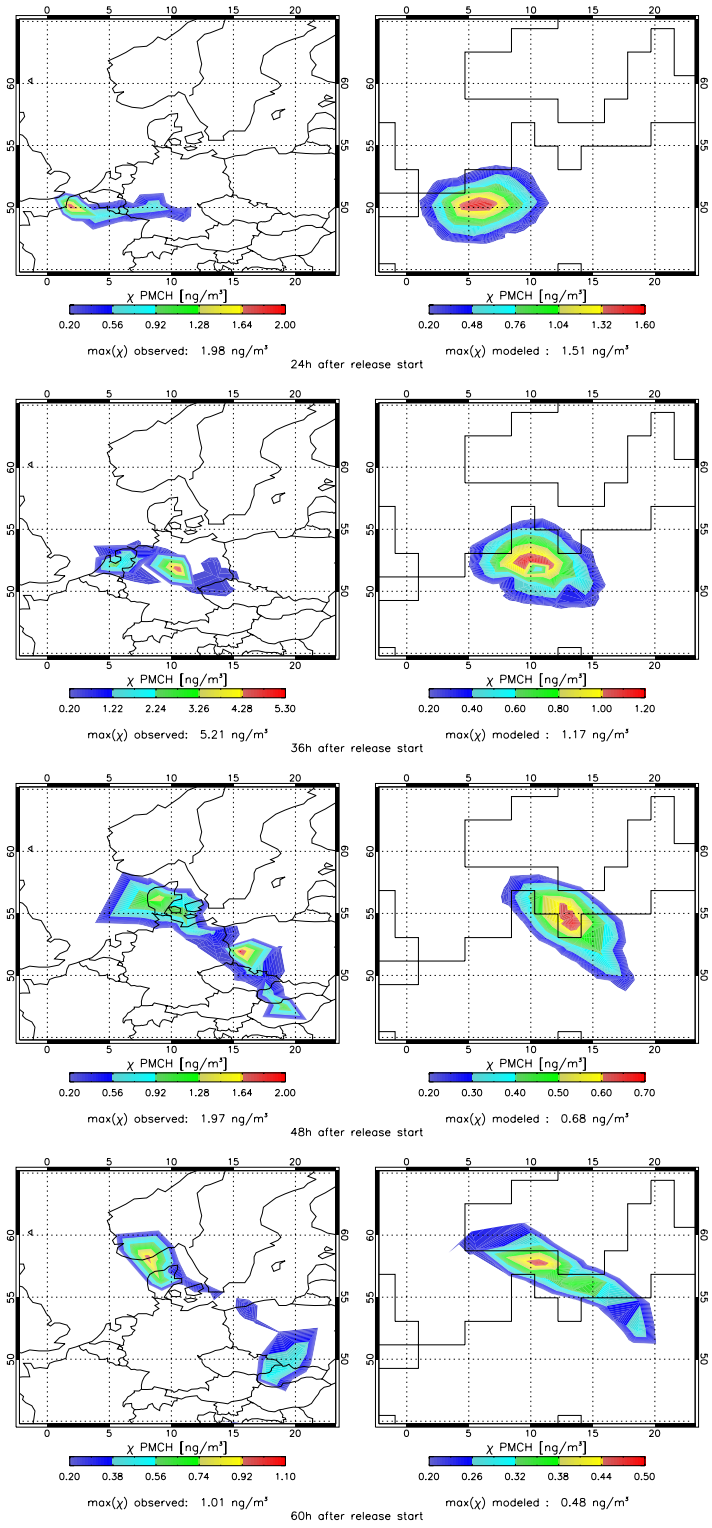


Figure F.4: PMCH surface concentrations, as observed (left column) and modeled by TM3 (right column, different scale!), for 24, 36, 48 and 60 hours after release start. Contouring method was triangular interpolation.

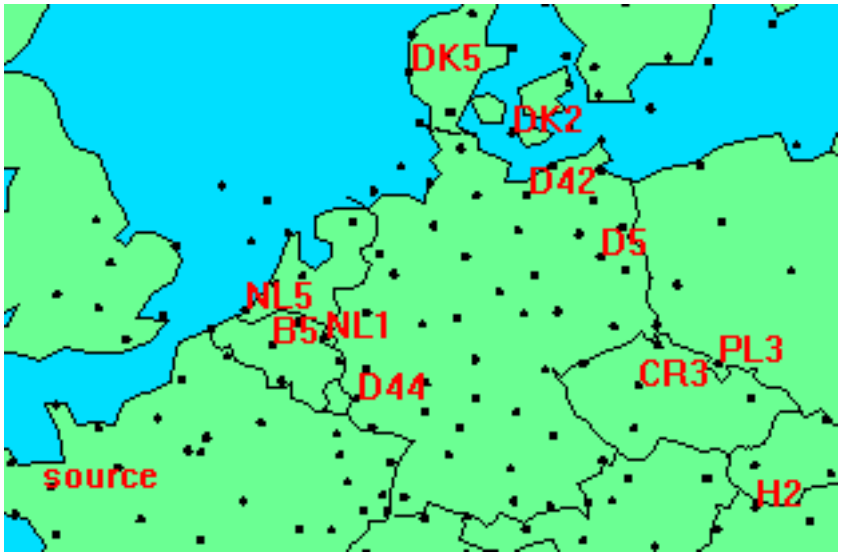


Figure F.5: Location of the stations selected for temporal analysis (figure taken from [ETEX homepage]).

Appendix G

Getting the Code

To get the most recent TM3 sources, preprocessed meteorological input files and the up-to-date manual please contact

Martin Heimann
Max Planck Institute for Biochemistry
P.O. Box 10 01 64
07701 Jena
GERMANY

martin.heimann@bgc-jena.mpg.de

There exists a TM3 mailing list to swap experiences and make suggestions. To sign in, please contact Martin Heimann.

Bibliography

[Denning et al., 1999] A. Scott Denning et al., Three-Dimensional Transport and Concentration of SF_6: A Model Intercomparison Study (TransCom 2). Tellus Series B - Chemical & Physical Meteorology **51**(2) (1999), 266–297

[ETEX homepage] ETEX homepage at http://rem.jrc.cec.eu.int/etex

[Heimann and Keeling, 1989] M. Heimann and C.D. Keeling: A three-dimensional model of atmospheric CO_2 transport based on observed winds: 2. Model description and simulated tracer experiments. Aspects of Climate Variability in the Pacific and the Western Americas (1989), 277–303

[Louis, 1979] Jean-François Louis: A Parametric Model of Vertical Eddy Fluxes in the Atmosphere. Boundary Layer Meteorology **17** (1979), 187–202

[Maiss et al., 1996] Maiss, M., L.P. Steele, R.J. Francey, P.J. Fraser, R.L. Langenfels, N.B.A. Trivett and I. Levin, Sulfur hexafluoride - a powerful new atmospheric tracer. Atmosph. Environment **30** (1996), 1621-1629

[Mosca et al., 1998] S. Mosca, R. Bianconi, R. Bellasio, G. Graziani, W. Klug, ATMES II - Evaluation of Long-Range Dispersion Models using Data of the 1st ETEX release. ISBN 92-828-3655-X (1998)

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

[Russel and Lerner, 1981] Gary L. Russell, Jean A. Learner: A New Finite-Differencing Scheme for the Tracer Transport Equation. Journal of Applied Meteorology **20** (1981), 1483–1499

[Tiedke, 1989] M. Tiedke: A Comprehensive Mass Flux Scheme for Cumulus Parametrization in Large-Scale Models. Monthly Weather Review **117** (1989), 1779–1800